Documentação do Sistema de Controle de Estacionamento

Introdução

Este documento descreve o sistema de controle de estacionamento desenvolvido em Java com interface gráfica (Swing) e acesso a banco de dados MySQL. O objetivo do sistema é gerenciar carros em um estacionamento, permitindo adicionar, atualizar, excluir e listar registros de carros.

Estrutura do Projeto

1. Banco de Dados

Criação da Tabela

O banco de dados utilizado é o MySQL, e a tabela `carros` foi criada com a seguinte estrutura:

sql

Copiar código

```
CREATE TABLE carros (
    id INT AUTO_INCREMENT PRIMARY KEY,
    marca VARCHAR(50),
    placa VARCHAR(20) UNIQUE,
    cor VARCHAR(30),
    horaEntrada INT,
    horaSaida INT
);
```

2. Classes Java

`Carro`

Representa a entidade de um carro.

Atributos:

• `marca` (String): Marca do carro.

• `placa` (String): Placa do carro (única).

• `cor` (String): Cor do carro.

• `horaEntrada` (int): Hora de entrada no estacionamento.

• `horaSaida` (int): Hora de saída do estacionamento.

Métodos:

• Construtores: Cinco construtores para inicialização do objeto.

• Getters e Setters: Métodos para acessar e modificar os atributos.

`CarroDAO`

Classe responsável pelas operações CRUD (Create, Read, Update, Delete) no banco de dados.

Métodos:

• `adicionarCarro(Carro carro)`: Adiciona um carro ao banco de dados.

• `listarCarros()`: Lista todos os carros cadastrados.

• `atualizarCarro(Carro carro)`: Atualiza as informações de um carro.

• `excluirCarro(String placa)`: Remove um carro do banco de dados com base na placa.

`ConexaoBD`

Classe responsável pela conexão com o banco de dados MySQL.

Métodos:

• `conectar()`: Estabelece uma conexão com o banco de dados e retorna um objeto `Connection`.

`EstacionamentoGUI`

Classe que fornece a interface gráfica do usuário (GUI) para interagir com o sistema.

Componentes:

• Campos de texto (`JTextField`): Para entrada de dados do carro.

• Botões (`JButton`): Para adicionar, listar, atualizar e excluir carros.

• Área de texto (`JTextArea`): Para exibir o relatório de carros cadastrados.

Funcionalidades:

• Adicionar: Permite adicionar um novo carro ao banco de dados.

• Listar: Mostra todos os carros cadastrados no estacionamento.

• Atualizar: Atualiza as informações de um carro existente.

• Excluir: Remove um carro do banco de dados com base na placa.

3. Configuração do Projeto

Maven

O projeto usa o Maven para gerenciamento de dependências. As dependências são especificadas no arquivo `pom.xml`.

Exemplo de Dependência:

xml

Copiar código

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
```

<version>8.0.30</version>
</dependency>

Execução

1.

Compilar o Projeto:

• No IntelliJ IDEA, selecione `Build > Rebuild Project` para compilar o código.

2.

Executar a Aplicação:

• Clique com o botão direito na classe `EstacionamentoGUI` e selecione `Run 'EstacionamentoGUI.main()'`.

• A interface gráfica será exibida em uma janela separada.

4. Uso do Sistema

Ao iniciar o aplicativo, uma janela com a interface gráfica será exibida. Os usuários podem:

• Adicionar Carro: Preencher os campos e clicar no botão "Adicionar" para incluir um carro no banco de dados.

• Listar Carros: Clicar no botão "Listar" para exibir todos os carros cadastrados.

• Atualizar Carro: Preencher os campos e clicar no botão "Atualizar" para modificar as informações de um carro existente.

• Excluir Carro: Inserir a placa do carro e clicar no botão "Excluir" para removê-lo do banco de dados.