

1. Visão Geral do Projeto

Este projeto é uma aplicação de gestão de estacionamento, onde veículos são registrados, controlados e monitorados através de uma interface web. Ele utiliza Firebase Firestore para armazenar os dados dos veículos e interage com a API **SheetDB** para enviar relatórios diários.

Funcionalidades Principais:

- **Cadastro de veículos:** Registra veículos com informações como placa, tipo, proprietário e horário de entrada.
- **Exclusão de veículos:** Permite a exclusão de veículos, movendo-os para a coleção "saida" antes de excluí-los da coleção "entrada".
- **Relatório diário:** Envia todos os dados da coleção "saida" para a API **SheetDB** e deleta os dados da coleção.
- **Cálculo do valor de estacionamento:** Calcula o valor a ser pago pelos veículos com base no tipo de veículo.
- **Exibição de veículos:** Carrega e exibe os veículos registrados no estacionamento em tempo real.

2. Tecnologias Utilizadas

- **Frontend:**
 - HTML, CSS e JavaScript para a interface do usuário.
 - **Fetch API** para comunicação com a API de backend.
- **Backend:**
 - **Node.js** com o framework **Express** para a criação da API.
 - **Firebase Firestore** como banco de dados.
 - **SheetDB API** para integração de dados com planilhas Google.

3. Endpoints da API

3.1 POST /api/entrada

- **Descrição:** Registra um veículo na coleção "entrada".
- **Requisição:**
 - **Body:**

```
json
Copiar código
{
  "placa": "ABC1234",
  "dono": "João Silva",
  "tipo": "carro",
  "entrada": "2024-11-11T10:00:00Z",
  "valor": 15.0
}
```

- **Resposta:**

```
json
Copiar código
{
  "id": "12345",
  "message": "Veículo cadastrado com sucesso"
}
```

3.2 GET /api/entrada

- **Descrição:** Retorna todos os veículos da coleção "entrada".
- **Resposta:**

```
json
Copiar código
[
  {
    "id": "12345",
    "placa": "ABC1234",
    "dono": "João Silva",
    "tipo": "carro",
    "entrada": "2024-11-11T10:00:00Z",
    "valor": 15.0
  }
]
```

3.3 GET /api/saida

- **Descrição:** Retorna todos os veículos da coleção "saida".
- **Resposta:**

```
json
Copiar código
[
  {
    "id": "54321",
    "placa": "XYZ9876",
    "dono": "Maria Oliveira",
    "tipo": "moto",
    "entrada": "2024-11-11T09:00:00Z",
    "saida": "2024-11-11T12:00:00Z",
    "valor": 10.0
  }
]
```

3.4 POST /api/saida

- **Descrição:** Envia dados de um veículo para a coleção "saida".
- **Requisição:**
 - **Body:**

```
json
Copiar código
{
  "placa": "XYZ9876",
  "dono": "Maria Oliveira",
  "tipo": "moto",
  "entrada": "2024-11-11T09:00:00Z",
}
```

```
    "saida": "2024-11-11T12:00:00Z",  
    "valor": 10.0  
  }
```

3.5 DELETE /api/entrada/

- **Descrição:** Exclui um veículo da coleção "entrada".
- **Parâmetro:** id (ID do veículo a ser excluído).
- **Resposta:**

```
json  
Copiar código  
{  
  "message": "Veículo excluído com sucesso"  
}
```

3.6 DELETE /api/saida

- **Descrição:** Deleta todos os documentos da coleção "saida".
- **Resposta:**

```
json  
Copiar código  
{  
  "message": "Todos os veículos da coleção 'saida' foram  
excluídos"  
}
```

3.7 REPORT <https://sheetdb.io/api/v1/{endpoint}> e DELETE /api/saida

- **Descrição:** Envia todos os dados da coleção "saida" para o SheetDB e apaga os dados da coleção "saida".
- **Resposta:**

```
json  
Copiar código  
{  
  "message": "Relatório enviado com sucesso para o SheetDB e  
dados excluídos"  
}
```

4. Funcionalidades no Frontend

4.1 Cadastro de Veículos

- O usuário pode preencher um formulário para registrar um veículo, informando os seguintes campos:
 - **Placa**
 - **Proprietário**
 - **Tipo de veículo**

Ao clicar em "Cadastrar", as informações são enviadas para a rota POST /api/entrada.

4.2 Exclusão de Veículos

- O usuário pode excluir um veículo, o que moverá o veículo para a coleção `saida` e, em seguida, excluirá o registro da coleção `entrada`.

4.3 Relatório Diário

- Ao clicar no botão de "Relatório Diário", todos os dados da coleção `saida` serão enviados para o **SheetDB**, e os dados da coleção `saida` serão excluídos no Firestore.

4.4 Tabela de Veículos

- Exibe todos os veículos da coleção "entrada" ou "saida" em uma tabela HTML. A tabela contém as informações de:
 - **Placa**
 - **Tipo**
 - **Proprietário**
 - **Hora de Entrada**
 - **Hora de Saída**
 - **Valor (R\$)**

Quando um veículo é excluído, ele é removido da tabela e adicionado à coleção "saida".

4.5 Cálculo do Valor

- O valor do estacionamento é calculado com base no tipo do veículo, exemplo:
 - Carro: R\$4,00
 - Moto: R\$2,00
 - caminhonete: R\$6,00

O valor é enviado ao Firestore junto com os dados do veículo.

4.6 Exibição do Valor Total

- O valor total do dia (soma dos valores dos veículos registrados) é exibido dinamicamente em uma área específica da interface.

5. Estrutura do Projeto

```
.
├── app
│   ├── assets
│   │   ├── css
│   │   └── js
│   ├── index.html
│   ├── diario.html
│   ├── login.html
│   └── procurar.html
└── config
```

```
| | routes.js
| data
| | configurações.js
| | usuário.js
| doc
| | documentação.js
| .gitignore
| firestore.js
| main.js
| package-lock.json
| package.json
```

- **index.html:** A página principal do frontend.
- **css:** Arquivos CSS para estilizar a página.
- **js:** Arquivos JavaScript contendo a lógica de interação com a API.
- **routes.js:** Arquivo que contém as rotas do servidor.
- **firestore.js:** Arquivo principal do servidor que configura o Express e conecta ao Firestore.
- **package.json:** Arquivo de configuração do Node.js que inclui dependências.

6. Instalação

6.1 Pré-requisitos

- Node.js e npm instalados.
- Conta Firebase configurada com Firestore.
- Conta no **SheetDB**.

6.2 Instalação do Backend

1. Clone o repositório do projeto.
2. Instale as dependências:

```
bash
Copiar código
npm install
```

3. Configure as credenciais do Firebase no arquivo `firestore.js`.
4. Execute o servidor:

```
bash
Copiar código
node firestore.js
```

6.3 Instalação do Frontend

1. Abra o arquivo `index.html` em um navegador.
2. Certifique-se de que o backend esteja em execução para que as requisições funcionem corretamente.

7. Considerações Finais

Este projeto oferece uma forma simples de gerenciar veículos em um estacionamento, com integração com o Firebase para armazenamento e o SheetDB para geração de relatórios. A API RESTful foi construída com o Express e utiliza Firestore para persistência de dados.