

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

```
1 import axios, { AxiosInstance } from "axios";
2
3 import { Condutor } from "@model/condutor";
4 import { PageRequest } from "@model/page/page-request";
5 import { PageResponse } from "@model/page/page-response";
6
7 export class CondutorCliente {
8
9     private axiosClient: AxiosInstance;
10
11     constructor() {
12         this.axiosClient = axios.create({
13             baseURL: 'http://localhost:8081/estacionamento',
14             headers: { 'Content-type' : 'application/json' }
15         });
16     }
17
18     public async findById(id : number) : Promise<Condutor> {
19         try {
20             return (await this.axiosClient.get<Condutor>(`/${id}`)).data
21         } catch (error : any) {
22             return Promise.reject(error.response)
23         }
24     }
25
26     public async cadastrar(condutor : Condutor) : Promise<void> {
27         try {
28             return (await this.axiosClient.post('/', condutor))
29         } catch (error : any) {
30             return Promise.reject(error.response)
31         }
32     }
33
34
35     public async editar(condutor : Condutor) : Promise<void> {
36         try {
37             return (await this.axiosClient.put(`/${condutor.id}`, condutor)).data
38         } catch (error : any) {
39             return Promise.reject(error.response)
40         }
41     }
42
43     public async deletar(condutor : Condutor) : Promise<void> {
44         try {
45             return (await this.axiosClient.delete(`/${condutor.id}`)).data
46         } catch (error : any) {
47             return Promise.reject(error.response)
48         }
49     }
50
51     public async findByFiltrosPaginado(pageRequest : PageRequest) : Promise<PageResponse<Condutor>> {
52         try {
53             let requestPath = ''
54
55             requestPath += `?page=${pageRequest.currentPage}`
56             requestPath += `&size=${pageRequest.pageSize}`
57             requestPath += `&sort=${pageRequest.sortField === undefined ? '' : pageRequest.sortField}, ${pageRequest.direction}`
58
59             return (await this.axiosClient.get<PageResponse<Condutor>>(requestPath, {params : {filtros : pageRequest.filter} })).data
60         } catch (error : any) {
61             return Promise.reject(error.response)
62         }
63     }
64
65 }
```



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

```
1 import axios, { AxiosInstance } from "axios";
2
3 import { Configuracao } from "@model/configuracao";
4 import { PageRequest } from "@model/page/page-request";
5 import { PageResponse } from "@model/page/page-response";
6
7 export class ConfiguracaoClient {
8
9     private axiosClient : AxiosInstance
10
11     constructor(){
12         this.axiosClient = axios.create({
13             baseURL: 'http://localhost:8080/api/configuracao',
14             headers: { 'Content-Type' : 'application/json' }
15         });
16     }
17
18     public async findById(id : number) : Promise<Configuracao> {
19         try {
20             return (await this.axiosClient.get<Configuracao>(`/${id}`)).data
21         } catch (error : any) {
22             return Promise.reject(error.response)
23         }
24     }
25
26     public async cadastrar(configuracao : Configuracao) : Promise<void> {
27         try {
28             return (await this.axiosClient.post('/', configuracao))
29         } catch (error : any) {
30             return Promise.reject(error.response)
31         }
32     }
33
34
35     public async editar(configuracao : Configuracao) : Promise<void> {
36         try {
37             return (await this.axiosClient.put(`/${configuracao.id}`, configuracao)).data
38         } catch (error : any) {
39             return Promise.reject(error.response)
40         }
41     }
42
43     public async findByFiltrosPaginado(pageRequest : PageRequest) : Promise<PageResponse<Configuracao>> {
44         try {
45             let requestPath = ''
46
47             requestPath += `?page=${pageRequest.currentPage}`
48             requestPath += `&size=${pageRequest.pageSize}`
49             requestPath += `&sort=${pageRequest.sortField === undefined ? '' : pageRequest.sortField}, ${pageRequest.direction}`
50
51             return (await this.axiosClient.get<PageResponse<Configuracao>>(requestPath, {params : {filtros : pageRequest.filter} })).data
52         } catch (error : any) {
53             return Promise.reject(error.response)
54         }
55     }
56 }
```



TS marca.client.ts U X

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

```
1 import axios, { AxiosInstance } from "axios";
2
3 import { Marca } from "@model/marca";
4 import { PageRequest } from "@model/page/page-request";
5 import { PageResponse } from "@model/page/page-response";
6
7 export class MarcaClient {
8
9     private axiosClient : AxiosInstance
10
11     constructor(){
12         this.axiosClient = axios.create({
13             baseURL: 'http://localhost:8080/api/marca',
14             headers: {'Content-Type' : 'application/json'}
15         });
16     }
17
18     public async findById(id : number) : Promise<Marca> {
19         try {
20             return (await this.axiosClient.get<Marca>(`/ ${id}`)).data
21         } catch (error : any) {
22             return Promise.reject(error.response)
23         }
24     }
25
26     public async listAll() : Promise<Marca[]> {
27         try {
28             return (await this.axiosClient.get<Marca[]>(`/ lista`)).data
29         } catch (error : any) {
30             return Promise.reject(error.response)
31         }
32     }
33
34     public async findByAtivo() : Promise<Marca[]> {
35         try {
36             return (await this.axiosClient.get<Marca[]>(`/ ativo`)).data
37         } catch (error : any) {
38             return Promise.reject(error.response)
39         }
40     }
41
42     public async cadastrar(marca : Marca) : Promise<void> {
43         try {
44             return (await this.axiosClient.post('/', marca))
45         } catch (error : any) {
46             return Promise.reject(error.response)
47         }
48     }
49
50     public async editar(marca : Marca) : Promise<void> {
51         try {
52             return (await this.axiosClient.post(`/ ${marca.id}`, marca)).data
53         } catch (error : any) {
54             return Promise.reject(error.response)
55         }
56     }
57
58     public async deletar(marca : Marca) : Promise<string> {
59         try {
60             return (await this.axiosClient.delete(`/ ${marca.id}`)).data
61         } catch (error : any) {
62             return Promise.reject(error.response)
63         }
64     }
65
66     public async findByFiltrosPaginado(pageRequest : PageRequest) : Promise<PageResponse<Marca>> {
```

TS modelo.client.ts U X

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

```
1 import axios, { AxiosInstance } from "axios";
2
3 import { Modelo } from "@model/modelo";
4 import { PageRequest } from "@model/page/page-request";
5 import { PageResponse } from "@model/page/page-response";
6
7 export class ModeloClient {
8
9     private axiosClient : AxiosInstance;
10
11     constructor(){
12         this.axiosClient = axios.create({
13             baseURL: 'http://localhost:8080/api/modelo',
14             headers: {'Content-Type' : 'application/json'}
15         });
16     }
17
18     public async findById(id : number) : Promise<Modelo> {
19         try {
20             return (await this.axiosClient.get<Modelo>(`/ ${id}`)).data
21         } catch (error : any) {
22             return Promise.reject(error.response)
23         }
24     }
25
26     public async listAll() : Promise<Modelo[]> {
27         try {
28             return (await this.axiosClient.get<Modelo[]>(`/ lista`)).data
29         } catch (error : any) {
30             return Promise.reject(error.response)
31         }
32     }
33
34     public async findByAtivo() : Promise<Modelo[]> {
35         try {
36             return (await this.axiosClient.get<Modelo[]>(`/ ativo`)).data
37         } catch (error : any) {
38             return Promise.reject(error.response)
39         }
40     }
41
42     public async cadastrar(modelo : Modelo) : Promise<void> {
43         try {
44             return (await this.axiosClient.post('/', modelo))
45         } catch (error : any) {
46             return Promise.reject(error.response)
47         }
48     }
49
50     public async editar(modelo : Modelo) : Promise<void> {
51         try {
52             return (await this.axiosClient.put(`/ ${modelo.id}`, modelo)).data
53         } catch (error : any) {
54             return Promise.reject(error.response)
55         }
56     }
57
58     public async findByFiltrosPaginado(pageRequest : PageRequest) : Promise<PageResponse<Modelo>> {
59         try {
60             let requestPath = ''
61
62             requestPath += `?page=${pageRequest.currentPage}`
63             requestPath += `&size=${pageRequest.pageSize}`
64             requestPath += `&sort=${pageRequest.sortField === undefined ? '' : pageRequest.sortField}, ${pageRequest.direction}`
65
66             return (await this.axiosClient.get<PageResponse<Modelo>>)(requestPath, {params : {filtros : pageRequest.filter } })).data
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

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

No issues found.


```

estacionamento-client > src > client > TS movimentacao.clients.ts > ...
1 import axios, { AxiosInstance } from "axios";
2
3 import { Movimentacao } from "@model/movimentacao";
4 import { PageRequest } from "@model/page/page-request";
5 import { PageResponse } from "@model/page/page-response";
6
7 export class MovimentacaoClient {
8
9     private axiosClient : AxiosInstance;
10
11     constructor(){
12         this.axiosClient = axios.create({
13             baseURL: 'http://localhost:8080/api/movimentacao',
14             headers: {'Content-type' : 'application/json'}
15         });
16     }
17
18     public async findById(id : number) : Promise<Movimentacao> {
19         try {
20             return (await this.axiosClient.get<Movimentacao>(`/ ${id}`)).data
21         } catch (error : any) {
22             return Promise.reject(error.response)
23         }
24     }
25
26     public async listAll() : Promise<Movimentacao[]> {
27         try {
28             return (await this.axiosClient.get<Movimentacao[]>(`/ lista`)).data
29         } catch (error : any) {
30             return Promise.reject(error.response)
31         }
32     }
33
34     public async findByAbertas() : Promise<Movimentacao[]> {
35         try {
36             return (await this.axiosClient.get<Movimentacao[]>(`/ ativo`)).data
37         } catch (error : any) {
38             return Promise.reject(error.response)
39         }
40     }
41
42     public async cadastrar(movimentacao : Movimentacao) : Promise<void> {
43         try {
44             return (await this.axiosClient.post('/', movimentacao))
45         } catch (error : any) {
46             return Promise.reject(error.response)
47         }
48     }
49
50     public async editar(movimentacao : Movimentacao) : Promise<void> {
51         try {
52             return (await this.axiosClient.put(`/ ${movimentacao.id}`, movimentacao)).data
53         } catch (error : any) {
54             return Promise.reject(error.response)
55         }
56     }
57
58     public async fecharMovimentacao(movimentacao : Movimentacao) : Promise<void> {
59         try {
60             return (await this.axiosClient.put(`/ saida/${movimentacao.id}`, movimentacao)).data
61         } catch (error : any) {
62             return Promise.reject(error.response)
63         }
64     }
65
66     public async deletar(movimentacao : Movimentacao) : Promise<string> {

```

```

estacionamento-client > src > client > TS veiculo.clients.ts > ...
1 import axios, { AxiosInstance } from "axios";
2
3 import { Veiculo } from "@model/veiculo";
4 import { PageRequest } from "@model/page/page-request";
5 import { PageResponse } from "@model/page/page-response";
6 import { Movimentacao } from "@model/movimentacao";
7
8 export class VeiculoClient {
9
10     private axiosClient : AxiosInstance;
11
12     constructor(){
13         this.axiosClient = axios.create({
14             baseURL: 'http://localhost:8080/api/veiculo',
15             headers: {'Content-type' : 'application/json'}
16         });
17     }
18
19     public async findById(id : number) : Promise<Veiculo> {
20         try {
21             return (await this.axiosClient.get<Veiculo>(`/ ${id}`)).data
22         } catch (error : any) {
23             return Promise.reject(error.response)
24         }
25     }
26
27     public async listAll() : Promise<Veiculo[]> {
28         try {
29             return (await this.axiosClient.get<Veiculo[]>(`/ lista`)).data
30         } catch (error : any) {
31             return Promise.reject(error.response)
32         }
33     }
34
35     public async findByAtivo() : Promise<Veiculo[]> {
36         try {
37             return (await this.axiosClient.get<Veiculo[]>(`/ ativo`)).data
38         } catch (error : any) {
39             return Promise.reject(error.response)
40         }
41     }
42
43     public async cadastrar(veiculo : Veiculo) : Promise<void> {
44         try {
45             return (await this.axiosClient.post('/', veiculo))
46         } catch (error : any) {
47             return Promise.reject(error.response)
48         }
49     }
50
51     public async editar(veiculo : Veiculo) : Promise<void> {
52         try {
53             return (await this.axiosClient.put(`/ ${veiculo.id}`, veiculo)).data
54         } catch (error : any) {
55             return Promise.reject(error.response)
56         }
57     }
58
59     public async deletar(veiculo : Veiculo) : Promise<string> {
60         try {
61             return (await this.axiosClient.delete(`/ ${veiculo.id}`)).data
62         } catch (error : any) {
63             return Promise.reject(error.response)
64         }
65     }
66
67     public async fecharVeiculo(veiculo : Veiculo) : Promise<void> {

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

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

No issues found.