

Realizando Requisições com o Axios

EC021 - Tópicos Avançados II Sistemas Distribuídos

Prof. Adauto Mendes | E-mail: adauto.mendes@inatel.br

O Axios

- O Axios é uma biblioteca NodeJS que é capaz de realizar requisições HTTP. Todas as requisições são encapsuladas em Promises para garantir que a aplicação aguarde a resposta.
- O Axios é baseado em Promises e, portanto, podemos tirar proveito do mecanismo de async/await e tornar o código mais legível.
- Também podemos interceptar e cancelar solicitações, e há proteção interna do cliente contra falsificação de solicitações entre sites.

Instalando o Axios

 Para instalar o Mongoose na nossa aplicação devemos executar o comando abaixo:

```
npm install axios
```

 Feito isto, adicione a importação da biblioteca no arquivo de entrada da aplicação:

```
const axios = require('axios');
```

- As requisições com o Axios geralmente recebem 2 ou 3 parâmetros:
 - URL
 - Uma string contendo o endpoint que será chamado.
 - Data (para requests POST, PUT e PATCH)
 - Um objeto json com o body da requisição.
 - Config
 - A configuração da requisição.
 - Neste parâmetro enviamos um objeto json contendo, por exemplo, os headers da requisição.
 - No caso de requisições que não suportem 3 parâmetros utilizaremos este parâmetro para definir o body a ser enviado.

Configuração

 É possível, ao configurar o Axios, definir uma URL base para ser utilizada nas requisições. Para isto criaremos uma instância do Axios utilizando o método create. Como parâmetro definiremos a baseURL:

```
const axiosInstance = axios.create({
   baseURL: 'http://localhost:5000/toddy',
});
```



 Para requisições dos tipos POST, PUT e PATCH é possível passar 3 parâmetros:

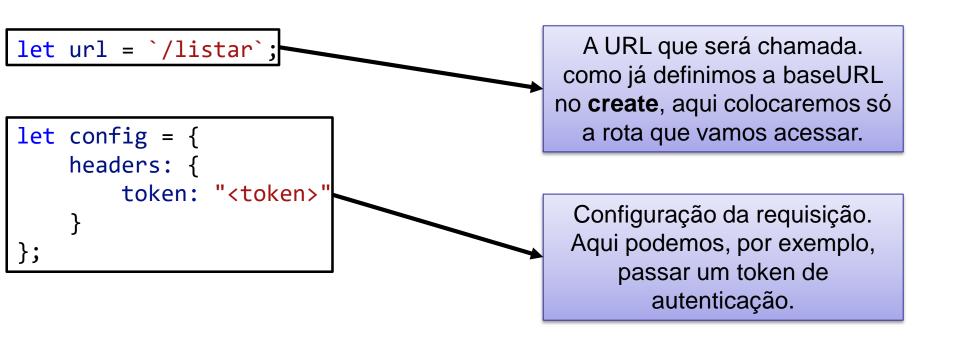
```
A URL que será chamada.
let url = `/salvar`;
                                                como já definimos a baseURL
                                               no create, aqui colocaremos só
                                                  a rota que vamos acessar.
let data = {
    lote: "ABC",
    conteudo: 200,
                                            O body que será enviado na requisição
    validade: "20/12/2021"
                                                        Configuração da
let config = {
                                                   requisição. Aqui podemos,
    headers: {
                                                    por exemplo, passar um
        token: "<token>"
                                                     token de autenticação.
    },
```

 Para realizar a requisição chamaremos a instância criada do Axios:

```
let url = `/salvar`;
let data = {
    lote: "ABC",
    conteudo: 200,
    validade: "20/12/2021"
                                  axiosInstance.<método>(url, data, config)
                                       .then((response) => {
                                           return res.json(response.data);
                                       })
let config = {
                                       .catch((error) => {
    headers: {
                                           return res.json(error.response.data);
        token: "<token>"
                                      });
    },
```



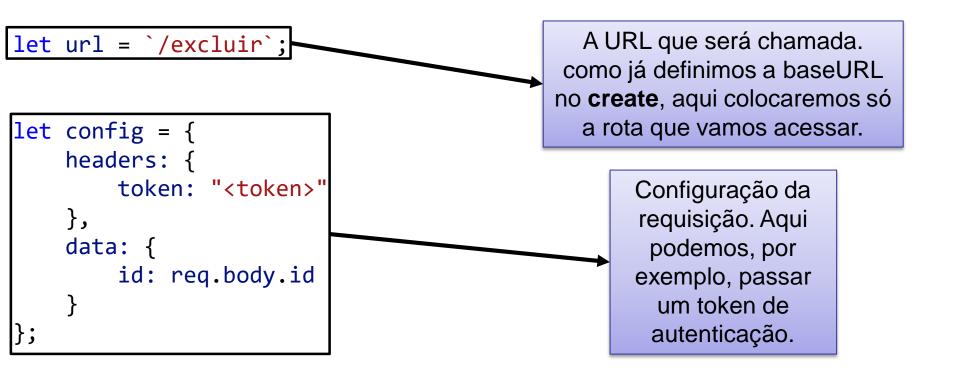
 Para requisições dos tipos GET é possível passar 2 parâmetros:



 Para realizar a requisição chamaremos a instância criada do Axios:

```
let url = `/listar`;
let config = {
    headers: {
        token: "<token>"
                                    axiosInstance.get(url, config)
                                        .then((response) => {
                                            return res.json(response.data);
                                        })
                                        .catch((error) => {
                                            return res.json(error.response.data);
                                        });
```

Requisições do tipo DELETE também recebem 2 parâmetros.
 Em casos onde é necessário passar dados no body, isto deve ser feito no parâmetro config:



 Para realizar a requisição chamaremos a instância criada do Axios:

```
let url = `/excluir`;
let config = {
    headers: {
        token: "<token>"
    },
    data: {
        id: req.body.id
                                   axiosInstance.delete(url, config)
                                       .then((response) => {
                                           return res.json(response.data);
                                       .catch((error) => {
                                           return res.json(error.response.data);
                                       });
```

Tratando as Respostas

- Independentemente do tipo de requisição feita, o Axios trabalha baseado em Promise. Isto quer dizer que devemos utilizar as funções then/catch para tratar a resposta das requisições de maneira síncrona.
 - Em alguns casos, também é possível fazer este tratamento com async/await.
- A função then recebe como parâmetro a response.
- A função catch recebe como parâmetro error.



Tratando as Respostas

```
axiosInstance.<método>(url, ...)
    .then((response) => {
        return res.json(response.data);
})

.catch((error) => {
        return res.json(error.response.data);
});

Trata casos de sucesso

O objeto error possui
    um objeto response
    interna a ele.
```



Tratando as Respostas

 De acordo com a documentação do Axios a response possui os seguintes campos:

```
{
    data: {},
    status: 200,
    statusText: 'OK',
    headers: {},
    config: {},
    request: {}
}
```



- Data: resposta fornecida pelo servidor
- Status: Código HTTP da resposta
- StatusText: Mensagem HTTP da resposta
- Headers: Cabeçalho da resposta
- Config: Configuração fornecida para o Axios na requisição
- Request: Requisição que originou esta resposta