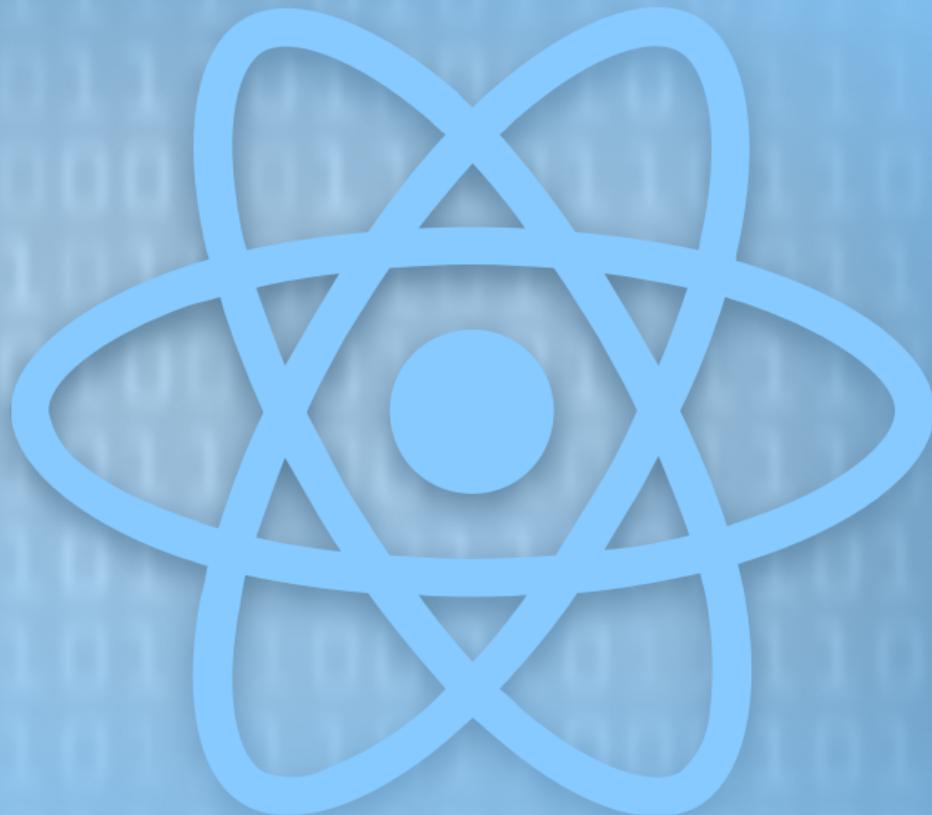


AXIOS



Quando utilizamos algum framework de Javascript para criação de algum sistema, em algum momento necessitaremos da utilização de API, neste caso entra o Axios.

O Axios é uma biblioteca que simplifica as requisições a serem requisitadas na API, de forma descomplicada e simples para o usuário.



A biblioteca funciona da seguinte forma:



1 - IMPORTAMOS A BIBLIOTECA

```
1 import axios from "axios";
```

2 - CRIAMOS UMA FUNÇÃO PARA ACESSAR O ENDPOINT DA API

```
Login = async (email: string, password: string) => {
  return axios
    .post("http://localhost:5000/Auth/SignIn",
      { email: email, password: password })
    .then((res) => res.data)
};
```

A biblioteca funciona da seguinte forma:



3 - CRIAMOS UMA FUNÇÃO DENTRO DO COMPONENTE E ARMAZENAMOS O RETORNO DA API DENTRO DA VARIÁVEL

```
const userLogin = async () => {  
  const response = await this.Login('email', 'senha');  
  console.log(response);  
}  
  
userLogin();
```

Executando dessa forma conseguimos utilizar o retorno dos dados desta variavel dentro do componente da maneira que desejarmos, podemos mandar diretamente como props, ou atribuir os dados diretamente no retorno do componente.



Essas são as requisições disponíveis a serem realizadas pelo axios (o método do endpoint da API deve ser idêntico ao requisitado pelo axios):

```
axios.request(config)
```

```
axios.get(url[, config])
```

```
axios.delete(url[, config])
```

```
axios.head(url[, config])
```

```
axios.options(url[, config])
```

```
axios.post(url[, data[, config]])
```

```
axios.put(url[, data[, config]])
```

```
axios.patch(url[, data[, config]])
```

AXIOS GET

Endpoint no swagger:

The screenshot shows a Swagger UI interface for a 'Pokemon' endpoint. The method is 'GET' and the URL is '/GetPokemons'. The 'Parameters' section indicates 'No parameters'. A large blue button at the bottom is labeled 'Execute'. Below the execute button, under 'Responses', is a 'Curl' section containing a command: 'curl -X GET "http://localhost:5000/GetPokemons" -H "accept: */*' and a 'Request URL' field with the value 'http://localhost:5000/GetPokemons'.

Exemplo de função GET no axios:

```
 GetAllPokemon = async (): Promise<PokedexList[]> =>  
    axios  
        .get(pokedex.getAll)  
        .then((res) => res.data);
```

neste caso a string do endpoint está sendo passada através de um objeto

```
const APIUrl = 'http://localhost:5000/';  
  
export const pokedex = {  
    getAll: `${APIUrl}GetPokemons`,  
    getByName: `${APIUrl}GetPokemonsByName`,  
    getByNameXML: `${APIUrl}GetPokemonsByNameXML`,  
};
```

Você deve ter notado a utilização de :Promise nas funções, ela serve para mapear o retorno da requisição da API, possibilitando a utilização das respostas em toda a aplicação, o mapeamento das Promises podem ser realizada com type ou interface.

Resposta da API

```
{  
  "id": "b62ecaac-8672-409c-85df-d1ae71d83196",  
  "name": "Bulbasaur",  
  "type": "Grass",  
  "total": 318,  
  "hp": 45,  
  "attack": 49,  
  "defense": 49,  
  "spAtck": 65,  
  "spDef": 65,  
  "speed": 45,  
  "generation": 1,  
  "type2": "Poison",  
  "legendary": false,  
  "pokemonId": 1  
},
```

Interface

```
export interface PokedexList {  
  attack?: number;  
  defense?: number;  
  id?: string;  
  generation?: number;  
  hp?: number;  
  legendary?: boolean;  
  name?: string;  
  pokemonId?: number;  
  spAtck?: number;  
  spDef?: number;  
  speed?: number;  
  total?: number;  
  type?: string;  
  type2?: string;  
}
```

GET by ID

Swagger

GET /api/Area/{id}

Parameters

Name	Description
id * required string (path)	id

Responses

Essa é a maneira correta de realizar a requisição GET by ID

```
GetById = async (id: string) =>
  axios.get(`https://localhost:5001/api/Area/${id}`)
    .then((res) => res.data);
```

OBS: este modo apesar irá funcionar caso o endpoint contenha {} e o nome do parametro dentro, neste caso {id}

GET com parametros

Swagger

GET /GetPokemonsByName

Parameters

Name	Description
name	string (query) <input type="text" value="name"/>

Caso o endpoint solicite um ou mais parametros para requisição, eles deverão ser enviados através de objeto

```
GetPokemonByName = async (pokemonName: string): Promise<PokedexList> =>
  axios
    .get<PokedexList>(`http://localhost:5000/GetPokemonsByName`,
      {
        params: {
          name: pokemonName
        }
      })
    .then((res) => res.data);
```

POST

Swagger

POST /api/Usuarios/Login

Parameters

No parameters

Request body

Example Value | Schema

```
{  
  "userName": "string",  
  "email": "string",  
  "password": "string",  
  "rememberMe": true  
}
```

Assim como funciona no GET, no POST também devemos passar o corpo como objeto, porém aqui não precisamos do params para isso.

```
SignIn = async (user: UserLogin): Promise<User> =>  
  axios.post<User>(auth.signIn, user)  
  .then((res) => res.data);  
  
SignIn2 = async (  
  password: string,  
  rememberMe: boolean,  
  email?: string,  
  userName?: string,  
): Promise<User> =>  
  axios.post<User>(auth.signIn,  
  {  
    userName,  
    email,  
    password,  
    rememberMe  
  })  
  .then((res) => res.data);
```

```
SignIn3 = async (  
  passwordStr: string,  
  rememberMeBool: boolean,  
  emailStr?: string,  
  userNameStr?: string,  
): Promise<User> =>  
  axios.post<User>(auth.signIn,  
  {  
    userName: userNameStr,  
    email: emailStr,  
    password: passwordStr,  
    rememberMe: rememberMeBool  
  })  
  .then((res) => res.data);
```

auth.signIn = "http://localhost:5000/api/Usuarios/Login"