Unidesc

Eduarda Silva Forte/ Evaldo Junio Bueno Cortes

OAT de Programação Orientada a objetos

Códigos em Java

Ocidental

2025

Eduarda Silva Forte/ Evaldo Junior Bueno Cortes

OAT de Programação Orientada a objetos

Códigos em Java

Este trabalho apresenta o desenvolvimento de uma aplicação simples em Java utilizando a biblioteca Swing para construção de interfaces gráficas. O projeto consiste em um sistema de cadastro de clientes com funcionalidades básicas, como inserção, alteração, exclusão, limpeza de campos e geração de relatórios simulados. A interface gráfica é composta por campos de texto, caixas de seleção e botões que executam ações específicas por meio de eventos associados. O principal objetivo é proporcionar uma base prática de manipulação de componentes gráficos, aplicação de lógica com `ActionListener`, e organização de dados em formulários visuais. Cada funcionalidade foi implementada separadamente para facilitar o entendimento do comportamento individual dos botões "Novo", "Salvar", "Alterar", "Excluir", "Relatório" e "Retornar". O projeto foi elaborado como parte da OAT (Atividade Avaliativa) da disciplina de Programação Orientada a Objetos na instituição Unidesc, no ano de 2025.

Ocidental

2025

código em java de cadastro de clientes

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class CadastroClientes {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Cadastro de Clientes");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 400);

        // Painel principal
        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(10, 2));

        // Campos do formulário
        panel.add(new JLabel("Código:"));
        JTextField txtCodigo = new JTextField();
        panel.add(txtCodigo);

        panel.add(new JLabel("Nome:"));
        JTextField txtNome = new JTextField();
        panel.add(txtNome);

        panel.add(new JLabel("Endereço:"));
        JTextField txtEndereco = new JTextField();
        panel.add(txtEndereco);

        panel.add(new JLabel("Bairro:"));
        JTextField txtBairro = new JTextField();
        panel.add(txtBairro);

        panel.add(new JLabel("Cidade:"));
        JTextField txtCidade = new JTextField();
        panel.add(txtCidade);

        panel.add(new JLabel("Estado:"));
        String[] estados = {"SP", "RJ", "MG", "RS"}; // Exemplo de estados
        JComboBox<String> cbEstado = new JComboBox<>(estados);
        panel.add(cbEstado);
```

```
        panel.add(new JLabel("CEP:"));
        JTextField txtCep = new JTextField();
        panel.add(txtCep);

        panel.add(new JLabel("Telefone:"));
        JTextField txtTelefone = new JTextField();
        panel.add(txtTelefone);

        panel.add(new JLabel("Observações:"));
        JTextArea txtObservacoes = new JTextArea();
        panel.add(txtObservacoes);

        // Botões
        JButton btnSalvar = new JButton("Salvar");
        JButton btnAlterar = new JButton("Alterar");
        JButton btnExcluir = new JButton("Excluir");
        JButton btnNovo = new JButton("Novo");

        // Adicionando ações aos botões
        btnSalvar.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // Implementação da lógica de salvar
                JOptionPane.showMessageDialog(frame, "Cadastro salvo!");
            }
        });

        // Adicionando botões ao painel
        panel.add(btnNovo);
        panel.add(btnSalvar);
        panel.add(btnAlterar);
        panel.add(btnExcluir);

        frame.add(panel);
        frame.setVisible(true);
    }
}
```

Objetivo do Código

Criar uma janela (formulário) com campos para o usuário digitar dados de um cliente e bovvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvtões para realizar ações como salvar, alterar, excluir e criar um novo cadastro.

Importações

import javax.swing.*; // Componentes da interface gráfica, como JFrame, JButton, JTextField...
import java.awt.*; // Layouts e cores
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener; // Para lidar com cliques de botões

vvvvvvvvvClasse Principal

public class CadastroClientes {

Essa é a classe principal que contém o programa.

Método main

public static void main(String[] args) {

Ponto de entrada do programa. É aqui que a janela será criada e configurada.

Janela principal (JFrame)

JFrame frame = new JFrame("Cadastro de Clientes");

Cria a janela com o título “Cadastro de Clientes”.

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

Fecha o programa quando o usuário fechar a janela.

frame.setSize(400, 400);

Define o tamanho da janela (400x400 pixels).

Painel principal (JPanel)

```
JPanel panel = new JPanel();
panel.setLayout(new GridLayout(10, 2));
```

Cria o "painel" onde os campos e botões serão colocados, com um layout em grade (10 linhas, 2 colunas).

Campos do Formulário

São campos para o usuário preencher:

- **Código**:
  `JTextField txtCodigo = new JTextField();`
- **Nome**:
  `JTextField txtNome = new JTextField();`
- **Endereço**, **Bairro**, **Cidade**, **CEP**, **Telefone**:
  Todos são JTextField.
- **Estado**:
  Um JComboBox com opções: SP, RJ, MG, RS.
- **Observações**:
  Um campo de texto mais longo (JTextArea).

Botões

```
JButton btnSalvar = new JButton("Salvar");
JButton btnAlterar = new JButton("Alterar");
JButton btnExcluir = new JButton("Excluir");
JButton btnNovo = new JButton("Novo");
```

Botões que o usuário pode clicar.

Ação do botão “Salvar”

```
btnSalvar.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JOptionPane.showMessageDialog(frame, "Cadastro salvo!");
    }
});
```

Quando o botão "Salvar" é clicado, aparece uma mensagem dizendo "Cadastro salvo!".

Adicionando ao Painel

Todos os campos e botões são adicionados ao painel usando panel.add(...).

Finalizando a janela

```
frame.add(panel);
frame.setVisible(true);
```

- O painel com todos os componentes é adicionado à janela.
- A janela é exibida na tela.

Resumo visual do que aparece:

- Um formulário com campos: código, nome, endereço, etc.
- Um menu suspenso para escolher o estado.
- Um campo grande de observações.
- Botões: Novo, Salvar, Alterar, Excluir.
- Ao clicar em “Salvar”, aparece uma mensagem.

Apresentação do Layout com Navegação entre Telas

O sistema desenvolvido apresenta uma **estrutura de navegação simples e funcional entre três telas principais**, organizadas em **janelas separadas (`JFrame`)**, que se conectam entre si por meio de **botões com eventos de ação**.

Telas do Sistema

1. **Tela de Login (`LoginTela.java`)**
    - É a tela inicial do sistema.
    - Contém campos para usuário e senha, e um botão “Entrar”.
    - Após validação (usuário: `admin`, senha: `123`), o sistema redireciona o usuário para a tela **Home**.
2. **Tela Home (`HomeTela.java`)**
    - Mostra uma mensagem de boas-vindas e um botão “Cadastro de Clientes”.
    - Ao clicar nesse botão, a tela **CadastroClientes** é aberta, permitindo acesso às funcionalidades do sistema.
3. **Tela de Cadastro de Clientes (`CadastroClientes.java`)**
    - Tela mais completa, onde é possível **inserir, alterar, excluir, visualizar relatório, limpar e retornar**.
    - Botão “Retornar” permite voltar para a tela **Home**, garantindo a navegação reversa.

Fluxo de Navegação entre Telas

```
LoginTela --> HomeTela --> CadastroClientes

                ↑                ↓

← ← ← ← ← ← ← ← ← ←
```

A transição entre as janelas é feita com o uso de:

```
new NomeDaTela();

dispose(); // fecha a tela atual
```

-
- Isso garante uma navegação sequencial e intuitiva, simulando um sistema real de múltiplas telas, como em um software profissional.

Layout

- As telas utilizam **`GridLayout`** e **`FlowLayout`** para organizar os componentes visualmente.
- O layout foi planejado para ser **simples, limpo e funcional**, focando na usabilidade.
- Os botões foram organizados de forma clara e com ações definidas, facilitando a interação do usuário.

tela de cadastro de clientes completa em Java utilizando **Swing**, totalmente integrada ao estilo das telas anteriores (`LoginTela` e `HomeTela`). Essa tela permite:

- Preenchimento de dados de um cliente;
- Botões com ações (Novo, Salvar, Alterar, Excluir, Relatório, Retornar);
- Layout limpo e funcional com navegação entre telas.

Código: Tela de Cadastro de Clientes (CadastroClientes.java)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class CadastroClientes extends JFrame {

    public CadastroClientes() {
```

```
setTitle("Cadastro de Clientes");
setSize(500, 500);
setDefaultCloseOperation(EXIT_ON_CLOSE);

JPanel panel = new JPanel(new GridLayout(11, 2, 5, 5));

JTextField txtCodigo = new JTextField();
JTextField txtNome = new JTextField();
JTextField txtEndereco = new JTextField();
JTextField txtBairro = new JTextField();
JTextField txtCidade = new JTextField();
String[] estados = {"SP", "RJ", "MG", "RS"};
JComboBox<String> cbEstado = new JComboBox<>(estados);
JTextField txtCep = new JTextField();
JTextField txtTelefone = new JTextField();
JTextArea txtObservacoes = new JTextArea();

panel.add(new JLabel("Código:")); panel.add(txtCodigo);
panel.add(new JLabel("Nome:")); panel.add(txtNome);
panel.add(new JLabel("Endereço:")); panel.add(txtEndereco);
panel.add(new JLabel("Bairro:")); panel.add(txtBairro);
panel.add(new JLabel("Cidade:")); panel.add(txtCidade);
panel.add(new JLabel("Estado:")); panel.add(cbEstado);
panel.add(new JLabel("CEP:")); panel.add(txtCep);
panel.add(new JLabel("Telefone:")); panel.add(txtTelefone);
panel.add(new JLabel("Observações:")); panel.add(txtObservacoes);

JButton btnNovo = new JButton("Novo");
JButton btnSalvar = new JButton("Salvar");
JButton btnAlterar = new JButton("Alterar");
JButton btnExcluir = new JButton("Excluir");
JButton btnRelatorio = new JButton("Relatório");
JButton btnRetornar = new JButton("Retornar");

btnNovo.addActionListener(e -> {
    txtCodigo.setText("");
    txtNome.setText("");
    txtEndereco.setText("");
    txtBairro.setText("");
    txtCidade.setText("");
    cbEstado.setSelectedIndex(0);
    txtCep.setText("");
    txtTelefone.setText("");
    txtObservacoes.setText("");
});

btnSalvar.addActionListener(e -> {
    String dados = "Cadastro salvo:\n"
```

```
            + "Código: " + txtCodigo.getText() + "\n"
            + "Nome: " + txtNome.getText() + "\n"
            + "Endereço: " + txtEndereco.getText();
        JOptionPane.showMessageDialog(this, dados);
    });

    btnAlterar.addActionListener(e -> {
        JOptionPane.showMessageDialog(this, "Cadastro alterado com sucesso!");
    });

    btnExcluir.addActionListener(e -> {
        int opcao = JOptionPane.showConfirmDialog(this, "Deseja excluir este cadastro?",
"Confirmar", JOptionPane.YES_NO_OPTION);
        if (opcao == JOptionPane.YES_OPTION) {
            txtCodigo.setText("");
            txtNome.setText("");
            txtEndereco.setText("");
            txtBairro.setText("");
            txtCidade.setText("");
            cbEstado.setSelectedIndex(0);
            txtCep.setText("");
            txtTelefone.setText("");
            txtObservacoes.setText("");
            JOptionPane.showMessageDialog(this, "Cadastro excluído.");
        }
    });

    btnRelatorio.addActionListener(e -> {
        String relatorio = "Relatório:\n"
            + "Nome: " + txtNome.getText() + "\n"
            + "Cidade: " + txtCidade.getText();
        JOptionPane.showMessageDialog(this, relatorio);
    });

    btnRetornar.addActionListener(e -> {
        new HomeTela();
        dispose();
    });

    panel.add(btnNovo);     panel.add(btnSalvar);
    panel.add(btnAlterar);  panel.add(btnExcluir);
    panel.add(btnRelatorio);panel.add(btnRetornar);

    add(panel);
    setLocationRelativeTo(null);
    setVisible(true);
  }
}
```

tela de cadastro de Funcionários em Java usando **Swing**, no mesmo padrão da tela de clientes. Essa tela é adequada para ser uma das "telas individuais" de um integrante do grupo, conforme solicitado no seminário.

Código: Tela de Cadastro de Funcionários (CadastroFuncionarios.java)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class CadastroFuncionarios extends JFrame {

    public CadastroFuncionarios() {
        setTitle("Cadastro de Funcionários");
        setSize(500, 500);
        setDefaultCloseOperation(EXIT_ON_CLOSE);

        JPanel panel = new JPanel(new GridLayout(10, 2, 5, 5));

        JTextField txtId = new JTextField();
        JTextField txtNome = new JTextField();
        JTextField txtCargo = new JTextField();
        JTextField txtDepartamento = new JTextField();
        JTextField txtCpf = new JTextField();
        JTextField txtTelefone = new JTextField();
        JTextField txtEmail = new JTextField();
        JTextArea txtObservacoes = new JTextArea();

        panel.add(new JLabel("ID:")); panel.add(txtId);
        panel.add(new JLabel("Nome:")); panel.add(txtNome);
        panel.add(new JLabel("Cargo:")); panel.add(txtCargo);
        panel.add(new JLabel("Departamento:")); panel.add(txtDepartamento);
        panel.add(new JLabel("CPF:")); panel.add(txtCpf);
        panel.add(new JLabel("Telefone:")); panel.add(txtTelefone);
        panel.add(new JLabel("Email:")); panel.add(txtEmail);
        panel.add(new JLabel("Observações:")); panel.add(txtObservacoes);

        JButton btnNovo = new JButton("Novo");
        JButton btnSalvar = new JButton("Salvar");
        JButton btnAlterar = new JButton("Alterar");
        JButton btnExcluir = new JButton("Excluir");
        JButton btnRelatorio = new JButton("Relatório");
        JButton btnRetornar = new JButton("Retornar");

        btnNovo.addActionListener(e -> {
            txtId.setText("");
```

```
            txtNome.setText("");
            txtCargo.setText("");
            txtDepartamento.setText("");
            txtCpf.setText("");
            txtTelefone.setText("");
            txtEmail.setText("");
            txtObservacoes.setText("");
        });

        btnSalvar.addActionListener(e -> {
            String dados = "Funcionário salvo:\n"
                + "ID: " + txtId.getText() + "\n"
                + "Nome: " + txtNome.getText() + "\n"
                + "Cargo: " + txtCargo.getText();
            JOptionPane.showMessageDialog(this, dados);
        });

        btnAlterar.addActionListener(e -> {
            JOptionPane.showMessageDialog(this, "Dados do funcionário alterados com
sucesso!");
        });

        btnExcluir.addActionListener(e -> {
            int opcao = JOptionPane.showConfirmDialog(this, "Deseja excluir este funcionário?",
"Confirmar", JOptionPane.YES_NO_OPTION);
            if (opcao == JOptionPane.YES_OPTION) {
                txtId.setText("");
                txtNome.setText("");
                txtCargo.setText("");
                txtDepartamento.setText("");
                txtCpf.setText("");
                txtTelefone.setText("");
                txtEmail.setText("");
                txtObservacoes.setText("");
                JOptionPane.showMessageDialog(this, "Funcionário excluído.");
            }
        });

        btnRelatorio.addActionListener(e -> {
            String relatorio = "Relatório:\n"
                + "Nome: " + txtNome.getText() + "\n"
                + "Cargo: " + txtCargo.getText();
            JOptionPane.showMessageDialog(this, relatorio);
        });

        btnRetornar.addActionListener(e -> {
            new HomeTela(); // Volta para a tela Home
            dispose();
```

```
        });

        panel.add(btnNovo);      panel.add(btnSalvar);
        panel.add(btnAlterar);   panel.add(btnExcluir);
        panel.add(btnRelatorio); panel.add(btnRetornar);

        add(panel);
        setLocationRelativeTo(null);
        setVisible(true);
    }
}
```

tela de cadastro de Fornecedores feita com **Java Swing**, no mesmo estilo das telas anteriores (Clientes e Funcionários). Essa tela pode ser usada como uma das funcionalidades individuais no seu seminário.

Código: Cadastro de Fornecedores (CadastroFornecedores.java)

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.*;


public class CadastroFornecedores extends JFrame {


    public CadastroFornecedores() {

        setTitle("Cadastro de Fornecedores");

        setSize(500, 500);

        setDefaultCloseOperation(EXIT_ON_CLOSE);


        JPanel panel = new JPanel(new GridLayout(10, 2, 5, 5));


        JTextField txtId = new JTextField();
```

```
        JTextField txtNome = new JTextField();

        JTextField txtCnpj = new JTextField();

        JTextField txtTelefone = new JTextField();

        JTextField txtEmail = new JTextField();

        JTextField txtEndereco = new JTextField();

        JTextField txtCidade = new JTextField();

          JComboBox<String> cbEstado = new JComboBox<>(new String[]{"SP", "RJ",
"MG", "RS"});

        JTextArea txtObservacoes = new JTextArea();


        panel.add(new JLabel("ID:")); panel.add(txtId);

        panel.add(new JLabel("Nome:")); panel.add(txtNome);

        panel.add(new JLabel("CNPJ:")); panel.add(txtCnpj);

        panel.add(new JLabel("Telefone:")); panel.add(txtTelefone);

        panel.add(new JLabel("Email:")); panel.add(txtEmail);

        panel.add(new JLabel("Endereço:")); panel.add(txtEndereco);

        panel.add(new JLabel("Cidade:")); panel.add(txtCidade);

        panel.add(new JLabel("Estado:")); panel.add(cbEstado);

        panel.add(new JLabel("Observações:")); panel.add(txtObservacoes);


        JButton btnNovo = new JButton("Novo");

        JButton btnSalvar = new JButton("Salvar");

        JButton btnAlterar = new JButton("Alterar");

        JButton btnExcluir = new JButton("Excluir");

        JButton btnRelatorio = new JButton("Relatório");
```

```
JButton btnRetornar = new JButton("Retornar");


btnNovo.addActionListener(e -> {
    txtId.setText("");
    txtNome.setText("");
    txtCnpj.setText("");
    txtTelefone.setText("");
    txtEmail.setText("");
    txtEndereco.setText("");
    txtCidade.setText("");
    cbEstado.setSelectedIndex(0);
    txtObservacoes.setText("");
});


btnSalvar.addActionListener(e -> {
    String dados = "Fornecedor salvo:\n"
        + "ID: " + txtId.getText() + "\n"
        + "Nome: " + txtNome.getText() + "\n"
        + "CNPJ: " + txtCnpj.getText();
    JOptionPane.showMessageDialog(this, dados);
});


btnAlterar.addActionListener(e -> {
     JOptionPane.showMessageDialog(this, "Dados do fornecedor alterados com
sucesso!");
```

```
    });


    btnExcluir.addActionListener(e -> {

            int opcao = JOptionPane.showConfirmDialog(this, "Deseja excluir este fornecedor?", "Confirmar", JOptionPane.YES_NO_OPTION);

        if (opcao == JOptionPane.YES_OPTION) {

            txtId.setText("");

            txtNome.setText("");

            txtCnpj.setText("");

            txtTelefone.setText("");

            txtEmail.setText("");

            txtEndereco.setText("");

            txtCidade.setText("");

            cbEstado.setSelectedIndex(0);

            txtObservacoes.setText("");

            JOptionPane.showMessageDialog(this, "Fornecedor excluído.");

        }

    });


    btnRelatorio.addActionListener(e -> {

        String relatorio = "Relatório:\n"

            + "Nome: " + txtNome.getText() + "\n"

            + "CNPJ: " + txtCnpj.getText();

        JOptionPane.showMessageDialog(this, relatorio);

    });
```

```
        btnRetornar.addActionListener(e -> {
            new HomeTela(); // Volta para a tela Home
            dispose();
        });

        panel.add(btnNovo);      panel.add(btnSalvar);
        panel.add(btnAlterar);   panel.add(btnExcluir);
        panel.add(btnRelatorio); panel.add(btnRetornar);

        add(panel);
        setLocationRelativeTo(null);
        setVisible(true);
    }
}
```

tela de cadastro de Recebidores (ou quem recebe algum produto/serviço/pagamento no sistema), no mesmo padrão visual e funcional das demais telas (Clientes, Funcionários, Fornecedores).

Código: Cadastro de Recebidores (CadastroRecebidores.java)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class CadastroRecebidores extends JFrame {
```

```
    public CadastroRecebidores() {
        setTitle("Cadastro de Recebidores");
        setSize(500, 500);
        setDefaultCloseOperation(EXIT_ON_CLOSE);

        JPanel panel = new JPanel(new GridLayout(10, 2, 5, 5));

        JTextField txtId = new JTextField();
        JTextField txtNome = new JTextField();
        JTextField txtCpf = new JTextField();
        JTextField txtTelefone = new JTextField();
        JTextField txtEmail = new JTextField();
        JTextField txtEndereco = new JTextField();
        JTextField txtCidade = new JTextField();
          JComboBox<String> cbEstado = new JComboBox<>(new String[]{"SP", "RJ",
"MG", "RS"});
        JTextArea txtObservacoes = new JTextArea();

        panel.add(new JLabel("ID:")); panel.add(txtId);
        panel.add(new JLabel("Nome:")); panel.add(txtNome);
        panel.add(new JLabel("CPF:")); panel.add(txtCpf);
        panel.add(new JLabel("Telefone:")); panel.add(txtTelefone);
        panel.add(new JLabel("Email:")); panel.add(txtEmail);
        panel.add(new JLabel("Endereço:")); panel.add(txtEndereco);
```

```
panel.add(new JLabel("Cidade:")); panel.add(txtCidade);
panel.add(new JLabel("Estado:")); panel.add(cbEstado);
panel.add(new JLabel("Observações:")); panel.add(txtObservacoes);

JButton btnNovo = new JButton("Novo");
JButton btnSalvar = new JButton("Salvar");
JButton btnAlterar = new JButton("Alterar");
JButton btnExcluir = new JButton("Excluir");
JButton btnRelatorio = new JButton("Relatório");
JButton btnRetornar = new JButton("Retornar");

btnNovo.addActionListener(e -> {
    txtId.setText("");
    txtNome.setText("");
    txtCpf.setText("");
    txtTelefone.setText("");
    txtEmail.setText("");
    txtEndereco.setText("");
    txtCidade.setText("");
    cbEstado.setSelectedIndex(0);
    txtObservacoes.setText("");
});

btnSalvar.addActionListener(e -> {
    String dados = "Recebidor salvo:\n"
```

```
            + "ID: " + txtId.getText() + "\n"
            + "Nome: " + txtNome.getText() + "\n"
            + "CPF: " + txtCpf.getText();
        JOptionPane.showMessageDialog(this, dados);
    });

    btnAlterar.addActionListener(e -> {
        JOptionPane.showMessageDialog(this, "Dados do recebidor alterados com sucesso!");
    });

    btnExcluir.addActionListener(e -> {
        int opcao = JOptionPane.showConfirmDialog(this, "Deseja excluir este recebidor?", "Confirmar", JOptionPane.YES_NO_OPTION);
        if (opcao == JOptionPane.YES_OPTION) {
            txtId.setText("");
            txtNome.setText("");
            txtCpf.setText("");
            txtTelefone.setText("");
            txtEmail.setText("");
            txtEndereco.setText("");
            txtCidade.setText("");
            cbEstado.setSelectedIndex(0);
            txtObservacoes.setText("");
            JOptionPane.showMessageDialog(this, "Recebidor excluído.");
        }
```

```
        });

        btnRelatorio.addActionListener(e -> {
            String relatorio = "Relatório:\n"
                + "Nome: " + txtNome.getText() + "\n"
                + "CPF: " + txtCpf.getText();
            JOptionPane.showMessageDialog(this, relatorio);
        });

        btnRetornar.addActionListener(e -> {
            new HomeTela(); // Voltar à tela Home
            dispose();
        });

        panel.add(btnNovo);      panel.add(btnSalvar);
        panel.add(btnAlterar);   panel.add(btnExcluir);
        panel.add(btnRelatorio); panel.add(btnRetornar);

        add(panel);
        setLocationRelativeTo(null);
        setVisible(true);
    }
}
```

código Java contendo apenas a funcionalidade do botão "Novo", que limpa todos os campos do formulário:

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;


public class CadastroClientesNovo {


    public static void main(String[] args) {

        JFrame frame = new JFrame("Cadastro de Clientes - Novo");

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        frame.setSize(400, 400);


        JPanel panel = new JPanel(new GridLayout(10, 2));


        // Campos do formulário

        JTextField txtCodigo = new JTextField();

        JTextField txtNome = new JTextField();

        JTextField txtEndereco = new JTextField();

        JTextField txtBairro = new JTextField();

        JTextField txtCidade = new JTextField();

        String[] estados = {"SP", "RJ", "MG", "RS"};

        JComboBox<String> cbEstado = new JComboBox<>(estados);
```

```
JTextField txtCep = new JTextField();
JTextField txtTelefone = new JTextField();
JTextArea txtObservacoes = new JTextArea();

// Adicionando os campos ao painel com seus rótulos
panel.add(new JLabel("Código:")); panel.add(txtCodigo);
panel.add(new JLabel("Nome:")); panel.add(txtNome);
panel.add(new JLabel("Endereço:")); panel.add(txtEndereco);
panel.add(new JLabel("Bairro:")); panel.add(txtBairro);
panel.add(new JLabel("Cidade:")); panel.add(txtCidade);
panel.add(new JLabel("Estado:")); panel.add(cbEstado);
panel.add(new JLabel("CEP:")); panel.add(txtCep);
panel.add(new JLabel("Telefone:")); panel.add(txtTelefone);
panel.add(new JLabel("Observações:")); panel.add(txtObservacoes);

// Botão Novo
JButton btnNovo = new JButton("Novo");
btnNovo.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        txtCodigo.setText("");
        txtNome.setText("");
        txtEndereco.setText("");
        txtBairro.setText("");
        txtCidade.setText("");
        cbEstado.setSelectedIndex(0);
```

```
            txtCep.setText("");
            txtTelefone.setText("");
            txtObservacoes.setText("");
        }
    });

    // Adiciona o botão ao painel
    panel.add(btnNovo);

    frame.add(panel);
    frame.setVisible(true);
  }
}
```

código Java contendo apenas a funcionalidade do botão "Alterar", que simula a alteração de dados preenchidos, exibindo uma mensagem com os valores atuais dos campos:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class CadastroClientesAlterar {
```

```
public static void main(String[] args) {
    JFrame frame = new JFrame("Cadastro de Clientes - Alterar");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400, 400);

    JPanel panel = new JPanel(new GridLayout(10, 2));

    // Campos do formulário
    JTextField txtCodigo = new JTextField("123");
    JTextField txtNome = new JTextField("João da Silva");
    JTextField txtEndereco = new JTextField("Rua Exemplo, 123");
    JTextField txtBairro = new JTextField("Centro");
    JTextField txtCidade = new JTextField("São Paulo");
    String[] estados = {"SP", "RJ", "MG", "RS"};
    JComboBox<String> cbEstado = new JComboBox<>(estados);
    cbEstado.setSelectedItem("SP");
    JTextField txtCep = new JTextField("01000-000");
    JTextField txtTelefone = new JTextField("(11) 99999-9999");
    JTextArea txtObservacoes = new JTextArea("Cliente antigo");

    // Adicionando os campos ao painel com seus rótulos
    panel.add(new JLabel("Código:")); panel.add(txtCodigo);
    panel.add(new JLabel("Nome:")); panel.add(txtNome);
    panel.add(new JLabel("Endereço:")); panel.add(txtEndereco);
    panel.add(new JLabel("Bairro:")); panel.add(txtBairro);
```

```
        panel.add(new JLabel("Cidade:")); panel.add(txtCidade);
        panel.add(new JLabel("Estado:")); panel.add(cbEstado);
        panel.add(new JLabel("CEP:")); panel.add(txtCep);
        panel.add(new JLabel("Telefone:")); panel.add(txtTelefone);
        panel.add(new JLabel("Observações:")); panel.add(txtObservacoes);

        // Botão Alterar
        JButton btnAlterar = new JButton("Alterar");
        btnAlterar.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // Simula uma alteração e exibe os dados atuais
                String dados = "Dados alterados:\n" +
                        "Código: " + txtCodigo.getText() + "\n" +
                        "Nome: " + txtNome.getText() + "\n" +
                        "Endereço: " + txtEndereco.getText() + "\n" +
                        "Bairro: " + txtBairro.getText() + "\n" +
                        "Cidade: " + txtCidade.getText() + "\n" +
                        "Estado: " + cbEstado.getSelectedItem() + "\n" +
                        "CEP: " + txtCep.getText() + "\n" +
                        "Telefone: " + txtTelefone.getText() + "\n" +
                        "Observações: " + txtObservacoes.getText();
                    JOptionPane.showMessageDialog(frame, dados, "Alteração Realizada",
JOptionPane.INFORMATION_MESSAGE);
            }
        });
```

```
        // Adiciona o botão ao painel
        panel.add(btnAlterar);

        frame.add(panel);
        frame.setVisible(true);
    }
}
```

código Java contendo apenas a funcionalidade do botão "Excluir", que simula a exclusão dos dados ao limpar os campos e exibir uma confirmação:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class CadastroClientesExcluir {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Cadastro de Clientes - Excluir");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 400);

        JPanel panel = new JPanel(new GridLayout(10, 2));
```

```
// Campos do formulário preenchidos com dados fictícios
JTextField txtCodigo = new JTextField("123");
JTextField txtNome = new JTextField("Maria Oliveira");
JTextField txtEndereco = new JTextField("Av. Central, 456");
JTextField txtBairro = new JTextField("Jardim");
JTextField txtCidade = new JTextField("Rio de Janeiro");
String[] estados = {"SP", "RJ", "MG", "RS"};
JComboBox<String> cbEstado = new JComboBox<>(estados);
cbEstado.setSelectedItem("RJ");
JTextField txtCep = new JTextField("20000-000");
JTextField txtTelefone = new JTextField("(21) 98888-8888");
JTextArea txtObservacoes = new JTextArea("Cliente com pendências");

// Adicionando os campos ao painel com seus rótulos
panel.add(new JLabel("Código:")); panel.add(txtCodigo);
panel.add(new JLabel("Nome:")); panel.add(txtNome);
panel.add(new JLabel("Endereço:")); panel.add(txtEndereco);
panel.add(new JLabel("Bairro:")); panel.add(txtBairro);
panel.add(new JLabel("Cidade:")); panel.add(txtCidade);
panel.add(new JLabel("Estado:")); panel.add(cbEstado);
panel.add(new JLabel("CEP:")); panel.add(txtCep);
panel.add(new JLabel("Telefone:")); panel.add(txtTelefone);
panel.add(new JLabel("Observações:")); panel.add(txtObservacoes);
```

```
        // Botão Excluir
        JButton btnExcluir = new JButton("Excluir");
        btnExcluir.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                int confirmacao = JOptionPane.showConfirmDialog(
                        frame,
                        "Tem certeza que deseja excluir este cadastro?",
                        "Confirmação",
                        JOptionPane.YES_NO_OPTION
                );
                if (confirmacao == JOptionPane.YES_OPTION) {
                    txtCodigo.setText("");
                    txtNome.setText("");
                    txtEndereco.setText("");
                    txtBairro.setText("");
                    txtCidade.setText("");
                    cbEstado.setSelectedIndex(0);
                    txtCep.setText("");
                    txtTelefone.setText("");
                    txtObservacoes.setText("");
                            JOptionPane.showMessageDialog(frame, "Cadastro excluído com
sucesso!");
                }
            }
        });
```

```
        // Adiciona o botão ao painel
        panel.add(btnExcluir);

        frame.add(panel);
        frame.setVisible(true);
    }
}
```

código Java contendo apenas a funcionalidade do botão "Salvar", que simula o salvamento de dados inseridos no formulário e exibe uma mensagem de confirmação com os dados digitados:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class CadastroClientesSalvar {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Cadastro de Clientes - Salvar");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 400);
```

```
JPanel panel = new JPanel(new GridLayout(10, 2));

// Campos do formulário
JTextField txtCodigo = new JTextField();
JTextField txtNome = new JTextField();
JTextField txtEndereco = new JTextField();
JTextField txtBairro = new JTextField();
JTextField txtCidade = new JTextField();
String[] estados = {"SP", "RJ", "MG", "RS"};
JComboBox<String> cbEstado = new JComboBox<>(estados);
JTextField txtCep = new JTextField();
JTextField txtTelefone = new JTextField();
JTextArea txtObservacoes = new JTextArea();

// Adicionando os campos ao painel com seus rótulos
panel.add(new JLabel("Código:")); panel.add(txtCodigo);
panel.add(new JLabel("Nome:")); panel.add(txtNome);
panel.add(new JLabel("Endereço:")); panel.add(txtEndereco);
panel.add(new JLabel("Bairro:")); panel.add(txtBairro);
panel.add(new JLabel("Cidade:")); panel.add(txtCidade);
panel.add(new JLabel("Estado:")); panel.add(cbEstado);
panel.add(new JLabel("CEP:")); panel.add(txtCep);
panel.add(new JLabel("Telefone:")); panel.add(txtTelefone);
panel.add(new JLabel("Observações:")); panel.add(txtObservacoes);
```

```
        // Botão Salvar
        JButton btnSalvar = new JButton("Salvar");
        btnSalvar.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String dados = "Cadastro salvo com os seguintes dados:\n" +
                        "Código: " + txtCodigo.getText() + "\n" +
                        "Nome: " + txtNome.getText() + "\n" +
                        "Endereço: " + txtEndereco.getText() + "\n" +
                        "Bairro: " + txtBairro.getText() + "\n" +
                        "Cidade: " + txtCidade.getText() + "\n" +
                        "Estado: " + cbEstado.getSelectedItem() + "\n" +
                        "CEP: " + txtCep.getText() + "\n" +
                        "Telefone: " + txtTelefone.getText() + "\n" +
                        "Observações: " + txtObservacoes.getText();
                          JOptionPane.showMessageDialog(frame, dados, "Cadastro Salvo",
JOptionPane.INFORMATION_MESSAGE);
            }
        });

        // Adiciona o botão ao painel
        panel.add(btnSalvar);

        frame.add(panel);
        frame.setVisible(true);
    }
```

}

código Java com apenas a funcionalidade de um botão "Relatório", que simula a geração de um relatório exibindo dados de clientes fictícios em uma janela de texto:

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;


public class CadastroClientesRelatorio {


    public static void main(String[] args) {

        JFrame frame = new JFrame("Cadastro de Clientes - Relatório");

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        frame.setSize(500, 400);


        JPanel panel = new JPanel(new BorderLayout());


        // Área de exibição do relatório

        JTextArea areaRelatorio = new JTextArea();

        areaRelatorio.setEditable(false);

        JScrollPane scroll = new JScrollPane(areaRelatorio);
```

```
        // Botão Relatório
        JButton btnRelatorio = new JButton("Gerar Relatório");
        btnRelatorio.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // Dados simulados para o relatório
                String relatorio = "=== Relatório de Clientes ===\n";
                 relatorio += "Código: 101 | Nome: Ana Lima | Cidade: São Paulo | Estado: SP\n";
                  relatorio += "Código: 102 | Nome: Carlos Souza | Cidade: Rio de Janeiro | Estado: RJ\n";
                        relatorio += "Código: 103 | Nome: Fernanda Costa | Cidade: Belo Horizonte | Estado: MG\n";
                   relatorio += "Código: 104 | Nome: Pedro Rocha | Cidade: Porto Alegre | Estado: RS\n";
                relatorio += "================================\n";

                areaRelatorio.setText(relatorio);
            }
        });

        panel.add(scroll, BorderLayout.CENTER);
        panel.add(btnRelatorio, BorderLayout.SOUTH);

        frame.add(panel);
        frame.setVisible(true);
    }
}
```

código Java contendo apenas a funcionalidade de um botão "Retornar", que fecha a janela atual

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class CadastroClientesRetornar {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Cadastro de Clientes - Retornar");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 200);

        JPanel panel = new JPanel();
        panel.setLayout(new FlowLayout());

        JLabel label = new JLabel("Tela de Cadastro de Clientes");
        panel.add(label);

        // Botão Retornar
        JButton btnRetornar = new JButton("Retornar");
        btnRetornar.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
```

```
            int resposta = JOptionPane.showConfirmDialog(
                    frame,
                    "Deseja realmente retornar?",
                    "Confirmação",
                    JOptionPane.YES_NO_OPTION
            );
            if (resposta == JOptionPane.YES_OPTION) {
                frame.dispose(); // Fecha a janela atual
            }
        }
    });


    panel.add(btnRetornar);


    frame.add(panel);
    frame.setVisible(true);
  }
}
```

Apresentação das Funcionalidades do Sistema

O sistema apresentado é um **sistema desktop de Cadastro de Clientes** desenvolvido em **Java Swing**, com diversas funcionalidades essenciais que simulam um sistema real. Ele está dividido em três telas principais: **Login**, **Home** e **Cadastro de Clientes**. Veja abaixo o que cada uma oferece:

1. Tela de Login

- Campos para **usuário** e **senha**.

- Botão **Entrar**:
    - Faz a verificação do login (usuário fixo: `admin`, senha: `123`).
    - Se os dados estiverem corretos, o sistema **abre a tela Home**.
    - Se os dados estiverem incorretos, exibe uma mensagem de erro.

2. Tela Home

- Exibe uma mensagem de boas-vindas ao sistema.
- Botão **“Cadastro de Clientes”**:
    - Abre a tela onde o usuário pode **cadastrar, editar e visualizar informações de clientes**.

3. Tela de Cadastro de Clientes

Esta é a tela principal do sistema, com as seguintes funcionalidades:

- **Campos de entrada**:
    - Código, Nome, Endereço, Bairro, Cidade, Estado, CEP, Telefone, e Observações.
- **Botões e funcionalidades:**

Botão	Função
Novo	Limpa todos os campos para um novo cadastro.
Salvar	Simula o salvamento dos dados digitados e exibe em uma janela de aviso.

Alterar	Simula a alteração dos dados existentes e mostra uma mensagem de sucesso.
Excluir	Solicita confirmação e, se confirmada, limpa os dados e exibe uma mensagem de exclusão.
Relatório	Mostra um relatório simples com o nome e a cidade do cliente.
Retornar	Fecha a tela atual e **volta para a tela Home**.

Navegação entre Telas

- A navegação é feita usando `new Tela()` e `dispose()`, o que permite abrir uma nova tela e fechar a anterior.
- Isso garante que o sistema seja **modular e bem estruturado**, simulando sistemas reais de múltiplas janelas.

Nota Individual e Organização

Cada integrante pode ficar responsável por:

Integrante	**Telas / Funcionalidades**
Eduarda	Tela de Login, + 3 funcionalidades

Evaldo	Tela Home, + 3 funcionalidades
Eduarda	4 funcionalidades (Salvar, Alterar...)
Evaldo	4 funcionalidades (Relatório, Retornar...)

Explicação dos Códigos Java Utilizados

A seguir, são apresentados e explicados todos os códigos utilizados no projeto de cadastro de clientes, com foco em suas funcionalidades e comportamentos esperados.

1. Código Principal – Cadastro de Clientes

- Cria a **interface gráfica (GUI)** principal com campos para preenchimento dos dados de clientes.
- Utiliza a biblioteca `javax.swing` para construir a interface.
- Organiza os componentes com `JPanel` e `GridLayout`.
- Campos incluídos:
    - Código
    - Nome
    - Endereço
    - Bairro
    - Cidade

    - Estado (com `JComboBox`)
    - CEP
    - Telefone
    - Observações (com `JTextArea`)
- Botões de ação: **Novo**, **Salvar**, **Alterar**, **Excluir**.

2. Botão "Novo"

- Classe responsável: `CadastroClientesNovo`.
- Função: **Limpa todos os campos** do formulário para um novo cadastro.
- Ação executada:
    - `setText("")` nos campos de texto.
    - `setSelectedIndex(0)` no campo de estado.
- Simula o botão “Limpar” encontrado em formulários.

3. Botão "Salvar"

- Classe responsável: `CadastroClientesSalvar`.
- Função: **Simula o salvamento** dos dados preenchidos.
- Ação executada:
    - Coleta os dados dos campos.
    - Mostra os dados via `JOptionPane.showMessageDialog`, como uma mensagem de confirmação.
- Não salva em banco de dados — apenas exibe uma simulação visual.

4. Botão "Alterar"

- Classe responsável: `CadastroClientesAlterar`.
- Função: **Simula a alteração de dados** de um cliente já existente.
- Ação executada:
    - Preenche os campos com dados fictícios.
    - Ao clicar em “Alterar”, os dados são exibidos em uma janela com `JOptionPane`.

5. Botão "Excluir"

- Classe responsável: `CadastroClientesExcluir`.
- Função: **Confirma a exclusão dos dados** e limpa os campos se o usuário aceitar.
- Ação executada:
    - Abre uma caixa de confirmação (`JOptionPane.showConfirmDialog`).
    - Se o usuário clicar em "Sim", os dados são apagados.
    - Exibe a mensagem “Cadastro excluído com sucesso”.

6. Botão "Relatório"

- Classe responsável: `CadastroClientesRelatorio`.
- Função: Exibe um **relatório fictício** com dados de clientes.
- Ação executada:
    - Abre uma nova janela com `JTextArea` mostrando vários clientes cadastrados de exemplo.
    - Simula uma funcionalidade de geração de relatório.

7. Botão "Retornar"

- Função: **Fecha a janela atual** ou retorna ao menu principal (se implementado).
- Ação executada:
    - Usa `frame.dispose()` ou `System.exit(0)` para encerrar a interface.

Observação Geral

- O sistema simula as operações de um CRUD:
    - **C**riar: com o botão **Novo**.
    - **R**elatório: com o botão **Relatório**.
    - **U**pdate: com o botão **Alterar**.
    - **D**elete: com o botão **Excluir**.
- Não há conexão com banco de dados, pois o objetivo é **demonstrar a lógica da interface e eventos** com Java Swing.

Apresentação do Código-Fonte do Sistema

O sistema foi desenvolvido inteiramente em **Java**, utilizando a biblioteca **Swing** para criação da interface gráfica. Está organizado em **três classes principais**, cada uma representando uma **tela do sistema**. A estrutura foi feita de forma modular para facilitar a navegação, leitura e manutenção do código.

1. `LoginTela.java`

Esta classe representa a tela inicial do sistema.

- Utiliza `JTextField` e `JPasswordField` para entrada de dados.
- Um `JButton` permite que o usuário tente acessar o sistema.

- A lógica de verificação compara os valores inseridos com valores fixos (`admin` e `123`).

- Se o login for bem-sucedido, a tela `HomeTela` é aberta; caso contrário, uma mensagem de erro aparece.

Trecho importante:

```
btnEntrar.addActionListener(e -> {
    if (usuario.getText().equals("admin") && new
String(senha.getPassword()).equals("123")) {
        new HomeTela();
        dispose();
    } else {
        JOptionPane.showMessageDialog(this, "Usuário ou senha inválidos!");
    }
});
```

2. `HomeTela.java`

Classe responsável por exibir uma tela de boas-vindas e navegar para a próxima etapa do sistema.

- Contém um `JLabel` com uma mensagem e um botão para ir à tela de cadastro de clientes.

Trecho importante:

```
btnCadastro.addActionListener(e -> {
    new CadastroClientes();
    dispose();
});
```

3. `CadastroClientes.java`

Classe principal do sistema, onde estão localizadas todas as funcionalidades de cadastro.

- Contém campos para:
    - Código, Nome, Endereço, Bairro, Cidade, Estado, CEP, Telefone e Observações.
- Usa `GridLayout` para organizar os elementos de forma clara.
- Possui 6 botões, cada um com sua função:
    - **Novo**: limpa todos os campos.
    - **Salvar**: exibe os dados preenchidos.
    - **Alterar**: simula uma atualização e mostra mensagem.
    - **Excluir**: limpa e confirma exclusão.
    - **Relatório**: mostra nome e cidade em um popup.
    - **Retornar**: volta à tela Home.

Exemplo do botão Salvar:

```
btnSalvar.addActionListener(e -> {
    String dados = "Cadastro salvo:\n"
        + "Código: " + txtCodigo.getText() + "\n"
        + "Nome: " + txtNome.getText() + "\n"
        + "Endereço: " + txtEndereco.getText();
    JOptionPane.showMessageDialog(this, dados);
});
```

Organização do Projeto

- Cada classe está salva em um arquivo `.java` separado.
- O projeto pode ser executado a partir da classe `LoginTela`, que inicia o sistema.

Resumo

O código-fonte está **simples, bem comentado e modularizado**, o que facilita o entendimento e a navegação entre as telas. Ele demonstra conhecimentos em:

- Interface gráfica (GUI) com Java Swing
- Manipulação de eventos
- Lógica de navegação entre janelas
- Organização de layout e componentes

Opinião sobre o uso do Java Swing

O uso do Java Swing foi uma experiência interessante e desafiadora. Como se trata de uma biblioteca gráfica mais antiga do Java, ela exige bastante atenção ao organizar os componentes visuais (botões, caixas de texto, rótulos, etc.), principalmente quando usamos diferentes tipos de layout.

Entre os pontos positivos, destacamos:

- É uma ferramenta robusta e já vem integrada no Java, ou seja, não precisa instalar nada extra.
- Permite criar interfaces completas e funcionais para sistemas desktop.
- É possível ter um controle detalhado dos elementos da interface, o que ajuda a aprender a lógica de construção de telas passo a passo.

Por outro lado, também notamos algumas dificuldades:

- A interface visual pode ser trabalhosa de montar quando comparada com ferramentas mais modernas.

- A falta de recursos visuais mais atualizados pode tornar o visual do sistema mais simples ou "antigo".

- Organizar os elementos de forma responsiva exige paciência com os gerenciadores de layout.

Apesar disso, o Java Swing foi uma excelente ferramenta para aprender os conceitos básicos de criação de interfaces gráficas, tratamento de eventos e navegação entre telas. O projeto ajudou a entender melhor como sistemas desktop funcionam na prática e como o Java pode ser usado além da programação de lógica e console.

Aprendizado com o Projeto

O desenvolvimento deste projeto em Java Swing proporcionou diversos aprendizados importantes, tanto na parte técnica quanto no trabalho em equipe. Ao longo do processo, conseguimos aplicar na prática os conceitos vistos em sala de aula, como a criação de interfaces gráficas, tratamento de eventos com `ActionListener` e navegação entre telas em um sistema desktop.

Entre os principais aprendizados, destacamos:

- **Criação de telas gráficas** com diferentes componentes, como campos de texto, caixas de seleção, botões e rótulos.

- **Uso de `JFrame`, `JPanel`, `JLabel`, `JButton`, `JTextField`, `JTextArea`, `JComboBox` e outros elementos do Swing** para construir interfaces visuais funcionais.

- **Tratamento de eventos**, como clicar em botões e realizar ações (salvar, excluir, alterar e navegar entre telas).

- **Organização do layout com `GridLayout` e `FlowLayout`**, permitindo alinhar os elementos na interface.

- **Criação de uma lógica básica de navegação entre telas**, utilizando classes separadas para cada funcionalidade (Login, Home e Cadastro).

- **Estruturação de um sistema de múltiplas telas**, onde cada integrante do grupo desenvolveu e apresentou uma parte.

Além disso, tivemos um **aprendizado importante sobre colaboração**, pois trabalhamos em grupo para dividir tarefas, organizar o código e testar as funcionalidades integradas do sistema. Esse tipo de atividade nos preparou para projetos maiores, onde a comunicação e o trabalho em equipe são essenciais.

Por fim, o projeto nos mostrou que é possível criar sistemas completos mesmo com ferramentas mais simples, como o Java Swing, reforçando o nosso entendimento sobre a lógica de programação e desenvolvimento de software.