

UNIVERSIDADE PAULISTA

MATEUS PINHEIRO MENDES

GUILHERME REIS

KENNEDY ANDERSON

ATIVIDADES PRÁTICAS SUPERVISIONADAS

SISTEMA PARA CADASTRO DE INFORMAÇÕES

GOIÂNIA – GO

2019

MATEUS PINHEIRO MENDES – N281BA-4

GUILHERME REIS – N27162-0

KENNEDY ANDERSON – N316FA-1

ATIVIDADES PRÁTICAS SUPERVISIONADAS

SISTEMA PARA CADASTRO DE INFORMAÇÕES

Atividades Práticas Supervisionadas (APS) é apresentada para a avaliação do 3º Semestre do curso de Ciência da Computação da Universidade Paulista (UNIP), sobre as diretrizes do mestre do semestre.

Mestre: Prof. Jeferson Silva.

GOIÂNIA-GO

2019

Sumário

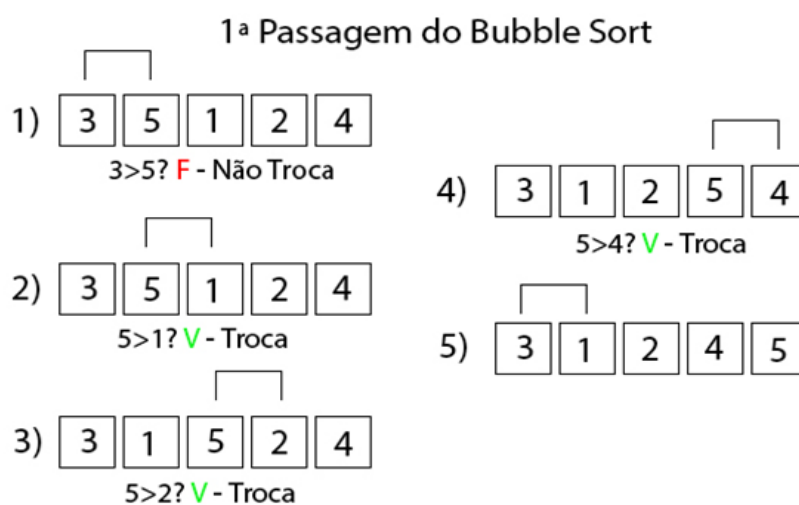
1.Introdução.....	Pg.4-6
2.Referencial Teórico.....	Pg.7-9
2.1 Troca de posições (SWAP).....	Pg.9
3.Desenvolvimento.....	Pg.10-23
3.1. A História do Banco de Dados.....	Pg.10-12
3.2 Banco de Dados Escolhido.....	Pg12-14
3.3 Criação do Banco de Dados.....	Pg.14-16
3.4 Programação Orientada a Objetos (POO).....	Pg.17-21
3.5 Criação do Sistema de Cadastro.....	Pg.21-23
4.Resultados e Discussões.....	Pg24-29
5.Considerações Finais.....	Pg.30
6.Referências Bibliográficas.....	Pg.31-32
7.Código Fonte.....	Pg.33-105

1. INTRODUÇÃO

Os métodos de ordenação são utilizados na parte organizacional dos dados na área da computação, eles podem ser classificados como os de ordenação interna, e ordenação externa. Os que possuem ordenação interna são os que possibilitam que os dados sejam acessados instantaneamente, e ficam todos armazenados na memória principal. Já nos de ordenação externa, os dados só podem ser acessados por meio de grandes blocos, ou de forma sequencial, além de serem muito grandes para serem armazenados na memória principal.

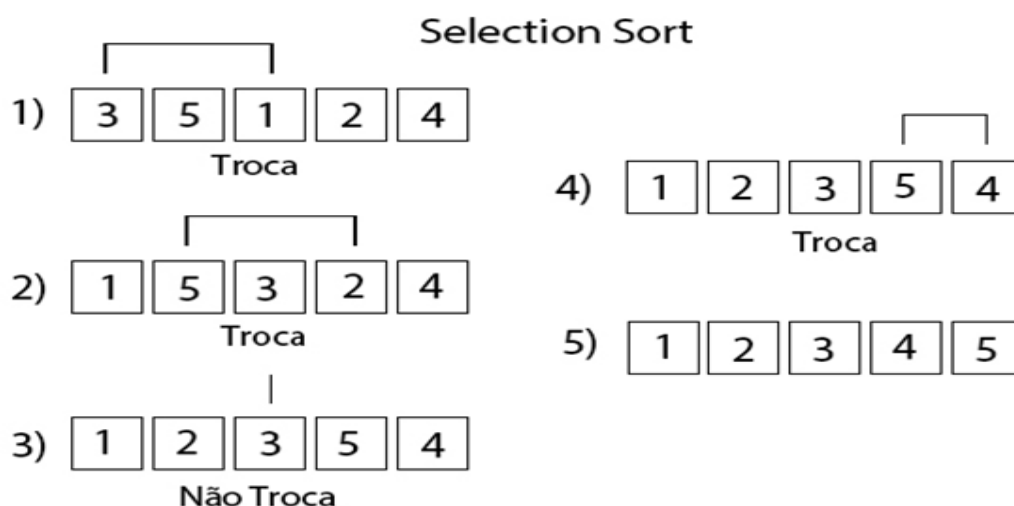
Bubble Sort

Está entre um dos mais simples dos algoritmos de ordenação, e também um dos menos eficientes. Ele compara determinado elemento de uma posição x , por outro de posição $x + 1$, isso significa que um elemento de posição 4 é comparado ao de posição 5, e se o da posição 4 for maior, eles trocam de posição, e assim por diante. Como o vetor precisa ser percorrido quantas vezes preciso for, ele pode se mostra ineficiente para listas de tamanhos superiores. Caso a lista em questão já esteja organizada de forma crescente, não será necessária nenhuma movimentação, porém, muito tempo e muitas comparações serão feitos, o que o torna ineficiente.



Selection Sort

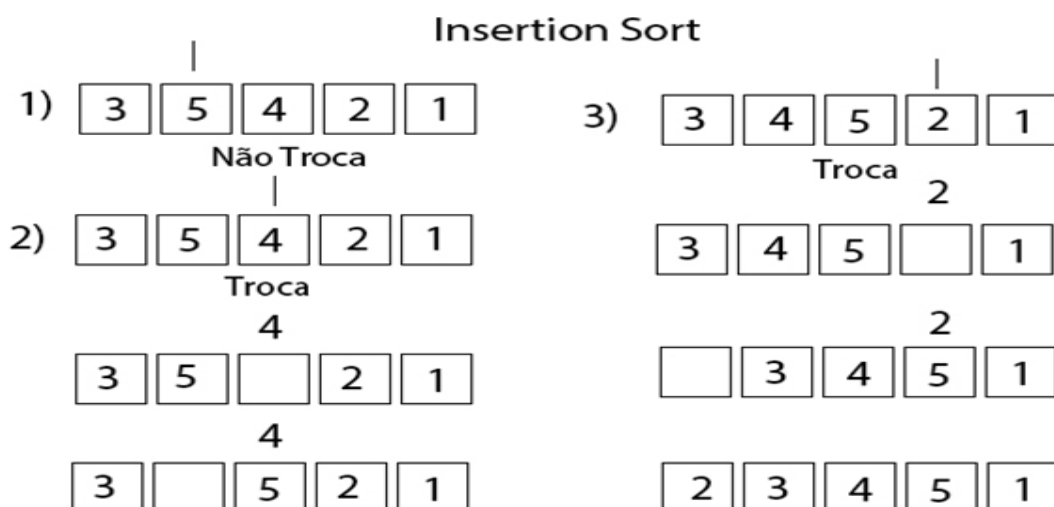
Seu método de funcionamento consiste em passar o dado de menor ou maior (varia de acordo com a ordem a ser seguida) valor do vetor para a primeira posição, e na segunda posição ocupa o segundo dados de menor valor, e assim segue até terminar os dados. Ele realiza essa operação escolhendo um dado que parte do primeiro, e comparando ele sempre ao da direita, caso encontre um menor do que ele, esse dado passa a ocupar sua posição, e passa a ser o novo número escolhido para ser comparado aos demais, e caso não tenha nenhum número menor do que o usado para comparação, ele passa para o primeiro lugar, e a comparação é feita com o número a sua direita, e assim sucessivamente até o termino de ordenação da lista.



Insertion Sort

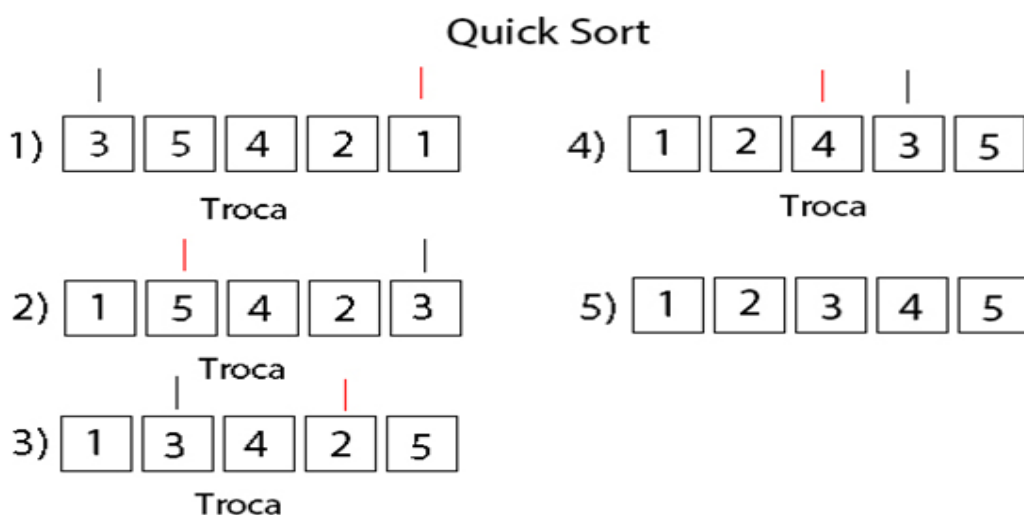
Mais comumente usados em pequenas listas, se mostra eficaz e simples, ele faz a leitura do vetor da esquerda para a direita, e conforme vai

lendo os dados, os organiza à esquerda dos elementos ordenados.



Quick Sort

No quesito de ordenação por comparação, este se mostra como o mais eficiente, principalmente nas listas que não possuem nenhum tipo de ordenação pois gasta bem menos tempo do que os demais, ele determina certo dado escolhido como pivô, e começa a organizar a lista, de forma que todos os dados que venham antes deste pivô, sejam menores que ele, e todos os que venham depois, sejam maiores. Quando ele termina de organizar os elementos, o escolhido como pivô estará em sua posição final, e todos os outros elementos terão passado por esta organização, e a lista estará ordenada.



2. Referencial Teórico

Um dos mais eficientes algoritmos de ordenação é o Quicksort, bastante utilizado por programadores profissionais, ele se destaca perante os outros, tendo como objetivo a ordenação dos dados contidos em uma lista, funcionando através da categoria de Exchange Sort, também conhecida como ordenação por troca. Utilizando o paradigma chamado de divide and conquer (dividir para conquistar), este paradigma se consiste em ramificar diversas vezes a entrada do algoritmo, transformando uma grande tarefa em várias tarefas menores.

Ele usa o método `partition()` para realizar a implementação, este método escolhe um dado que é chamado de pivô, depois organiza todos os dados de forma que eles fiquem posicionados à esquerda dele, o pivô pode ser escolhido de forma aleatória, vamos utilizar de exemplo a sequência `<34512>`, e escolheremos como pivô o número 3, e ao percorrer a lista, todos os números encontrados menores que ele, ficarão à sua esquerda, estabelecendo desta forma a sequência particionada. Este método não organizará a lista completa de uma única vez, porém, todos os dados estarão organizados em relação ao pivô escolhido, desta forma, basta repetir o processo, até que todos os elementos estejam ordenados.



Esta imagem representa como é o funcionamento do método citado, o `partition()`, e como ele realiza seu particionamento. No início do exemplo acima, é escolhido o último dado como pivô, e o retorno da execução deste método é a variável `i`. Como citado anteriormente, ele compara o dado escolhido com todos os outros, posicionando à sua esquerda todos os menores que ele, e ao final do loop, basta trocar o pivô e repetir até que se finalize a ordenação da lista. Desta forma, todos os elementos que estiverem à esquerda e todos à direita do pivô serão menores e maiores do que o pivô.

Exemplo:

```
/**
 * Método de particionamento que é o núcleo do algoritmo Quicksort.
 * É a implementação utilizando o paradigma dividir para conquistar
 */

int partition (const int start, const int end)
{
    int l = start;

    for (int j = start; j < end; j++) {
        /* Elemento atual menor ou igual ao pivô? */

        If (elements [j] <= elements[end]) {

            Swap (i++, j);

        }

    }

    Swap (i, end);

    Return l;
}
```


2.1 Troca de posições (SWAP)

O método swap utilizado no código acima para realizar a troca das posições dentro da sequência vem da biblioteca do C++ Vector.

```
/**
 * Método para fazer a troca de dados entre duas posições no vetor
 */
void swap (const int i, const int j) {
    int k = elements[i]; elements[i] = elements[j]; elements[j] = k;
}
```

Como citado anteriormente, particionar apenas uma vez, não é o suficiente para ordenar todos os elementos a esquerda e a direita do pivô, então o Quicksort utiliza o paradigma chamado de divide and conquer, que particiona o vetor várias vezes, até que a lista esteja ordenada.

```
/**
 * Método privado que implementa o Quicksort recursivamente */
void quicksort (const int start, const int end) {
    if (start >= end) return;
    int pivot = partition(start, end);
    quicksort(start, pivot - 1);
    quicksort(pivot + 1, end);
}
```

3. DESENVOLVIMENTO

3.1. A História do Banco de Dados

Algumas décadas atrás víamos que os dados das empresas eram armazenados em fichas de papel e essas fichas sendo organizadas em pastas dentro de gavetas. Além de ter que estruturar cada pasta dentro das gavetas o acesso a essas informações era bem limitada pois dependia da localidade de onde os arquivos físicos estavam.

Depois disso, os dados que eram armazenados de forma arcaica começaram a ser manipulados em softwares que eram bem simples. Esses programas apenas permitiam que o usuário realizasse operações de cadastro, alteração, exclusão e uma consulta de dados nos arquivos que estavam salvos digitalmente.

No decênio de 60 a IBM (*International Business Machines*) começou suas pesquisas para o aperfeiçoamento de softwares que eram destinados a facilitar o relacionamento entre os arquivos digitais, como um produto X pertencer a fornecedor Y. Depois de várias pesquisas surgiram alguns tipos de banco de dados como o hierárquico e rede.

3.1.2 Banco de Dados Hierárquico

O banco de dados hierárquico é baseado em um conjunto de registros que são interligados uns aos outros através de ligações. Um registro é um acervo de campos podendo ser de diferentes tipos e cada um tem um valor de dados. A ligação em si consiste em uma associação entre precisamente dois registros. O modelo de banco de dados hierárquico é parecido com o modelo de rede por conta de ser exercido também através de registros e ligações, porém o que difere os dois é na organização dos arquivos através de coleções de árvores que ocorre no hierárquico e no banco de dados em rede ocorre como grafos arbitrários.

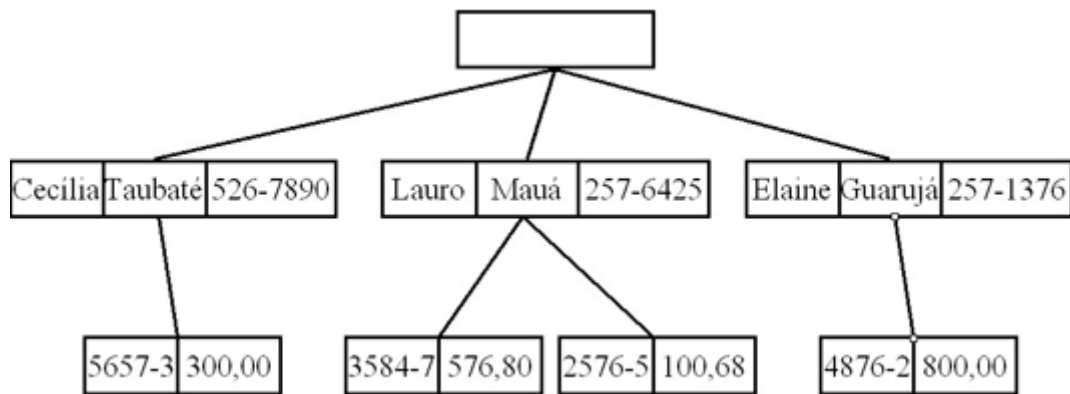


Figura 1 – Exemplo de Banco de Dados Hierárquico

3.1.3 Banco de Dados Em Redes

O modelo de banco de dados em redes pode ser considerado uma ampliação do modelo hierárquico. Onde o modelo de redes eliminou a estrutura de hierarquia em banco de dados e passou a ser permitido o mesmo registro envolver várias associações. Neste modelo os registros possuem uma ordenação em grafos arbitrários, que possuem uma só associação que é do tipo 1:X. Sendo assim, com dois relacionamentos 1:X você pode construir um relacionamento M:N.

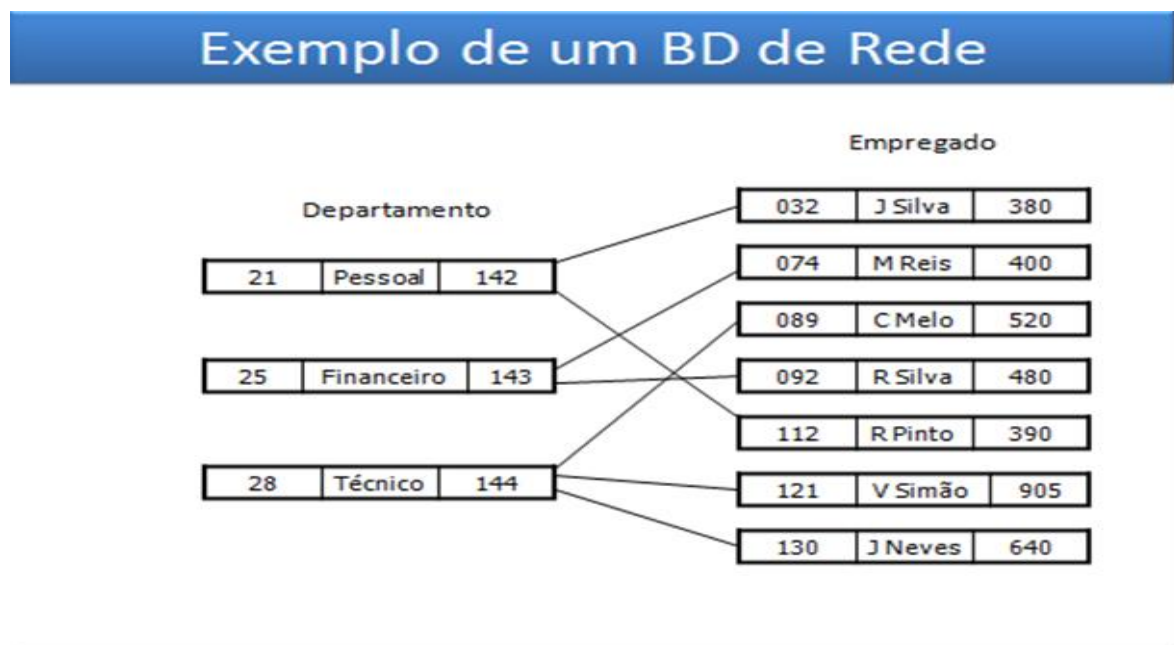


Figura 2 – Exemplo de Banco de Dados em Redes

3.1.4 Banco de Dados Relacional

O banco de dados relacional foi apresentado por *Edgar Frank “Ted” Codd* funcionário da IBM. No começo não foi tão utilizado, mas no final da década de 70 foi criado um sistema que tinha como sua base as ideias centrais de Codd, o “Sistema R”. Adicionado ao sistema foi desenvolvido a linguagem de consulta estruturada SQL (*Structured Query Language*) que até os dias atuais é a linguagem que domina os bancos de dados relacionais.

Depois de algum tempo, gigantes como a Oracle anunciaram seus próprios banco de dados como o Oracle 2 e a outra gigante IBM apresentou o SQL/DS, os dois são sistemas comerciais de banco de dados.

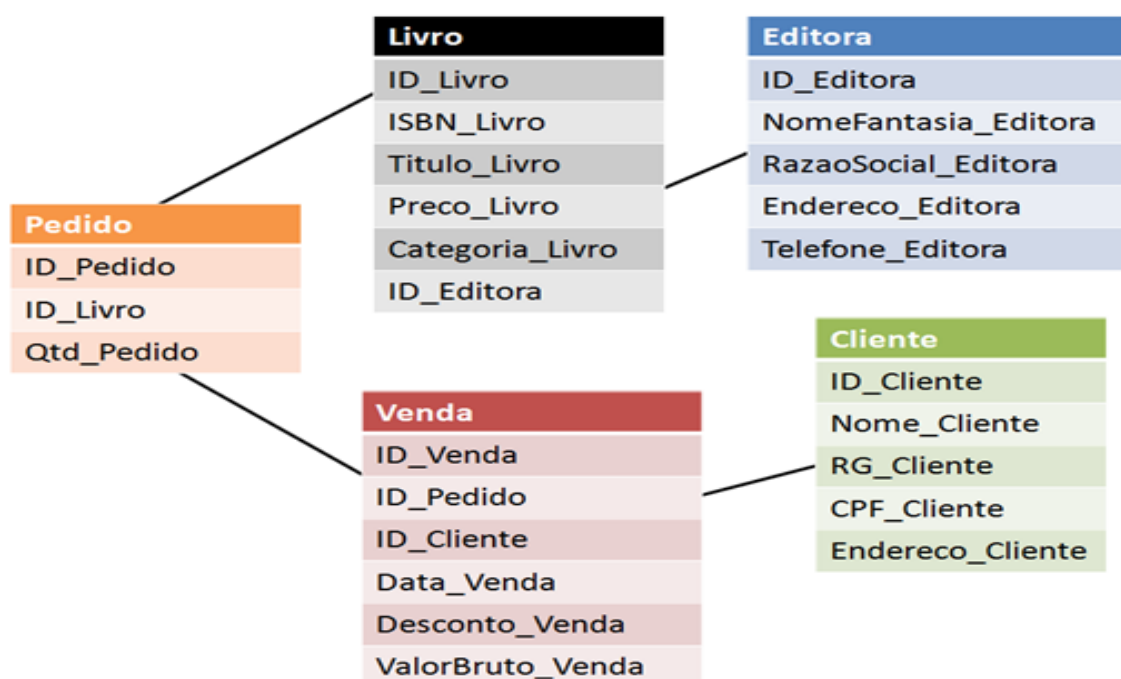


Figura 3 – Exemplo de Banco de Dados Relacional

3.2 Banco de Dados Escolhido

O banco de dados escolhido foi um banco de dados relacional, o MySQL. Ele é um SGBD (*Sistema de Gerenciamento de Banco de Dados*) que opera com a linguagem SQL como uma forma de interface. O MySQL foi criado na Suécia pela empresa MySQL AB, que no ano de 2008 foi comprada pela Sun Microsystems, por um valor recorde nesse setor. Logo depois no ano de 2009 a

Sun Microsystems foi vendida para Oracle que hoje é quem mantém e atualiza o MySQL.



Figura 4 - Logo MySQL

3.2.1 MySQL Workbench

Na utilização do MySQL é essencial que tenha um servidor e uma aplicação cliente. O papel do servidor manter os dados, ter que lidar com as requisições. Já o cliente utiliza a linguagem SQL para a comunicação com o servidor. Na instalação do MySQL a Oracle já oferece uma interface gráfica cliente gratuita que é chamada MySQL Workbench, estando disponível para sistemas Windows, macOS e Linux.

O Workbench possui uma interface de modelagem de base de dados e nela pode ser definida as entidades da database, os atributos e relacionamentos. Em qualquer banco de dados, deve ser definido ajustes importantes, como as chaves primárias e as chaves estrangeiras.

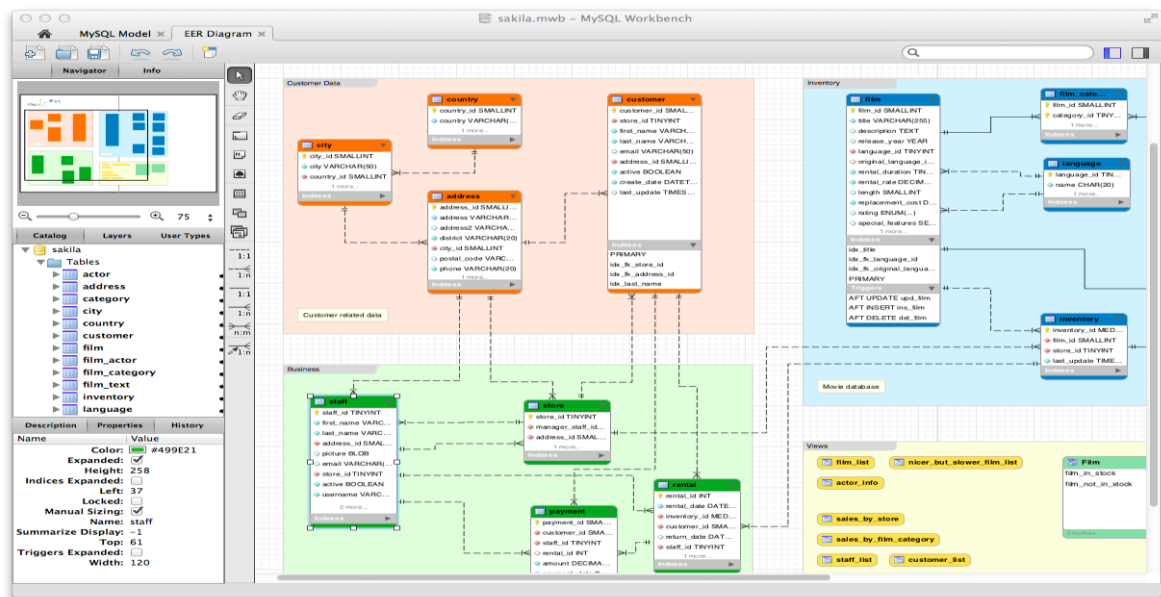


Figura 5 - Exemplo da interface do MySQL Workbench

3.3 Criação do Banco de Dados

3.3.1 Criando a Database

Nesta parte iniciamos a criação do banco de dados, primeiramente criando a database (schema), para assim criarmos o as tabelas e sucessivamente e colunas para ter campos específicos para cada parte do programa. Criamos um database chamado “dados_aps”

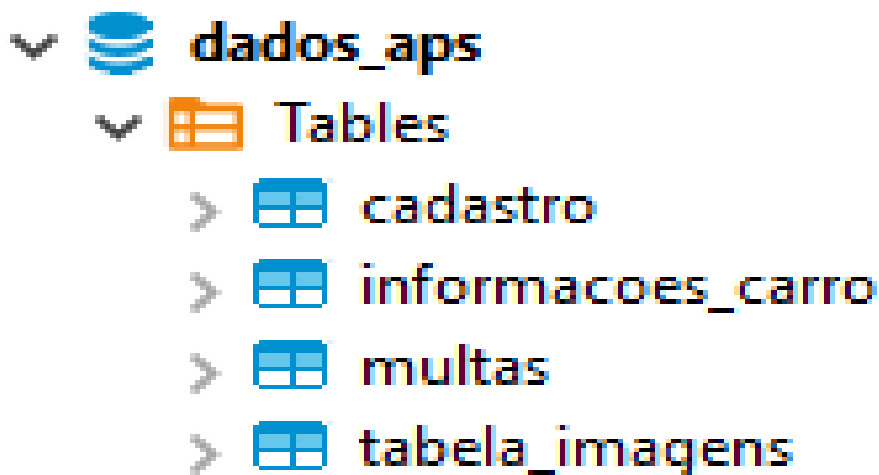


3.3.2 Criando as Tabelas

Aqui nós criamos cada tabela para ser específica para cada parte do trabalho e ficar bem dividido os dados que são enviados para a database. No trabalho a database foi dividida em quatro (4) tabelas que são elas:

1. cadastro
2. informacoes_carro
3. multas
4. tabela_imagens

Onde a tabela “cadastro” é restrita para as informações dos usuários que tenham uma conta no sistema. Já a tabela de “informações_carro” é designada para armazenar informações sobre veículos que são cadastrados no sistema. A tabela “multas” é onde é armazenados carros que contém multas ou infrações de trânsito. Por último, a tabela “tabela_imagens” ficará responsável por armazenar as imagens.








3.3.3 Criando as Colunas






Nessa parte do desenvolvimento criamos as colunas referentes a cada tabela do banco de dados. Cada tabela tem um número específico de colunas que tem a mesma finalidade das tabelas, organizar o armazenamento de dados. A tabela “cadastro” possui seis (6) colunas:









A tabela “informacoes_carro” possui oito (8) colunas:

- ▼  informacoes_carro
 - ▼  Columns
 - 123  ID_Carro (int)
 - ABC modelo (varchar(15))
 - ABC dono (varchar(255))
 - ABC cor (varchar(15))
 - 123 anoDoCarro (smallint)
 - ABC placa (varchar(7))
 - 123  ID_Imagem (int)
 - 123  ID_Multa (int)

A tabela “multas” também possui oito (8) colunas:

- ▼  multas
 - ▼  Columns
 - 123  ID_Multa (int)
 - 123 valorMulta (float)
 - ABC nome (varchar(255))
 - ABC cpf (varchar(12))
 - ABC tipoMulta (varchar(50))
 - ABC placaVeiculo (varchar(7))
 - 123  ID_Carro (int)
 - 123  ID_Imagem (int)

Já a tabela de imagens é a que possui menos colunas são quatro (4):

- ▼  tabela_imagens
 - ▼  Columns
 - 123  codigolmg (int)
 -  imagem (longblob)
 - 123  ID_Carro (int)
 - 123  ID_Multa (int)

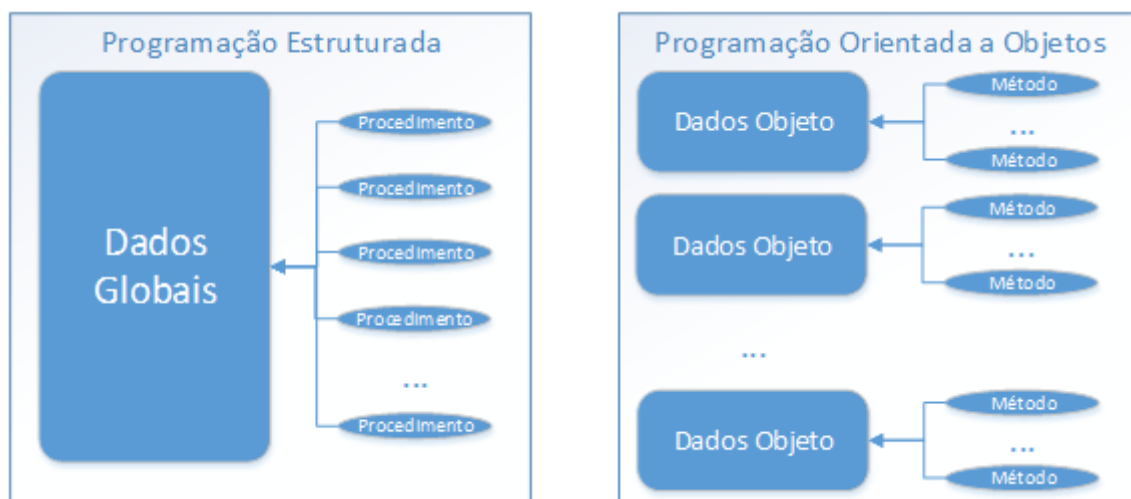
3.4 Programação Orientada a Objetos (POO)

Entre as várias linguagens de programação que estão em uso no mercado voltado ao desenvolvimento de software, discorreremos sobre o paradigma Orientação a Objetos, este que nos dias atuais é o mais utilizado. Esse favoritismo é devido ao padrão evoluído deste paradigma, principalmente no aspecto da segurança, e do reaproveitamento do código, aspectos fundamentais em qualquer desenvolvimento de software nos dias atuais.

A POO é utilizada por grandes linguagens, como no C# e no Java. Adiante, mostraremos as distinções entre a Programação Orientada a Objetos, e a Programação estruturada, bastante utilizada a alguns anos atrás, mais comumente pela linguagem c. A POO se mantém sobre 4 pilares principais.

3.4.1 Diferenças entre Programação Estruturada e Programação Orientada a Objetos

Em relação aos dados, a Programação Estruturada aplica globalmente as funções nas aplicações criadas, já na POO, as ações serão aplicados aos dados dos determinados objetos.



A principal linguagem utilizada na programação estruturada é a c, que é uma linguagem de baixo nível, já nos dias atuais não é utilizada em grandes projetos. Já nos sistemas embarcados, ou nos que a compreensão de hardware é necessária para um ótimo programa, ela é mais utilizada, devido ao seu baixo nível. Uma observação importante, é que a programação estruturada, possui um melhor desempenho do que a POO. Isso é devido a ela

ser um paradigma sequencial, em que as linhas de código são executadas uma após a outra, sem os desvios feitos na POO, outro fator importante é o auxílio do hardware, que é permitido pela programação estruturada, e resulta em um melhor desempenho.

Porém, como o desempenho destas aplicações não é o foco na maior parte das aplicações, devido às máquinas que estão a cada dia mais potentes em relação ao poder de processamento, a POO traz alguns benefícios que chamam mais atenção nas aplicações atuais, como sua capacidade de reaproveitamento do código, e pela representação de seu sistema ser muito mais parecida do que veríamos no mundo real, fatores que contribuíram muita para sua difusão no mercado.

3.4.2 Os 4 pilares da Programação Orientada a Objetos



3.4.2.1 Abstração

É o principal ponto dentro de toda linguagem POO. Ele é o responsável por definir o que o objeto real poderá fazer dentro do sistema. São 3 pontos a serem considerados durante a abstração.

A primeira parte é a identidade do objeto a ser criado, ela deve ser única, para não haver conflitos no sistema. A segunda, é referente as características do objeto, pois todo e qualquer objeto é definido por seus elementos, e na POO

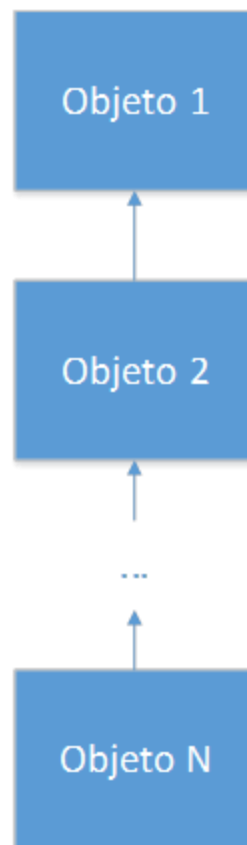
elas são chamadas de propriedades. E a terceira, são as ações que este objeto será capaz de realizar, também chamado de métodos.

3.4.2.2 Encapsulamento

Ele é o responsável por esconder as propriedades de um objeto, o que o torna o agente que traz proteção e segurança a aplicação, como se fosse uma caixa preta, sendo uma das principais técnicas deste paradigma. Grande parte das linguagens de POO o utilizam baseado em propriedades privadas, sendo diretamente relacionados à métodos especiais, os getters e setters, que são os responsáveis por retornarem e setarem valores das propriedades em questão, evitando assim que os atributos sejam acessados de forma direta pelo usuário, ação que adiciona proteção à aplicação.

3.4.2.3 Herança

A reutilização do código é um dos principais atrativos da POO, e grande parte disto se deve às heranças, pois elas reduzem as linhas de código, e otimizam a produção da aplicação. O objeto que estiver abaixo na hierarquia, irá herdar as características de todos os objetos que estiverem acima dele, a característica herdada do objeto logo acima é a herança direta, as demais são heranças indiretas.



A herança varia de uma linguagem para outra, no C++ por exemplo, existe a herança múltipla, o que permite que o objeto receba características de mais de um “Ancestral” de forma simultânea e direta. Porém, tal prática não foi difundida nas linguagens atuais, por apresentar problemas, elas optaram por utilizar outras formas de criar algo como a herança múltipla.

3.4.2.4 Polimorfismo

No meio natural, vemos animais capazes de se adaptarem conforme a situação em que se encontram, e é daí que vem o polimorfismo da POO. Sua função é alterar o funcionamento interno de uma ação ou método, que tenha sido herdada de um objeto pai. Como seu funcionamento é diretamente dependente da herança, é necessário o domínio dos dois. Assim como na herança, as linguagens têm suas diferenças na hora de implementar o polimorfismo, o C# utiliza métodos virtuais que possibilitam o reimplante nas classes herdeiras.

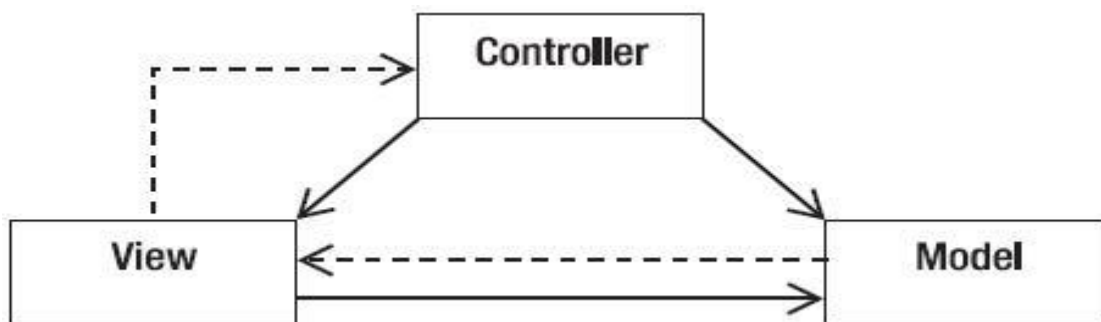
Esses são os 4 pilares fundamentais no entendimento de qualquer linguagem de POO, mesmo que cada uma os implemente de uma forma, sua essência será

sempre a mesma. Apenas as heranças apresentam mudanças mais expressivas.

3.5 Criação do Sistema de Cadastro

3.5.1 O que foi usado?

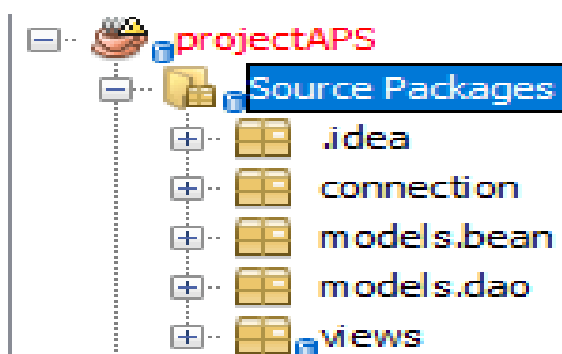
Na criação do sistema foi utilizado o Padrão MVC, com ele foi possível fazer a divisão do projeto em packages bem definidos. Ele é dividido entre, o Model, o Controller e a View cada um tem funções bem definidas e só executam elas, nada além que isso. Ele é baseado nisso:



No projeto também foi usado o DAO que por si só é um padrão de projetos também, ele foi usado junto ao MVC. E basicamente as regras desse padrão são:

- provê uma interface que omite o ingresso aos dados;
- lê e grava a partir do limiar dos dados (banco de dados, arquivo, memória, etc.); e
 - encapsula o ingresso aos dados, de modo que as demais classes não precisam ter o conhecimento sobre isso.

Sendo assim, veja como ficou o projeto:

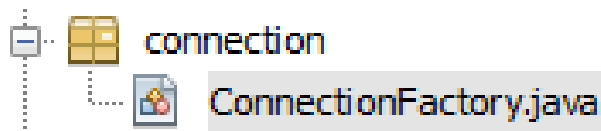


3.5.2 Falando sobre os packages

Bom nesse tópico iremos falar um pouco sobre cada package e o que contém em cada um deles.

3.5.2.1 Connection

O pacote connection foi criado com objetivo de ter apenas uma classe dentro dele, e essa classe consiga fazer a conexão com o banco de dados, nuvem ou qualquer outro serviço que esteja utilizando para o armazenamento de dados. Neste caso como foi mostrado no começo do desenvolvimento foi utilizado o banco de dados MySQL, ou seja, o pacote tem a função de fazer a conexão com a database do MySQL.

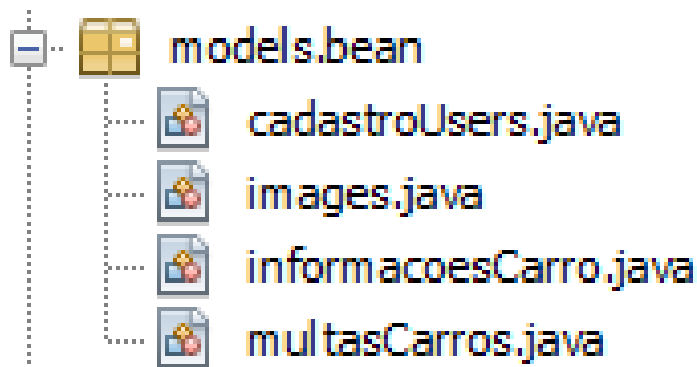


3.5.2.2 Models

O pacote Models tem um diferencial que, ele é subdividido em dois outros pacotes o Bean e o DAO. Então vamos começar falando sobre o Models.Bean.

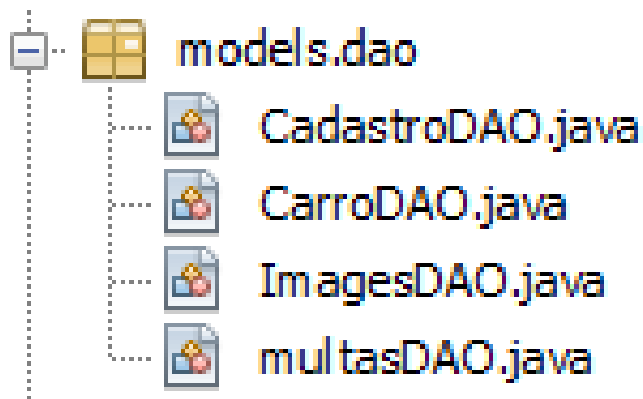
3.5.2.2.1 Models Bean

No Models Bean ele tem como principal objetivo dividir cada classe com seus respectivos atributos e métodos, ou seja, ele faz um encapsulamento dos atributos já que só ele é responsável por isso, nesse projeto temos quatro (4) classes, cada uma com seus atributos e métodos.



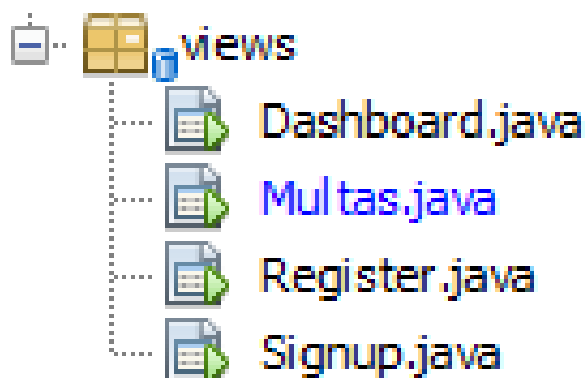
3.5.2.2.2 Models DAO

No Models DAO seu objetivo com auxílio do Models Bean é passar os dados que o usuário inseriu para o banco de dados ou qualquer outro sistema que esteja armazenando dados. Nesse caso as classes de Models DAO tem junto a sintaxe do Java a linguagem SQL para passar as informações como o cadastro de um usuário, atualização de alguma informação, deletar entre outros vários métodos.



3.5.2.2 Views

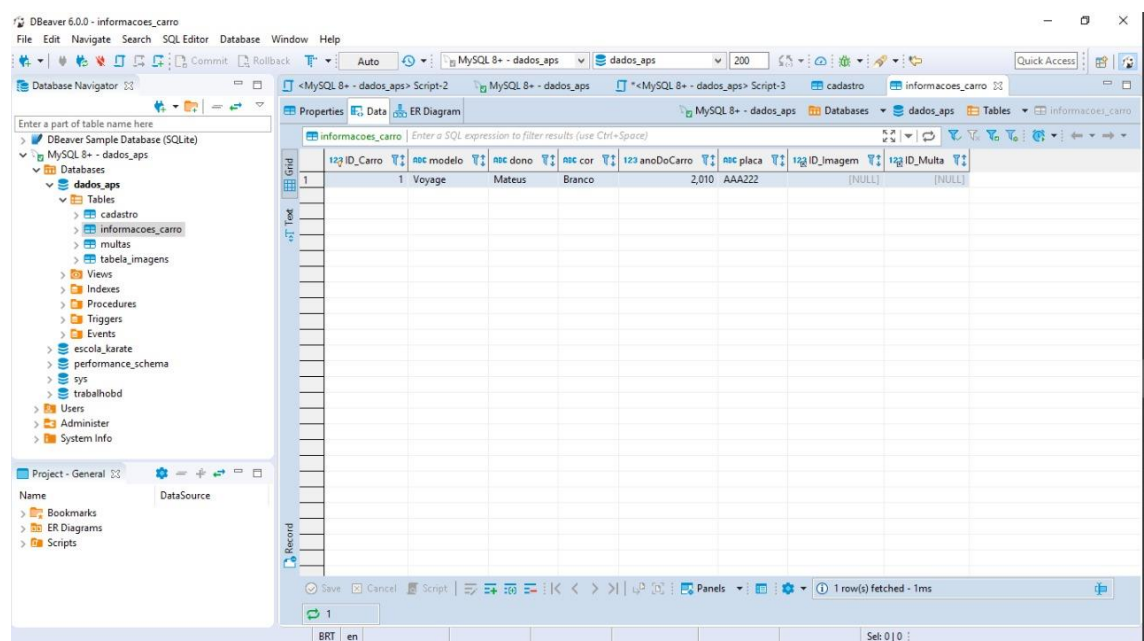
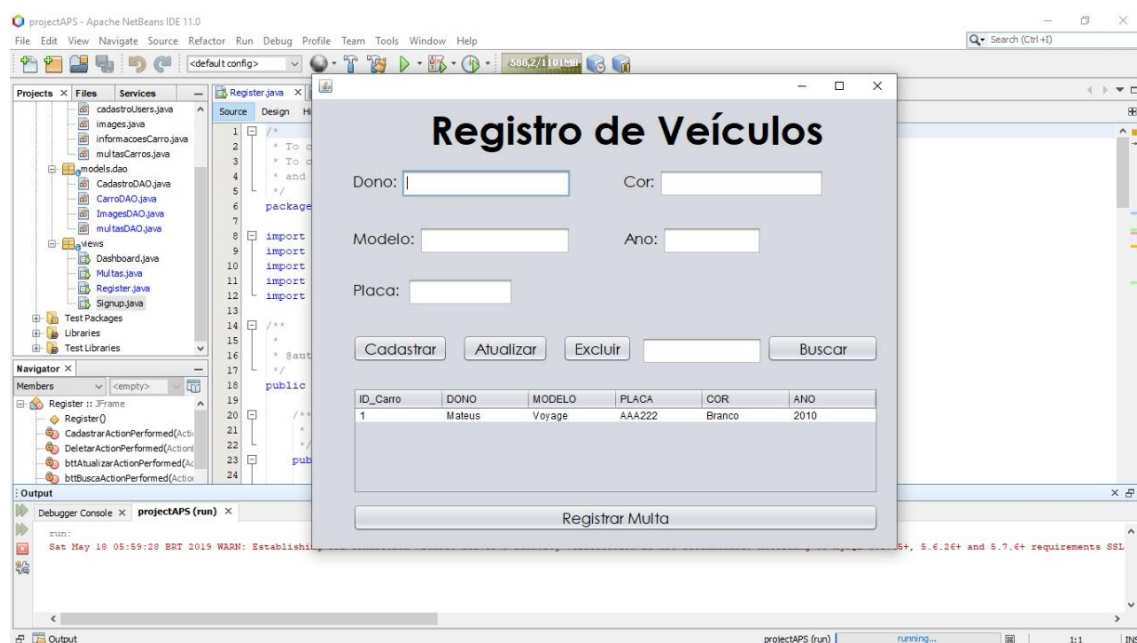
No package de Views é onde ficam todas as interfaces gráficas, é um pacote essencial pois é nele que o usuário entra em contato com o sistema e assim pode colocar suas informações para armazenar no banco de dados. No projeto são quatro (4) interfaces gráficas, cada uma com sua respectiva função e interação com o usuário para assim não haver algum tipo de confusão de estar na interface errada.



4. Resultados e Discussões

Realizando os testes com o programa em execução:

Registrando os veículos:

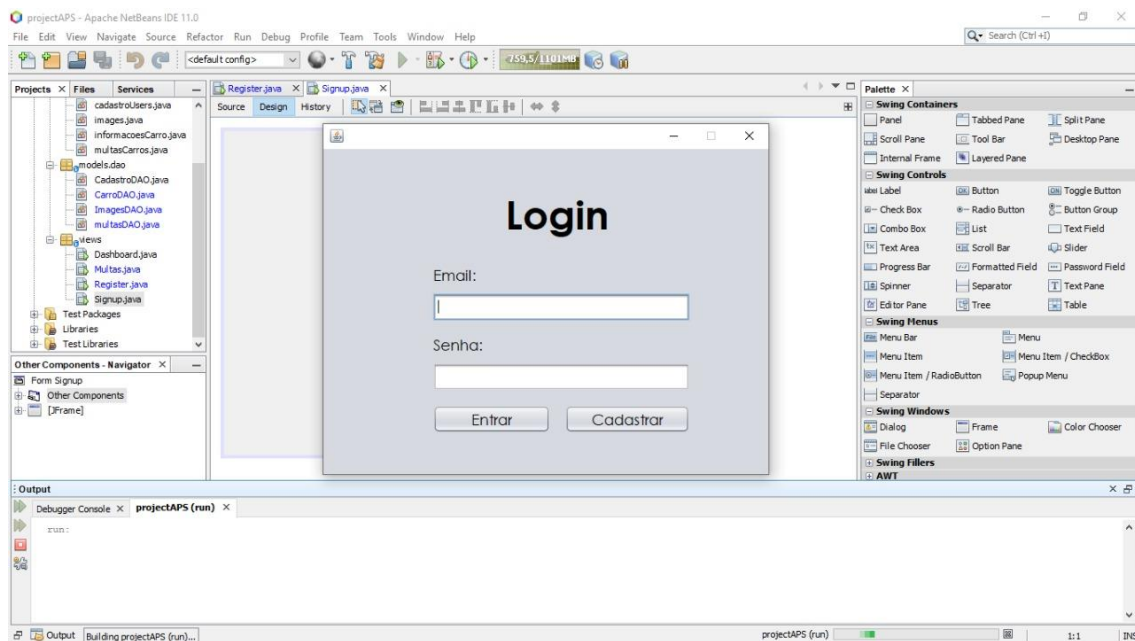


Nessa janela, o usuário irá informar o nome do dono do veículo, especificará a cor do mesmo, o modelo, o ano e a placa.

A janela conta com 5 botões, que são os de cadastrar os dados de veículo informado acima no banco de dados, o de atualizar os dados sobre algum veículo previamente cadastrado, o de excluir algum veículo cadastrado, e o campo de busca.

Por final, uma tabela mostrando os veículos cadastrados no banco de dados, e o botão de registrar multa, que avançará para a próxima janela.

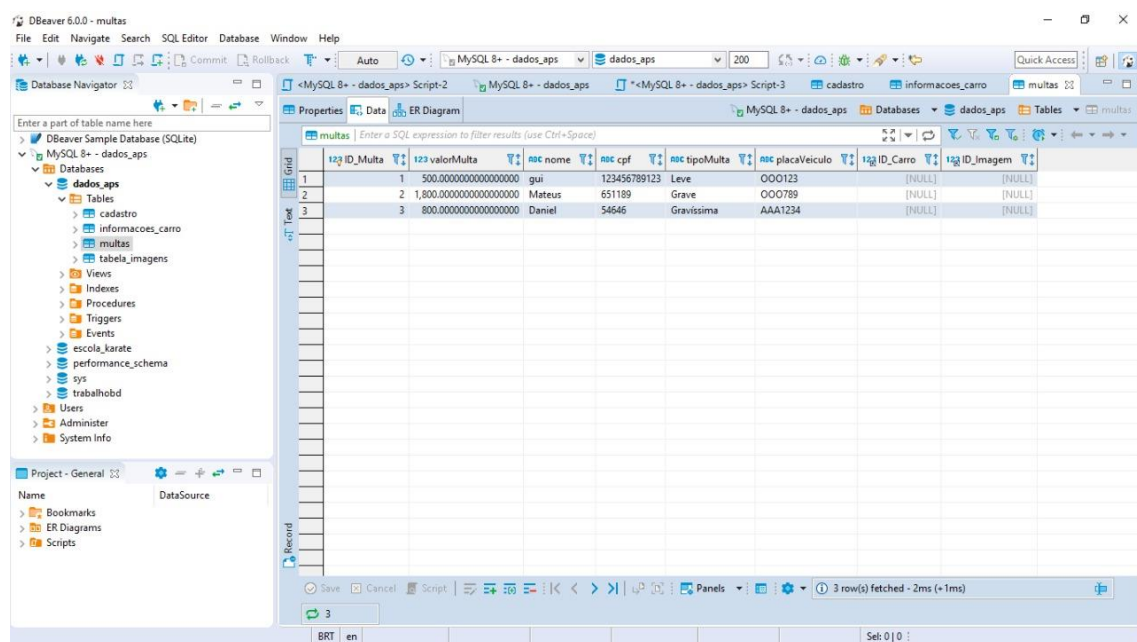
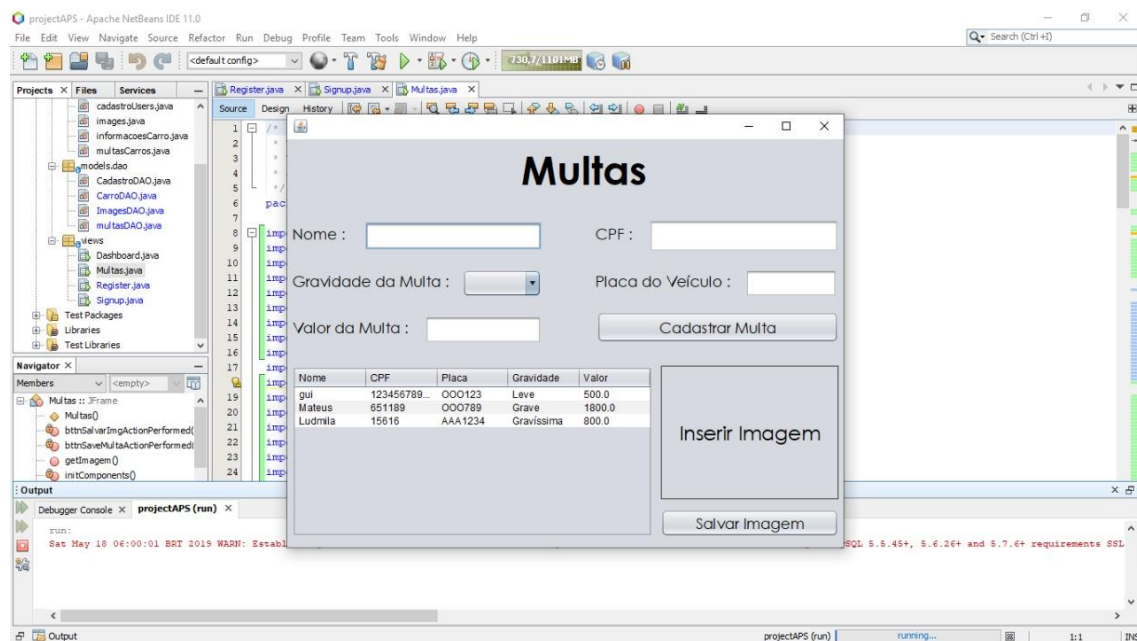
Parte responsável pelo login dos clientes:

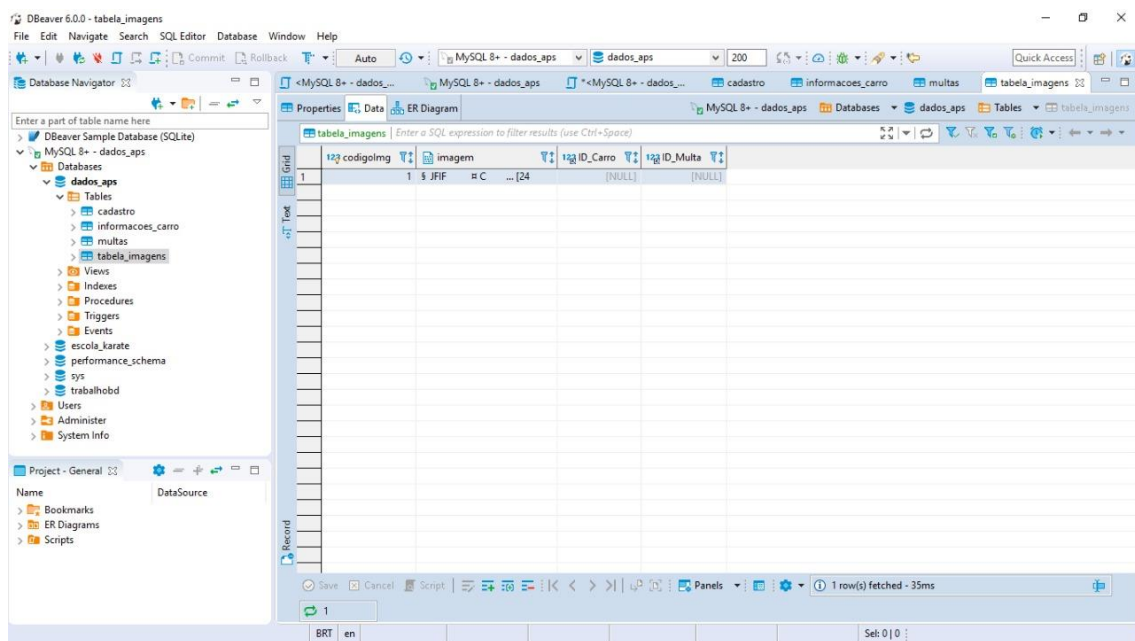


Nesta janela o usuário irá informar seu email e senha, e ao clicar no botão de entrar, irá acionar uma verificação, para saber se os dados informados estão cadastrados no banco de dados, e se o usuário já possui uma conta previamente cadastrada.

Caso o usuário não tenha uma conta, ele poderá clicar no botão cadastrar, que o levará para a janela de cadastro

Parte que possibilita o cadastro das multas:



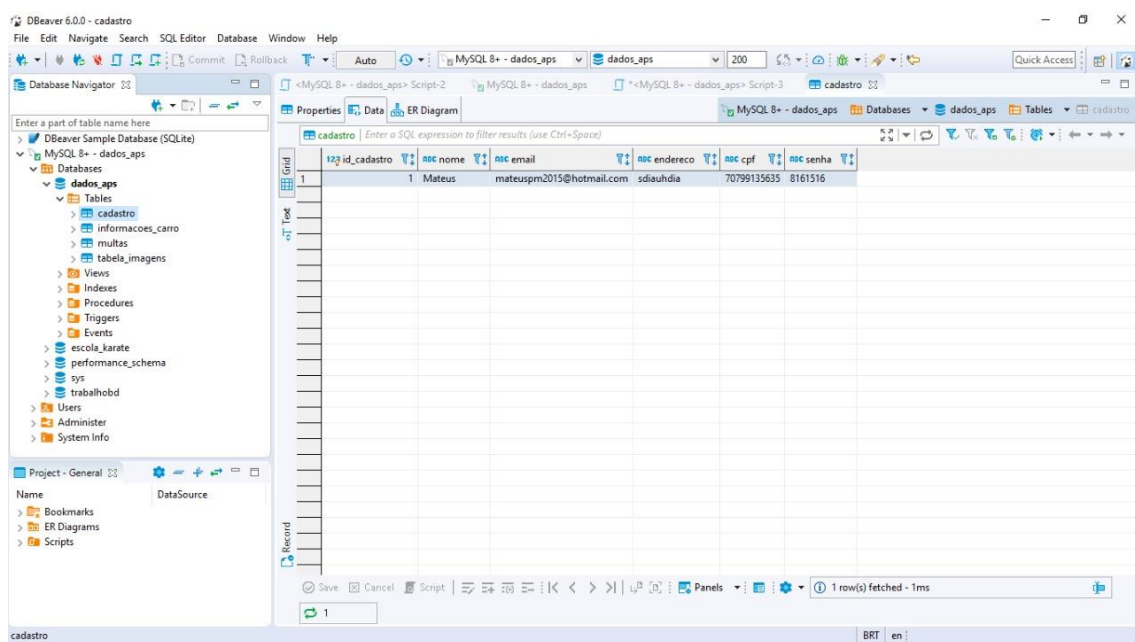
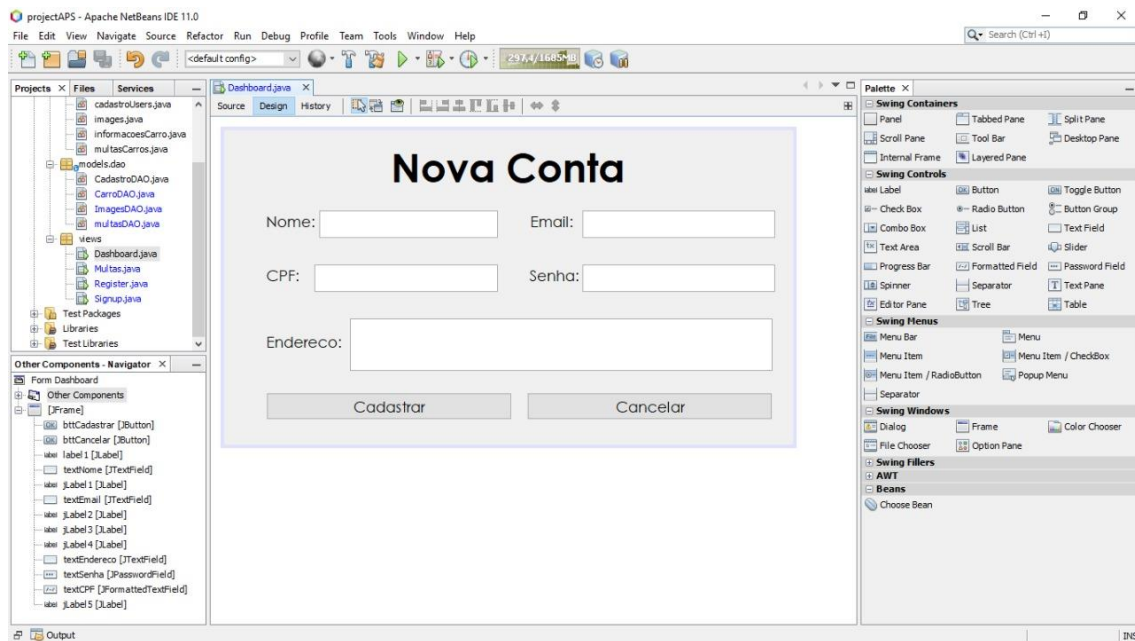


Nesta janela, o cliente irá preencher as informações sobre a multa em questão. Ela possibilita a informação do nome do infrator, o seu CPF, qual a gravidade da multa e a placa do veículo pra qual a multa se destina.

Conta também com os botões de cadastrar a multa no banco de dados e o de salvar a imagem da placa do veículo para o qual a multa foi emitida.

Possui uma tabela listando todas as multas já cadastrados no banco de dados, e o campo para o usuário inserir a imagem da placa do veículo, com uma tabela do banco de dados dedicada para ela.

Janela de criação de novas contas:



Aqui é onde o usuário pode abrir uma nova conta, realizando seu cadastro, tendo os campos para receber o seu nome, email, CPF, senha e endereço.

O botão de cadastrar irá enviar todas as informações para o banco de dados, criando a nova conta do usuário, e o cancelar irá fechar a janela.

5.Considerações Finais

Utilizando o paradigma de programação orientada a objetos, tivemos como objetivo a criação de um programa capaz de receber uma imagem da placa de um carro em um campo do tipo binário (BLOB) no banco de dados, previamente cadastrada, e a partir dela, detectar se há alguma infração de trânsito cometida por este veículo no banco de dados, utilizando a linguagem Java.

Ao decorrer do desenvolvimento do trabalho, viemos a entrar em contato com diversas novas tecnologias, métodos e paradigmas, possibilitando o aprendizado de novas técnicas, e algumas aplicadas até mesmo neste trabalho em questão. Vários algoritmos de ordenação foram vistos e levados em consideração, levando o grupo a chegar à conclusão de que o melhor a ser aplicado seria o Quicksort.

O paradigma de linguagem como havia sido previamente estabelecido foi o Orientado a Objetos, e a linguagem escolhida para levar o programa até a sua fase final foi o Java. Existiram testes e rascunhos de projetos em C#, mas sendo abandonados após reuniões e discussões feitas pelo grupo, devido as inviabilidades apresentadas pela linguagem, como sua incapacidade de declarar as exceções que uma rotina pode apresentar.

6.Referências Bibliográficas

<https://www.devmedia.com.br/os-4-pilares-da-programacao-orientada-a-objetos/9264>

[21:28, 5/19/2019] Mateus: <https://www.devmedia.com.br/principais-conceitos-da-programacao-orientada-a-objetos/32285/>

<https://www.devmedia.com.br/encapsulamento-polimorfismo-heranca-em-java/12991>

https://netbeans.org/kb/docs/java/quickstart-gui_pt_BR.html

<http://www.caelum.com.br/apostila-java-testes-xml-design-patterns/interfaces-graficas-com-swing/>

<https://stackoverflow.com/>

Livro - Use a Cabeça Java Sierra,Kathy - Alta Books

<https://www.devmedia.com.br/criando-uma-conexao-java-mysql-server/16753>

<https://dev.mysql.com/doc/refman/8.0/en/>

<http://www.webcodefree.com.br/blog/?p=870>

<https://www.devmedia.com.br/guia/guia-completo-de-sql/38314>

<https://www.devmedia.com.br/os-4-pilares-da-programacao-orientada-a-objetos/9264>

<https://becode.com.br/programacao-orientada-a-objetos-poo/>

<https://medium.com/@TDamiao/16-conceitos-poo-programa%C3%A7%C3%A3o-orientada-a-objeto-6cdc72ac3ee2>

7.Código Fonte

O código fonte será apresentado por partes seguindo cada classe de cada package.

Package Connection

Classe ConnectionFactory.java

```
package connection;

// importações de bibliotecas
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;

// fazendo a classe de conexão
public class ConnectionFactory {
    // declarando os logins do banco de dados
    private static final String DRIVER = "com.mysql.jdbc.Driver";
    private static final String URL = "jdbc:mysql://localhost:3308/dados_aps";
    private static final String USER = "root";
    private static final String PASS = "";

    // método para abrir uma conexão
    public static Connection getConnection() {
        try {
            Class.forName(DRIVER);

            return DriverManager.getConnection(URL, USER, PASS);
        } catch (ClassNotFoundException | SQLException ex) {
            throw new RuntimeException("Connection error: " ,ex);
        }
    }
}
```

```

// primeiro método estático para tratar uma exceção no fechamento
public static void closeConnection(Connection connection) {
    try {
        if(connection != null) {
            connection.close();
        }
    } catch (SQLException ex) {

        Logger.getLogger(ConnectionFactory.class.getName()).log(Level.SEVERE,
        null, ex);
    }
}

// segundo método estático para tratar uma exceção no fechamento
public static void closeConnection(Connection connection,
PreparedStatement stmt) {

    closeConnection(connection);

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

        Logger.getLogger(ConnectionFactory.class.getName()).log(Level.SEVERE,
        null, ex);
    }
}

// terceiro método estático para tratar uma exceção no fechamento
public static void closeConnection(Connection connection,
PreparedStatement stmt, ResultSet rs) {

    closeConnection(connection, stmt);

```

```

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

Logger.getLogger(ConnectionFactory.class.getName()).log(Level.SEVERE,
null, ex);
    }
}
}

```

Package Models.Bean

Classe cadastroUsers.java

```

package models.bean;

public class cadastroUsers {
    private int id_cadastro;
    private String nome;
    private String email;
    private String endereco;
    private String cpf;
    private String senha;

    public int getId_cadastro() {
        return id_cadastro;
    }

    public void setId_cadastro(int id_cadastro) {
        this.id_cadastro = id_cadastro;
    }

    public String getNome() {
        return nome;
    }
}

```

```
public void setNome(String nome) {  
    this.nome = nome;  
}
```

```
public String getEmail() {  
    return email;  
}
```

```
public void setEmail(String email) {  
    this.email = email;  
}
```

```
public String getEndereco() {  
    return endereco;  
}
```

```
public void setEndereco(String endereco) {  
    this.endereco = endereco;  
}
```

```
public String getCpf() {  
    return cpf;  
}
```

```
public void setCpf(String cpf) {  
    this.cpf = cpf;  
}
```

```
public String getSenha() {  
    return senha;  
}
```

```
public void setSenha(String senha) {  
    this.senha = senha;  
}
```

```
}
```

Package Models.Bean

Classe informacoesCarro.java

```
package models.bean;
```

```
public class informacoesCarro {  
    private String modelo;  
    private String dono;  
    private String cor;  
    private int anoDoCarro;  
    private String placa;  
    private int ID_Carro;  
  
    public String getModelo() {  
        return modelo;  
    }  
  
    public void setModelo(String modelo) {  
        this.modelo = modelo;  
    }  
  
    public String getDono() {  
        return dono;  
    }  
  
    public void setDono(String dono) {  
        this.dono = dono;  
    }  
  
    public String getCor() {  
        return cor;  
    }  
  
    public void setCor(String cor) {
```

```
        this.cor = cor;
    }

    public int getAnoDoCarro() {
        return anoDoCarro;
    }

    public void setAnoDoCarro(int anoDoCarro) {
        this.anoDoCarro = anoDoCarro;
    }

    public String getPlaca() {
        return placa;
    }

    public void setPlaca(String placa) {
        this.placa = placa;
    }

    public int getID_Carro() {
        return ID_Carro;
    }

    public void setID_Carro(int ID_Carro) {
        this.ID_Carro = ID_Carro;
    }
}
```

Package Models.Bean

Classe images.java

```
package models.bean;

public class images {
    private byte[] imagem;
```

```

private int codigoImg;

public byte[] getImagem() {
    return imagem;
}

public void setImagem(byte[] imagem) {
    this.imagem = imagem;
}

public int getCodigoImg() {
    return codigoImg;
}

public void setCodigoImg(int ID_Imagem) {
    this.codigoImg = ID_Imagem;
}
}

```

Package Models.Bean

Classe multasCarros.java

```
package models.bean;
```

```

public class multasCarros {
    private int ID_Multa;
    private float valorMulta;
    private String nome;
    private String cpf;
    private String tipoMulta;
    private String placaVeiculo;

    public int getID_Multa() {
        return ID_Multa;
    }
}

```

```
public void setID_Multa(int ID_Multa) {  
    this.ID_Multa = ID_Multa;  
}  
  
public float getValorMulta() {  
    return valorMulta;  
}  
  
public void setValorMulta(float valorMulta) {  
    this.valorMulta = valorMulta;  
}  
  
public String getNome() {  
    return nome;  
}  
  
public void setNome(String nome) {  
    this.nome = nome;  
}  
  
public String getCpf() {  
    return cpf;  
}  
  
public void setCpf(String cpf) {  
    this.cpf = cpf;  
}  
  
public String getTipoMulta() {  
    return tipoMulta;  
}  
  
public void setTipoMulta(String tipoMulta) {  
    this.tipoMulta = tipoMulta;  
}
```



```

public String getPlacaVeiculo() {
    return placaVeiculo;
}

public void setPlacaVeiculo(String placaVeiculo) {
    this.placaVeiculo = placaVeiculo;
}
}

```

Package Models.dao

Classe cadastroDAO.java

```

package models.dao;

import connection.ConnectionFactory;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JOptionPane;
import models.bean.cadastroUsers;

public class CadastroDAO {

    public void create(cadastroUsers registerUser) {
        Connection connect = ConnectionFactory.getConnection();
        PreparedStatement stmt = null;

        try {
            stmt = connect.prepareStatement("INSERT INTO cadastro (nome, email, endereco, cpf, senha) VALUES (?, ?, ?, ?, ?)");

            stmt.setString(1, registerUser.getNome());
            stmt.setString(2, registerUser.getEmail());

```

```

        stmt.setString(3, registerUser.getEndereco());
        stmt.setString(4, registerUser.getCpf());
        stmt.setString(5, registerUser.getSenha());

        stmt.executeUpdate();

        JOptionPane.showMessageDialog(null, "Usuário cadastro com
sucesso!");
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Erro ao cadastrar usuário!");
    } finally {
        ConnectionFactory.closeConnection(connect, stmt);
    }
}

// metodo para checar se existe usuario e fazer o login
public boolean checkUsers(String email, String senha) {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;
    ResultSet rs = null;
    boolean check = false;

    try {
        stmt = connect.prepareStatement("SELECT * FROM cadastro WHERE
email = ? and senha = ?");
        stmt.setString(1, email);
        stmt.setString(2, senha);

        rs = stmt.executeQuery();

        if (rs.next()) {
            check = true;
        }
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Usuario ou senha incorretos");
    } finally {

```

```

        ConnectionFactory.closeConnection(connect, stmt, rs);
    }

    return check;
}
}

```

Package Models.dao

Classe carroDAO.java

```

package models.dao;

import connection.ConnectionFactory;
import java.util.List;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import javax.swing.JOptionPane;
import models.bean.informacoesCarro;

// classe principal do programa
public class CarroDAO {
    // metodo que cadastrada um novo carro no banco de dados
    public void create (informacoesCarro informationCars) {
        Connection connect = ConnectionFactory.getConnection();
        PreparedStatement stmt = null;

        try {
            stmt = connect.prepareStatement("INSERT INTO informacoes_carro
(modelo, dono, cor, anoDoCarro, placa)"
            + "VALUES(?, ?, ?, ?, ?)");

            stmt.setString(1, informationCars.getModelo());
            stmt.setString(2, informationCars.getDono());

```

```

        stmt.setString(3, informationCars.getCor());
        stmt.setInt(4, informationCars.getAnoDoCarro());
        stmt.setString(5, informationCars.getPlaca());

        stmt.executeUpdate();

        JOptionPane.showMessageDialog(null, "Informações salvas com
sucesso!");
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Desculpe, houve um erro ao
salvar as informações!");
    } finally {
        ConnectionFactory.closeConnection(connect, stmt);
    }
}

// metodo para retornar uma array list com os carros ja cadastrados dentro
da tabela
public List<informacoesCarro> read() {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;
    ResultSet rs = null;

    List<informacoesCarro> listaInformacoes = new ArrayList<>();

    try {
        stmt = connect.prepareStatement("SELECT * FROM
informacoes_carro");
        rs = stmt.executeQuery();

        while (rs.next()) {
            informacoesCarro informations = new informacoesCarro();

            informations.setID_Carro(rs.getInt("ID_Carro"));
            informations.setDono(rs.getString("dono"));
            informations.setAnoDoCarro(rs.getInt("anoDoCarro"));

```

```

        informations.setCor(rs.getString("cor"));
        informations.setModelo(rs.getString("modelo"));
        informations.setPlaca(rs.getString("placa"));

        listaInformacoes.add(informations);
    }
} catch (SQLException ex) {
    JOptionPane.showMessageDialog(null, "Erro ao mostrar a lista!");
} finally {
    ConnectionFactory.closeConnection(connect, stmt, rs);
}

return listaInformacoes;
}

// metodo para buscar o carro dentro de uma array list com os carros ja
// cadastrados dentro da tabela
public List<informacoesCarro> buscarCarro(String placa) {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;
    ResultSet rs = null;

    List<informacoesCarro> listaInformacoes = new ArrayList<>();

    try {
        stmt = connect.prepareStatement("SELECT * FROM informacoes_carro
        WHERE placa LIKE ?");
        stmt.setString(1, "%" + placa + "%");
        rs = stmt.executeQuery();

        while (rs.next()) {
            informacoesCarro informations = new informacoesCarro();

            informations.setDono(rs.getString("dono"));
            informations.setAnoDoCarro(rs.getInt("anoDoCarro"));
            informations.setCor(rs.getString("cor"));
            informations.setModelo(rs.getString("modelo"));

```

```

        informations.setPlaca(rs.getString("placa"));

        listaInformacoes.add(informations);
    }
} catch (SQLException ex) {
    JOptionPane.showMessageDialog(null, "Erro ao mostrar a lista!");
} finally {
    ConnectionFactory.closeConnection(connect, stmt, rs);
}

return listaInformacoes;
}

// metodo para dar o update de novas informacoes de carros ja
cadastrados
public void update (informacoesCarro informationCars) {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;

    try {
        stmt = connect.prepareStatement("UPDATE informacoes_carro SET
dono = ?, cor = ? WHERE ID_Carro = ?");

        stmt.setString(1, informationCars.getDono());
        stmt.setString(2, informationCars.getCor());
        stmt.setInt(3, informationCars.getID_Carro());

        stmt.executeUpdate();

        JOptionPane.showMessageDialog(null, "Informações atualizadas com
sucesso!");
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Desculpe, houve um erro ao
atualizar as informações!");
    } finally {
        ConnectionFactory.closeConnection(connect, stmt);
    }
}

```

```

    }
}

// metodo para deletar um certo carro no banco de dados
public void delete (informacoesCarro informationCars) {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;

    try {
        stmt = connect.prepareStatement("DELETE FROM informacoes_carro
        WHERE ID_Carro = ?");

        stmt.setInt(1, informationCars.getID_Carro());

        stmt.executeUpdate();

        JOptionPane.showMessageDialog(null, "Informações excluídas com
        sucesso!");
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Desculpe, houve um erro ao
        excluir as informações!");
    } finally {
        ConnectionFactory.closeConnection(connect, stmt);
    }
}
}

```

Package Models.dao

Classe imagesDAO.java

```

package models.dao;

import connection.ConnectionFactory;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

```

```

import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.swing.JOptionPane;
import models.bean.images;

public class ImagesDAO {
    public void Save (images imagem) {
        Connection connect = ConnectionFactory.getConnection();
        PreparedStatement stmt = null;

        try {
            stmt = connect.prepareStatement("INSERT INTO tabela_imagens
(imagem) VALUES (?)");

            // converter a imagem Buffered em bytes
            /*
                ByteArrayOutputStream bytesImg = new ByteArrayOutputStream();
                ImageIO.write((BufferedImage)img, "jpg", bytesImg);
                bytesImg.flush();
                byte[] byteArray = bytesImg.toByteArray();
                bytesImg.close();
            */

            stmt.setBytes(1, imagem.getImagem());

            stmt.executeUpdate();

            JOptionPane.showMessageDialog(null, "Imagem cadastrada com
sucesso");
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(null, "Erro ao cadastrar imagem!");
        } finally {
            ConnectionFactory.closeConnection(connect, stmt);
        }
    }
}

```



```

public List<images> listar() {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;
    ResultSet rs = null;

    List<images> listaImagens = new ArrayList<>();

    try {
        stmt = connect.prepareStatement("SELECT * FROM tabela_imagens");
        rs = stmt.executeQuery();

        while (rs.next()) {
            images imagem = new images();

            imagem.setImagem(rs.getBytes("imagem"));
            imagem.setCodigoImg(rs.getInt("codigoImg"));

            listaImagens.add(imagem);
        }
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Erro ao mostrar a lista!");
    } finally {
        ConnectionFactory.closeConnection(connect, stmt, rs);
    }

    return listaImagens;
}

public void delete (images imagem) {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;

    try {

```

```
stmt = connect.prepareStatement("DELETE FROM tabela_imagens
WHERE codigolmg = ?");
```

```
stmt.setInt(1, imagem.getCodigolmg());
stmt.executeUpdate();
```

```
JOptionPane.showMessageDialog(null, "Informações excluídas com
sucesso!");
```

```
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Desculpe, houve um erro ao
excluir as informações!");
    } finally {
        ConnectionFactory.closeConnection(connect, stmt);
    }
}
}
```

Package Models.dao

Classe multasDAO.java

```
package models.dao;
```

```
import connection.ConnectionFactory;
import java.util.List;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.SQLException;
import java.util.ArrayList;
import javax.swing.JOptionPane;
import models.bean.multasCarros;
```

```
public class multasDAO {
    public void create (multasCarros multaCarro) {
        Connection connect = ConnectionFactory.getConnection();
```

```
PreparedStatement stmt = null;
```

```
try {
    stmt = connect.prepareStatement("INSERT INTO multas (valorMulta,
nome, cpf, tipoMulta, placaVeiculo)"
    + "VALUES(?, ?, ?, ?, ?)");

    stmt.setFloat(1, multaCarro.getValorMulta());
    stmt.setString(2, multaCarro.getNome());
    stmt.setString(3, multaCarro.getCpf());
    stmt.setString(4, multaCarro.getTipoMulta());
    stmt.setString(5, multaCarro.getPlacaVeiculo());

    stmt.executeUpdate();

    JOptionPane.showMessageDialog(null, "Multa cadastrada com
sucesso!");
} catch (SQLException ex) {
    JOptionPane.showMessageDialog(null, "Erro ao cadastrar a multa!");
} finally {
    ConnectionFactory.closeConnection(connect, stmt);
}
}
```

```
public List<multasCarros> read() {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;
    ResultSet rs = null;

    List<multasCarros> listaInformacoes = new ArrayList<>();

    try {
        stmt = connect.prepareStatement("SELECT * FROM multas");
        rs = stmt.executeQuery();

        while (rs.next()) {
```

```

        multasCarros informations = new multasCarros();

        informations.setID_Multa(rs.getInt("ID_Multa"));
        informations.setNome(rs.getString("nome"));
        informations.setCpf(rs.getString("cpf"));
        informations.setPlacaVeiculo(rs.getString("placaVeiculo"));
        informations.setTipoMulta(rs.getString("tipoMulta"));
        informations.setValorMulta(rs.getFloat("valorMulta"));

        listaInformacoes.add(informations);
    }
} catch (SQLException ex) {
    JOptionPane.showMessageDialog(null, "Erro ao mostrar a lista!");
} finally {
    ConnectionFactory.closeConnection(connect, stmt, rs);
}

return listaInformacoes;
}

public void update(multasCarros attMulta) {
    Connection connect = ConnectionFactory.getConnection();
    PreparedStatement stmt = null;

    try {
        stmt = connect.prepareStatement("UPDATE multas SET nome = ?, cpf
= ? WHERE ID_Multa = ?");

        stmt.setString(1, attMulta.getNome());
        stmt.setString(2, attMulta.getCpf());
        stmt.setInt(3, attMulta.getID_Multa());

        stmt.executeUpdate();

        JOptionPane.showMessageDialog(null, "Informações atualizadas com
sucesso!");
    }
}

```

```

    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Desculpe, houve um erro ao
        atualizar as informações!");
    } finally {
        ConnectionFactory.closeConnection(connect, stmt);
    }
}
}

```

Package Views

View Dashboard.java

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package views;

import java.text.ParseException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.text.MaskFormatter;
import models.bean.cadastroUsers;
import models.dao.CadastroDAO;

/**
 *
 * @author mateus
 */
public class Dashboard extends javax.swing.JFrame {

    /**
     * Creates new form Dashboard
     */
    public Dashboard() {

```

```

        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        bttCadastrar = new javax.swing.JButton();
        bttCancelar = new javax.swing.JButton();
        label1 = new javax.swing.JLabel();
        textNome = new javax.swing.JTextField();
        jLabel1 = new javax.swing.JLabel();
        textEmail = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        textEndereco = new javax.swing.JTextField();
        textSenha = new javax.swing.JPasswordField();
        textCPF = new javax.swing.JFormattedTextField();
        jLabel5 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        bttCadastrar.setFont(new java.awt.Font("Century Gothic", 0, 18)); //
        NOI18N
        bttCadastrar.setText("Cadastrar");
        bttCadastrar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                bttCadastrarActionPerformed(evt);
            }
        }
    }

```

```
});
```

```
    bttCancelar.setFont(new java.awt.Font("Century Gothic", 0, 18)); //
NOI18N
```

```
    bttCancelar.setText("Cancelar");
    bttCancelar.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            bttCancelarActionPerformed(evt);
        }
    });
```

```
    label1.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
    label1.setText("Nome:");
```

```
    textNome.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            textNomeActionPerformed(evt);
        }
    });
```

```
    jLabel1.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
    jLabel1.setText("Email:");
```

```
    jLabel2.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
    jLabel2.setText("Senha:");
```

```
    jLabel3.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
    jLabel3.setText("CPF:");
```

```
    jLabel4.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
    jLabel4.setText("Endereco:");
```

```
    textSenha.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            textSenhaActionPerformed(evt);
        }
    });
```

$$\} \\ \}));$$

```
textCPF.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        textCPFActionPerformed(evt);
    }
});
```

```
jLabel5.setFont(new java.awt.Font("Century Gothic", 1, 48)); // NOI18N
jLabel5.setText("Nova Conta");
```

```

    javax.swing.GroupLayout layout = new
    javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(201, 201, 201)
                .addComponent(jLabel5)
                .addGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addGap(51, 51, 51)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(201, 201, 201)
                        .addComponent(jLabel2)
                        .addGap(51, 51, 51))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addGap(51, 51, 51))
                )
            )
    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(51, 51, 51)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel1)
                    .addComponent(jLabel2)
                )
                .addGap(51, 51, 51)
            )
    );
}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

```



```

        .addComponent(label1)
        .addGap(5, 5, 5)
        .addComponent(textNome,
javax.swing.GroupLayout.PREFERRED_SIZE, 215,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(39, 39, 39)
        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 59,
javax.swing.GroupLayout.PREFERRED_SIZE)))
        .addGroup(layout.createSequentialGroup())
        .addComponent(jLabel3)
        .addGap(18, 18, 18)
        .addComponent(textCPF,
javax.swing.GroupLayout.PREFERRED_SIZE, 221,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

        .addComponent(textEmail,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addComponent(textSenha)))
        .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
false)

        .addGroup(layout.createSequentialGroup())
        .addComponent(jLabel4)
        .addGap(10, 10, 10)
        .addComponent(textEndereco,
javax.swing.GroupLayout.PREFERRED_SIZE, 509,
javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup())
        .addComponent(bttCadastrar,
javax.swing.GroupLayout.PREFERRED_SIZE, 296,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(18, 18, 18)
        .addComponent(bttCancelar,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))))

```

```

        .addGap(0, 22, Short.MAX_VALUE))
    );
    layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
            .addGap(14, 14, 14)
            .addComponent(jLabel5)
            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
            .addComponent(label1,
javax.swing.GroupLayout.PREFERRED_SIZE, 32,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(textNome,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(textEmail,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 34,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
            .addComponent(jLabel3)
            .addComponent(textCPF,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(jLabel2)
            .addComponent(textSenha,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)

```

```

        .addComponent(jLabel4)
        .addComponent(textEndereco,
javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(26, 26, 26)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
        .addComponent(bttCadastrar,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addComponent(bttCancelar,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))
        .addContainerGap(30, Short.MAX_VALUE))
    );

    pack();
} // </editor-fold>

```

```

private void textNomeActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

```

```

private void bttCadastrarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    // instanciando as classes do modelo de Bean e de DAO
    cadastroUsers cadastroUser = new cadastroUsers();
    CadastroDAO cadastro = new CadastroDAO();

    // setando e colocando cada valor em sua variavel
    cadastroUser.setNome(textNome.getText());
    cadastroUser.setEmail(textEmail.getText());
    cadastroUser.setSenha(new String (textSenha.getPassword()));
    cadastroUser.setCpf(textCPF.getText());
    cadastroUser.setEndereco(textEndereco.getText());

    // criando o usuario no banco de dados
}

```

```

cadastro.create(cadastroUser);

// limpando os textos
textNome.setText("");
textEmail.setText("");
textSenha.setText("");
textCPF.setText("");
textEndereco.setText("");

// redirecionando o usuario para fazer o login
new Signup().setVisible(true);
this.dispose();

/*

*/

}

private void bttCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.dispose();
}

private void textSenhaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void textCPFActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    /*
    try {
        // TODO add your handling code here:
        MaskFormatter mascara = new MaskFormatter("###.###.###-##");
        mascara.setPlaceholderCharacter('_');
    }
    */
}

```

```

        textCPF.setFormatterFactory(new
javax.swing.text.DefaultFormatterFactory(mascara));
    } catch (ParseException ex) {
    }
    */
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default
look and feel.
    * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Dashboard.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);

    } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Dashboard.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);

    } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Dashboard.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);

```

```

    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Dashboard.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
    }
//</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new Dashboard().setVisible(true);
    }
});
}

// Variables declaration - do not modify
private javax.swing.JButton bttCadastrar;
private javax.swing.JButton bttCancelar;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel label1;
private javax.swing.JFormattedTextField textCPF;
private javax.swing.JTextField textEmail;
private javax.swing.JTextField textEndereco;
private javax.swing.JTextField textNome;
private javax.swing.JPasswordField textSenha;
// End of variables declaration
}

```

Package Views

View Multas.java

```
/*
```

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates

* and open the template in the editor.

*/

package views;

import java.awt.Image;

import java.awt.image.BufferedImage;

import java.io.ByteArrayOutputStream;

import java.io.File;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.imageio.ImageIO;

import javax.swing.ImageIcon;

import javax.swing.JFileChooser;

import javax.swing.JOptionPane;

import javax.swing.filechooser.FileNameExtensionFilter;

import javax.swing.table.DefaultTableModel;

import javax.swing.table.TableRowSorter;

import models.bean.images;

import models.bean.multasCarros;

import models.dao.ImagesDAO;

import models.dao.multasDAO;

/**

*

* @author Guilherme

*/

public class Multas extends javax.swing.JFrame {

 BufferedImage imagemBuffer = null;

 ByteArrayOutputStream bytesImagem = new ByteArrayOutputStream();

 byte[] arrayByte;

 /**

 * Creates new form Multas

```

*/
public Multas() {
    initComponents();

    DefaultTableModel modeloTabela = (DefaultTableModel)
tabelaInformacoes.getModel();
    tabelaInformacoes.setRowSorter(new TableRowSorter(modeloTabela));

    readTabela();
}

public void readTabela() {
    DefaultTableModel modeloTabela = (DefaultTableModel)
tabelaInformacoes.getModel();
    modeloTabela.setNumRows(0);

    multasDAO infoMultas = new multasDAO();

    infoMultas.read().forEach((multas) -> {
        modeloTabela.addRow(new Object[]{
            multas.getID_Multa(),
            multas.getNome(),
            multas.getCpf(),
            multas.getPlacaVeiculo(),
            multas.getTipoMulta(),
            multas.getValorMulta(),
        });
    });
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")

```



```
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jCheckBoxMenuItem1 = new javax.swing.JCheckBoxMenuItem();
    jFileChooser1 = new javax.swing.JFileChooser();
    jLabel1 = new javax.swing.JLabel();
    jLabel2 = new javax.swing.JLabel();
    textNome = new javax.swing.JTextField();
    jLabel3 = new javax.swing.JLabel();
    textCpf = new javax.swing.JTextField();
    jLabel4 = new javax.swing.JLabel();
    comboMulta = new javax.swing.JComboBox<>();
    jLabel5 = new javax.swing.JLabel();
    textValorMulta = new javax.swing.JTextField();
    btnSalvarImg = new javax.swing.JButton();
    jScrollPane1 = new javax.swing.JScrollPane();
    tabelaInformacoes = new javax.swing.JTable();
    jLabel7 = new javax.swing.JLabel();
    textPlaca = new javax.swing.JTextField();
    labelImagem = new javax.swing.JLabel();
    btnSaveMulta = new javax.swing.JButton();
    btnVoltar = new javax.swing.JButton();
    btnAtualizarMulta = new javax.swing.JButton();

    jCheckBoxMenuItem1.setSelected(true);
    jCheckBoxMenuItem1.setText("jCheckBoxMenuItem1");

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

    jLabel1.setFont(new java.awt.Font("Century Gothic", 1, 48)); // NOI18N
    jLabel1.setText("Multas");

    jLabel2.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
    jLabel2.setText("Nome :");
```

```
jLabel3.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
jLabel3.setText("CPF :");
```

```
textCpf.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        textCpfActionPerformed(evt);
    }
});
```

```
jLabel4.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
jLabel4.setText("Gravidade da Multa :");
```

```
comboMulta.setFont(new java.awt.Font("DialogInput", 0, 12)); // NOI18N
comboMulta.setModel(new javax.swing.DefaultComboBoxModel<>(new
String[] { "", "Leve", "Média", "Grave", "Gravíssima" }));
```

```
jLabel5.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
jLabel5.setText("Valor da Multa :");
```

```
btnnSalvarImg.setFont(new java.awt.Font("Century Gothic", 0, 18)); //
NOI18N
btnnSalvarImg.setText("Salvar Imagem");
btnnSalvarImg.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnnSalvarImgActionPerformed(evt);
    }
});
```

```
tabelaInformacoes.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {

    },
    new String [] {
        "ID_Multa", "Nome", "CPF", "Placa", "Gravidade", "Valor"
    }
});
```

```

    ) {
        boolean[] canEdit = new boolean [] {
            false, true, false, false, false, false
        };

        public boolean isCellEditable(int rowIndex, int columnIndex) {
            return canEdit [columnIndex];
        }
    });
    tabelaInformacoes.addMouseListener(new java.awt.event.MouseAdapter()
    {
        public void mouseClicked(java.awt.event.MouseEvent evt) {
            tabelaInformacoesMouseClicked(evt);
        }
    });
    tabelaInformacoes.addKeyListener(new java.awt.event.KeyAdapter() {
        public void keyReleased(java.awt.event.KeyEvent evt) {
            tabelaInformacoesKeyReleased(evt);
        }
    });
    jScrollPane1.setViewportViewView(tabelaInformacoes);
    if (tabelaInformacoes.getColumnModel().getColumnCount() > 0) {
tabelaInformacoes.getColumnModel().getColumn(0).setResizable(false);
    }

    jLabel7.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
    jLabel7.setText("Placa do Veículo :");

    textPlaca.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            textPlacaActionPerformed(evt);
        }
    });

```

```

        labelImagem.setFont(new java.awt.Font("Century Gothic", 0, 24)); //
        NOI18N

```

```

        labelImagem.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        labelImagem.setText("Inserir Imagem");
        labelImagem.setBorder(javax.swing.BorderFactory.createLineBorder(new
        java.awt.Color(51, 51, 51)));
        labelImagem.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                labelImagemMouseClicked(evt);
            }
        });

```

```

        btnSaveMulta.setFont(new java.awt.Font("Century Gothic", 0, 18)); //
        NOI18N
        btnSaveMulta.setText("Cadastrar Multa");
        btnSaveMulta.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSaveMultaActionPerformed(evt);
            }
        });

```

```

        btnVoltar.setFont(new java.awt.Font("Century Gothic", 0, 14)); // NOI18N
        btnVoltar.setText("Voltar");
        btnVoltar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnVoltarActionPerformed(evt);
            }
        });

```

```

        btnAtualizarMulta.setFont(new java.awt.Font("Century Gothic", 0, 18)); //
        NOI18N
        btnAtualizarMulta.setText("Atualizar Multa");
        btnAtualizarMulta.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAtualizarMultaActionPerformed(evt);
            }
        });

```



```

        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 434,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(10, 10, 10)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

        .addComponent(labelImagem,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

        .addComponent(btnSalvarImg,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

        .addComponent(btnVoltar)

        .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 82,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

        .addComponent(textNome,
javax.swing.GroupLayout.PREFERRED_SIZE, 212,
javax.swing.GroupLayout.PREFERRED_SIZE))

        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

        .addComponent(textValorMulta))

        .addGroup(layout.createSequentialGroup()

```

```

        .addComponent(jLabel4,
javax.swing.GroupLayout.PREFERRED_SIZE, 199,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(comboMulta,
javax.swing.GroupLayout.PREFERRED_SIZE, 95,
javax.swing.GroupLayout.PREFERRED_SIZE)))

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(btnAtualizarMulta)))

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(btnSaveMulta))))))
        .addContainerGap())
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
        .addGap(0, 0, Short.MAX_VALUE)
        .addComponent(jLabel1)
        .addGap(237, 237, 237))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
        .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jLabel1))
        .addGroup(layout.createSequentialGroup()
        .addGap(19, 19, 19)
        .addComponent(btnVoltar,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)))
        .addGap(29, 29, 29)

```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
    .addComponent(jLabel2,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 33,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(textNome,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 33,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(jLabel3,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 37,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(textCpf,  
    javax.swing.GroupLayout.DEFAULT_SIZE, 39, Short.MAX_VALUE))
```

```
    .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
    .addComponent(jLabel4,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 33,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(comboMulta,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 33,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(jLabel7,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 33,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(textPlaca,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 33,  
    javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
    .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
    .addComponent(jLabel5,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 33,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(btnSaveMulta,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 37,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(btnAtualizarMulta,  
    javax.swing.GroupLayout.PREFERRED_SIZE, 37,  
    javax.swing.GroupLayout.PREFERRED_SIZE)
```



```

        .addComponent(textValorMulta,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(27, 27, 27)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup())

        .addComponent(labelImagem,
javax.swing.GroupLayout.PREFERRED_SIZE, 161,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

        .addComponent(btnSalvarImg,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))

        .addComponent(jScrollPane1,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 205,
javax.swing.GroupLayout.PREFERRED_SIZE))

        .addGap(13, 13, 13))
    );

    pack();
} // </editor-fold>

private void textCpfActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

public void getImagem() throws IOException {
    JFileChooser adicionarImagem = new JFileChooser();

    adicionarImagem.setFileFilter(new FileNameExtensionFilter("imagem",
"bmp", "png", "jpg", "jpeg"));
    adicionarImagem.setAcceptAllFileFilterUsed(false);
    adicionarImagem.setDialogTitle("Selecionar imagem");
    adicionarImagem.showOpenDialog(this);
}

```

```

        if (adicionarImagem.getSelectedFile() != null) {
            String diretorio = "" +
adicionarImagem.getSelectedFile().getAbsolutePath();

            imagemBuffer = ImageIO.read(new File(diretorio));
            Image redimensionarImagem = imagemBuffer.getScaledInstance(220,
100, 0);

            labelImagem.setText("");
            labelImagem.setIcon(new ImageIcon(redimensionarImagem));
        }
    }

private void btnnSalvarImgActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    images imagem = new images();
    ImagesDAO inserirImagem = new ImagesDAO();

    try {
        ImageIO.write(imagemBuffer, "jpg", bytesImagem);
        bytesImagem.flush();
        arrayByte = bytesImagem.toByteArray();
        bytesImagem.close();

        imagem.setImagem(arrayByte);
        inserirImagem.Save(imagem);
    } catch (IOException ex) {
        Logger.getLogger(Multas.class.getName()).log(Level.SEVERE, null, ex);
    }
}

private void textPlacaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

```

```

private void labelImagemMouseClicked(java.awt.event.MouseEvent evt) {
    try {
        this.getImagem();
    } catch (IOException ex) {
        Logger.getLogger(Multas.class.getName()).log(Level.SEVERE, null, ex);
    }
}

```

```

private void btnnSaveMultaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    multasCarros newMulta = new multasCarros();
    multasDAO saveMultaBD = new multasDAO();

    newMulta.setValorMulta(Float.parseFloat(textValorMulta.getText()));
    newMulta.setNome(textNome.getText());
    newMulta.setCpf(textCpf.getText());
    newMulta.setTipoMulta((String) comboMulta.getSelectedItem());
    newMulta.setPlacaVeiculo(textPlaca.getText());

    saveMultaBD.create(newMulta);

    textValorMulta.setText("");
    textNome.setText("");
    textCpf.setText("");
    comboMulta.setSelectedItem("");
    textPlaca.setText("");

    readTabela();
}

```

```

private void tabelaInformacoesKeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here
    if (tabelaInformacoes.getSelectedRow() != -1) {

```

```
textNome.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 1).toString());
```

```
textCpf.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 2).toString());
```

```
textPlaca.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 3).toString());
```

```
textValorMulta.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 5).toString());
```

```
    }
```

```
}
```

```
private void tabelaInformacoesMouseClicked(java.awt.event.MouseEvent evt) {
```

```
    // TODO add your handling code here:
```

```
    if (tabelaInformacoes.getSelectedRow() != -1) {
```

```
textNome.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 1).toString());
```

```
textCpf.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 2).toString());
```

```
textPlaca.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 3).toString());
```

```
textValorMulta.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 5).toString());
```

```
    }
```

```
}
```

```
private void btnVoltarActionPerformed(java.awt.event.ActionEvent evt) {
```

```
    // TODO add your handling code here:
```

```
    new Register().setVisible(true);
```

```
    this.dispose();
```

```
}
```

```

private void btnAtualizarMultaActionPerformed(java.awt.event.ActionEvent
evt) {
    // TODO add your handling code here:
    if (tabelaInformacoes.getSelectedRow() != -1) {
        multasCarros attMulta = new multasCarros();
        multasDAO updateMulta = new multasDAO();

        attMulta.setNome(textNome.getText());
        attMulta.setCpf(textCpf.getText());

        attMulta.setID_Multa((int)tabelaInformacoes.getValueAt(tabelaInformacoes.get
SelectedRow(), 0));

        updateMulta.update(attMulta);

        textValorMulta.setText("");
        textNome.setText("");
        textCpf.setText("");
        comboMulta.setSelectedItem("");
        textPlaca.setText("");

        readTabela();
    }
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default
look and feel.
    * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */

```

```

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
                javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Multas.class.getName()).log(java.util.logging.
Level.SEVERE, null, ex);

        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Multas.class.getName()).log(java.util.logging.
Level.SEVERE, null, ex);

        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Multas.class.getName()).log(java.util.logging.
Level.SEVERE, null, ex);

        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Multas.class.getName()).log(java.util.logging.
Level.SEVERE, null, ex);

        }
    }
}
//</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new Multas().setVisible(true);
    }
});
}

// Variables declaration - do not modify
private javax.swing.JButton btnAtualizarMultas;
private javax.swing.JButton btnSalvarImg;

```

```

private javax.swing.JButton btnSaveMulta;
private javax.swing.JButton btnVoltar;
private javax.swing.JComboBox<String> comboMulta;
private javax.swing.JCheckBoxMenuItem jCheckBoxMenuItem1;
private javax.swing.JFileChooser jFileChooser1;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel7;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JLabel labelImagem;
private javax.swing.JTable tabelaInformacoes;
private javax.swing.JTextField textCpf;
private javax.swing.JTextField textNome;
private javax.swing.JTextField textPlaca;
private javax.swing.JTextField textValorMulta;
// End of variables declaration
}

```

Package Views

View Register.java

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package views;

import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableRowSorter;
import models.bean.informacoesCarro;
import models.dao.CarroDAO;

```

```

/**
 *
 * @author mateus
 */
public final class Register extends javax.swing.JFrame {

    /**
     * Creates new form Register
     */
    public Register() {
        initComponents();
        DefaultTableModel modeloTabela = (DefaultTableModel)
tabelainformacoes.getModel();
        tabelainformacoes.setRowSorter(new TableRowSorter(modeloTabela));

        readTabela();
    }

    public void readTabela() {
        DefaultTableModel modeloTabela = (DefaultTableModel)
tabelainformacoes.getModel();
        modeloTabela.setNumRows(0);
        CarroDAO infoDAO = new CarroDAO();

        infoDAO.read().forEach((carro) -> {
            modeloTabela.addRow(new Object[]{
                carro.getID_Carro(),
                carro.getDono(),
                carro.getModelo(),
                carro.getPlaca(),
                carro.getCor(),
                carro.getAnoDoCarro(),});
        });
    }
}

```



```

public void readTabelaForPlaca(String placa) {
    DefaultTableModel modeloTabela = (DefaultTableModel)
tabelaInformacoes.getModel();
    modeloTabela.setNumRows(0);
    CarroDAO infoDAO = new CarroDAO();

    for (informacoesCarro carro : infoDAO.buscarCarro(placa)) {
        modeloTabela.addRow(new Object[]{
            carro.getID_Carro(),
            carro.getDono(),
            carro.getModelo(),
            carro.getPlaca(),
            carro.getCor(),
            carro.getAnoDoCarro(),});
    }
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jScrollPane1 = new javax.swing.JScrollPane();
    tabelaInformacoes = new javax.swing.JTable();
    Cadastrar = new javax.swing.JButton();
    Deletar = new javax.swing.JButton();
    textBusca = new javax.swing.JTextField();
    bttBusca = new javax.swing.JButton();
    jLabel2 = new javax.swing.JLabel();
    textDono = new javax.swing.JTextField();
    jLabel1 = new javax.swing.JLabel();

```

```

textModelo = new javax.swing.JTextField();
jLabel3 = new javax.swing.JLabel();
textPlaca = new javax.swing.JTextField();
jLabel4 = new javax.swing.JLabel();
textCor = new javax.swing.JTextField();
jLabel5 = new javax.swing.JLabel();
textAno = new javax.swing.JTextField();
bttAtualizar = new javax.swing.JButton();
jLabel6 = new javax.swing.JLabel();
bttMultas = new javax.swing.JButton();

```

```

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

```

```

tabelaInformacoes.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {

    },
    new String [] {
        "ID_Carro", "DONO", "MODELO", "PLACA", "COR", "ANO"
    }
) {
    boolean[] canEdit = new boolean [] {
        false, false, false, false, false, false
    };

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return canEdit [columnIndex];
    }
});
tabelaInformacoes.addMouseListener(new java.awt.event.MouseAdapter()
{
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        tabelaInformacoesMouseClicked(evt);
    }
});

```

```

tabelaInformacoes.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyReleased(java.awt.event.KeyEvent evt) {
        tabelaInformacoesKeyReleased(evt);
    }
});
jScrollPane1.setViewportViewView(tabelaInformacoes);
if (tabelaInformacoes.getColumnModel().getColumnCount() > 0) {

tabelaInformacoes.getColumnModel().getColumn(5).setResizable(false);
}

Cadastrar.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
Cadastrar.setText("Cadastrar");
Cadastrar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        CadastrarActionPerformed(evt);
    }
});

Deletar.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
Deletar.setText("Excluir");
Deletar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        DeletarActionPerformed(evt);
    }
});

textBusca.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        textBuscaActionPerformed(evt);
    }
});

btbBusca.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
btbBusca.setText("Buscar");

```

```

bttBusca.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        bttBuscaActionPerformed(evt);
    }
});

```

```

jLabel2.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
jLabel2.setText("Dono:");

```

```

jLabel1.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
jLabel1.setText("Modelo:");

```

```

jLabel3.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
jLabel3.setText("Placa:");

```

```

textPlaca.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        textPlacaActionPerformed(evt);
    }
});

```

```

jLabel4.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
jLabel4.setText("Cor:");

```

```

jLabel5.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
jLabel5.setText("Ano:");

```

```

textAno.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        textAnoActionPerformed(evt);
    }
});

```

```

bttAtualizar.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
bttAtualizar.setText("Atualizar");

```

```

bttAtualizar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        bttAtualizarActionPerformed(evt);
    }
});

jLabel6.setFont(new java.awt.Font("Century Gothic", 1, 48)); // NOI18N
jLabel6.setText("Registro de Veículos");

bttMultas.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
bttMultas.setText("Registrar Multa");
bttMultas.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        bttMultasActionPerformed(evt);
    }
});

javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .add(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .add(jLabel6)
                .add(bttAtualizar)
                .add(bttMultas)
            )
            .addContainerGap(142, true)
        )
);

```

```

        .addComponent(jLabel6)))
    .addGroup(layout.createSequentialGroup())
        .addGap(49, 49, 49)

    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
        .addComponent(bttMultas,
            javax.swing.GroupLayout.DEFAULT_SIZE,
            javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addComponent(jScrollPane1)
        .addGroup(layout.createSequentialGroup()

    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
        .addGroup(layout.createSequentialGroup()
            .addComponent(jLabel1)

    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(textModelo))
        .addGroup(layout.createSequentialGroup()
            .addComponent(jLabel2)

    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(textDono,
            javax.swing.GroupLayout.PREFERRED_SIZE, 203,
            javax.swing.GroupLayout.PREFERRED_SIZE))

    .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
        layout.createSequentialGroup()
            .addComponent(jLabel3,
                javax.swing.GroupLayout.PREFERRED_SIZE, 60,
                javax.swing.GroupLayout.PREFERRED_SIZE)

    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(textPlaca,
            javax.swing.GroupLayout.PREFERRED_SIZE, 127,
            javax.swing.GroupLayout.PREFERRED_SIZE)))
        .addGap(64, 64, 64)

    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

```

```

        .addComponent(jLabel4,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGroup(layout.createSequentialGroup())
        .addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(textAno,
javax.swing.GroupLayout.PREFERRED_SIZE, 119,
javax.swing.GroupLayout.PREFERRED_SIZE))))
        .addGroup(layout.createSequentialGroup())
        .addComponent(Cadastrar)
        .addGap(18, 18, 18)
        .addComponent(btAtualizar)
        .addGap(18, 18, 18)
        .addComponent(Deletar)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATE
D)
        .addComponent(textBusca,
javax.swing.GroupLayout.PREFERRED_SIZE, 145,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(btBusca,
javax.swing.GroupLayout.PREFERRED_SIZE, 134,
javax.swing.GroupLayout.PREFERRED_SIZE))))
        .addContainerGap(22, Short.MAX_VALUE)
    );
    layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jLabel6)
            .addGap(16, 16, 16)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)

```

```

        .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(textDono,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(jLabel4,
javax.swing.GroupLayout.PREFERRED_SIZE, 26,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(textCor,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))

        .addGap(35, 35, 35)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)

        .addComponent(jLabel1)

        .addComponent(textModelo,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(jLabel5,
javax.swing.GroupLayout.PREFERRED_SIZE, 29,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(textAno,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))

        .addGap(29, 29, 29)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)

        .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 24,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(textPlaca,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))

        .addGap(35, 35, 35)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)

        .addComponent(Cadastrar,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

```



```

        .addComponent(bttAtualizar,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(Deletar,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(textBusca,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(bttBusca,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))

        .addGap(28, 28, 28)

        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 128,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATE
D)

        .addComponent(bttMultas,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addContainerGap(21, Short.MAX_VALUE)

    );

    pack();
} // </editor-fold>

```

```

private void CadastrarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    informacoesCarro carro = new informacoesCarro();
    CarroDAO carroDAO = new CarroDAO();

    carro.setDono(textDono.getText());
    carro.setModelo(textModelo.getText());
    carro.setPlaca(textPlaca.getText());
    carro.setCor(textCor.getText());
    carro.setAnoDoCarro(Integer.parseInt(textAno.getText()));
}

```

```

        carroDAO.create(carro);
        // aq embaixo vc limpa os campos
        textDono.setText("");
        textModelo.setText("");
        textPlaca.setText("");
        textCor.setText("");
        textAno.setText("");

        readTabela();
    }

    private void tabelaInformacoesKeyReleased(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        if (tabelaInformacoes.getSelectedRow() != -1) {

            textDono.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelected
            Row(), 1).toString());

            textModelo.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelect
            edRow(), 2).toString());

            textPlaca.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelecte
            dRow(), 3).toString());

            textCor.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedR
            ow(), 4).toString());

            textAno.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelected
            Row(), 5).toString());
        }
    }

    private void DeletarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        if (tabelaInformacoes.getSelectedRow() != -1) {
            informacoesCarro carro = new informacoesCarro();
            CarroDAO carroDAO = new CarroDAO();

```

```

        carro.setID_Carro((int)
tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 0));

        carroDAO.delete(carro);

        textDono.setText("");
        textModelo.setText("");
        textPlaca.setText("");
        textCor.setText("");
        textAno.setText("");

        readTabela();
    } else {
        JOptionPane.showMessageDialog(null, "Selecione um item para ser
excluido");
    }
}

private void textBuscaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void bttBuscaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    readTabelaForPlaca(textBusca.getText());
}

private void textPlacaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void textAnoActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void bttAtualizarActionPerformed(java.awt.event.ActionEvent evt) {

```

```

// TODO add your handling code here:
if (tabelaInformacoes.getSelectedRow() != -1) {
    informacoesCarro carro = new informacoesCarro();
    CarroDAO carroDAO = new CarroDAO();

    carro.setDono(textDono.getText());
    carro.setCor(textCor.getText());

    carro.setID_Carro((int)tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 0));

    carroDAO.update(carro);
    // aq embaixo vc limpa os campos
    textDono.setText("");
    textModelo.setText("");
    textPlaca.setText("");
    textCor.setText("");
    textAno.setText("");

    readTabela();
}
}

private void tabelaInformacoesMouseClicked(java.awt.event.MouseEvent
evt) {
    // TODO add your handling code here:
    if (tabelaInformacoes.getSelectedRow() != -1) {

        textDono.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 1).toString());

        textModelo.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 2).toString());

        textPlaca.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 3).toString());

        textCor.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelectedRow(), 4).toString());
    }
}

```

```
textAno.setText(tabelaInformacoes.getValueAt(tabelaInformacoes.getSelected
Row(), 5).toString());
```

```
    }
}
```

```
private void bttMultasActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new Multas().setVisible(true);
    this.dispose();
}
```

```
/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default
look and feel.
    * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    }
    catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Register.class.getName()).log(java.util.loggin
g.Level.SEVERE, null, ex);

    } catch (InstantiationException ex) {
```

```
java.util.logging.Logger.getLogger(Register.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
    } catch (IllegalAccessException ex) {
```

```
java.util.logging.Logger.getLogger(Register.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
```

```
java.util.logging.Logger.getLogger(Register.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
    }
```

```
//</editor-fold>
```

```
/* Create and display the form */
```

```
java.awt.EventQueue.invokeLater(new Runnable() {
```

```
    public void run() {
```

```
        new Register().setVisible(true);
```

```
    }
```

```
});
```

```
}
```

```
// Variables declaration - do not modify
```

```
private javax.swing.JButton Cadastrar;
```

```
private javax.swing.JButton Deletar;
```

```
private javax.swing.JButton bttAtualizar;
```

```
private javax.swing.JButton bttBusca;
```

```
private javax.swing.JButton bttMultas;
```

```
private javax.swing.JLabel jLabel1;
```

```
private javax.swing.JLabel jLabel2;
```

```
private javax.swing.JLabel jLabel3;
```

```
private javax.swing.JLabel jLabel4;
```

```
private javax.swing.JLabel jLabel5;
```

```
private javax.swing.JLabel jLabel6;
```

```
private javax.swing.JScrollPane jScrollPane1;
```

```
private javax.swing.JTable tabelaInformacoes;
```

```
private javax.swing.JTextField textAno;
```

```

private javax.swing.JTextField textBusca;
private javax.swing.JTextField textCor;
private javax.swing.JTextField textDono;
private javax.swing.JTextField textModelo;
private javax.swing.JTextField textPlaca;
// End of variables declaration
}

```

Package Views

View Signup.java

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package views;

import javax.swing.JOptionPane;
import models.dao.CadastroDAO;

/**
 *
 * @author mateus
 */
public class Signup extends javax.swing.JFrame {

    /**
     * Creates new form Signup
     */
    public Signup() {
        initComponents();
    }

    /**

```

```

* This method is called from within the constructor to initialize the form.
* WARNING: Do NOT modify this code. The content of this method is always
* regenerated by the Form Editor.
*/

```

```
@SuppressWarnings("unchecked")
```

```
// <editor-fold defaultstate="collapsed" desc="Generated Code">
```

```
private void initComponents() {
```

```
    jLabel1 = new javax.swing.JLabel();
```

```
    textLogin = new javax.swing.JTextField();
```

```
    jLabel2 = new javax.swing.JLabel();
```

```
    textPassword = new javax.swing.JPasswordField();
```

```
    bttLogin = new javax.swing.JButton();
```

```
    bttRegister = new javax.swing.JButton();
```

```
    jLabel3 = new javax.swing.JLabel();
```

```

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
setResizable(false);

```

```
jLabel1.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
```

```
jLabel1.setText("Email:");
```

```
jLabel2.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
```

```
jLabel2.setText("Senha:");
```

```
textPassword.addActionListener(new java.awt.event.ActionListener() {
```

```
    public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        textPasswordActionPerformed(evt);
```

```
    }
```

```
});
```

```
textPassword.addKeyListener(new java.awt.event.KeyAdapter() {
```

```
    public void keyPressed(java.awt.event.KeyEvent evt) {
```

```
        textPasswordKeyPressed(evt);
```

```
    }
```



```
});
```

```
bttLogin.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
bttLogin.setText("Entrar");
bttLogin.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        bttLoginActionPerformed(evt);
    }
});
```

```
bttRegister.setFont(new java.awt.Font("Century Gothic", 0, 18)); // NOI18N
bttRegister.setText("Cadastrar");
bttRegister.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        bttRegisterActionPerformed(evt);
    }
});
```

```
jLabel3.setFont(new java.awt.Font("Century Gothic", 1, 48)); // NOI18N
jLabel3.setText("Login");
```

```
javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addGap(132, 132, 132)

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
            .addComponent(jLabel2)
            .addComponent(jLabel1)
            .addComponent(textLogin)
            .addComponent(textPassword,
                javax.swing.GroupLayout.DEFAULT_SIZE, 309, Short.MAX_VALUE)
```

```

        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

        .addComponent(bttLogin,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

        .addGap(18, 18, 18)

        .addComponent(bttRegister,
javax.swing.GroupLayout.PREFERRED_SIZE, 150,
javax.swing.GroupLayout.PREFERRED_SIZE)))

        .addContainerGap(95, Short.MAX_VALUE))

        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

        .addComponent(jLabel3)

        .addGap(192, 192, 192))

    );
    layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

        .addGroup(layout.createSequentialGroup()

            .addGap(48, 48, 48)

            .addComponent(jLabel3)

            .addGap(27, 27, 27)

            .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

            .addComponent(textLogin,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATE
D)

            .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

```

```

        .addComponent(textPassword,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)

        .addComponent(bttLogin,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addComponent(bttRegister,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE))
        .addContainerGap(50, Short.MAX_VALUE))
    );

    pack();
} // </editor-fold>

private void textPasswordActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void bttLoginActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    CadastroDAO checkCadastro = new CadastroDAO();

    if (checkCadastro.checkUsers(textLogin.getText(), new String
(textPassword.getPassword())) {
        new Register().setVisible(true);
        this.dispose();
    } else {
        JOptionPane.showMessageDialog(null, "Usuario nao encontrado");
    }
}

private void textPasswordKeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:

```

```

    }

    private void bttRegisterActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        new Dashboard().setVisible(true);
        this.dispose();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default
look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

            java.util.logging.Logger.getLogger(Signup.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);

        } catch (InstantiationException ex) {

            java.util.logging.Logger.getLogger(Signup.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);

        } catch (IllegalAccessException ex) {

```

```

java.util.logging.Logger.getLogger(Signup.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

```

```

java.util.logging.Logger.getLogger(Signup.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
    }
//</editor-fold>

```

```

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new Signup().setVisible(true);
    }
});
}

```

```

// Variables declaration - do not modify
private javax.swing.JButton bttLogin;
private javax.swing.JButton bttRegister;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JTextField textLogin;
private javax.swing.JPasswordField textPassword;
// End of variables declaration
}

```

Script Banco de Dados:

```

-- Usar a database da APS
use dados_aps;

```

```

-- Criando a tabela que contém as informações dos usuários
create table if not exists `cadastro` (

```

```

        `id_cadastro` integer primary key auto_increment,
        `nome` varchar(255) not null,
        `email` varchar(350) not null,
        `endereco` varchar(350) not null,
        `cpf` varchar(12) not null,
        `senha` varchar(20) not null
    ) Engine = innodb;

```

-- Criando a tabela que tem as informações dos carros

```

create table if not exists `informacoes_carro` (
    `ID_Carro` integer primary key auto_increment,
    `modelo` varchar(15) not null,
    `dono` varchar(255) not null,
    `cor` varchar(15) not null,
    `anoDoCarro` smallint not null,
    `placa` varchar(6) not null,
    `ID_Imagem` integer,
    `ID_Multa` integer
) Engine = innodb;

```

-- Criando a tabela de imagens

```

create table if not exists `tabela_imagens` (
    `codigoImg` integer primary key auto_increment,
    `imagem` longblob not null,
    `ID_Carro` integer,
    `ID_Multa` integer
) Engine = innodb;

```

-- Criando a tabela de multas

```

create table if not exists `multas` (
    `ID_Multa` integer primary key auto_increment,
    `valorMulta` float not null,
    `nome` varchar(255) not null,
    `cpf` varchar(12) not null,
    `tipoMulta` varchar(50) not null,

```

```

        `placaVeiculo` varchar(6) not null,
        `ID_Carro` integer,
        `ID_Imagem` integer
    ) Engine = innodb;

```

-- Adicionando as chaves estrangeiras na tabela de informações dos carros

```

alter table `informacoes_carro` add constraint fk_CarroImagem foreign key
(`ID_Imagem`) references `tabela_imagens` (`codigoImg`);

```

```

alter table `informacoes_carro` add constraint fk_CarroMulta foreign key
(`ID_Multa`) references `multas` (`ID_Multa`);

```

-- Adicionando as chaves estrangeiras na tabela de multas

```

alter table `multas` add constraint fk_MultaImagem foreign key (`ID_Imagem`)
references `tabela_imagens` (`codigoImg`);

```

```

alter table `multas` add constraint fk_MultaCarro foreign key (`ID_Carro`)
references `informacoes_carro` (`ID_Carro`);

```

-- Adicionando as chaves estrangeiras na tabela de imagens

```

alter table `tabela_imagens` add constraint fk_ImagemCarro foreign key
(`ID_Carro`) references `informacoes_carro` (`ID_Carro`);

```

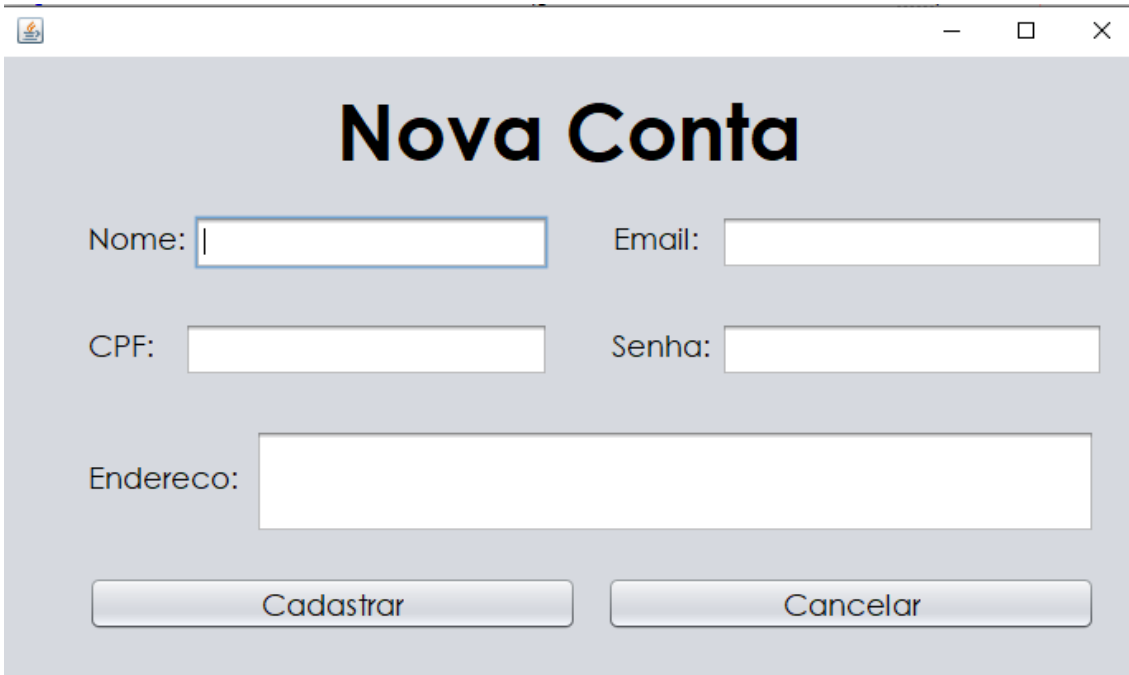
```

alter table `tabela_imagens` add constraint fk_ImagemMulta foreign key
(`ID_Multa`) references `multas` (`ID_Multa`);

```

Aqui imagens de como ficaram as interfaces gráficas:

1- Dashboard



Nova Conta

Nome: Email:

CPF: Senha:

Endereco:

2- Multas



Multas

Nome : CPF :

Gravidade da Multa : Placa do Veículo :

Valor da Multa :

ID_Multa	Nome	CPF	Placa	Gravidade	Valor
1	gui	1234567...	000123	Leve	500.0
2	Mateus	651189	000789	Grave	1800.0
3	Daniel	54646	AAA1234	Gravíssima	800.0

3- Register



A screenshot of a web application window titled "Registro de Veículos". The window has a light gray background and standard window controls (minimize, maximize, close) in the top right corner. The main content area contains several input fields and buttons. At the top, the title "Registro de Veículos" is displayed in a large, bold, black font. Below the title, there are five input fields: "Dono:" (with a blue border), "Cor:", "Modelo:", "Ano:", and "Placa:". Below these fields, there are four buttons: "Cadastrar", "Atualizar", "Excluir", and "Buscar". To the right of the "Excluir" button is an empty input field. Below the buttons is a table with six columns: "ID_Carro", "DONO", "MODELO", "PLACA", "COR", and "ANO". The table contains one row of data: "1", "Mateus", "Voyage", "AAA222", "Branco", and "2010". Below the table is a large, empty rectangular area. At the bottom of the window is a button labeled "Registrar Multa".

Registro de Veículos

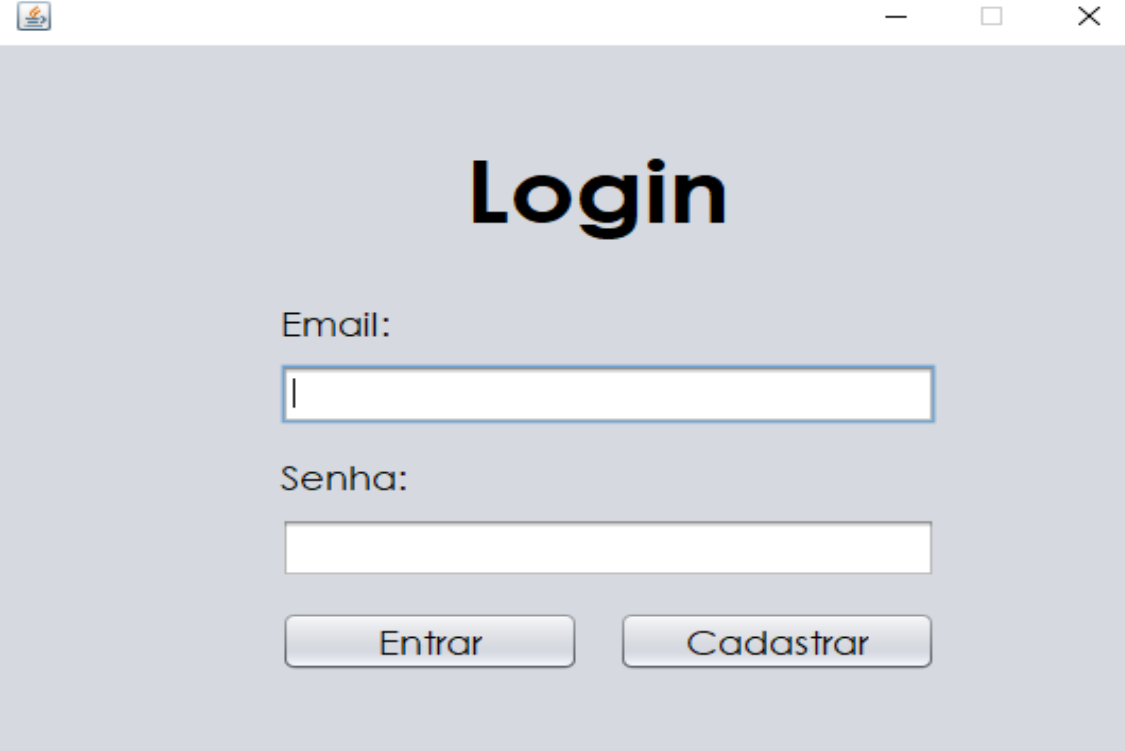
Dono: Cor:

Modelo: Ano:

Placa:

ID_Carro	DONO	MODELO	PLACA	COR	ANO
1	Mateus	Voyage	AAA222	Branco	2010

4- Signup



A screenshot of a web application window titled "Login". The window has a light gray background and standard window controls (minimize, maximize, close) in the top right corner. The main content area contains two input fields and two buttons. At the top, the title "Login" is displayed in a large, bold, black font. Below the title, there are two input fields: "Email:" and "Senha:". Below these fields are two buttons: "Entrar" and "Cadastrar".

Login

Email:

Senha: