Universidade Estácio - Unidade Santo André - Centro - SP

Análise e Desenvolvimento de Sistemas

Dennis Araujo Serafim Dias

Eduardo de Assis Vallini

Julio Francisco Bernardino

Victor Luiz Umeki Pozato

PRGRAMAÇÃO EM JAVA:

SISTEMA DE GESTÃO DE RETÍFICA DE AUTOMÓVEIS

Santo André

2024

Dennis Araujo Serafim Dias
Eduardo de Assis Vallini
Julio Francisco Bernardino
Victor Luiz Umeki Pozato

PRGRAMAÇÃO EM JAVA:
SISTEMA DE GESTÃO DE RETÍFICA DE AUTOMÓVEIS

Relatório apresentado como requisito parcial para a obtenção do título de Sistema de Gestão de Retífica de Automóveis em Java em Orientação a Objetos em Java, sob a orientação do Prof. Paulo Daniel da Cruz.

Santo André
2024

SUMÁRIO

PRGRAMAÇÃO EM JAVA:

SISTEMA DE GESTÃO DE RETÍFICA DE AUTOMÓVEIS

1. INTRODUÇÃO

O presente relatório descreve o desenvolvimento de um sistema de gestão para uma retífica de automóveis, utilizando a linguagem de programação Java e o banco de dados SQLite. Este projeto foi concebido com o objetivo de atender às necessidades específicas do negócio, proporcionando uma solução integrada e eficiente para a administração de orçamentos.

A retífica de automóveis é um setor que demanda precisão e organização na gestão de informações, como dados dos clientes, peças utilizadas e custos associados. A centralização dessas informações em um sistema informatizado não apenas facilita o acesso e a atualização dos dados, mas também melhora a transparência e a comunicação com os clientes.

O sistema desenvolvido permite o registro detalhado de orçamentos, a geração automática de relatórios financeiros, o monitoramento do histórico de clientes e a integração com outros sistemas administrativos da organização. A interface gráfica foi projetada para ser intuitiva e de fácil utilização, garantindo uma experiência de usuário agradável e eficiente.

Este projeto foi implementado utilizando o ambiente de desenvolvimento integrado (IDE) NetBeans, que oferece uma plataforma robusta e versátil para a programação em Java. A modelagem de dados foi realizada com o auxílio do software Brmodelos, assegurando uma estrutura de banco de dados bem-organizada e consistente.

Em suma, o sistema de gestão desenvolvido neste projeto representa uma ferramenta essencial para a retífica de automóveis, contribuindo para a melhoria dos processos operacionais, a satisfação dos clientes e a tomada de decisões estratégicas fundamentadas.

2. DIAGNÓSTICO E TEORIZAÇÃO

2.1. Identificação das partes envolvidas e parceiros

O desenvolvimento do sistema de orçamentos e serviços foi idealizado para atender às necessidades da REMOVEL Retífica de Motores. A empresa trabalha com serviços de manutenção e reparos de motores, oferecendo orçamentos detalhados e personalizados para cada cliente.

O foco do sistema é simplificar e auxiliar os processos internos relacionados ao gerenciamento de orçamentos e serviços, proporcionando maior eficiência e simplicidade no acompanhamento dos serviços. Além disso, busca-se oferecer uma interface amigável para facilitar o uso pelos funcionários, independentemente do nível de familiaridade com tecnologia.

Os principais usuários do sistema são:

Funcionários administrativos: responsáveis pelo acompanhamento, contato de suporte e pagamento.

Gerente e Dono: supervisores que utilizam o sistema para monitorar o andamento dos serviços e gerar relatórios.

2.2. Objetivos e resultados a serem alcançados

Desenvolver um sistema integrado de gestão de orçamentos e serviços que permita o controle eficiente e organizado dos dados relacionados à empresa.

Facilitar o processo de elaboração, consulta e atualização de orçamentos, reduzindo erros manuais e otimizando o tempo dos funcionários.

Expandir o conhecimento dos usuários sobre ferramentas tecnológicas modernas, como JavaFX e SQLite, através de uma interface intuitiva e amigável.

Promover a autonomia dos funcionários na gestão de informações do sistema, eliminando a necessidade de conhecimento avançado em SQL ou outras tecnologias de banco de dados.

2.3. Demanda organizacional e motivação acadêmica

A demanda foi fundamentada na necessidade de automatizar processos e aumentar a eficiência da retífica. O desenvolvimento do sistema proporcionou aos estudantes uma experiência prática em integração de sistemas e modelagem de banco de dados.

2.4. Recursos utilizados no desenvolvimento

2.4.1. Hardware: Computadores para desenvolvimento e testes.

Qqsawdaw

2.4.2. Software

BRModelos (para modelagem e diagramas de entidade-relacionamento).

SQLite (para a implementação do banco de dados).

Bibliotecas e APIs: JavaFX para a interface gráfica.

Conexões JDBC para comunicação com o banco de dados.

Conexões JDBC para comunicação com o banco de dados.

Repositório: Utilização do GitHub para controle de versões e colaboração da equipe.

3. SISTEMA AUXILIAR EM JAVA PARA RETÍFICA AUTOMOTIVA

Este projeto simboliza um sistema sólido e integrado criado em Java, interligado a um banco de dados para fornecer uma solução integral para a administração de uma retífica de automóveis. O software foi desenvolvido para satisfazer as demandas específicas do negócio, simplificando a rotina diária do proprietário da retífica ao centralizar e estruturar informações vitais de forma eficaz.

O programa tem como principal característica possibilitar o registro minucioso e a gestão dos orçamentos dos clientes. Isso engloba a recolha e conservação de informações relevantes, tais como dados pessoais dos clientes, as peças requeridas para cada conserto e seus respectivos custos. Assim, o sistema não apenas aprimora a eficácia do processo orçamentário, mas também garante a disponibilidade de todas as informações pertinentes em um único local, diminuindo a chance de falhas e intensificando a transparência com os clientes.

Ademais, o software proporciona recursos extras que tornam a administração da retífica ainda mais eficaz. Dentre as vantagens, destacam-se a produção automática de relatórios financeiros, o monitoramento do histórico de clientes e a capacidade de integrar essas informações com outros sistemas administrativos da organização. Com essas habilidades, o proprietário da retífica pode obter uma visão completa e minuciosa do funcionamento da sua empresa, possibilitando decisões mais fundamentadas e estratégicas.

Em suma, este software criado em Java e integrado a um banco de dados é um recurso essencial para qualquer oficina de retífica que deseja aprimorar seus procedimentos, aprimorar o serviço ao cliente e assegurar uma administração mais eficiente e profissional de seus serviços e orçamentos.

4. MODELAGEM DE DADOS

Este projeto realizou a modelagem de dados através do software Brmodelos, que forneceu a fundação para a implementação eficiente do banco de dados no SQLite. Este procedimento de modelagem é essencial para assegurar que todas as demandas e particularidades do sistema de gestão de pedidos sejam satisfeitas de maneira eficaz e organizada.

O sistema de administração de pedidos é composto por diversas entidades, cada uma com suas próprias características. Estas entidades foram meticulosamente estabelecidas e interligadas para assegurar que todas as operações e interações no sistema possam ocorrer de forma suave e sem complicações. O diagrama contempla:

Usuario:

- Id_Usuario: INT
- Nome: VARCHAR
- Senha: VARCHAR

Orcamento:

- Id_Orcamento: NUMERIC
- Nome_Cliente: VARCHAR
- Email_Cliente: VARCHAR
- Telefone: VARCHAR
- Total: INT

Essas entidades e seus atributos são essenciais para a organização e funcionamento do sistema de gestão de pedidos, possibilitando uma administração eficiente e unificada das informações. A utilização do Brmodelos na modelagem de dados assegura uma implementação precisa do banco de dados no SQLite, simplificando a consulta, atualização e manutenção dos dados.

5. TEORIZAÇÃO DO BANCO DE DADOS

SQLite é um sistema de gerenciamento de banco de dados relacional (RDBMS) leve e autônomo, amplamente utilizado em aplicativos que necessitam de um banco de dados embutido, como aplicativos móveis, navegadores web e dispositivos embarcados.

5.1. Caracterização do SQLite:

I. Autônomo: SQLite é uma biblioteca C que implementa um banco de dados SQL embutido. Ele não requer um servidor separado para operar.
II. Leve: O tamanho do binário do SQLite é pequeno, geralmente menos de 1 MB.
III. Zero Configuração: Não há necessidade de instalação ou configuração complexa. Basta incluir a biblioteca no seu projeto e começar a usar.
IV. Transações ACID: SQLite suporta transações atômicas, consistentes, isoladas e duráveis (ACID), garantindo a integridade dos dados.
V. Multiplataforma: Funciona em várias plataformas, incluindo Windows, macOS, Linux, iOS e Android.

5.2. Teorização do SQLite:

5.2.1. Modelo Relacional

SQLite segue o modelo relacional, onde os dados são organizados em tabelas, linhas e colunas. Cada tabela possui um esquema definido que especifica os tipos de dados e as restrições.

5.2.2. SQL (Structured Query Language)

SQLite usa SQL como sua linguagem de consulta. SQL permite a definição, manipulação e consulta de dados de forma declarativa.

5.2.3. Arquitetura de Armazenamento

SQLite armazena dados em um único arquivo de banco de dados. Esse arquivo contém todas as tabelas, índices, gatilhos e visualizações do banco de dados.

5.2.4. Journaling

Para garantir a integridade dos dados, SQLite usa um mecanismo de journaling. Existem dois modos principais de journaling:

Rollback Journal: Mantém um registro das transações pendentes para permitir a reversão em caso de falha.

Write-Ahead Logging (WAL): Permite que as transações sejam registradas em um log separado antes de serem aplicadas ao banco de dados principal, melhorando o desempenho em cenários de alta concorrência.

5.2.5. Índices

SQLite suporta a criação de índices para acelerar as consultas. Índices são estruturas de dados que permitem acesso rápido aos dados com base em colunas específicas.

5.2.6. Gatilhos e Visualizações

SQLite suporta gatilhos (triggers) e visualizações (views). Gatilhos são ações automáticas executadas em resposta a eventos específicos no banco de dados, enquanto visualizações são consultas armazenadas que podem ser tratadas como tabelas.

6. IMPLEMENTAÇÃO NO SQLITE

A implementação meticulosa da modelagem de dados para este projeto no SQLite assegurou que todas as restrições de integridade e as relações entre as entidades fossem estritamente preservadas. Este procedimento garante a consistência e a integridade das informações, componentes fundamentais para a operação eficaz e segura do sistema.

6.1. Modelagem de dados

A construção do banco de dados iniciou-se com a elaboração do modelo de dados por meio do programa Brmodelos. Este programa foi crucial para criar um diagrama de entidade-relacionamento (ERD), que atuou como um guia inicial para a execução. Com o Brmodelos, as entidades e suas relações foram estabelecidas de forma precisa e transparente, simplificando a compreensão da estrutura do banco de dados.

6.2. Geração de instruções SQL

Depois de finalizar a modelagem no Brmodelos, foram geradas automaticamente as instruções SQL necessárias para a criação das tabelas e a definição das restrições de integridade. Estas diretrizes englobam comandos para a criação de tabelas, estabelecimento de chaves primárias e estrangeiras, além de definir outras limitações de integridade, tais como as normas de unicidade e os valores padrão.

6.3. Execução das instruções SQL

Depois de criar as instruções SQL, o passo subsequente foi executá-las no ambiente de desenvolvimento SQLite, usando o NetBeans. O NetBeans é um instrumento potente que facilita a gestão e a administração do SQLite de maneira intuitiva e eficaz. Com a utilização do NetBeans, todas as instruções SQL geradas pelo Brmodelos foram aplicadas, levando à formação das estruturas de banco de dados conforme o previsto.

Exemplo de algumas instruções SQL utilizadas:

```
package Classe;

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

public class CriarBancoDeDados {
    public static void main(String[] args) {
        // URL de conexão ao banco de dados SQLite
        String url = "jdbc:sqlite:C:/Users/julio/Documents/NetBeansProjects/ProjetoJava-main (1)/ProjetoJava-main/db_retifica.db";

        // Carregar explicitamente o driver JDBC do SQLite
        try {
            Class.forName("org.sqlite.JDBC");
            System.out.println("Driver carregado com sucesso!");
        } catch (ClassNotFoundException e) {
            System.out.println("Driver JDBC do SQLite não encontrado!");
            e.printStackTrace();
            return;
        }

        try (Connection conn = DriverManager.getConnection(url);
             Statement stmt = conn.createStatement()) {

            // Verificar se a tabela "usuarios" já existe
            ResultSet rsUsuarios = conn.getMetaData().getTables(null, null, "usuarios", null);
            if (!rsUsuarios.next()) { ...3 lines } else {
                System.out.println("Tabela 'usuarios' já existe.");
            }

            // Verificar se a tabela "orcamento" já existe
            ResultSet rsOrcamento = conn.getMetaData().getTables(null, null, "orcamento", null);
            if (!rsOrcamento.next()) {
                // Código SQL para criar a tabela "orcamento"
                String sqlOrcamento = "CREATE TABLE orcamento ("
                        + "idorcamento INTEGER PRIMARY KEY AUTOINCREMENT, "
                        + "nomecliente VARCHAR(50) NOT NULL, "
                        + "emailcliente VARCHAR(50) NOT NULL, "
                        + "telefonewhats VARCHAR(14) NOT NULL, "
                        + "total NUMERIC(10,2) CHECK(total > 0) NOT NULL);";
                stmt.executeUpdate(sqlOrcamento);
                System.out.println("Tabela 'orcamento' criada com sucesso!");
            } else {
                System.out.println("Tabela 'orcamento' já existe.");
            }

            // Remover as tabelas que não serão usadas
            String[] tabelasParaDrop = {"clientes", "endereco", "servico", "peca", "pecasorcamento", "servicosorcamento"};
            for (String table : tabelasParaDrop) {
                String sqlDrop = "DROP TABLE IF EXISTS " + table + ";";
                stmt.executeUpdate(sqlDrop);
            }
            System.out.println("Tabelas não utilizadas removidas com sucesso!");

        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

6.4. Implementação em Java no NetBeans

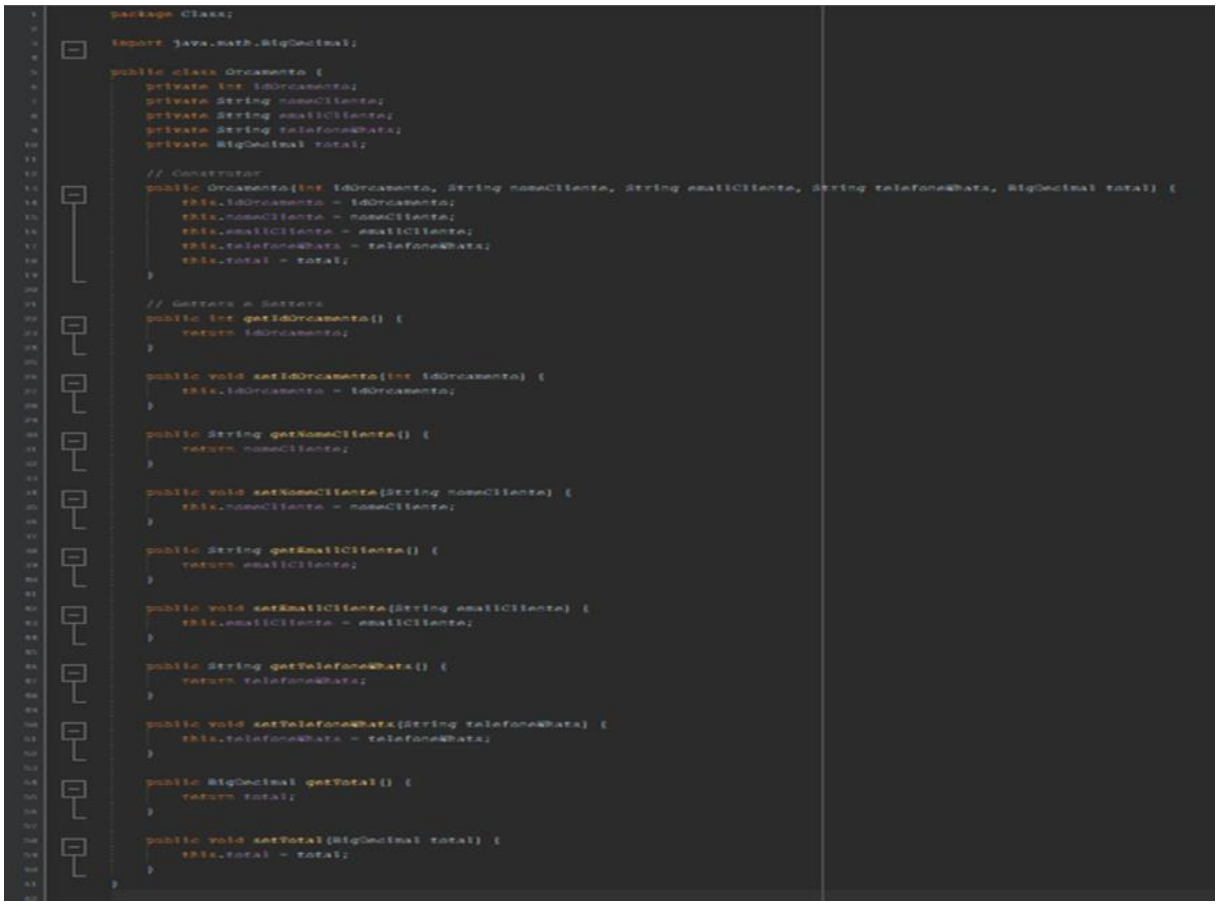
A execução da interface visual e da lógica de aplicação deste projeto foi realizada utilizando o NetBeans, um ambiente de desenvolvimento integrado (IDE) amplamente reconhecido e utilizado para programação em Java. A escolha do NetBeans como principal ferramenta para este projeto se deve à sua robustez, facilidade de uso e diversas funcionalidades que simplificam o desenvolvimento e a manutenção do código.

No projeto, utilizou-se o JavaFX para desenvolver os elementos da interface gráfica. O JavaFX é uma biblioteca padrão do Java para a criação de interfaces gráficas do usuário (GUIs) por meio de arquivos XML, permitindo uma clara separação entre o design da interface e a lógica de funcionamento do aplicativo. Com o JavaFX, foi possível criar uma interface gráfica interativa e responsiva que atende às necessidades dos usuários finais.

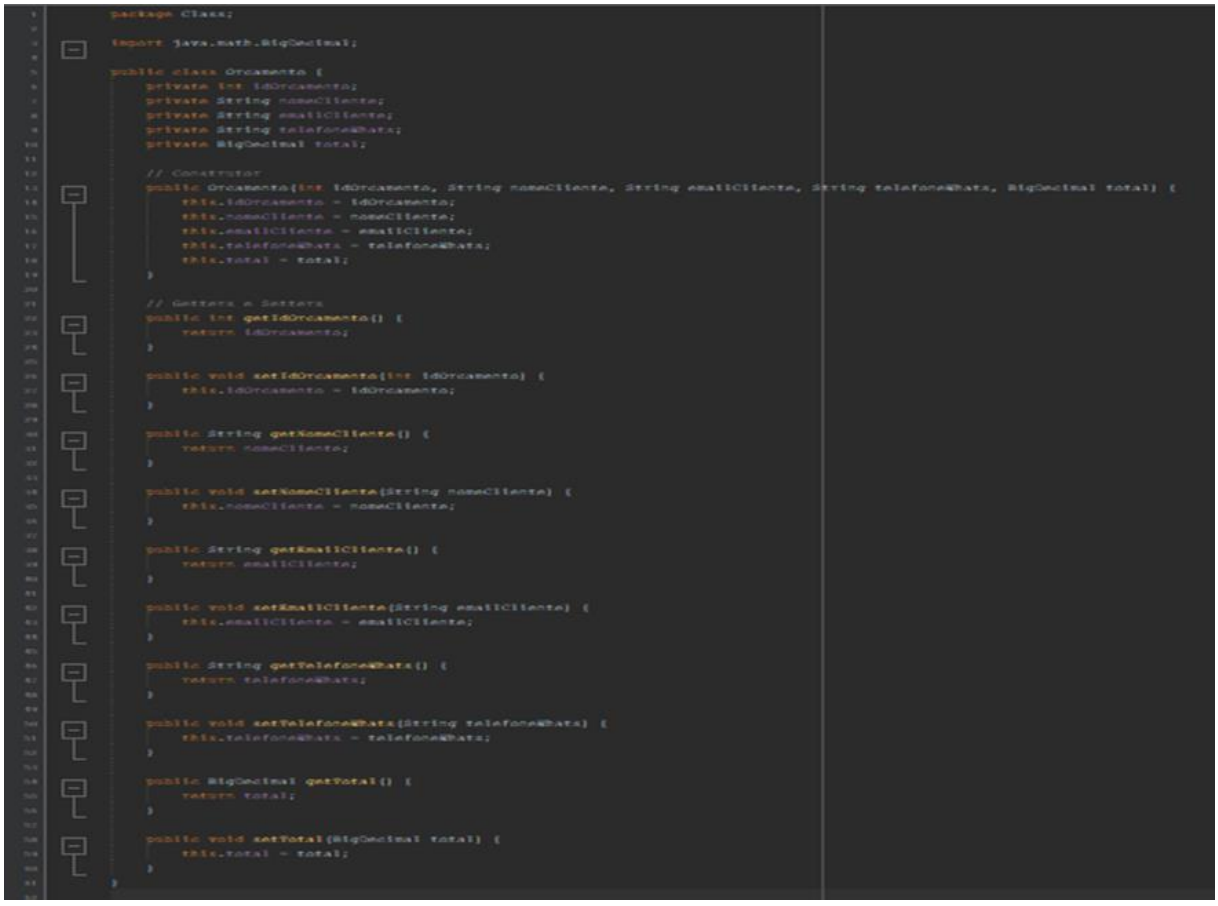
O planejamento e a organização da interface gráfica foram realizados com o objetivo de proporcionar uma experiência de usuário (UX) intuitiva e agradável. Cada componente foi cuidadosamente posicionado e organizado para garantir uma navegação fluida pelo sistema, permitindo que os usuários encontrem e utilizem as funcionalidades de maneira simples e eficiente. O layout foi projetado para ser intuitivo e estruturado, reduzindo o tempo de aprendizado e otimizando a usabilidade.

Além dos elementos visuais, também foi desenvolvida a lógica de aplicação no NetBeans, garantindo que as ações dos usuários na interface gráfica fossem corretamente interpretadas e processadas pelo sistema. Isso inclui a captura de eventos como cliques em botões, digitação de texto e seleção de opções, além de assegurar que as informações sejam transmitidas e recebidas com precisão do banco de dados SQLite.

Em resumo, o uso do NetBeans como ambiente de desenvolvimento integrado e do JavaFX para a criação da interface gráfica resultou em um sistema bem organizado, com uma interface de usuário eficaz e agradável. Este projeto demonstra como a combinação cuidadosa de ferramentas e tecnologias pode levar a um produto final de alta qualidade, que atende às expectativas dos usuários e simplifica suas interações com o sistema.



```java
package Class;

import java.math.BigDecimal;

public class Orcamento {
    private int idOrcamento;
    private String nomeCliente;
    private String emailCliente;
    private String telefoneWhats;
    private BigDecimal total;

    // Construtor
    public Orcamento(int idOrcamento, String nomeCliente, String emailCliente, String telefoneWhats, BigDecimal total) {
        this.idOrcamento = idOrcamento;
        this.nomeCliente = nomeCliente;
        this.emailCliente = emailCliente;
        this.telefoneWhats = telefoneWhats;
        this.total = total;
    }

    // Getters e Setters
    public int getIdOrcamento() {
        return idOrcamento;
    }

    public void setIdOrcamento(int idOrcamento) {
        this.idOrcamento = idOrcamento;
    }

    public String getNomeCliente() {
        return nomeCliente;
    }

    public void setNomeCliente(String nomeCliente) {
        this.nomeCliente = nomeCliente;
    }

    public String getEmailCliente() {
        return emailCliente;
    }

    public void setEmailCliente(String emailCliente) {
        this.emailCliente = emailCliente;
    }

    public String getTelefoneWhats() {
        return telefoneWhats;
    }

    public void setTelefoneWhats(String telefoneWhats) {
        this.telefoneWhats = telefoneWhats;
    }

    public BigDecimal getTotal() {
        return total;
    }

    public void setTotal(BigDecimal total) {
        this.total = total;
    }
}
```



```java
package Class;

import java.math.BigDecimal;

public class Orcamento {
    private int idOrcamento;
    private String nomeCliente;
    private String emailCliente;
    private String telefoneWhats;
    private BigDecimal total;

    // Construtor
    public Orcamento(int idOrcamento, String nomeCliente, String emailCliente, String telefoneWhats, BigDecimal total) {
        this.idOrcamento = idOrcamento;
        this.nomeCliente = nomeCliente;
        this.emailCliente = emailCliente;
        this.telefoneWhats = telefoneWhats;
        this.total = total;
    }

    // Getters e Setters
    public int getIdOrcamento() {
        return idOrcamento;
    }

    public void setIdOrcamento(int idOrcamento) {
        this.idOrcamento = idOrcamento;
    }

    public String getNomeCliente() {
        return nomeCliente;
    }

    public void setNomeCliente(String nomeCliente) {
        this.nomeCliente = nomeCliente;
    }

    public String getEmailCliente() {
        return emailCliente;
    }

    public void setEmailCliente(String emailCliente) {
        this.emailCliente = emailCliente;
    }

    public String getTelefoneWhats() {
        return telefoneWhats;
    }

    public void setTelefoneWhats(String telefoneWhats) {
        this.telefoneWhats = telefoneWhats;
    }

    public BigDecimal getTotal() {
        return total;
    }

    public void setTotal(BigDecimal total) {
        this.total = total;
    }
}
```

7. BENEFÍCIOS

O principal propósito da solução criada é simplificar a gestão de orçamentos e acelerar o processo de atendimento na retífica de automóveis. Este sistema foi concebido para proporcionar uma estratégia segura e prática na gestão dos registros, contribuindo diretamente para as decisões estratégicas e operacionais da empresa.

O sistema possui recursos sólidos que foram meticulosamente implementados para satisfazer as demandas específicas da retífica. Dentre essas características, destaca-se a habilidade de organizar e controlar orçamentos de clientes de maneira unificada, mantendo todas as informações cruciais, tais como dados dos clientes, detalhes dos serviços solicitados, peças empregadas e custos associados em um único local. Isso não apenas simplifica o acesso a essas informações, mas também aprimora a exatidão dos registros, diminuindo consideravelmente a chance de falhas.

A interface do sistema foi concebida pensando no usuário, proporcionando uma experiência intuitiva e prazerosa. Por meio de uma interface intuitiva, os usuários conseguem se movimentar facilmente pelo sistema, localizando rapidamente as funcionalidades que necessitam e executando suas atividades de forma eficaz. Este projeto intuitivo favorece um aprendizado mais rápido, possibilitando que novos usuários se adaptem rapidamente à utilização do sistema.

8. GitHub

Todo conteudo dentro do repositório GitHub:
https://github.com/ProjetoJav/ProjetoJava

9. CARTA DE AUTORIZAÇÃO

CARTA DE AUTORIZAÇÃO

Eu, Jefferson Ascava Nespoli, Administrador/Proprietário da empresa Removel retifica de motores LTDA, situada no endereço, Rua onze de junho, N°155/165, Casa Branca – Santo André/SP autorizo a realização das seguintes atividades acadêmicas extensionistas associada a disciplinas, da Universidade Estácio - Unidade Santo André - Centro - SP, sob orientação do Prof. Paulo Daniel da Cruz:

Atividades: Proposta para o desenvolvimento de um sistema operante, construído por Programação Orientada em JAVA e Banco de Dados, com o propósito de auxiliar na venda e orçamento. A atividade será feita de acordo com o pedido do cliente/autorizador engajado no projeto e do que foi acordado com os alunos.

Conforme combinado em contato prévio, as atividades acima descritas são autorizadas para os seguintes alunos:

Nome do Aluno	Curso
Dennis Araujo Serafim Dias	Análise e Desenvolvimento de Sistemas
Eduardo de Assis Vallini	Análise e Desenvolvimento de Sistemas
Julio Francisco Bernardino	Análise e Desenvolvimento de Sistemas
Victor Luiz Umeki Pozato	Análise e Desenvolvimento de Sistemas

Declaro que fui informado por meio da **Carta de Apresentação** sobre as características e objetivos das atividades que serão realizadas na organização a qual represento e afirmo estar ciente de tratar-se de uma atividade realizada com intuito exclusivo de ensino de alunos de graduação, sem a finalidade de exercício profissional.

Desta forma, autorizo:

☑ o acesso a informações e dados que forem necessários à execução da atividade;

☒ o registro de imagem por meio de fotografias;

Santo André/SP 26 de Setembro de 2024.

REMOVEL Retifica de Motores Ltda