Documentação (1)

API Automóveis

API para gerenciar carros, vendedores, clientes, vendas e revisões de automóveis.

Sumário

Pré-requisitos

- **Java 17** ou superior
- **Maven** (para gerenciamento de dependências)
- **PostgreSQL** (banco de dados)

Instalação

1. **Clonar o repositório**:

   ```
   git clone <URL_DO_REPOSITORIO_GIT>
   cd api_automoveis
   ```

2. **Instalar dependências**:

   ```
   mvn clean install
   ```

Configuração do Banco de Dados

1. Inicie o PostgreSQL e crie o banco de dados:

```
CREATE DATABASE api_automoveis;
```

2. (Opcional) Crie um usuário específico para o projeto:

```
CREATE USER api_automoveis_user WITH PASSWORD 'sua_senh
a';
ALTER DATABASE api_automoveis OWNER TO api_automoveis_us
er;
```

3. Configure as credenciais no arquivo `src/main/resources/application.properties`:

```
spring.datasource.url=jdbc:postgresql://localhost:5432/a
pi_automoveis
spring.datasource.username=api_automoveis_user
spring.datasource.password=sua_senha
```

Execução

Para rodar o projeto, execute o seguinte comando no terminal:

```
mvn spring-boot:run
```

A aplicação estará disponível em: `http://localhost:8080`.

Ordem de Requisições

Para evitar erros de relacionamento ao usar a API, siga esta ordem para as requisições POST:

1. **Vendedor**:
   - Crie os vendedores primeiro, pois eles estarão associados às vendas.

```
{
  "nome": "Carlos Vendas",
  "cnpj": "12345678000123",
```

```
  "email": "carlos.vendas@example.com",
  "telefone": "(11) 98765-4321",
  "endereco": "Rua das Vendas, 123"
}
```

2. **Cliente**:
   - Crie os clientes, que estarão associados às vendas.

```
{
  "nome": "Maria Oliveira",
  "cpf": "12345678901",
  "email": "maria.oliveira@example.com",
  "telefone": "(21) 91234-5678",
  "endereco": "Avenida Principal, 456"
}
```

3. **Carro**:
   - Crie os carros antes das vendas e revisões.

```
{
  "modelo": "Civic",
  "marca": "Honda",
  "ano": 2022
}
```

4. **Venda**:
   - Registre as vendas associando carro, cliente e vendedor.

```
{
  "carroId": 1,
  "clienteId": 1,
  "vendedorId": 1,
  "dataVenda": "2024-11-22",
  "valorVenda": 75000.00
}
```

5. **Revisão**:

- Registre revisões para os carros.

```
{
  "descricao": "Troca de óleo e filtros",
  "dataRevisao": "2024-12-01",
  "carroId": 1
}
```

Testes no Postman

1. Importe a documentação da API no Postman usando a URL do OpenAPI (Swagger):

   ```
   <http://localhost:8080/api-docs.json>
   ```

2. Configure a variável `{{base_url}}` como `http://localhost:8080` para facilitar o uso dos endpoints.
3. Execute as requisições seguindo a ordem definida acima para evitar erros de relacionamento.

Documentação

A documentação completa da API é gerada automaticamente pelo Swagger e pode ser acessada em:

```
<http://localhost:8080/docs>
```
