1º/2º Ciência da Computação (CC)

SISTEMA DE CONTROLE DE PESSOAS DESAPARECIDAS

Evelyn de Souza e Manuella de Fátima Kuiawa

Sumário

1. OBJETIVO

Este trabalho tem como foco desenvolver, documentar e analisar um sistema de computação para o controle e monitoramento de pessoas desaparecidas. O principal objetivo é aplicar tanto conhecimentos teóricos quanto práticos de forma estruturada, permitindo uma compreensão ampla do ciclo de desenvolvimento de sistemas e das decisões técnicas envolvidas.

O projeto visa oferecer uma solução prática que permite cadastrar, consultar, atualizar e gerenciar registros de desaparecimentos, facilitando o acesso das instituições de segurança pública às informações essenciais para acompanhar esses casos. Para isso, o sistema conta com gerenciamento do estado dos registros, uma organização intuitiva das informações, busca avançada e validações rigorosas.

Além disso, o trabalho destaca,como uma definição clara impacta diretamente no planejamento e desenvolvimento. Assim, tanto o sistema quanto o conteúdo teórico têm como objetivo mostrar que uma boa delimitação de requisitos e funcionalidades é essencial para evitar falhas, retrabalhos e inconsistências durante o processo de desenvolvimento.

Por fim, a intenção deste trabalho é promover a integração entre teoria e prática, reforçando habilidades programação orientada a objetos e banco de dados.

Essa proposta não só contribui para a formação acadêmica, mas também para entender o papel crítico que os sistemas de informações exercem em contextos sociais delicados, como o desaparecimento de pessoas.

2. INTRODUÇÃO

O desaparecimento de pessoas é um assunto muito complicado e sensível, que impacta milhares de famílias e envolve várias instituições de segurança pública em todo o Brasil. Todo ano, os dados oficiais mostram que há um número considerável de novos casos, o que demanda ações que vão desde investigações policiais até campanhas de conscientização e mobilização social.

Nesse contexto desafiador, é crucial ter informações precisas, atualizadas e fáceis de acessar, pois isso aumenta as chances de encontrar e acompanhar os desaparecidos.

Este trabalho explora o tema tanto de um ponto de vista teórico quanto prático, abordando o desenvolvimento de um Sistema de Controle de Pessoas Desaparecidas, criado em Java com uma interface gráfica em Swing e um banco de dados MySQL.

Na parte teórica, o trabalho contextualiza o problema social, discute a relevância da gestão de informações e faz comparações que ajudam a entender o impacto da tecnologia nesse tipo de registro. Depois, na parte prática, são apresentados detalhes sobre o sistema desenvolvido, incluindo um relatório técnico que explica os principais trechos do código.

Assim, o objetivo é mostrar como a tecnologia e boas práticas de desenvolvimento podem ajudar a criar uma ferramenta que seja útil, organizada e que realmente possa auxiliar na luta contra esse problema tão sério.

3. CONCEITOS GERAIS

O Sistema de Controle de Pessoas Desaparecidas é fundamentado em princípios básicos de organização e gestão de dados. Ele opera com uma arquitetura cliente-servidor, tratando informações sensíveis e controlando o fluxo de dados para assegurar que os registros sejam mantidos, atualizados e acessados de maneira eficiente.

3.1 Estrutura de Dados

Cada pessoa registrada no sistema tem informações essenciais, como: nome, idade, sexo, data de desaparecimento, local, descrição e status. Esses dados formam um registro organizado que é guardado em tabelas MySQL, o que possibilita consultas rápidas e várias operações como criar, ler, atualizar e excluir.

3.2 Segurança das Informações

Mesmo que o sistema não utilize técnicas de criptografia avançadas, ele ainda se baseia em princípios fundamentais de segurança da informação, como a integridade dos registros, controle de acesso e armazenamento seguro em banco de dados.

3.3 Fluxo de Cadastro e Consulta

O fluxo de operações do sistema assegura que todas as etapas — inserir, atualizar, consultar e gerenciar status — sejam feitas de maneira sequencial, evitando problemas de inconsistência e melhorando o processamento dos dados.

4. FUNCIONALIDADES DO SISTEMA

4.1 Cadastro Completo

O sistema conta com um módulo de cadastro bem estruturado, que serve para registrar informações essenciais sobre cada pessoa desaparecida. Esse processo abrange dados pessoais, detalhes sobre as circunstâncias do desaparecimento e outras informações relevantes que podem ajudar na identificação e acompanhamento do caso.

O formulário foi feito para ser claro, direto e fácil de preencher, permitindo que os responsáveis insiram rapidamente todos os dados necessários, sem perder a precisão das informações.

4.2 Busca Avançada

O sistema oferece um mecanismo de busca avançada que permite encontrar registros de forma rápida, precisa e eficaz. Essa ferramenta é crucial em ambientes que lidam com um grande volume de dados, pois facilita a filtragem inteligente das informações conforme critérios específicos.

Os usuários podem fazer consultas usando vários parâmetros ao mesmo tempo, como nome, data do desaparecimento, gênero e status do caso. A combinação desses filtros torna a busca muito eficiente, reduzindo o tempo necessário para localizar informações e aumentando a certeza dos resultados apresentados.

4.3 Gerenciamento de Status

O gerenciamento de status é um dos pilares do sistema, permitindo acompanhar a evolução de cada caso de forma clara e organizada. Cada registro pode ser classificado em três estados diferentes:

Ativo: quando o caso está em aberto e ainda se busca informações atualizadas.

Pendente: quando há informações incompletas, que estão em análise ou aguardando validação antes de avançar.

Encerrado: quando o caso é finalizado, seja pela localização da pessoa, pela conclusão da investigação ou por outros motivos relevantes.

Essa estrutura de estados facilita a visualização geral do cenário e ajuda no planejamento das ações necessárias. Além disso, a classificação permite priorizar casos, distribuir demandas e otimizar a atuação dos responsáveis.

Todos os status podem ser atualizados a qualquer momento, garantindo que o sistema continue dinâmico e sempre alinhado à situação real de cada desaparecimento.

5. DISSERTAÇÃO – FUNDAMENTAÇÃO DO SISTEMA

5.1 Estrutura Lógica

O sistema foi projetado com arquitetura modular, dividindo interface, regras de negócio e persistência de dados. Essa separação facilita a compreensão, manutenção e futuras atualizações. A modularidade também permite que cada parte evolua de forma independente, favorecendo escalabilidade e integração de novos recursos.

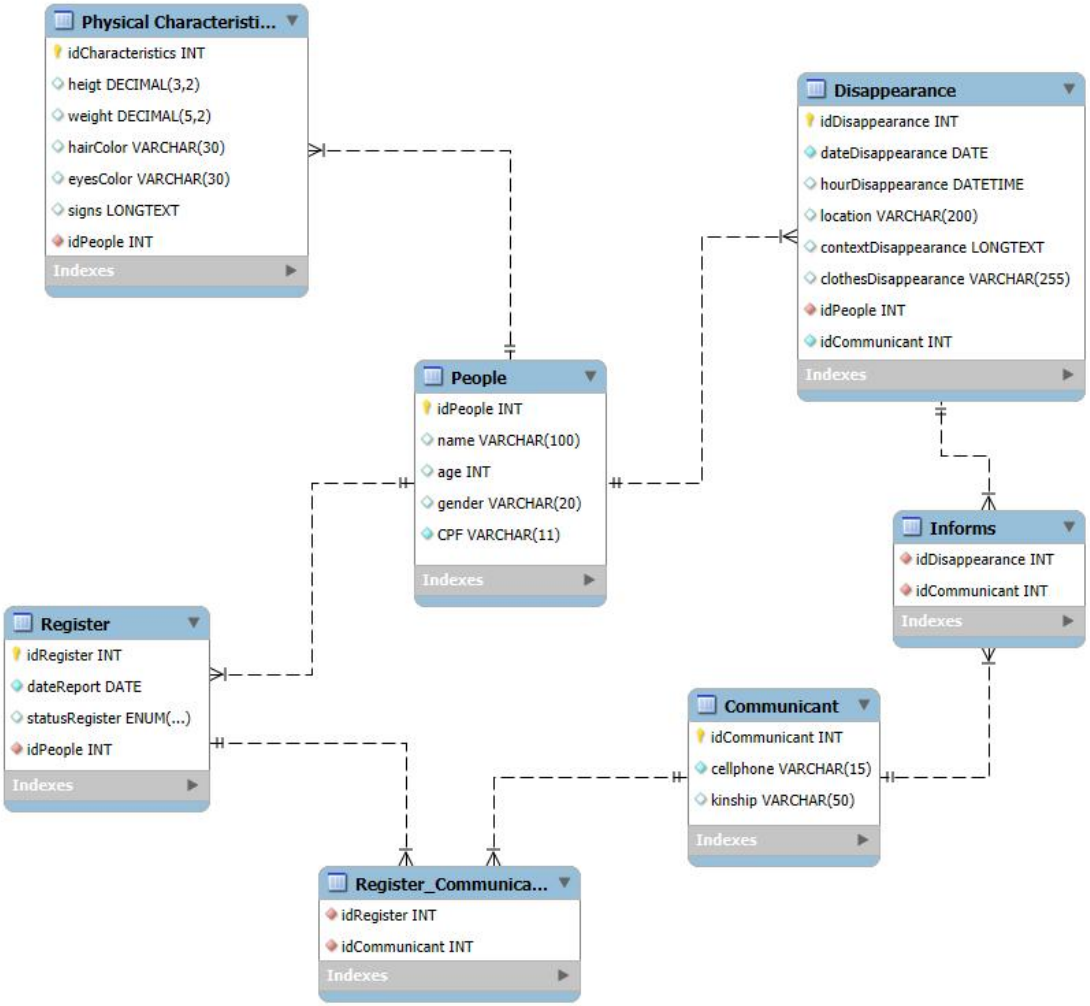
5.2 Importância Social

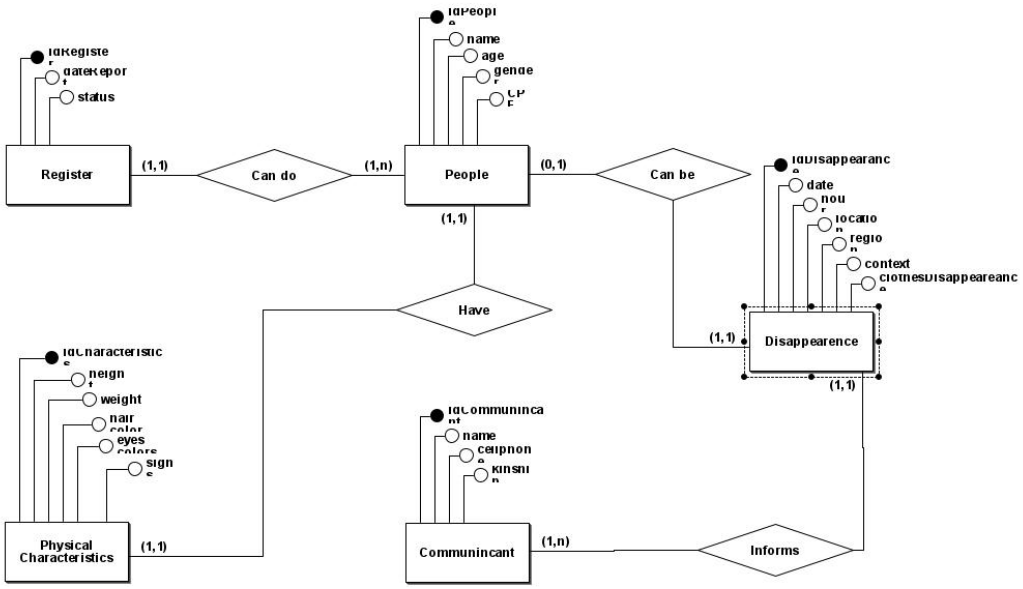
Os casos de desaparecimento de pessoas têm recebido cada vez mais atenção. A falta de informações centralizadas dificulta o trabalho de familiares e autoridades, prejudicando as buscas. Nesse contexto, sistemas bem organizados ajudam a reunir dados, agilizar consultas e apoiar ações de investigação, contribuindo diretamente para melhorar processos de localização e acompanhamento de casos.

5.3 Justificativa do Projeto

O projeto se justifica pela necessidade de uma ferramenta simples e acessível que auxilie tanto cidadãos quanto instituições responsáveis pelas buscas. Ao centralizar registros e padronizar informações, o sistema se torna útil nas etapas de prevenção, monitoramento e investigação, reduzindo retrabalho e mostrando como a tecnologia pode apoiar ações voltadas ao bem-estar social.

6. BANCO DE DADOS





```sql
CREATE DATABASE sistemaDesaparecimentos;
USE sistemaDesaparecimentos;

CREATE TABLE peoples (
    id_peoples INT PRIMARY KEY AUTO_INCREMENT,
    personName VARCHAR(100) NOT NULL,
    age INT,
    gender VARCHAR(20),
    CPF VARCHAR(11) UNIQUE
);

CREATE TABLE register (
    id_register INT PRIMARY KEY AUTO_INCREMENT,
    dateReport DATETIME,
    statusRegister ENUM('andamento', 'pendente', 'encerrado'),
    id_peoples INT,
    FOREIGN KEY (id_peoples) REFERENCES peoples(id_peoples)
);

CREATE TABLE physicalCharacteristics (
    id_characteristics INT PRIMARY KEY AUTO_INCREMENT,
    height DECIMAL(4,2),
    weight DECIMAL(5,2),
    hairColor VARCHAR(30),
    eyesColor VARCHAR(30),
    signs LONGTEXT,
    id_peoples INT,
    FOREIGN KEY (id_peoples) REFERENCES peoples(id_peoples)
);

CREATE TABLE disappearance (
    id_disappearance INT PRIMARY KEY AUTO_INCREMENT,
    dateDisappearance DATE,
    hourDisappearance TIME,
    location VARCHAR(200),
    contextDisappearance LONGTEXT,
    clothesDisappearance VARCHAR(255),
    id_peoples INT,
    FOREIGN KEY (id_peoples) REFERENCES peoples(id_peoples)
);

CREATE TABLE communicant (
    id_communicant INT PRIMARY KEY AUTO_INCREMENT,
    cellphone VARCHAR(15),
    kinship VARCHAR(50)
);

CREATE TABLE informs (
    id_communicant INT,
    id_disappearance INT,
    FOREIGN KEY (id_communicant) REFERENCES communicant(id_communicant),
    FOREIGN KEY (id_disappearance) REFERENCES disappearance(id_disappearance
);

CREATE TABLE RegisterCommunicant (
    id_register INT,
    id_communicant INT,
    FOREIGN KEY (id_register) REFERENCES Register(id_register),
    FOREIGN KEY (id_communicant) REFERENCES Communicant(id_communicant)
);

ALTER TABLE physicalCharacteristics ADD url LONGTEXT;
ALTER TABLE physicalCharacteristics DROP COLUMN url;

ALTER TABLE disappearance CHANGE timeDisappearance hourDisappearance TIME;

ALTER TABLE informs DROP COLUMN id_informs;

ALTER TABLE peoples ADD address VARCHAR(150);
```

7. JAVA

Classe de Conexão com Banco de Dados

```java
package dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class BaseDao {

    private static final String URL = "jdbc:mysql://localhost:3306/sistemaDesaparecimentos";
    private static final String USER = "root";
    private static final String PASS = "0166";


    public Connection getConnection() throws SQLException {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            throw new SQLException("Driver JDBC não encontrado!", e);
        }

        return DriverManager.getConnection(URL, USER, PASS);
    }


    public int getLastInsertedId(Connection conn) throws SQLException {
        String sql = "SELECT LAST_INSERT_ID() AS id";

        try (PreparedStatement stmt = conn.prepareStatement(sql);
             ResultSet rs = stmt.executeQuery()) {

            if (rs.next()) {
                return rs.getInt("id");
            } else {
                return -1;
            }
        }
    }


    public void close(Connection conn) {
        if (conn != null) {
            try {
                if (!conn.isClosed()) conn.close();
            } catch (SQLException e) {
                System.out.println("Erro ao fechar conexão: " + e.getMessage());
            }
        }
    }
}
```

PeopleDao.java

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;

import model.People;

public class PeopleDao {

    public int insert(Connection conn, People p) throws SQLException {

        String sql = "INSERT INTO peoples (personName, age, gender, CPF, address) "
                   + "VALUES (?, ?, ?, ?, ?)";

        try (PreparedStatement stmt = conn.prepareStatement(sql, Statement.RETURN_GENERATED_KEYS)) {

            stmt.setString(1, p.getPersonName());
            stmt.setInt(2, p.getAge());
            stmt.setString(3, p.getGender());
            stmt.setString(4, p.getCPF());
            stmt.setString(5, p.getAddress());

            stmt.executeUpdate();

            ResultSet rs = stmt.getGeneratedKeys();
            if (rs.next()) return rs.getInt(1);

            return -1;
        }
    }
}
```

CharacteristicsDao.java

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

import model.PhysicalCharacteristics;

public class CharacteristicsDao {

    public void insert(Connection conn, PhysicalCharacteristics pc) throws SQLException {

        if (conn == null) {
            throw new SQLException("Conexão recebida é nula! Verifique o BaseDao.getConexao().");
        }

        String sql = "INSERT INTO physicalCharacteristics "
                   + "(height, weight, hairColor, eyesColor, signs, id_peoples) "
                   + "VALUES (?, ?, ?, ?, ?, ?)";

        try (PreparedStatement stmt = conn.prepareStatement(sql)) {

            stmt.setDouble(1, pc.getHeight());
            stmt.setDouble(2, pc.getWeight());
            stmt.setString(3, pc.getHairColor());
            stmt.setString(4, pc.getEyesColor());
            stmt.setString(5, pc.getSigns());
            stmt.setInt(6, pc.getIdPeople());

            stmt.executeUpdate();

            System.out.println("Características físicas cadastradas com sucesso!");
        } catch (SQLException e) {
            System.err.println("Erro ao inserir características físicas: " + e.getMessage());
            throw e;
        }
    }
}
```

CommunicantDao.java

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import model.Communicant;

public class CommunicantDao {

    public int insert(Connection conn, Communicant c) throws SQLException {

        if (conn == null) {
            throw new SQLException("Conexão nula recebida no CommunicantDao!");
        }

        String sql = "INSERT INTO communicant (cellphone, kinship) VALUES (?, ?)";

        try (PreparedStatement stmt = conn.prepareStatement(sql, Statement.RETURN_GENERATED_KEYS)) {

            stmt.setString(1, c.getCellphone());
            stmt.setString(2, c.getKinship());

            stmt.executeUpdate();

            try (ResultSet rs = stmt.getGeneratedKeys()) {
                if (rs.next()) {
                    System.out.println("Communicant cadastrado! ID = " + rs.getInt(1));
                    return rs.getInt(1);
                }
            }

            System.err.println("Nenhum ID retornado ao inserir Communicant!");
            return -1;

        } catch (SQLException e) {
            System.err.println("Erro ao inserir Communicant: " + e.getMessage());
            throw e;
        }
    }
}
```

DashboardDao.java

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class DashboardDao extends BaseDao {

    // Contar todos os casos
    public int contarTotalCasos() {
        String sql = "SELECT COUNT(*) AS total FROM register";
        try (Connection conn = getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql);
             ResultSet rs = stmt.executeQuery()) {
            if (rs.next()) {
                return rs.getInt("total");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return 0;
    }

    // Contar casos por status
    public int contarCasosPorStatus(String status) {
        String sql = "SELECT COUNT(*) AS total FROM register WHERE statusRegister = ?";
        try (Connection conn = getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setString(1, status);
            try (ResultSet rs = stmt.executeQuery()) {
                if (rs.next()) {
                    return rs.getInt("total");
                }
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return 0;
    }
}
```

DisappearanceDao.java

```java
package dao;

import java.sql.Connection;
import java.sql.Date;
import java.sql.Time;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import model.Disappearance;

public class DisappearanceDao extends BaseDao {

    public int insert(Connection conn, Disappearance d) throws SQLException {

        String sql = "INSERT INTO disappearance "
                + "(dateDisappearance, hourDisappearance, location, contextDisappearance, clothesDisappearance, id_peoples) "
                + "VALUES (?, ?, ?, ?, ?, ?)";

        try (PreparedStatement stmt = conn.prepareStatement(sql, PreparedStatement.RETURN_GENERATED_KEYS)) {

            stmt.setDate(1, Date.valueOf(d.getDateDisappearance()));
            stmt.setTime(2, Time.valueOf(d.getHourDisappearance()));
            stmt.setString(3, d.getLocation());
            stmt.setString(4, d.getContextDisappearance());
            stmt.setString(5, d.getClothesDisappearance());
            stmt.setInt(6, d.getId_peoples());

            stmt.executeUpdate();

            ResultSet rs = stmt.getGeneratedKeys();
            if (rs.next()) return rs.getInt(1);

            return -1;
        }
    }
```

```java
    public List<Disappearance> getAll() {

        List<Disappearance> lista = new ArrayList<>();

        String sql = "SELECT * FROM disappearance";

        try (Connection conn = getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql);
             ResultSet rs = stmt.executeQuery()) {

            while (rs.next()) {

                Disappearance d = new Disappearance();

                d.setId_disappearance(rs.getInt("id_disappearance"));
                d.setDateDisappearance(rs.getDate("dateDisappearance").toLocalDate());
                d.setHourDisappearance(rs.getTime("hourDisappearance").toLocalTime());
                d.setLocation(rs.getString("location"));
                d.setContextDisappearance(rs.getString("contextDisappearance"));
                d.setClothesDisappearance(rs.getString("clothesDisappearance"));
                d.setId_peoples(rs.getInt("id_peoples"));

                lista.add(d);
            }

        } catch (SQLException e) {
            e.printStackTrace();
        }

        return lista;
    }
}
```

DisappearanceListDao.java

```java
package dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;

import model.DisappearanceListItem;
import model.StatusRegister;

public class DisappearanceListDao {

    private Connection getConnection() throws Exception {
        Class.forName("com.mysql.cj.jdbc.Driver");
        return DriverManager.getConnection(
                "jdbc:mysql://localhost:3306/sistemaDesaparecimentos",
                "root",
                "0166"
        );
    }

    public List<DisappearanceListItem> listarTodos() throws Exception {

        String sql =
            "SELECT p.id_peoples, r.id_register, p.personName, p.age, p.gender, " +
            "pc.height, pc.weight, pc.hairColor, pc.eyesColor, " +
            "d.contextDisappearance, d.location, d.clothesDisappearance, " +
            "c.kinship, c.cellphone, r.statusRegister " +
            "FROM peoples p " +
            "LEFT JOIN disappearance d ON p.id_peoples = d.id_peoples " +
            "LEFT JOIN physicalCharacteristics pc ON p.id_peoples = pc.id_peoples " +
            "LEFT JOIN informs i ON d.id_disappearance = i.id_disappearance " +
            "LEFT JOIN communicant c ON i.id_communicant = c.id_communicant " +
            "LEFT JOIN register r ON p.id_peoples = r.id_peoples";

        List<DisappearanceListItem> lista = new ArrayList<>();
```

```java
        try (Connection conn = getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql);
             ResultSet rs = stmt.executeQuery()) {

            while (rs.next()) {

                StatusRegister status =
                        StatusRegister.fromString(rs.getString("statusRegister"));

                DisappearanceListItem item = new DisappearanceListItem(
                        rs.getInt("id_peoples"),
                        rs.getInt("id_register"),
                        rs.getString("personName"),
                        rs.getInt("age"),
                        rs.getString("gender"),
                        rs.getDouble("height"),
                        rs.getDouble("weight"),
                        rs.getString("hairColor"),
                        rs.getString("eyesColor"),
                        rs.getString("clothesDisappearance"),
                        rs.getString("location"),
                        rs.getString("contextDisappearance"),
                        rs.getString("kinship"),
                        rs.getString("cellphone"),
                        status
                );

                lista.add(item);
            }
        }

        return lista;
    }

    public void atualizarStatusRegister(int idRegister, StatusRegister novoStatus) throws Exception {
        String sql = "UPDATE register SET statusRegister = ? WHERE id_register = ?";

        try (Connection conn = getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql)) {

            stmt.setString(1, novoStatus.toDatabaseValue());
            stmt.setInt(2, idRegister);
            stmt.executeUpdate();
        }
    }
}
```

Informs.Dao

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class InformsDao {

    public void insert(Connection conn, int idCommunicant, int idDisappearance) throws SQLException {
        String sql = "INSERT INTO informs (id_communicant, id_disappearance) VALUES (?, ?)";

        try (PreparedStatement stmt = conn.prepareStatement(sql)) {

            stmt.setInt(1, idCommunicant);
            stmt.setInt(2, idDisappearance);

            stmt.executeUpdate();
        }
    }
}
```

RegisterCommunicantDao.java

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class RegisterCommunicantDao {

    public void insert(Connection conn, int idRegister, int idCommunicant) throws SQLException {

        String sql = "INSERT INTO registerCommunicant (id_register, id_communicant) VALUES (?, ?)";

        try (PreparedStatement stmt = conn.prepareStatement(sql)) {

            stmt.setInt(1, idRegister);
            stmt.setInt(2, idCommunicant);

            stmt.executeUpdate();
            System.out.println("Ligação entre registro e comunicante salva!");
        }
    }
}
```

RegisterDao.java

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.sql.Timestamp;

import model.Register;

public class RegisterDao {

    public void insert(Connection conn, Register register) throws SQLException {

        String sql = "INSERT INTO register (dateReport, statusRegister, id_peoples) VALUES (?, ?, ?)";

        try (PreparedStatement stmt = conn.prepareStatement(sql)) {

            Timestamp ts = new Timestamp(register.getDateReport().getTime());
            stmt.setTimestamp(1, ts);

            stmt.setString(2, register.getStatusRegister().name());
            stmt.setInt(3, register.getIdPeople());

            stmt.executeUpdate();
            System.out.println("Registro salvo com sucesso!");
            System.out.println(".");
        }
    }
}
```

SearchMissingDao.java

```java
package dao;

import model.DisappearanceListItem;
import model.StatusRegister;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class SearchMissingDao {

    private final BaseDao baseDao = new BaseDao();

    public List<DisappearanceListItem> pesquisar(
            String nome,
            String idade,
            String altura,
            String cabelo,
            String olhos,
            String roupas,
            String local
    ) {
        List<DisappearanceListItem> lista = new ArrayList<>();

        // Query alinhada ao seu schema (peoples, physicalCharacteristics, disappearance, register, communicant)
        StringBuilder sql = new StringBuilder(
                "SELECT " +
                        "p.id_peoples, r.id_register, p.personName, p.age, p.gender, " +
                        "pc.height, pc.weight, pc.hairColor, pc.eyesColor, " +
                        "d.clothesDisappearance, d.location, d.contextDisappearance, " +
                        "c.kinship, c.cellphone, r.statusRegister " +
                "FROM peoples p " +
                "LEFT JOIN physicalCharacteristics pc ON p.id_peoples = pc.id_peoples " +
                "LEFT JOIN disappearance d ON p.id_peoples = d.id_peoples " +
                "LEFT JOIN informs i ON d.id_disappearance = i.id_disappearance " +
                "LEFT JOIN communicant c ON i.id_communicant = c.id_communicant " +
                "LEFT JOIN register r ON p.id_peoples = r.id_peoples " +
                "WHERE 1=1 "
        );
```

```java
List<Object> params = new ArrayList<>();

if (nome != null && !nome.isBlank()) {
    sql.append(" AND LOWER(p.personName) LIKE ?");
    params.add("%" + nome.toLowerCase() + "%");
}
if (idade != null && !idade.isBlank()) {
    sql.append(" AND p.age = ?");
    params.add(Integer.parseInt(idade));
}
if (altura != null && !altura.isBlank()) {
    sql.append(" AND pc.height = ?");
    params.add(Double.parseDouble(altura));
}
if (cabelo != null && !cabelo.isBlank()) {
    sql.append(" AND LOWER(pc.hairColor) LIKE ?");
    params.add("%" + cabelo.toLowerCase() + "%");
}
if (olhos != null && !olhos.isBlank()) {
    sql.append(" AND LOWER(pc.eyesColor) LIKE ?");
    params.add("%" + olhos.toLowerCase() + "%");
}
if (local != null && !local.isBlank()) {
    sql.append(" AND LOWER(d.location) LIKE ?");
    params.add("%" + local.toLowerCase() + "%");
}
if (roupas != null && !roupas.isBlank()) {
    sql.append(" AND LOWER(d.clothesDisappearance) LIKE ?");
    params.add("%" + roupas.toLowerCase() + "%");
}

try (Connection conn = baseDao.getConnection();
     PreparedStatement stmt = conn.prepareStatement(sql.toString())) {


    for (int i = 0; i < params.size(); i++) {
        Object p = params.get(i);
        if (p instanceof Integer) stmt.setInt(i + 1, (Integer) p);
        else if (p instanceof Double) stmt.setDouble(i + 1, (Double) p);
        else stmt.setString(i + 1, p.toString());
    }
```

```java
        try (ResultSet rs = stmt.executeQuery()) {
            while (rs.next()) {

                DisappearanceListItem item = new DisappearanceListItem();

                item.setIdPeoples(rs.getInt("id_peoples"));
                item.setIdRegister(rs.getInt("id_register"));
                item.setPersonName(rs.getString("personName"));
                item.setAge(rs.getInt("age"));
                item.setGender(rs.getString("gender"));


                double height = rs.getDouble("height");
                if (rs.wasNull()) item.setHeight(0.0);
                else item.setHeight(height);

                double weight = rs.getDouble("weight");
                if (rs.wasNull()) item.setWeight(0.0);
                else item.setWeight(weight);

                item.setHairColor(rs.getString("hairColor"));
                item.setEyesColor(rs.getString("eyesColor"));
                item.setClothesDisappearance(rs.getString("clothesDisappearance"));
                item.setLocation(rs.getString("location"));
                item.setContextDisappearance(rs.getString("contextDisappearance"));
                item.setKinship(rs.getString("kinship"));
                item.setCellphone(rs.getString("cellphone"));


                item.setStatus(StatusRegister.fromString(rs.getString("statusRegister")));

                lista.add(item);
            }
        }

    } catch (Exception e) {
        e.printStackTrace();
    }

    return lista;
}
}
```

Communicant.java

```java
package model;

public class Communicant {

    private int id_Communicant;
    private String cellphone;
    private String kinship;

    public Communicant() {

    }

    public Communicant(int idCommunicant, String cellphone, String kinship) {
        super();
        this.id_Communicant = idCommunicant;
        this.cellphone = cellphone;
        this.kinship = kinship;
    }

    public int getIdCommunicant() {
        return id_Communicant;
    }

    public void setIdCommunicant(int idCommunicant) {
        this.id_Communicant = idCommunicant;
    }

    public String getCellphone() {
        return cellphone;
    }

    public void setCellphone(String cellphone) {
        this.cellphone = cellphone;
    }

    public String getKinship() {
        return kinship;
    }

    public void setKinship(String kinship) {
        this.kinship = kinship;
    }

    @Override
    public String toString() {
        return "Communicant [idCommunicant=" + id_Communicant + ", cellphone=" + cellphone + ", kinship=" + kinship
                + "]";
    }

}
```

Disappearance.java

```java
package model;

import java.time.LocalDate;
import java.time.LocalTime;

public class Disappearance {

    private int id_disappearance;
    private LocalDate dateDisappearance;
    private LocalTime hourDisappearance;
    private String location;
    private String contextDisappearance;
    private String clothesDisappearance;
    private int id_peoples;

    public Disappearance() {}

    public Disappearance(int id_disappearance, LocalDate dateDisappearance, LocalTime hourDisappearance,
                         String location, String contextDisappearance, String clothesDisappearance, int id_peoples) {

        this.id_disappearance = id_disappearance;
        this.dateDisappearance = dateDisappearance;
        this.hourDisappearance = hourDisappearance;
        this.location = location;
        this.contextDisappearance = contextDisappearance;
        this.clothesDisappearance = clothesDisappearance;
        this.id_peoples = id_peoples;
    }

    public int getId_disappearance() {
        return id_disappearance;
    }

    public void setId_disappearance(int id_disappearance) {
        this.id_disappearance = id_disappearance;
    }

    public LocalDate getDateDisappearance() {
        return dateDisappearance;
    }

    public void setDateDisappearance(LocalDate dateDisappearance) {
        this.dateDisappearance = dateDisappearance;
    }

    public LocalTime getHourDisappearance() {
        return hourDisappearance;
    }

    public void setHourDisappearance(LocalTime hourDisappearance) {
        this.hourDisappearance = hourDisappearance;
    }

    public String getLocation() {
        return location;
    }
```

```
    public String getLocation() {
        return location;
    }

    public void setLocation(String location) {
        this.location = location;
    }

    public String getContextDisappearance() {
        return contextDisappearance;
    }

    public void setContextDisappearance(String contextDisappearance) {
        this.contextDisappearance = contextDisappearance;
    }

    public String getClothesDisappearance() {
        return clothesDisappearance;
    }

    public void setClothesDisappearance(String clothesDisappearance) {
        this.clothesDisappearance = clothesDisappearance;
    }

    public int getId_peoples() {
        return id_peoples;
    }

    public void setId_peoples(int id_peoples) {
        this.id_peoples = id_peoples;
    }

    @Override
    public String toString() {
        return "Disappearance{" +
                "id_disappearance=" + id_disappearance +
                ", dateDisappearance=" + dateDisappearance +
                ", hourDisappearance=" + hourDisappearance +
                ", location='" + location + '\'' +
                ", contextDisappearance='" + contextDisappearance + '\'' +
                ", clothesDisappearance='" + clothesDisappearance + '\'' +
                ", id_peoples=" + id_peoples +
                '}';
    }
}
```

DisappearanceListItem.java

```java
package model;

public class DisappearanceListItem {

    private int idPeoples;
    private int idRegister;
    private String personName;
    private int age;
    private String gender;
    private double height;
    private double weight;
    private String hairColor;
    private String eyesColor;
    private String clothesDisappearance;
    private String location;
    private String contextDisappearance;
    private String kinship;
    private String cellphone;
    private StatusRegister status;


    public DisappearanceListItem(
            int idPeoples,
            int idRegister,
            String personName,
            int age,
            String gender,
            double height,
            double weight,
            String hairColor,
            String eyesColor,
            String clothesDisappearance,
            String location,
            String contextDisappearance,
            String kinship,
            String cellphone,
            StatusRegister status
    ) {
        this.idPeoples = idPeoples;
        this.idRegister = idRegister;
        this.personName = personName;
        this.age = age;
        this.gender = gender;
        this.height = height;
        this.weight = weight;
        this.hairColor = hairColor;
        this.eyesColor = eyesColor;
        this.clothesDisappearance = clothesDisappearance;
        this.location = location;
        this.contextDisappearance = contextDisappearance;
        this.kinship = kinship;
        this.cellphone = cellphone;
        this.status = status;
    }
```

```java
public DisappearanceListItem() {}


public int getIdPeoples() { return idPeoples; }
public int getIdRegister() { return idRegister; }
public String getPersonName() { return personName; }
public int getAge() { return age; }
public String getGender() { return gender; }
public double getHeight() { return height; }
public double getWeight() { return weight; }
public String getHairColor() { return hairColor; }
public String getEyesColor() { return eyesColor; }
public String getClothesDisappearance() { return clothesDisappearance; }
public String getLocation() { return location; }
public String getContextDisappearance() { return contextDisappearance; }
public String getKinship() { return kinship; }
public String getCellphone() { return cellphone; }
public StatusRegister getStatus() { return status; }


public void setIdPeoples(int idPeoples) { this.idPeoples = idPeoples; }
public void setIdRegister(int idRegister) { this.idRegister = idRegister; }
public void setPersonName(String personName) { this.personName = personName; }
public void setAge(int age) { this.age = age; }
public void setGender(String gender) { this.gender = gender; }
public void setHeight(double height) { this.height = height; }
public void setWeight(double weight) { this.weight = weight; }
public void setHairColor(String hairColor) { this.hairColor = hairColor; }
public void setEyesColor(String eyesColor) { this.eyesColor = eyesColor; }
public void setClothesDisappearance(String clothesDisappearance) { this.clothesDisappearance = clothesDisappearance; }
public void setLocation(String location) { this.location = location; }
public void setContextDisappearance(String contextDisappearance) { this.contextDisappearance = contextDisappearance; }
public void setKinship(String kinship) { this.kinship = kinship; }
public void setCellphone(String cellphone) { this.cellphone = cellphone; }
public void setStatus(StatusRegister status) { this.status = status; }
```

```java
    public void setIdPeoples(int idPeoples) { this.idPeoples = idPeoples; }
    public void setIdRegister(int idRegister) { this.idRegister = idRegister; }
    public void setPersonName(String personName) { this.personName = personName; }
    public void setAge(int age) { this.age = age; }
    public void setGender(String gender) { this.gender = gender; }
    public void setHeight(double height) { this.height = height; }
    public void setWeight(double weight) { this.weight = weight; }
    public void setHairColor(String hairColor) { this.hairColor = hairColor; }
    public void setEyesColor(String eyesColor) { this.eyesColor = eyesColor; }
    public void setClothesDisappearance(String clothesDisappearance) { this.clothesDisappearance = clothesDisappearance;
    public void setLocation(String location) { this.location = location; }
    public void setContextDisappearance(String contextDisappearance) { this.contextDisappearance = contextDisappearance;
    public void setKinship(String kinship) { this.kinship = kinship; }
    public void setCellphone(String cellphone) { this.cellphone = cellphone; }
    public void setStatus(StatusRegister status) { this.status = status; }

    @Override
    public String toString() {
        return "DisappearanceListItem{" +
                "idPeoples=" + idPeoples +
                ", idRegister=" + idRegister +
                ", personName='" + personName + '\'' +
                ", age=" + age +
                ", gender='" + gender + '\'' +
                ", height=" + height +
                ", weight=" + weight +
                ", hairColor='" + hairColor + '\'' +
                ", eyesColor='" + eyesColor + '\'' +
                ", clothesDisappearance='" + clothesDisappearance + '\'' +
                ", location='" + location + '\'' +
                ", contextDisappearance='" + contextDisappearance + '\'' +
                ", kinship='" + kinship + '\'' +
                ", cellphone='" + cellphone + '\'' +
                ", status=" + status +
                '}';
    }
}
```

Informs.java

```java
package model;

public class Informs {

    private int id_communicant;
    private int id_disappearence;

    public Informs() {

    }

    public int getId_communicant() {
        return id_communicant;
    }

    public void setId_communicant(int id_communicant) {
        this.id_communicant = id_communicant;
    }

    public int getId_disappearence() {
        return id_disappearence;
    }

    public void setId_disappearence(int id_disappearence) {
        this.id_disappearence = id_disappearence;
    }

    @Override
    public String toString() {
        return "Informs [id_communicant=" + id_communicant + ", id_disappearence=" + id_disappearence + "]";
    }

}
```

People.java

```java
package model;

public class People {

    private int id_peoples;
    private String personName;
    private int age;
    private String gender;
    private String CPF;
    private String address;

    public People() {}

    public People(int id_peoples, String personName, int age, String gender, String CPF, String address) {
        this.id_peoples = id_peoples;
        this.personName = personName;
        this.age = age;
        this.gender = gender;
        this.CPF = CPF;
        this.address = address;
    }

    public int getId_peoples() {
        return id_peoples;
    }

    public void setId_peoples(int id_peoples) {
        this.id_peoples = id_peoples;
    }

    public String getPersonName() {
        return personName;
    }

    public void setPersonName(String personName) {
        this.personName = personName;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public String getGender() {
        return gender;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }

    public String getCPF() {
        return CPF;
    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String adress) {
        this.address = adress;
    }

    @Override
    public String toString() {
        return "People [id_peoples=" + id_peoples + ", personName=" + personName + ", age=" + age + ", gender=" + gender
                + ", CPF=" + CPF + ", address=" + address + "]";
    }
}
```

PhysicalCharacteristcsa.java

```java
package model;

public class PhysicalCharacteristics {
    private int id_characteristics;
    private double height;
    private double weight;
    private String hairColor;
    private String eyesColor;
    private String signs;
    private int idPeople;


    public PhysicalCharacteristics() {

    }

    public PhysicalCharacteristics(int idCharacteristics, double height, double weight, String hairColor,
            String eyesColor, String signs, int idPeople, String url) {
        super();

        this.id_characteristics = idCharacteristics;
        this.height = height;
        this.weight = weight;
        this.hairColor = hairColor;
        this.eyesColor = eyesColor;
        this.signs = signs;
        this.idPeople = idPeople;
    }



    public int getIdCharacteristics() {
        return id_characteristics;
    }

    public void setIdCharacteristics(int idCharacteristics) {
        this.id_characteristics = idCharacteristics;
    }

    public double getHeight() {
        return height;
    }

    public void setHeight(double height) {
        this.height = height;
    }

    public double getWeight() {
        return weight;
    }

    public void setWeight(double weight) {
        this.weight = weight;
    }
```

```java
    public String getHairColor() {
        return hairColor;
    }

    public void setHairColor(String hairColor) {
        this.hairColor = hairColor;
    }

    public String getEyesColor() {
        return eyesColor;
    }

    public void setEyesColor(String eyesColor) {
        this.eyesColor = eyesColor;
    }

    public String getSigns() {
        return signs;
    }

    public void setSigns(String signs) {
        this.signs = signs;
    }

    public int getIdPeople() {
        return idPeople;
    }

    public void setIdPeople(int idPeople) {
        this.idPeople = idPeople;
    }


    @Override
    public String toString() {
        return "PhysicalCharacteristics [id_characteristics=" + id_characteristics + ", height=" + height + ", weight="
                + weight + ", hairColor=" + hairColor + ", eyesColor=" + eyesColor + ", signs=" + signs + ", idPeople="
                + idPeople + "]";
    }

}
```

Register.java

```java
package model;

import java.util.Date;

public class Register {
    private int id_register;
    private Date dateReport;
    private StatusRegister statusRegister;
    private int idPeople;

    public Register() {

    }

    public Register(int idRegister, Date dateReport, StatusRegister statusRegister, int idPeople) {
        super();
        this.id_register = idRegister;
        this.dateReport = dateReport;
        this.statusRegister = statusRegister;
        this.idPeople = idPeople;
    }

    public int getIdRegister() {
        return id_register;
    }

    public void setIdRegister(int idRegister) {
        this.id_register = idRegister;
    }

    public Date getDateReport() {
        return dateReport;
    }

    public void setDateReport(Date dateReport) {
        this.dateReport = dateReport;
    }

    public StatusRegister getStatusRegister() {
        return statusRegister;
    }

    public void setStatusRegister(StatusRegister statusRegister) {
        this.statusRegister = statusRegister;
    }

    public int getIdPeople() {
        return idPeople;
    }

    public void setIdPeople(int idPeople) {
        this.idPeople = idPeople;
    }

    @Override
    public String toString() {
        return "Register [idRegister=" + id_register + ", dateReport=" + dateReport + ", statusRegister="
                + statusRegister + ", idPeople=" + idPeople + "]";
    }
```

RegisterCommunicant.java

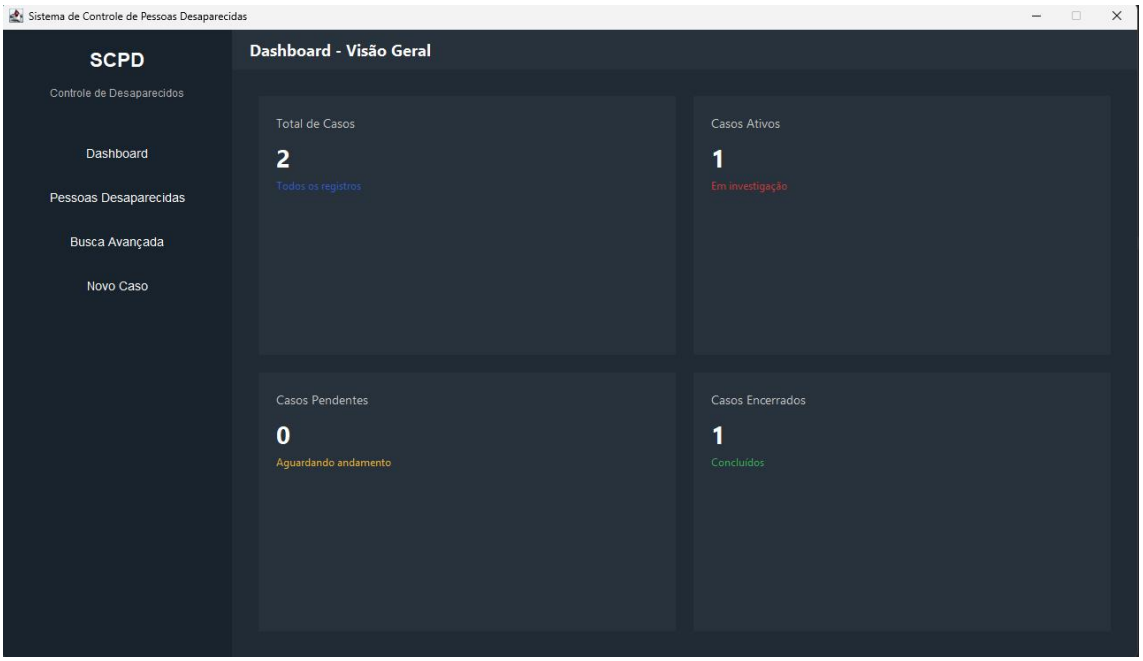
```java
package model;

public class RegisterCommunicant {

    private int id_register;
    private int i_communicant;

    public RegisterCommunicant() {

    }

    public RegisterCommunicant(int id_register, int i_communicant) {
        super();
        this.id_register = id_register;
        this.i_communicant = i_communicant;
    }

    public int getId_register() {
        return id_register;
    }

    public void setId_register(int id_register) {
        this.id_register = id_register;
    }

    public int getI_communicant() {
        return i_communicant;
    }

    public void setI_communicant(int i_communicant) {
        this.i_communicant = i_communicant;
    }

    @Override

    public String toString() {
        return "RegisterCommunicant [id_register=" + id_register + ", i_communicant=" + i_communicant + "]";
    }

}
```

StatusRegister.java

```java
package model;
public enum StatusRegister {
    ANDAMENTO("andamento"),
    PENDENTE("pendente"),
    ENCERRADO("encerrado");

    private final String dbValue;

    StatusRegister(String dbValue) {
        this.dbValue = dbValue;
    }

    public String toDatabaseValue() {
        return dbValue;
    }

    public static StatusRegister fromString(String value) {
        if (value == null) return null;
        for (StatusRegister s : values()) {
            if (s.dbValue.equalsIgnoreCase(value)) {
                return s;
            }
        }
        throw new IllegalArgumentException("Status inválido: " + value);
    }
}
```

8. TELAS SWING

Dashboardvisual.java




```java
package view;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.util.List;

import javax.swing.BorderFactory;
import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.SwingConstants;
import javax.swing.SwingUtilities;
import javax.swing.border.EmptyBorder;

import dao.DisappearanceListDao;
import model.DisappearanceListItem;
import model.StatusRegister;
```

```java
public class ScreenDisappearance extends JFrame {

    private JPanel painelAtivos;
    private JPanel listaPendentes;
    private JPanel listaEncerrados;

    private List<DisappearanceListItem> desaparecidos;

    public ScreenDisappearance() {
        setTitle("SCPD - Sistema de Controle de Pessoas Desaparecidas");
        setSize(1300, 750);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setResizable(false);
        setLocationRelativeTo(null);
        getContentPane().setLayout(null);
        getContentPane().setBackground(Color.decode("#011826"));

        //Sidebar
        JPanel sidebar = new JPanel();
        sidebar.setLayout(null);
        sidebar.setBackground(new Color(25, 35, 45));
        sidebar.setBounds(0, 0, 260, 750);
        getContentPane().add(sidebar);

        JLabel tituloSidebar = new JLabel("SCPD", SwingConstants.CENTER);
        tituloSidebar.setFont(new Font("Arial", Font.BOLD, 22));
        tituloSidebar.setForeground(Color.WHITE);
        tituloSidebar.setBounds(0, 20, 260, 30);
        sidebar.add(tituloSidebar);

        JLabel subtitulo = new JLabel("Controle de Desaparecidos", SwingConstants.CENTER);
        subtitulo.setFont(new Font("Arial", Font.PLAIN, 12));
        subtitulo.setForeground(new Color(180, 180, 180));
        subtitulo.setBounds(0, 55, 260, 35);
        sidebar.add(subtitulo);

        String[] menu = { "Dashboard", "Pessoas Desaparecidas", "Busca Avançada", "Novo Caso" };
        int y = 120;

        for (String item : menu) {
            JButton btn = new JButton(item);
            btn.setBounds(15, y, 230, 40);
            btn.setFocusPainted(false);
            btn.setForeground(Color.WHITE);
            btn.setBackground(new Color(25, 35, 45));
            btn.setFont(new Font("Arial", Font.PLAIN, 14));
            btn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
            btn.setBorder(BorderFactory.createEmptyBorder());

            btn.addMouseListener(new MouseAdapter() {
                @Override
                public void mouseEntered(MouseEvent e) {
                    btn.setBackground(new Color(35, 50, 65));
                }
```

```java
                @Override
                public void mouseExited(MouseEvent e) {
                    btn.setBackground(new Color(25, 35, 45));
                }
            });

            switch (item) {

            case "Novo Caso" -> btn.addActionListener(e -> SwingUtilities.invokeLater(() -> {
                RegisterPersonMissing tela = new RegisterPersonMissing();
                tela.abrir();
            }));

            case "Dashboard" -> btn.addActionListener(e -> SwingUtilities.invokeLater(() -> {
                new DashboardVisual();
                this.dispose();
            }));

            case "Pessoas Desaparecidas" -> btn.addActionListener(e ->
            SwingUtilities.invokeLater(() -> {
                new ScreenDisappearance().setVisible(true);
                this.dispose(); // se quiser fechar a tela atual igual ao Dashboard
            }));

            case "Busca Avançada" -> btn.addActionListener(e ->
            SwingUtilities.invokeLater(() -> {
                JFrame tela = new JFrame("Busca Avançada");
                tela.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                tela.setSize(1100, 700);
                tela.setLocationRelativeTo(null);

                tela.setContentPane(new SearchMissingPersons());

                tela.setVisible(true);

                this.dispose();
            })
        );

        }

            sidebar.add(btn);
            y += 50;
        }
```

```
        JPanel mainPanel = new JPanel();
        mainPanel.setBackground(new Color(17, 24, 38));
        mainPanel.setBounds(260, 0, 1040, 750);
        mainPanel.setLayout(new BorderLayout());
        getContentPane().add(mainPanel);

        // Header
        JPanel header = new JPanel(new BorderLayout());
        header.setBackground(new Color(17, 24, 38));
        header.setBorder(BorderFactory.createEmptyBorder(15, 20, 10, 20));
        JLabel lblTitulo = new JLabel("Busca Avançada por Características");
        lblTitulo.setForeground(Color.WHITE);
        lblTitulo.setFont(new Font("Segoe UI", Font.BOLD, 18));
        header.add(lblTitulo, BorderLayout.WEST);
        mainPanel.add(header, BorderLayout.NORTH);

        JPanel conteudo = new JPanel();
        conteudo.setLayout(new BoxLayout(conteudo, BoxLayout.Y_AXIS));
        conteudo.setBackground(new Color(17, 24, 38));
        conteudo.setBorder(new EmptyBorder(10, 25, 20, 25));

        JScrollPane scroll = new JScrollPane(conteudo);
        scroll.setBorder(null);
        mainPanel.add(scroll, BorderLayout.CENTER);

        //
        conteudo.add(criarTituloSecao("Casos Ativos"));
        conteudo.add(Box.createVerticalStrut(10));
        painelAtivos = new JPanel();
        painelAtivos.setLayout(new BoxLayout(painelAtivos, BoxLayout.Y_AXIS));
        painelAtivos.setOpaque(false);
        conteudo.add(painelAtivos);

        conteudo.add(Box.createVerticalStrut(50));
        conteudo.add(criarTituloSecao("Casos Pendentes"));
        listaPendentes = new JPanel();
        listaPendentes.setLayout(new BoxLayout(listaPendentes, BoxLayout.Y_AXIS));
        listaPendentes.setOpaque(false);
        conteudo.add(listaPendentes);

        conteudo.add(Box.createVerticalStrut(50));
        conteudo.add(criarTituloSecao("Casos Encerrados"));
        listaEncerrados = new JPanel();
        listaEncerrados.setLayout(new BoxLayout(listaEncerrados, BoxLayout.Y_AXIS));
        listaEncerrados.setOpaque(false);
        conteudo.add(listaEncerrados);
```

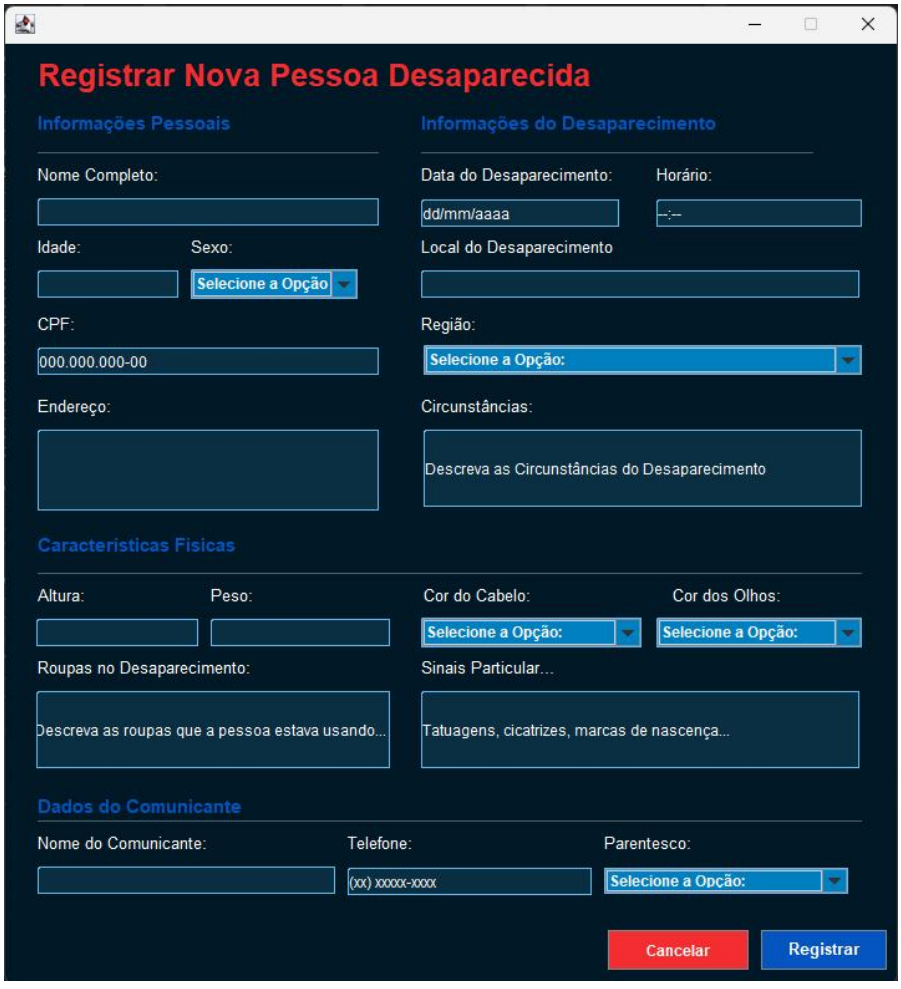
```
        // Carregar os dados
        try {
            DisappearanceListDao dao = new DisappearanceListDao();
            desaparecidos = dao.listarTodos();
            carregarCards();
        } catch (Exception e) {
            e.printStackTrace();
        }

        setVisible(true);
    }

    private JLabel criarTituloSecao(String txt) {
        JLabel label = new JLabel(txt);
        label.setFont(new Font("Arial", Font.BOLD, 24));
        label.setForeground(new Color(255, 184, 76));
        return label;
    }

    private void carregarCards() {
        painelAtivos.removeAll();
        listaPendentes.removeAll();
        listaEncerrados.removeAll();

        for (DisappearanceListItem d : desaparecidos) {
            StatusRegister status = d.getStatus();
            if (status == null)
                status = StatusRegister.PENDENTE;

            switch (status) {
            case ANDAMENTO:
                painelAtivos.add(criarCardAtivo(d));
                painelAtivos.add(Box.createVerticalStrut(15));
                break;
            case PENDENTE:
                listaPendentes.add(criarCardAtivo(d));
                listaPendentes.add(Box.createVerticalStrut(15));
                break;
            case ENCERRADO:
                listaEncerrados.add(criarCard(d, "CASO ENCERRADO", new Color(100, 100, 100)));
                listaEncerrados.add(Box.createVerticalStrut(15));
                break;
            }
        }

        revalidate();
        repaint();
    }
```

```java
// Cards de Ativos e Pendentes
private JPanel criarCardAtivo(DisappearanceListItem d) {
    // Header com status
    String titulo = "CASO ATIVO";
    if (d.getStatus() == StatusRegister.PENDENTE)
        titulo = "CASO PENDENTE";

    JPanel card = new JPanel();
    card.setLayout(null);
    card.setPreferredSize(new Dimension(500, 380));
    card.setMaximumSize(new Dimension(850, 380));
    card.setBackground(new Color(22, 25, 40));
    card.setBorder(BorderFactory.createLineBorder(new Color(40, 40, 55), 2));

    // Header
    JPanel header = new JPanel(null);
    header.setBounds(0, 0, 500, 60);
    header.setBackground(new Color(200, 60, 80));
    JLabel lblTitle = new JLabel(titulo);
    lblTitle.setForeground(Color.WHITE);
    lblTitle.setFont(new Font("Arial", Font.BOLD, 18));
    lblTitle.setBounds(20, 10, 450, 40);
    header.add(lblTitle);
    card.add(header);

    // Nome grande
    JLabel lblNome = new JLabel(d.getPersonName());
    lblNome.setForeground(Color.WHITE);
    lblNome.setFont(new Font("Arial", Font.BOLD, 22));
    lblNome.setBounds(20, 70, 450, 30);
    card.add(lblNome);

    // Idade, gênero e físico
    JLabel lblInfo = new JLabel("Idade: " + d.getAge() + " | Gênero: " + d.getGender());
    lblInfo.setForeground(Color.WHITE);
    lblInfo.setBounds(20, 105, 450, 25);
    card.add(lblInfo);

    JLabel lblFisico = new JLabel("Altura: " + d.getHeight() + " m | Peso: " + d.getWeight() + " kg | Cabelo: "
            + d.getHairColor() + " | Olhos: " + d.getEyesColor());
    lblFisico.setForeground(Color.WHITE);
    lblFisico.setBounds(20, 135, 450, 25);
    card.add(lblFisico);

    // Contato
    JLabel lblContato = new JLabel("Contato: " + d.getKinship() + " | " + d.getCellphone());
    lblContato.setForeground(Color.WHITE);
    lblContato.setBounds(20, 165, 450, 25);
    card.add(lblContato);

    // Local
    JLabel lblLocal = new JLabel("Local: " + d.getLocation());
    lblLocal.setForeground(Color.WHITE);
    lblLocal.setBounds(20, 195, 450, 25);
    card.add(lblLocal);
```

```java
    // Contexto
    JLabel lblDesc = new JLabel("<html>" + d.getContextDisappearance() + "</html>");
    lblDesc.setForeground(Color.LIGHT_GRAY);
    lblDesc.setBounds(20, 225, 460, 60);
    card.add(lblDesc);

    // Painel de botões
    JPanel painelBotoes = new JPanel();
    painelBotoes.setLayout(new GridLayout(2, 1, 0, 10));
    painelBotoes.setOpaque(false);
    painelBotoes.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));

    JButton btnPendente = new JButton("Marcar como Pendente");
    btnPendente.setBackground(new Color(255, 180, 50));
    btnPendente.setForeground(Color.BLACK);
    btnPendente.setFont(new Font("Arial", Font.BOLD, 16));
    btnPendente.setFocusPainted(false);

    JButton btnEncerrado = new JButton("Encerrar Caso");
    btnEncerrado.setBackground(new Color(200, 60, 80));
    btnEncerrado.setForeground(Color.WHITE);
    btnEncerrado.setFont(new Font("Arial", Font.BOLD, 16));
    btnEncerrado.setFocusPainted(false);

    painelBotoes.add(btnPendente);
    painelBotoes.add(btnEncerrado);

    card.add(painelBotoes);
    painelBotoes.setBounds(20, 295, 460, 70);

    btnPendente.addActionListener(e -> {
        d.setStatus(StatusRegister.PENDENTE);
        try {
            DisappearanceListDao dao = new DisappearanceListDao();
            dao.atualizarStatusRegister(d.getIdRegister(), StatusRegister.PENDENTE); // atualiza no banco
        } catch (Exception ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(this, "Erro ao atualizar status no banco!");
        }
        carregarCards(); // atualiza a interface
    });

    btnEncerrado.addActionListener(e -> {
        d.setStatus(StatusRegister.ENCERRADO);
        try {
            DisappearanceListDao dao = new DisappearanceListDao();
            dao.atualizarStatusRegister(d.getIdRegister(), StatusRegister.ENCERRADO);
        } catch (Exception ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(this, "Erro ao atualizar status no banco!");
        }
        carregarCards();
    });

    return card;
```

```java
// Cards Simples
private JPanel criarCard(DisappearanceListItem d, String titulo, Color corHeader) {
    JPanel card = new JPanel();
    card.setLayout(null);
    card.setPreferredSize(new Dimension(500, 380)); // de 300 para 380
    card.setMaximumSize(new Dimension(850, 380));
    card.setBackground(new Color(22, 25, 40));
    card.setBorder(BorderFactory.createLineBorder(new Color(40, 40, 55), 2));

    // Header
    JPanel header = new JPanel(null);
    header.setBounds(0, 0, 500, 60);
    header.setBackground(corHeader);

    JLabel lblTitle = new JLabel(titulo);
    lblTitle.setForeground(Color.WHITE);
    lblTitle.setFont(new Font("Arial", Font.BOLD, 18));
    lblTitle.setBounds(20, 10, 450, 40);
    header.add(lblTitle);

    card.add(header);

    // Nome grande
    JLabel lblNome = new JLabel(d.getPersonName());
    lblNome.setForeground(Color.WHITE);
    lblNome.setFont(new Font("Arial", Font.BOLD, 22));
    lblNome.setBounds(20, 80, 450, 30);
    card.add(lblNome);

    // Idade, gênero e físico
    JLabel lblInfo = new JLabel("Idade: " + d.getAge() + " | Gênero: " + d.getGender());
    lblInfo.setForeground(Color.WHITE);
    lblInfo.setBounds(20, 115, 450, 25);
    card.add(lblInfo);

    JLabel lblFisico = new JLabel("Altura: " + d.getHeight() + " m | Peso: " + d.getWeight() + " kg | Cabelo: "
            + d.getHairColor() + " | Olhos: " + d.getEyesColor());
    lblFisico.setForeground(Color.WHITE);
    lblFisico.setBounds(20, 145, 450, 25);
    card.add(lblFisico);

    // Contato
    JLabel lblContato = new JLabel("Contato: " + d.getKinship() + " | " + d.getCellphone());
    lblContato.setForeground(Color.WHITE);
    lblContato.setBounds(20, 175, 450, 25);
    card.add(lblContato);

    // Local
    JLabel lblLocal = new JLabel("Local: " + d.getLocation());
    lblLocal.setForeground(Color.WHITE);
    lblLocal.setBounds(20, 205, 450, 25);
    card.add(lblLocal);

    // Descrição
    JLabel lblDesc = new JLabel("<html>" + d.getContextDisappearance() + "</html>");
```

RegisterPersonMissing.java



```java
package view;

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.SQLException;
import java.time.LocalDate;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

import javax.swing.BorderFactory;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFormattedTextField;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JSeparator;
import javax.swing.JTextArea;
import javax.swing.JTextField;

import dao.BaseDao;
import dao.CharacteristicsDao;
import dao.CommunicantDao;
import dao.DisappearanceDao;
import dao.InformsDao;
import dao.PeopleDao;
import dao.RegisterCommunicantDao;
import dao.RegisterDao;
import model.Communicant;
import model.Disappearance;
import model.People;
import model.PhysicalCharacteristics;
import model.Register;
import model.StatusRegister;

public class RegisterPersonMissing {

    private JFrame frame;
    private JTextField inputNome;
    private JTextField inutIdade;
    private JTextField inputLocalDesaparecido;
    private JTextField inputAltura;
    private JTextField inputPeso;
    private JTextField inputNomeComunicante;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
```

```java
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                RegisterPersonMissing window = new RegisterPersonMissing();
                window.frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the application.
 */

// para abrir a tela na outra tela
public void abrir() {
    frame.setVisible(true);
}

public RegisterPersonMissing() {
    initialize();
}

/**
 * Initialize the contents of the frame.
 */
private void initialize() {
    frame = new JFrame();
    frame.getContentPane().setForeground(new Color(255, 255, 255));
    frame.getContentPane().setFont(new Font("Arial", Font.BOLD, 18));
    frame.setBounds(100, 100, 450, 300);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    frame.setSize(732, 794);
    frame.setLocationRelativeTo(null);
    frame.setResizable(false);
    frame.getContentPane().setLayout(null);
    frame.getContentPane().setBackground(Color.decode("#011826"));

    JLabel titleMain = new JLabel("Registrar Nova Pessoa Desaparecida");
    titleMain.setFont(new Font("Arial", Font.BOLD, 25));
    titleMain.setForeground(new Color(244, 45, 48));
    titleMain.setBounds(26, 10, 541, 32);
    frame.getContentPane().add(titleMain);
```

```
JLabel titulo1Cadastro = new JLabel("Informações Pessoais");
titulo1Cadastro.setFont(new Font("Arial", Font.BOLD, 15));
titulo1Cadastro.setForeground(new Color(4, 85, 191));
titulo1Cadastro.setBounds(26, 52, 183, 24);
frame.getContentPane().add(titulo1Cadastro);

JSeparator lineSeparator = new JSeparator();
lineSeparator.setForeground(new Color(255, 255, 255, 80));
lineSeparator.setBounds(26, 87, 274, 1);
frame.getContentPane().add(lineSeparator);

JLabel LabelNome = new JLabel("Nome Completo:");
LabelNome.setFont(new Font("Arial", Font.PLAIN, 13));
LabelNome.setForeground(new Color(255, 255, 255));
LabelNome.setBackground(new Color(255, 255, 255));
LabelNome.setBounds(26, 98, 113, 16);
frame.getContentPane().add(LabelNome);

inputNome = new JTextField();
inputNome.setForeground(new Color(255, 255, 255));
inputNome.setFont(new Font("Arial", Font.PLAIN, 14));
inputNome.setBackground(new Color(10, 46, 66));
inputNome.setBounds(26, 124, 274, 22);
frame.getContentPane().add(inputNome);
inputNome.setColumns(10);
inputNome.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));

JLabel labelidade = new JLabel("Idade: ");
labelidade.setForeground(Color.WHITE);
labelidade.setFont(new Font("Arial", Font.PLAIN, 13));
labelidade.setBackground(Color.WHITE);
labelidade.setBounds(26, 156, 113, 16);
frame.getContentPane().add(labelidade);

inutIdade = new JTextField();
inutIdade.setForeground(new Color(255, 255, 255));
inutIdade.setFont(new Font("Arial", Font.PLAIN, 14));
inutIdade.setColumns(10);
inutIdade.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inutIdade.setBackground(new Color(10, 46, 66));
inutIdade.setBounds(26, 182, 113, 22);
frame.getContentPane().add(inutIdade);

JLabel labelSexo = new JLabel("Sexo:");
labelSexo.setForeground(Color.WHITE);
labelSexo.setFont(new Font("Arial", Font.PLAIN, 13));
labelSexo.setBackground(Color.WHITE);
labelSexo.setBounds(149, 156, 113, 16);
frame.getContentPane().add(labelSexo);
```

```
JComboBox inputSexo = new JComboBox();
inputSexo.setForeground(new Color(255, 255, 255));
inputSexo.setModel(
        new DefaultComboBoxModel(new String[] {"Selecione a Opção", "Feminino ", "Masculino ", "Outro" }));
inputSexo.setBounds(149, 181, 134, 24);
frame.getContentPane().add(inputSexo);
inputSexo.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputSexo.setBackground(new Color(0, 128, 192));

JLabel labelCPF = new JLabel("CPF:");
labelCPF.setForeground(Color.WHITE);
labelCPF.setFont(new Font("Arial", Font.PLAIN, 13));
labelCPF.setBackground(Color.WHITE);
labelCPF.setBounds(26, 218, 113, 16);
frame.getContentPane().add(labelCPF);

JFormattedTextField inputCPF = new JFormattedTextField();
inputCPF.setForeground(new Color(255, 255, 255));
inputCPF.setText("000.000.000-00");
inputCPF.setBounds(26, 244, 274, 22);
inputCPF.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputCPF.setBackground(new Color(10, 46, 66));
frame.getContentPane().add(inputCPF);

JLabel labelEndereco = new JLabel("Endereço:");
labelEndereco.setForeground(Color.WHITE);
labelEndereco.setFont(new Font("Arial", Font.PLAIN, 13));
labelEndereco.setBackground(Color.WHITE);
labelEndereco.setBounds(26, 284, 113, 16);
frame.getContentPane().add(labelEndereco);

JTextArea inputEndereco = new JTextArea();
inputEndereco.setForeground(new Color(255, 255, 255));
inputEndereco.setBounds(26, 310, 274, 65);
inputEndereco.setBackground(new Color(10, 46, 66));
inputEndereco.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
frame.getContentPane().add(inputEndereco);

JLabel titulo2Cadastro = new JLabel("Informações do Desaparecimento");
titulo2Cadastro.setForeground(new Color(4, 85, 191));
titulo2Cadastro.setFont(new Font("Arial", Font.BOLD, 15));
titulo2Cadastro.setBounds(334, 52, 302, 24);
frame.getContentPane().add(titulo2Cadastro);

JSeparator lineSeparator_1 = new JSeparator();
lineSeparator_1.setForeground(new Color(255, 255, 255, 80));
lineSeparator_1.setBounds(334, 87, 315, 1);
frame.getContentPane().add(lineSeparator_1);

JLabel LabelDataDesaparecimento = new JLabel("Data do Desaparecimento:");
LabelDataDesaparecimento.setForeground(Color.WHITE);
LabelDataDesaparecimento.setFont(new Font("Arial", Font.PLAIN, 13));
LabelDataDesaparecimento.setBackground(Color.WHITE);
LabelDataDesaparecimento.setBounds(334, 98, 165, 17);
frame.getContentPane().add(LabelDataDesaparecimento);
```

```
JFormattedTextField inputDataDesaparecimento = new JFormattedTextField();
inputDataDesaparecimento.setForeground(new Color(255, 255, 255));
inputDataDesaparecimento.setBackground(new Color(10, 46, 66));
inputDataDesaparecimento.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputDataDesaparecimento.setText("dd/mm/aaaa");
inputDataDesaparecimento.setBounds(334, 125, 159, 22);
frame.getContentPane().add(inputDataDesaparecimento);

JLabel labelHorario = new JLabel("Horário: ");
labelHorario.setForeground(Color.WHITE);
labelHorario.setFont(new Font("Arial", Font.PLAIN, 13));
labelHorario.setBackground(Color.WHITE);
labelHorario.setBounds(523, 98, 165, 17);
frame.getContentPane().add(labelHorario);

JFormattedTextField inputHorarioDesaparecimento = new JFormattedTextField();
inputHorarioDesaparecimento.setForeground(new Color(255, 255, 255));
inputHorarioDesaparecimento.setText("--:--");
inputHorarioDesaparecimento.setBounds(523, 125, 165, 22);
inputHorarioDesaparecimento.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputHorarioDesaparecimento.setBackground(new Color(10, 46, 66));
frame.getContentPane().add(inputHorarioDesaparecimento);

JLabel LabelLocaalDesaparecido = new JLabel("Local do Desaparecimento");
LabellocaalDesaparecido.setForeground(Color.WHITE);
LabellocaalDesaparecido.setFont(new Font("Arial", Font.PLAIN, 13));
LabellocaalDesaparecido.setBackground(Color.WHITE);
LabelLocaalDesaparecido.setBounds(334, 156, 165, 17);
frame.getContentPane().add(labelLocaalDesaparecido);

inputLocalDesaparecido = new JTextField();
inputLocalDesaparecido.setForeground(new Color(255, 255, 255));
inputLocalDesaparecido.setFont(new Font("Arial", Font.PLAIN, 14));
inputLocalDesaparecido.setColumns(10);
inputLocalDesaparecido.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputLocalDesaparecido.setBackground(new Color(10, 46, 66));
inputLocalDesaparecido.setBounds(334, 182, 352, 22);
frame.getContentPane().add(inputLocalDesaparecido);

JLabel labelRegião = new JLabel("Região: ");
labelRegião.setForeground(Color.WHITE);
labelRegião.setFont(new Font("Arial", Font.PLAIN, 13));
labelRegião.setBackground(Color.WHITE);
labelRegião.setBounds(334, 218, 113, 16);
frame.getContentPane().add(labelRegião);

JComboBox inputRegiao = new JComboBox();
inputRegiao.setForeground(new Color(255, 255, 255));
inputRegiao.setModel(new DefaultComboBoxModel(
        new String[] { "Selecione a Opção: ", "Centro ", "Norte", "Sul", "Leste", "Oeste" }));
inputRegiao.setBounds(336, 242, 352, 24);
inputRegiao.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputRegiao.setBackground(new Color(0, 128, 192));
```

```
JLabel labelCircunstancias = new JLabel("Circunstâncias:");
labelCircunstancias.setForeground(Color.WHITE);
labelCircunstancias.setFont(new Font("Arial", Font.PLAIN, 13));
labelCircunstancias.setBackground(Color.WHITE);
labelCircunstancias.setBounds(334, 284, 113, 16);
frame.getContentPane().add(labelCircunstancias);

JFormattedTextField inputCircunstancias = new JFormattedTextField();
inputCircunstancias.setForeground(new Color(255, 255, 255));
inputCircunstancias.setText("Descreva as Circunstâncias do Desaparecimento");
inputCircunstancias.setBounds(336, 310, 352, 62);
inputCircunstancias.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true
inputCircunstancias.setBackground(new Color(10, 46, 66));

frame.getContentPane().add(inputCircunstancias);

JLabel lblCaractersticasFsicas = new JLabel("Características Físicas");
lblCaractersticasFsicas.setForeground(new Color(4, 85, 191));
lblCaractersticasFsicas.setFont(new Font("Arial", Font.BOLD, 15));
lblCaractersticasFsicas.setBounds(26, 391, 183, 24);
frame.getContentPane().add(lblCaractersticasFsicas);

JSeparator lineSeparator_2 = new JSeparator();
lineSeparator_2.setForeground(new Color(255, 255, 255, 80));
lineSeparator_2.setBounds(26, 425, 662, 1);
frame.getContentPane().add(lineSeparator_2);

inputAltura = new JTextField();
inputAltura.setForeground(new Color(255, 255, 255));
inputAltura.setFont(new Font("Arial", Font.PLAIN, 14));
inputAltura.setColumns(10);
inputAltura.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputAltura.setBackground(new Color(10, 46, 66));
inputAltura.setBounds(25, 462, 130, 22);
frame.getContentPane().add(inputAltura);

JLabel labelAltura = new JLabel("Altura: ");
labelAltura.setForeground(Color.WHITE);
labelAltura.setFont(new Font("Arial", Font.PLAIN, 13));
labelAltura.setBackground(Color.WHITE);
labelAltura.setBounds(26, 436, 113, 16);
frame.getContentPane().add(labelAltura);

JLabel labelPeso = new JLabel("Peso: ");
labelPeso.setForeground(Color.WHITE);
labelPeso.setFont(new Font("Arial", Font.PLAIN, 13));
labelPeso.setBackground(Color.WHITE);
labelPeso.setBounds(165, 436, 113, 16);
frame.getContentPane().add(labelPeso);
```

```
inputPeso = new JTextField();
inputPeso.setForeground(new Color(255, 255, 255));
inputPeso.setFont(new Font("Arial", Font.PLAIN, 14));
inputPeso.setColumns(10);
inputPeso.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputPeso.setBackground(new Color(10, 46, 66));
inputPeso.setBounds(165, 462, 144, 22);
frame.getContentPane().add(inputPeso);

JComboBox inputCordoCabelo = new JComboBox();
inputCordoCabelo.setForeground(new Color(255, 255, 255));
inputCordoCabelo.setModel(new DefaultComboBoxModel(
        new String[] { "Selecione a Opção: ", "Loiro ", "Castanho", "Preto", "Ruivo ", "Grisalho", "Careca" }));
inputCordoCabelo.setBounds(334, 461, 177, 24);
inputCordoCabelo.setBackground(new Color(0, 128, 192));
inputCordoCabelo.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
frame.getContentPane().add(inputCordoCabelo);

JLabel labelCorDoCabelo = new JLabel("Cor do Cabelo:");
labelCorDoCabelo.setForeground(Color.WHITE);
labelCorDoCabelo.setFont(new Font("Arial", Font.PLAIN, 13));
labelCorDoCabelo.setBackground(Color.WHITE);
labelCorDoCabelo.setBounds(336, 436, 113, 16);
frame.getContentPane().add(labelCorDoCabelo);

JLabel labelCorDosOlhos = new JLabel("Cor dos Olhos:");
labelCorDosOlhos.setForeground(Color.WHITE);
labelCorDosOlhos.setFont(new Font("Arial", Font.PLAIN, 13));
labelCorDosOlhos.setBackground(Color.WHITE);
labelCorDosOlhos.setBounds(536, 436, 113, 16);
frame.getContentPane().add(labelCorDosOlhos);

JComboBox inputCordosOlhos = new JComboBox();
inputCordosOlhos.setForeground(new Color(255, 255, 255));
inputCordosOlhos.setModel(new DefaultComboBoxModel(
        new String[] { "Selecione a Opção: ", "Castanhos ", "Azuis", "Verdes", "Pretos", "Mel " }));
inputCordosOlhos.setBounds(523, 461, 165, 24);
inputCordosOlhos.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputCordosOlhos.setBackground(new Color(0, 128, 192));
frame.getContentPane().add(inputCordosOlhos);

JLabel labelRoupasNoDesaparecimento = new JLabel("Roupas no Desaparecimento: ");
labelRoupasNoDesaparecimento.setForeground(Color.WHITE);
labelRoupasNoDesaparecimento.setFont(new Font("Arial", Font.PLAIN, 13));
labelRoupasNoDesaparecimento.setBackground(Color.WHITE);
labelRoupasNoDesaparecimento.setBounds(26, 494, 185, 16);
frame.getContentPane().add(labelRoupasNoDesaparecimento);

JLabel lblSinaisParticulares = new JLabel("Sinais Particulares: ");
lblSinaisParticulares.setForeground(Color.WHITE);
lblSinaisParticulares.setFont(new Font("Arial", Font.PLAIN, 13));
lblSinaisParticulares.setBackground(Color.WHITE);
lblSinaisParticulares.setBounds(334, 494, 113, 16);
```

```
JFormattedTextField inputSinais = new JFormattedTextField();
inputSinais.setForeground(new Color(255, 255, 255));
inputSinais.setText("Tatuagens, cicatrizes, marcas de nascença...");
inputSinais.setBounds(334, 520, 352, 62);
inputSinais.setBackground(new Color(10, 46, 66));
inputSinais.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
frame.getContentPane().add(inputSinais);

JFormattedTextField inputDescrevaAsRoupas = new JFormattedTextField();
inputDescrevaAsRoupas.setForeground(new Color(255, 255, 255));
inputDescrevaAsRoupas.setText("Descreva as roupas que a pessoa estava usando...");
inputDescrevaAsRoupas.setBounds(25, 520, 284, 62);
inputDescrevaAsRoupas.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputDescrevaAsRoupas.setBackground(new Color(10, 46, 66));
frame.getContentPane().add(inputDescrevaAsRoupas);

JLabel lblDadosDoComunicante = new JLabel("Dados do Comunicante");
lblDadosDoComunicante.setForeground(new Color(4, 85, 191));
lblDadosDoComunicante.setFont(new Font("Arial", Font.BOLD, 15));
lblDadosDoComunicante.setBounds(26, 601, 183, 24);
frame.getContentPane().add(lblDadosDoComunicante);

JSeparator lineSeparator_2_1 = new JSeparator();
lineSeparator_2_1.setForeground(new Color(255, 255, 255, 80));
lineSeparator_2_1.setBounds(26, 624, 662, 1);
frame.getContentPane().add(lineSeparator_2_1);

JLabel labelNomeComunicante = new JLabel("Nome do Comunicante:");
labelNomeComunicante.setForeground(Color.WHITE);
labelNomeComunicante.setFont(new Font("Arial", Font.PLAIN, 13));
labelNomeComunicante.setBackground(Color.WHITE);
labelNomeComunicante.setBounds(26, 635, 185, 16);
frame.getContentPane().add(labelNomeComunicante);

inputNomeComunicante = new JTextField();
inputNomeComunicante.setForeground(new Color(255, 255, 255));
inputNomeComunicante.setFont(new Font("Arial", Font.PLAIN, 14));
inputNomeComunicante.setColumns(10);
inputNomeComunicante.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1, true));
inputNomeComunicante.setBackground(new Color(10, 46, 66));
inputNomeComunicante.setBounds(26, 661, 239, 22);
frame.getContentPane().add(inputNomeComunicante);

JLabel lblTelefone = new JLabel("Telefone: ");
lblTelefone.setForeground(Color.WHITE);
lblTelefone.setFont(new Font("Arial", Font.PLAIN, 13));
lblTelefone.setBackground(Color.WHITE);
lblTelefone.setBounds(275, 635, 185, 16);
frame.getContentPane().add(lblTelefone);

JFormattedTextField inputTelefone = new JFormattedTextField();
inputTelefone.setForeground(new Color(255, 255, 255));
inputTelefone.setText("(xx) xxxxx-xxxx");
inputTelefone.setBounds(275, 662, 196, 22);
inputTelefone.setBackground(new Color(10, 46, 66));
```

```
JLabel labelParentesco = new JLabel("Parentesco: ");
labelParentesco.setForeground(Color.WHITE);
labelParentesco.setFont(new Font("Arial", Font.PLAIN, 13));
labelParentesco.setBackground(Color.WHITE);
labelParentesco.setBounds(481, 635, 185, 16);
frame.getContentPane().add(labelParentesco);

JComboBox inputParentesco = new JComboBox();
inputParentesco.setForeground(new Color(255, 255, 255));
inputParentesco.setModel(new DefaultComboBoxModel(new String[] { "Selecione a Opção: ", "Pai ", "Mãe",
        "Filho(a)", "Cônjuge", "Irmão(a)", "Amigo(a)", "Outro" }));
inputParentesco.setBounds(481, 662, 196, 21);
inputParentesco.setBorder(BorderFactory.createLineBorder(new Color(99, 187, 242), 1));
inputParentesco.setBackground(new Color(0, 128, 192));
frame.getContentPane().add(inputParentesco);

JButton btnCancelar = new JButton("Cancelar");
btnCancelar.setForeground(new Color(255, 255, 255));
btnCancelar.setBackground(new Color(244, 45, 48));
btnCancelar.setBounds(484, 712, 113, 32);
frame.getContentPane().add(btnCancelar);

// Para fechar, quando clica em cancelar
btnCancelar.addActionListener(e -> {
    frame.dispose();
});

JButton btnRegistrar = new JButton("Registrar");
btnRegistrar.setForeground(new Color(255, 255, 255));
```

```
    btnRegistrar.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {

            BaseDao base = new BaseDao();
            Connection conn = null;

            try {
                conn = base.getConnection();
                conn.setAutoCommit(false);

                // CADASTRAR PESSOA

                People pessoa = new People();
                pessoa.setPersonName(inputNome.getText());
                pessoa.setAge(Integer.parseInt(inutIdade.getText()));
                pessoa.setGender(inputSexo.getSelectedItem().toString());
                pessoa.setCPF(inputCPF.getText());
                pessoa.setAddress(inputEndereco.getText());

                PeopleDao pdao = new PeopleDao();
                pdao.insert(conn, pessoa);

                int idPeople = base.getLastInsertedId(conn);

                // CADASTRAR DESAPARECIMENTO

                Disappearance d = new Disappearance();

                try {
                    DateTimeFormatter df = DateTimeFormatter.ofPattern("dd/MM/yyyy");
                    DateTimeFormatter hf = DateTimeFormatter.ofPattern("HH:mm");

                    d.setDateDisappearance(LocalDate.parse(inputDataDesaparecimento.getText(), df));
                    d.setHourDisappearance(LocalTime.parse(inputHorarioDesaparecimento.getText(), hf));

                } catch (Exception ex) {
                    JOptionPane.showMessageDialog(null, "Data ou hora inválida!");
                    return;
                }

                d.setLocation(inputLocalDesaparecido.getText());
                d.setContextDisappearance(inputCircunstancias.getText());
                d.setClothesDisappearance(inputDescrevaAsRoupas.getText());
                d.setId_peoples(idPeople);

                DisappearanceDao ddao = new DisappearanceDao();
                ddao.insert(conn, d);

                int idDisappearance = base.getLastInsertedId(conn);

                // CARACTERISTICAS FÍSICAS

                PhysicalCharacteristics pc = new PhysicalCharacteristics();

                pc.setHeight(Double.parseDouble(inputAltura.getText()));
```

```
        pc.setHeight(Double.parseDouble(inputAltura.getText()));
        pc.setWeight(Double.parseDouble(inputPeso.getText()));
        pc.setHairColor(inputCordoCabelo.getSelectedItem().toString());
        pc.setEyesColor(inputCordosOlhos.getSelectedItem().toString());
        pc.setSigns(inputSinais.getText());
        pc.setIdPeople(idPeople);

        CharacteristicsDao cdao = new CharacteristicsDao();
        cdao.insert(conn, pc);

        // COMUNICANTE

        Communicant com = new Communicant();
        com.setCellphone(inputTelefone.getText());
        com.setKinship(inputParentesco.getSelectedItem().toString());

        CommunicantDao comdao = new CommunicantDao();
        comdao.insert(conn, com);

        int idCommunicant = base.getLastInsertedId(conn);

        // REGISTRO

        Register reg = new Register();
        reg.setDateReport(new java.util.Date());
        reg.setStatusRegister(StatusRegister.ANDAMENTO);
        reg.setIdPeople(idPeople);

        RegisterDao rdao = new RegisterDao();
        rdao.insert(conn, reg);

        int idRegister = base.getLastInsertedId(conn);

        // VINCULAR REGISTRO + COMUNICANTE

        RegisterCommunicantDao rcdao = new RegisterCommunicantDao();
        rcdao.insert(conn, idRegister, idCommunicant);

        // VINCULAR COMUNICANTE + DESAPARECIMENTO

        InformsDao idao = new InformsDao();
        idao.insert(conn, idCommunicant, idDisappearance);

        // FINALIZAR TRANSAÇÃO

        conn.commit();
        JOptionPane.showMessageDialog(null, "Cadastro realizado com sucesso!");

    } catch (Exception ex) {
        try {
            if (conn != null)
                conn.rollback();
        } catch (Exception e2) {
        }
```

```
                ex.printStackTrace();
                JOptionPane.showMessageDialog(null, "Erro ao registrar: " + ex.getMessage());

            } finally {
                try {
                    if (conn != null)
                        conn.close();
                } catch (SQLException ex) {
                }
            }

        }
    });

    btnRegistrar.setFont(new Font("Arial", Font.BOLD, 13));
    btnRegistrar.setBackground(new Color(4, 85, 191));
    btnRegistrar.setBounds(607, 712, 101, 32);
    frame.getContentPane().add(btnRegistrar);
}
}
```

ScreenDisappearence.java

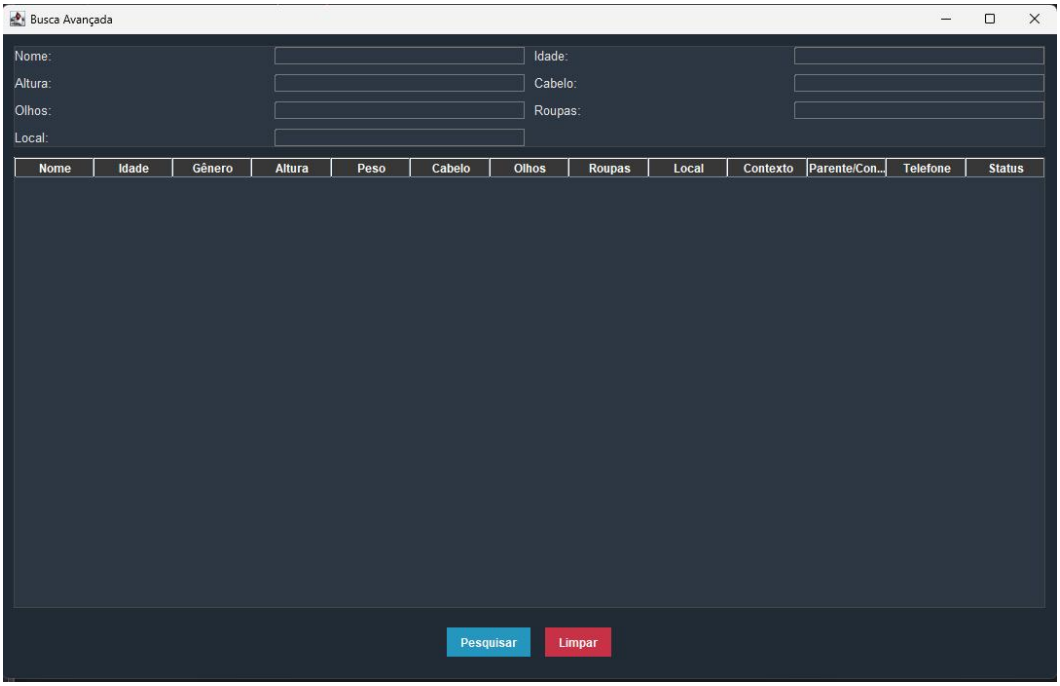

```java
package view;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.GridLayout;

import javax.swing.BorderFactory;
import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.SwingConstants;
import javax.swing.SwingUtilities;

import dao.DashboardDao;

import view.SearchMissingPersons;

public class DashboardVisual extends JFrame {

    private DashboardDao dao;

    public DashboardVisual() {
        dao = new DashboardDao();

        setTitle("Sistema de Controle de Pessoas Desaparecidas");
        setSize(1300, 750);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setResizable(false);
        setLocationRelativeTo(null);

        getContentPane().setLayout(new BorderLayout());
        getContentPane().setBackground(Color.decode("#011826"));

        //SideBar
        JPanel sidebar = new JPanel();
        sidebar.setPreferredSize(new Dimension(260, 750));
        sidebar.setBackground(new Color(25, 35, 45));
        sidebar.setLayout(null);
        getContentPane().add(sidebar, BorderLayout.WEST);

        JLabel tituloSidebar = new JLabel("SCPD", SwingConstants.CENTER);
        tituloSidebar.setFont(new Font("Arial", Font.BOLD, 22));
        tituloSidebar.setForeground(Color.WHITE);
        tituloSidebar.setBounds(0, 20, 260, 30);
        sidebar.add(tituloSidebar);

        JLabel subtitulo = new JLabel("Controle de Desaparecidos", SwingConstants.CENTER);
        subtitulo.setFont(new Font("Arial", Font.PLAIN, 12));
        subtitulo.setForeground(new Color(180, 180, 180));
        subtitulo.setBounds(0, 55, 260, 35);
        sidebar.add(subtitulo);
```

```java
        String[] menu = {"Dashboard", "Pessoas Desaparecidas", "Busca Avançada", "Novo Caso"};
        int y = 120;

        for (String item : menu) {
            JButton btn = new JButton(item);
            btn.setBounds(15, y, 230, 40);
            btn.setFocusPainted(false);
            btn.setForeground(Color.WHITE);
            btn.setBackground(new Color(25, 35, 45));
            btn.setFont(new Font("Arial", Font.PLAIN, 14));
            btn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
            btn.setBorder(BorderFactory.createEmptyBorder());


            switch (item) {

            case "Novo Caso" -> btn.addActionListener(e -> SwingUtilities.invokeLater(() -> {
                RegisterPersonMissing tela = new RegisterPersonMissing();
                tela.abrir();
            }));

            case "Dashboard" -> btn.addActionListener(e -> SwingUtilities.invokeLater(() -> {
                new DashboardVisual();
                this.dispose();
            }));

            case "Pessoas Desaparecidas" -> btn.addActionListener(e ->
            SwingUtilities.invokeLater(() -> {
                new ScreenDisappearance().setVisible(true);
                this.dispose(); // se quiser fechar a tela atual igual ao Dashboard
            }));

            case "Busca Avançada" -> btn.addActionListener(e ->
            SwingUtilities.invokeLater(() -> {
                JFrame tela = new JFrame("Busca Avançada");
                tela.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                tela.setSize(1100, 700);
                tela.setLocationRelativeTo(null);

                tela.setContentPane(new SearchMissingPersons());

                tela.setVisible(true);

                this.dispose();
            })
        );

        }

            sidebar.add(btn);
            y += 50;
        }
```

```
        //Painel Principal
        JPanel mainPanel = new JPanel(new BorderLayout());
        mainPanel.setBackground(new Color(33, 43, 54));
        getContentPane().add(mainPanel, BorderLayout.CENTER);

        //Cabeçalho
        JPanel header = new JPanel(new BorderLayout());
        header.setBackground(new Color(40, 50, 60));
        header.setBorder(BorderFactory.createEmptyBorder(10, 20, 10, 20));
        JLabel title = new JLabel("Dashboard - Visão Geral");
        title.setForeground(Color.WHITE);
        title.setFont(new Font("Segoe UI", Font.BOLD, 18));
        header.add(title, BorderLayout.WEST);
        mainPanel.add(header, BorderLayout.NORTH);

        //Banco
        JPanel centerPanel = new JPanel(new GridLayout(2, 2, 20, 20));
        centerPanel.setBackground(new Color(33, 43, 54));
        centerPanel.setBorder(BorderFactory.createEmptyBorder(30, 30, 30, 30));

        // Buscando os totais do banco
        int totalCasos = dao.contarTotalCasos();
        int casosAtivos = dao.contarCasosPorStatus("ANDAMENTO");
        int casosPendentes = dao.contarCasosPorStatus("PENDENTE");
        int casosEncerrados = dao.contarCasosPorStatus("ENCERRADO");

        centerPanel.add(createInfoCard("Total de Casos", String.valueOf(totalCasos), "Todos os registros", new Color(50, 90, 200)));
        centerPanel.add(createInfoCard("Casos Ativos", String.valueOf(casosAtivos), "Em investigação", new Color(200, 60, 60)));
        centerPanel.add(createInfoCard("Casos Pendentes", String.valueOf(casosPendentes), "Aguardando andamento", new Color(230, 180, 60)));
        centerPanel.add(createInfoCard("Casos Encerrados", String.valueOf(casosEncerrados), "Concluídos", new Color(60, 170, 90)));

        mainPanel.add(centerPanel, BorderLayout.CENTER);

        setVisible(true);
    }

    private JPanel createInfoCard(String title, String number, String desc, Color color) {
        JPanel card = new JPanel(new BorderLayout());
        card.setBackground(new Color(40, 50, 60));
        card.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

        JLabel lblTitle = new JLabel(title);
        lblTitle.setForeground(new Color(200, 200, 200));
        lblTitle.setFont(new Font("Segoe UI", Font.PLAIN, 14));

        JLabel lblNumber = new JLabel(number);
        lblNumber.setForeground(Color.WHITE);
        lblNumber.setFont(new Font("Segoe UI", Font.BOLD, 28));

        JLabel lblDesc = new JLabel(desc);
        lblDesc.setForeground(color);
        lblDesc.setFont(new Font("Segoe UI", Font.PLAIN, 12));

        JPanel info = new JPanel();
        info.setLayout(new BoxLayout(info, BoxLayout.Y_AXIS));
        info.setBackground(new Color(40, 50, 60));
        info.add(lblTitle);
```

```
        info.add(Box.createVerticalStrut(10));
        info.add(lblNumber);
        info.add(Box.createVerticalStrut(5));
        info.add(lblDesc);

        card.add(info, BorderLayout.WEST);
        return card;
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(DashboardVisual::new);
    }
}
```

SearchMissingPersons.java




```java
package view;

import dao.SearchMissingDao;
import model.DisappearanceListItem;

import javax.swing.*;
import javax.swing.border.EmptyBorder;
import javax.swing.border.LineBorder;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.util.List;

public class SearchMissingPersons extends JPanel {

    private final JTextField campoNome;
    private final JTextField campoIdade;
    private final JTextField campoAltura;
    private final JTextField campoCabelo;
    private final JTextField campoOlhos;
    private final JTextField campoRoupas;
    private final JTextField campoLocal;

    private final JTable tabela;
    private final DefaultTableModel modelo;

    private final SearchMissingDao dao;

    public SearchMissingPersons() {

        this.dao = new SearchMissingDao();

        setLayout(new BorderLayout(10, 10));
        setBorder(new EmptyBorder(12, 12, 12, 12));

        Color darkBg = new Color(33, 43, 54);
        Color darkPanel = new Color(45,55,67);
        Color textColor = new Color(230, 230, 230);

        setBackground(darkBg);

        JPanel topo = new JPanel(new GridLayout(4, 4, 10, 10));
        topo.setBackground(darkPanel);
        topo.setBorder(new LineBorder(new Color(70, 70, 70), 1, true));

        campoNome = criarCampo(darkPanel, textColor);
        campoIdade = criarCampo(darkPanel, textColor);
        campoAltura = criarCampo(darkPanel, textColor);
        campoCabelo = criarCampo(darkPanel, textColor);
        campoOlhos = criarCampo(darkPanel, textColor);
        campoRoupas = criarCampo(darkPanel, textColor);
        campoLocal = criarCampo(darkPanel, textColor);
```

```
        adicionarCampo(topo, "Nome:", campoNome, textColor);
        adicionarCampo(topo, "Idade:", campoIdade, textColor);
        adicionarCampo(topo, "Altura:", campoAltura, textColor);
        adicionarCampo(topo, "Cabelo:", campoCabelo, textColor);
        adicionarCampo(topo, "Olhos:", campoOlhos, textColor);
        adicionarCampo(topo, "Roupas:", campoRoupas, textColor);
        adicionarCampo(topo, "Local:", campoLocal, textColor);

        add(topo, BorderLayout.NORTH);

        JPanel painelBotoes = new JPanel(new FlowLayout(FlowLayout.CENTER, 15, 10));
        painelBotoes.setBackground(darkBg);

        JButton botaoPesquisar = criarBotao("Pesquisar", new Color(37, 150, 190));
        JButton botaoLimpar = criarBotao("Limpar", new Color(200, 50, 70));

        painelBotoes.add(botaoPesquisar);
        painelBotoes.add(botaoLimpar);

        add(painelBotoes, BorderLayout.SOUTH);

        modelo = new DefaultTableModel(new String[]{
                "Nome", "Idade", "Gênero",
                "Altura", "Peso", "Cabelo", "Olhos",
                "Roupas", "Local", "Contexto",
                "Parente/Contato", "Telefone", "Status"
        }, 0) {
            @Override
            public boolean isCellEditable(int row, int column) { return false; }
        };

        tabela = new JTable(modelo);
        tabela.setRowHeight(26);
        tabela.setShowGrid(false);
        tabela.setIntercellSpacing(new Dimension(0, 0));
        tabela.setAutoCreateRowSorter(true);

        tabela.setBackground(darkPanel);
        tabela.setForeground(textColor);
        tabela.setSelectionBackground(new Color(60, 60, 60));
        tabela.setSelectionForeground(Color.WHITE);

        tabela.getTableHeader().setBackground(new Color(55, 55, 55));
        tabela.getTableHeader().setForeground(Color.WHITE);
        tabela.getTableHeader().setFont(new Font("SansSerif", Font.BOLD, 12));

        JScrollPane scroll = new JScrollPane(tabela);
        scroll.setBorder(new LineBorder(new Color(70, 70, 70), 1, true));
        scroll.getViewport().setBackground(darkPanel);

        add(scroll, BorderLayout.CENTER);

        // Ações dos botões
        botaoPesquisar.addActionListener(e -> pesquisar());
        botaoLimpar.addActionListener(e -> limparCampos());
```

```
private JTextField criarCampo(Color bg, Color fg) {
    JTextField campo = new JTextField();
    campo.setBorder(new LineBorder(new Color(120, 120, 120), 1, true));
    campo.setBackground(bg);
    campo.setForeground(fg);
    campo.setCaretColor(Color.WHITE);
    return campo;
}


private void adicionarCampo(JPanel painel, String texto, JTextField campo, Color textColor) {
    JLabel label = new JLabel(texto);
    label.setForeground(textColor);
    label.setFont(new Font("SansSerif", Font.PLAIN, 13));
    painel.add(label);
    painel.add(campo);
}


private JButton criarBotao(String texto, Color cor) {
    JButton botao = new JButton(texto);
    botao.setFocusPainted(false);
    botao.setBackground(cor);
    botao.setForeground(Color.WHITE);
    botao.setCursor(new Cursor(Cursor.HAND_CURSOR));
    botao.setBorder(new EmptyBorder(7, 14, 7, 14));
    return botao;
}
```

```
private void pesquisar() {
    String nome = campoNome.getText().trim();
    String idade = campoIdade.getText().trim();
    String altura = campoAltura.getText().trim();
    String cabelo = campoCabelo.getText().trim();
    String olhos = campoOlhos.getText().trim();
    String roupas = campoRoupas.getText().trim();
    String local = campoLocal.getText().trim();

    List<DisappearanceListItem> lista = dao.pesquisar(
            nome, idade, altura, cabelo, olhos, roupas, local
    );

    modelo.setRowCount(0);
```