condutor.client.ts U

estacionamento-client > src > client > condutor.client.ts > CondutorCliente > constructor

```ts
import axios, { AxiosInstance } from "axios";

import { Condutor } from "@/model/condutor";
import { PageRequest } from "@/model/page/page-request";
import { PageResponse } from "@/model/page/page-response";

export class CondutorCliente {

    private axiosClient: AxiosInstance;

    constructor(){
        this.axiosClient = axios.create({
            baseURL: 'http://localhost:8081/estacionamento',
            headers: {'Content-type' : 'application/json'}
        });
    }

    public async findById(id : number) : Promise<Condutor> {
        try {
            return (await this.axiosClient.get<Condutor>(`/${id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async cadastrar(condutor : Condutor) : Promise<void> {
        try {
            return (await this.axiosClient.post('/', condutor))
        } catch (error : any) {
            return Promise.reject(error.response)
        }

    }

    public async editar(condutor : Condutor) : Promise<void> {
        try {
            return (await this.axiosClient.put(`/${condutor.id}`, condutor)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async deletar(condutor : Condutor) : Promise<void> {
        try {
            return (await this.axiosClient.delete(`/${condutor.id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async findByFiltrosPaginado(pageRequest : PageRequest) : Promise<PageResponse<Condutor>> {
        try {
            let requestPath = ''

            requestPath += `?page=${pageRequest.currentPage}`
            requestPath += `&size=${pageRequest.pageSize}`
            requestPath += `&sort=${pageRequest.sortField === undefined ? '' : pageRequest.sortField}, ${pageRequest.direction}`

            return (await this.axiosClient.get<PageResponse<Condutor>>(requestPath, {params : {filtros : pageRequest.filter } })).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }

    }
}
```

configuracao.client.ts U

estacionamento-client > src > client > configuracao.client.ts > ConfiguracaoClient

```ts
import axios, { AxiosInstance } from "axios";

import { Configuracao } from "@/model/configuracao";
import { PageRequest } from "@/model/page/page-request";
import { PageResponse } from "@/model/page/page-response";

export class ConfiguracaoClient {

    private axiosClient : AxiosInstance

    constructor(){
        this.axiosClient = axios.create({
            baseURL: 'http://localhost:8080/api/configuracao',
            headers: {'Content-Type' : 'application/json'}
        });
    }

    public async findById(id : number) : Promise<Configuracao> {
        try {
            return (await this.axiosClient.get<Configuracao>(`/${id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async cadastrar(configuracao : Configuracao) : Promise<void> {
        try {
            return (await this.axiosClient.post('/', configuracao))
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async editar(configuracao : Configuracao) : Promise<void> {
        try {
            return (await this.axiosClient.put(`/${configuracao.id}`, configuracao)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async findByFiltrosPaginado(pageRequest : PageRequest) : Promise<PageResponse<Configuracao>> {
        try {
            let requestPath = ''

            requestPath += `?page=${pageRequest.currentPage}`
            requestPath += `&size=${pageRequest.pageSize}`
            requestPath += `&sort=${pageRequest.sortField === undefined ? '' : pageRequest.sortField}, ${pageRequest.direction}`

            return (await this.axiosClient.get<PageResponse<Configuracao>>(requestPath, {params : {filtros : pageRequest.filter} })).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }
}
```

marca.client.ts U

estacionamento-client > src > client > marca.client.ts > ...

```ts
import axios, { AxiosInstance } from "axios";

import { Marca } from "@/model/marca";
import { PageRequest } from "@/model/page/page-request";
import { PageResponse } from "@/model/page/page-response";

export class MarcaClient {

    private axiosClient : AxiosInstance

    constructor(){
        this.axiosClient = axios.create({
            baseURL: 'http://localhost:8080/api/marca',
            headers: {'Content-Type' : 'application/json'}
        });
    }

    public async findById(id : number) : Promise<Marca> {
        try {
            return (await this.axiosClient.get<Marca>(`/${id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async listAll() : Promise<Marca[]> {
        try {
            return (await this.axiosClient.get<Marca[]>(`/lista`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async findByAtivo() : Promise<Marca[]> {
        try {
            return (await this.axiosClient.get<Marca[]>(`/ativo`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async cadastrar(marca : Marca) : Promise<void> {
        try {
            return (await this.axiosClient.post('/', marca))
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async editar(marca : Marca) : Promise<void> {
        try {
            return (await this.axiosClient.post(`/${marca.id}`, marca)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async deletar(marca : Marca) : Promise<string> {
        try {
            return (await this.axiosClient.delete(`/${marca.id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async findByFiltrosPaginado(pageRequest : PageRequest) : Promise<PageResponse<Marca>> {
```

modelo.client.ts U

estacionamento-client > src > client > modelo.client.ts > ...

```ts
import axios, { AxiosInstance } from "axios";

import { Modelo } from "@/model/modelo";
import { PageRequest } from "@/model/page/page-request";
import { PageResponse } from "@/model/page/page-response";

export class ModeloClient {

    private axiosClient : AxiosInstance;

    constructor(){
        this.axiosClient = axios.create({
            baseURL: 'http://localhost:8080/api/modelo',
            headers: {'Content-Type' : 'application/json'}
        });
    }

    public async findById(id : number) : Promise<Modelo> {
        try {
            return (await this.axiosClient.get<Modelo>(`/${id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async listAll() : Promise<Modelo[]> {
        try {
            return (await this.axiosClient.get<Modelo[]>(`/lista`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async findByAtivo() : Promise<Modelo[]> {
        try {
            return (await this.axiosClient.get<Modelo[]>(`/ativo`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async cadastrar(modelo : Modelo) : Promise<void> {
        try {
            return (await this.axiosClient.post('/', modelo))
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async editar(modelo : Modelo) : Promise<void> {
        try {
            return (await this.axiosClient.put(`/${modelo.id}`, modelo)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async findByFiltrosPaginado(pageRequest : PageRequest) : Promise<PageResponse<Modelo>> {
        try {
            let requestPath = ''

            requestPath += `?page=${pageRequest.currentPage}`
            requestPath += `&size=${pageRequest.pageSize}`
            requestPath += `&sort=${pageRequest.sortField === undefined ? '' : pageRequest.sortField}, ${pageRequest.direction}`

            return (await this.axiosClient.get<PageResponse<Modelo>>(requestPath, {params : {filtros : pageRequest.filter } })).data
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

node

```
App running at:
- Local:   http://localhost:8080/
- Network: http://192.168.100.103:8080/

No issues found.
```

movimentacao.client.ts U

estacionamento-client > src > client > movimentacao.client.ts > ...

```ts
import axios, { AxiosInstance } from "axios";

import { Movimentacao } from "@/model/movimentacao";
import { PageRequest } from "@/model/page/page-request";
import { PageResponse } from "@/model/page/page-response";

export class MovimentacaoClient {

    private axiosClient : AxiosInstance;

    constructor(){
        this.axiosClient = axios.create({
            baseURL: 'http://localhost:8080/api/movimentacao',
            headers: {'Content-type' : 'application/json'}
        });
    }

    public async findById(id : number) : Promise<Movimentacao> {
        try {
            return (await this.axiosClient.get<Movimentacao>(`/${id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async listAll() : Promise<Movimentacao[]> {
        try {
            return (await this.axiosClient.get<Movimentacao[]>(`/lista`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async findByAbertas() : Promise<Movimentacao[]> {
        try {
            return (await this.axiosClient.get<Movimentacao[]>(`/ativo`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async cadastrar(movimentacao : Movimentacao) : Promise<void> {
        try {
            return (await this.axiosClient.post('/', movimentacao))
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async editar(movimentacao : Movimentacao) : Promise<void> {
        try {
            return (await this.axiosClient.put(`/${movimentacao.id}`, movimentacao)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async fecharMovimentacao(movimentacao : Movimentacao) : Promise<void> {
        try {
            return (await this.axiosClient.put(`/saida/${movimentacao.id}`, movimentacao)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async deletar(movimentacao : Movimentacao) : Promise<string> {
```

veiculo.client.ts U

estacionamento-client > src > client > veiculo.client.ts > ...

```ts
import axios, { AxiosInstance } from "axios";

import { Veiculo } from "@/model/veiculo";
import { PageRequest } from "@/model/page/page-request";
import { PageResponse } from "@/model/page/page-response";
import { Movimentacao } from "@/model/movimentacao";

export class VeiculoClient {

    private axiosClient : AxiosInstance;

    constructor(){
        this.axiosClient = axios.create({
            baseURL: 'http://localhost:8080/api/veiculo',
            headers: {'Content-Type' : 'application/json'}
        });
    }

    public async findById(id : number) : Promise<Veiculo> {
        try {
            return (await this.axiosClient.get<Veiculo>(`/${id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async listAll() : Promise<Veiculo[]> {
        try {
            return (await this.axiosClient.get<Veiculo[]>(`/lista`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async findByAtivo() : Promise<Veiculo[]> {
        try {
            return (await this.axiosClient.get<Veiculo[]>(`/ativo`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async cadastrar(veiculo : Veiculo) : Promise<void> {
        try {
            return (await this.axiosClient.post('/', veiculo))
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async editar(veiculo : Veiculo) : Promise<void> {
        try {
            return (await this.axiosClient.put(`/${veiculo.id}`, veiculo)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }

    public async deletar(veiculo : Veiculo) : Promise<string> {
        try {
            return (await this.axiosClient.delete(`/${veiculo.id}`)).data
        } catch (error : any) {
            return Promise.reject(error.response)
        }
    }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
App running at:
- Local:   http://localhost:8080/
- Network: http://192.168.100.103:8080/

No issues found.
```