Felipe Barbour, Guilherme Menezes Grizão e Weslley de Santana Moreira

Sumário

Projeto

O Projeto consiste em um site que irá auxiliar o cliente, oferecendo uma solução, concedendo autonomia e até mesmo facilitando o acesso a pessoas que não saibam descrever totalmente seu problema. Será incrementado um site com uma aba de Login/Cadastro, onde o cliente poderá ter um cadastro, ou se cadastrar. Após o cadastro, o cliente irá se deparar com uma estrutura totalmente voltada para a resolução de seu problema. Com o apoio de uma IA (Inteligência Artificial) o site fará no máximo dois pré-diagnósticos do problema, para serem solucionados em uma oficina mecânica, e oferecerá algumas opções de “reparo” no local, caso o problema do cliente possa ser solucionado de forma fácil pelo próprio cliente. Dado o diagnóstico do problema do cliente, a IA (Inteligência Artificial) fará uma estimativa de valor em que o cliente pagará, após isso, irá localizar uma oficina mecânica parceira mais próxima para o melhor deslocamento do cliente. Junto a IA (Inteligência Artificial), haverá um campo para que o cliente possa descrever seu problema da melhor forma possível. Após o pré-diagnóstico do cliente, será feito um PDF com todas as informações necessárias como: problema, diagnóstico e estimativa de valor. Esse PDF será encaminhado tanto para o cliente quanto para a oficina mecânica parceira escolhida pelo cliente.

Justificativa

O Projeto tem como objetivo facilitar a descrição do problema do cliente utilizando meios que o auxiliem em todo o tempo. Usando uma Inteligência Artificial o problema do cliente será solucionado de uma forma mais fácil e eficiente, onde, o cliente possa entrar em detalhes do seu problema facilitando o entendimento da Inteligência Artificial e facilitando o diagnóstico de seu problema. Com um site totalmente voltado a uma estrutura de resolução do problema e solicitando um fácil acesso ao cliente, haverá praticidade, autonomia e acessibilidade.

Tabela dos EndPoints

Marca

- Listar Marcas: GET - http://localhost:8080/isolutions/marcas

Problema

- Listar Problemas: GET - http://localhost:8080/isolutions/problema
- Atualizar Problema: PUT - http://localhost:8080/isolutions/problema/1
- Criar Problema: POST - http://localhost:8080/isolutions/problema
- Buscar por Id: GET - http://localhost:8080/isolutions/problema/1

Veículo

- Listar Veículos: GET - http://localhost:8080/isolutions/veiculo
- Atualizar Veículo: PUT - http://localhost:8080/isolutions/veiculo/1
- Criar Veículo: POST - http://localhost:8080/isolutions/veiculo
- Buscar veículo por IdCliente: GET - http://localhost:8080/isolutions/veiculo/1
- Remover Veículo: DELETE - http://localhost:8080/isolutions/veiculo/1

Cliente

- Listar Clientes: GET - http://localhost:8080/isolutions/cliente
- Atualizar Cliente: PUT - http://localhost:8080/isolutions/cliente/1
- Buscar por CPF: GET - http://localhost:8080/isolutions/cliente/32165498700
- Criar Cliente: POST - http://localhost:8080/isolutions/cliente
- Deletar Cliente: DELETE - http://localhost:8080/isolutions/cliente/1
- Realizar Login: GET - http://localhost:8080/isolutions/cliente/login?cpf=98765432100&senha=senha456

Resumo da aplicação JAVA

Estrutura da Aplicação

- **DAO:** As classes DAO (ClienteDao, MarcaDao, ProblemaDao, VeiculoDao) são responsáveis por acessar e manipular dados no banco de dados para cada entidade específica (Cliente, Marca, Problema e Veículo). Cada DAO implementa métodos que realizam operações CRUD, como busca, inserção, atualização e remoção de dados.

- **Model:** As classes de modelo, localizadas na pasta model, representam as tabelas no banco de dados e possuem atributos correspondentes aos campos dessas tabelas. Os principais modelos incluem: Cliente, Marca, Problema e Veiculo

- **Recursos da API REST**: As classes ClienteResource, MarcaResource, ProblemaResource e VeiculoResource definem os endpoints REST para cada entidade. Esses recursos permitem que clientes enviem requisições HTTP (GET, POST, PUT, DELETE) para manipular os dados no banco. Cada endpoint é configurado para responder com JSON, facilitando a integração com front-ends ou outras aplicações.

- **Conexão com o Banco de Dados**: A classe ConexaoBanco gerencia a conexão com o banco de dados, utilizando JDBC. Ela fornece métodos para abrir e fechar conexões, e as DAOs a utilizam para interagir diretamente com o banco.

- **Filtros e Utilitários**:
  - CorsFilter: Implementa políticas de CORS, permitindo que a API seja consumida por clientes hospedados em diferentes origens.
  - ErrorResponse: Define um padrão de resposta de erro, garantindo que respostas de erro da API sejam consistentes e informativas.

- **Script SQL**: O arquivo script.sql contém comandos SQL que criam e populam as tabelas no banco de dados, estabelecendo a estrutura necessária para que a aplicação funcione corretamente. Ele define tabelas e seus relacionamentos, incluindo Cliente, Veiculo, Marca e Problema.

- **Main**: A aplicação inclui uma classe Main, usada para testar funcionalidades principais ou executar a aplicação em um ambiente local.

Funcionamento Geral da Aplicação

Essa aplicação permite que sistemas externos acessem e manipulem informações sobre veículos, problemas mecânicos, clientes e marcas. Utilizando endpoints REST, cada operação realizada sobre os dados (como consultar um veículo específico ou cadastrar um novo problema) é traduzida em comandos SQL que interagem com o banco de dados através das DAOs.

A implementação do CORS e o uso de padrões de resposta de erro facilitam a integração com front-ends e garantem uma experiência de uso consistente. A configuração da conexão com o banco de dados permite que a aplicação seja escalável e capaz de lidar com operações complexas de forma eficiente.