

# SW-II

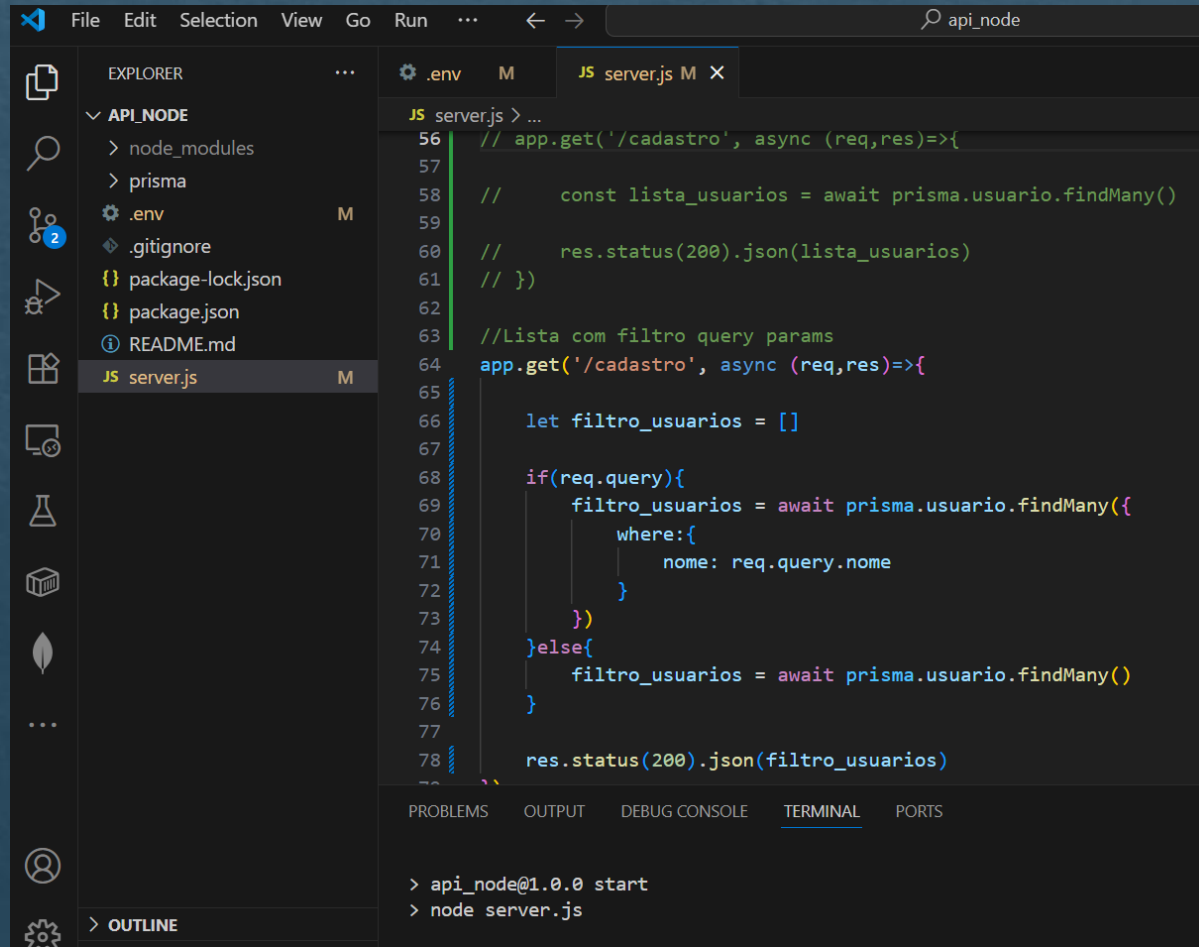
# SISTEMAS WEB II

Prof. Anderson Vanin

AULA 18 – Conectando o Frontend ao Backend – Parte 2

# Rodar o Servidor

- Abra uma aba do VS code com os arquivos de seu servidor

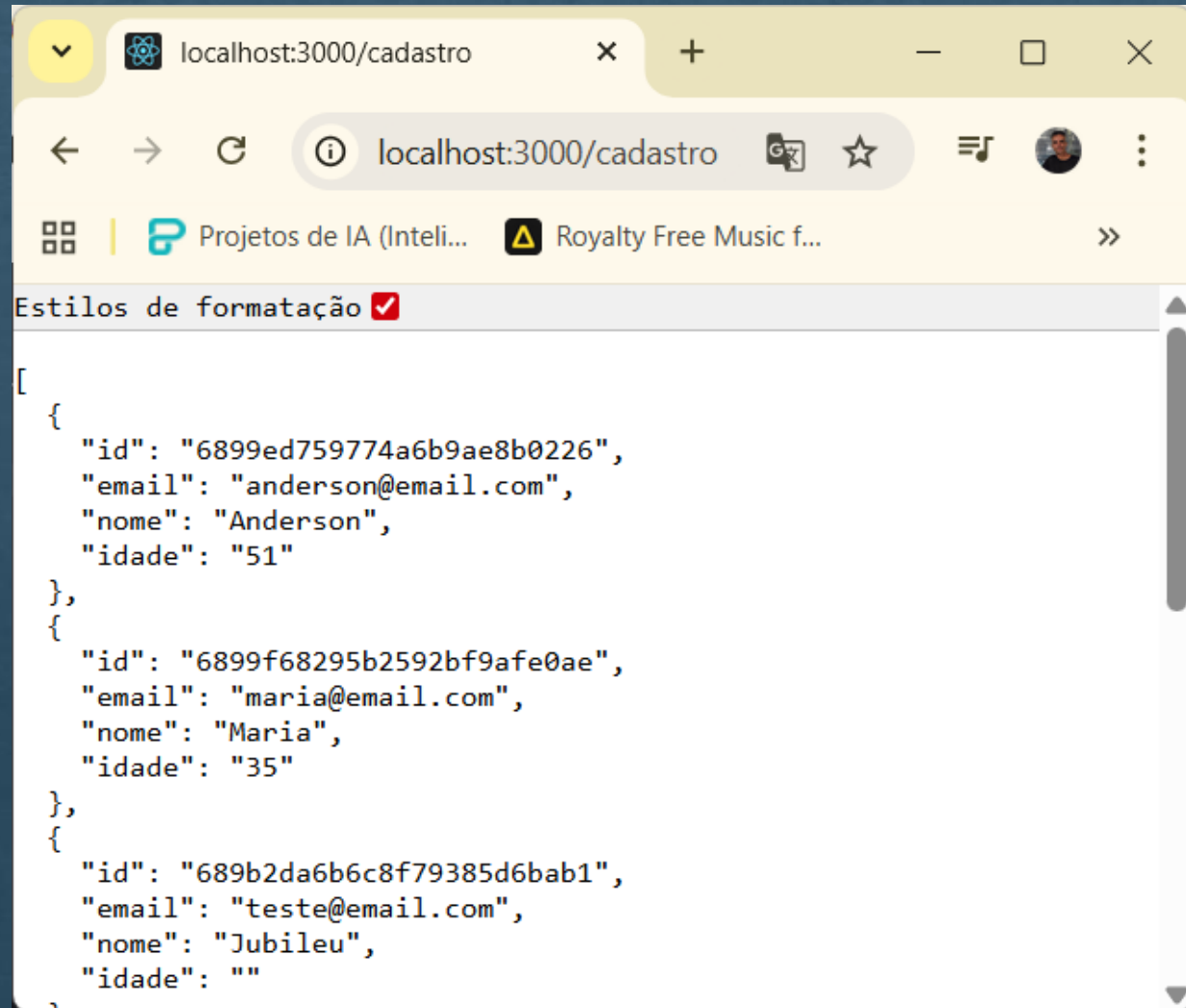


The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the project structure. The main editor window shows the `server.js` file with the following code:

```
56 // app.get('/cadastro', async (req,res)=>{
57
58 //     const lista_usuarios = await prisma.usuario.findMany()
59
60 //     res.status(200).json(lista_usuarios)
61 // })
62
63 //Lista com filtro query params
64 app.get('/cadastro', async (req,res)=>{
65
66     let filtro_usuarios = []
67
68     if(req.query){
69         filtro_usuarios = await prisma.usuario.findMany({
70             where:{
71                 nome: req.query.nome
72             }
73         })
74     }else{
75         filtro_usuarios = await prisma.usuario.findMany()
76     }
77
78     res.status(200).json(filtro_usuarios)
79 }
```

The terminal at the bottom shows the command to start the server:

```
> api_node@1.0.0 start
> node server.js
```

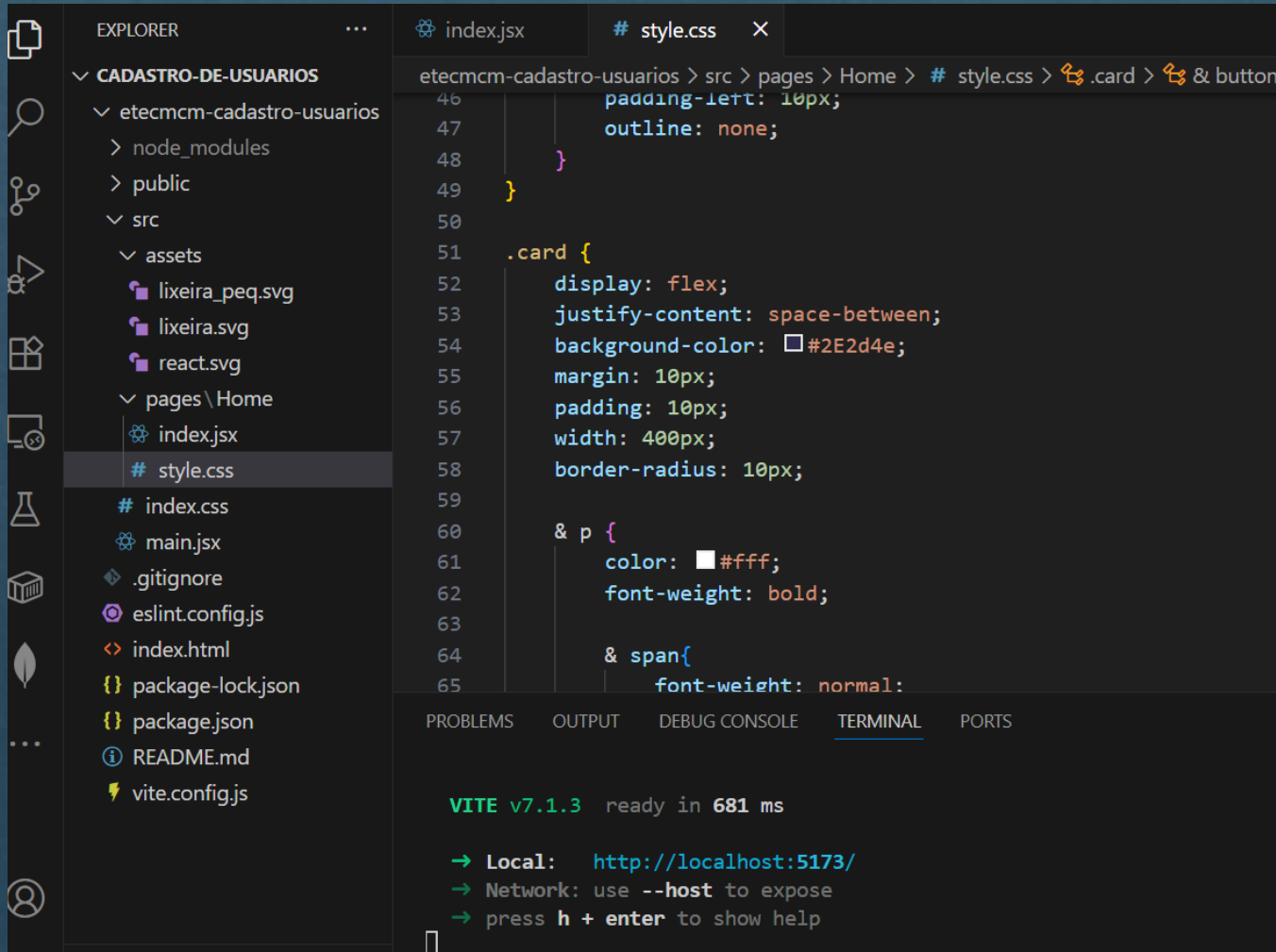


The screenshot shows a web browser window with the address bar displaying `localhost:3000/cadastro`. The browser's developer tools are open, showing the response of the API call. The response is a JSON array of three user objects:

```
[
  {
    "id": "6899ed759774a6b9ae8b0226",
    "email": "anderson@email.com",
    "nome": "Anderson",
    "idade": "51"
  },
  {
    "id": "6899f68295b2592bf9afe0ae",
    "email": "maria@email.com",
    "nome": "Maria",
    "idade": "35"
  },
  {
    "id": "689b2da6b6c8f79385d6bab1",
    "email": "teste@email.com",
    "nome": "Jubileu",
    "idade": ""
  }
]
```

# Abrir arquivos do Front end

- Abra outra aba do VS code com os arquivos de seu FRONT END

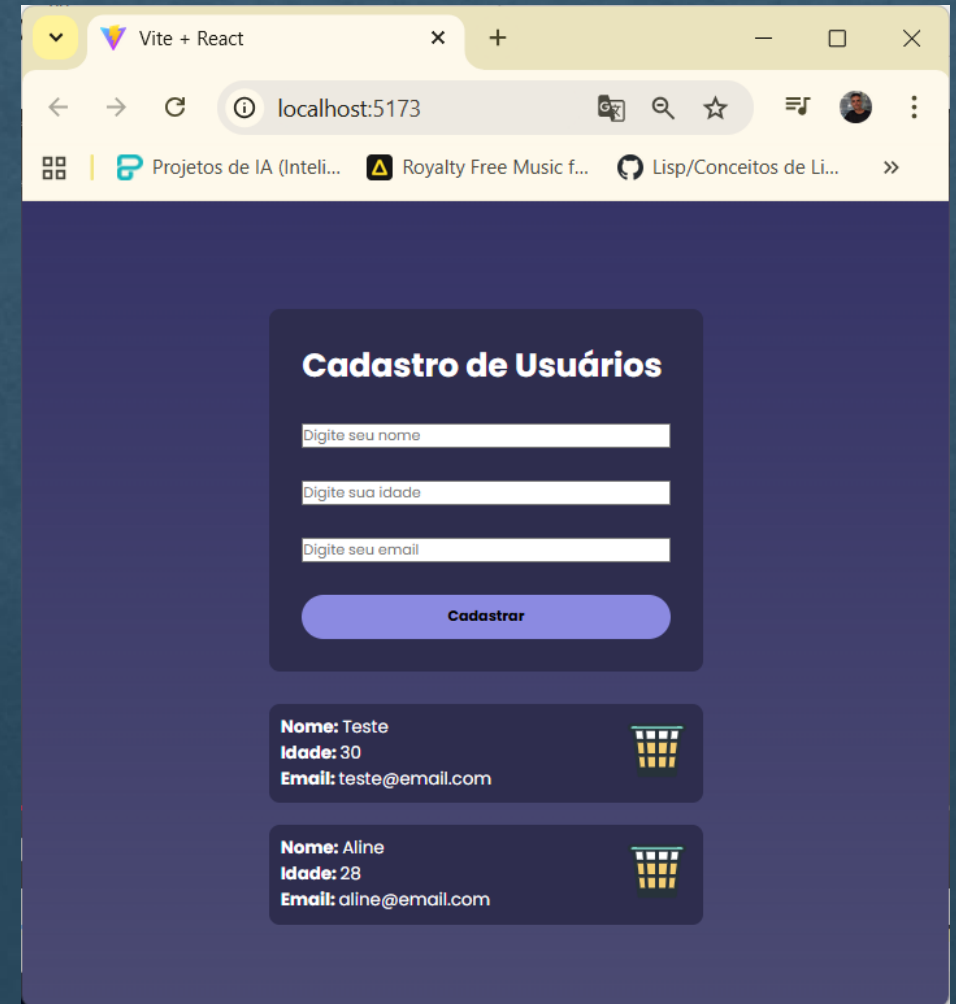


The screenshot shows the VS Code interface. The Explorer sidebar on the left displays a project structure for 'etecmcm-cadastro-usuarios'. The main editor area shows the 'style.css' file with the following content:

```
46 padding-left: 10px;
47 outline: none;
48 }
49 }
50
51 .card {
52   display: flex;
53   justify-content: space-between;
54   background-color: #2E2d4e;
55   margin: 10px;
56   padding: 10px;
57   width: 400px;
58   border-radius: 10px;
59
60   & p {
61     color: #fff;
62     font-weight: bold;
63
64     & span{
65       font-weight: normal;
```

The bottom panel shows the terminal with the following output:

```
VITE v7.1.3 ready in 681 ms
→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```



# AXIOS

- Agora vamos conectar os dois lados Back e Frontend e para isso vamos utilizar uma biblioteca do Node chamada Axios (<https://axios-http.com/ptbr/docs/intro>)






# Instalando Axios

- Pare o FrontEnd e instale a biblioteca axios (**INSTALAR NO FRONTEND**)
- `npm i axios`

```
C:\Users\Anderson\Desktop\cadastro-de-usuarios\etecmcm-cadastro-usuarios>npm i axios  
  
added 23 packages, and audited 176 packages in 3s  
  
39 packages are looking for funding  
  run `npm fund` for details
```

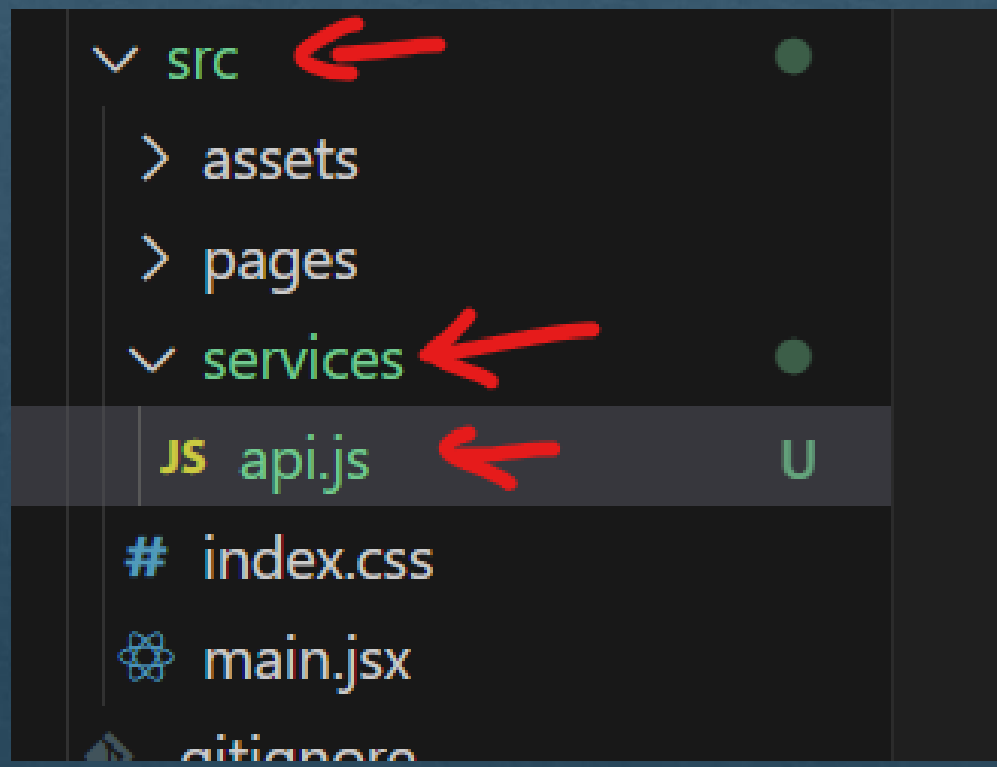
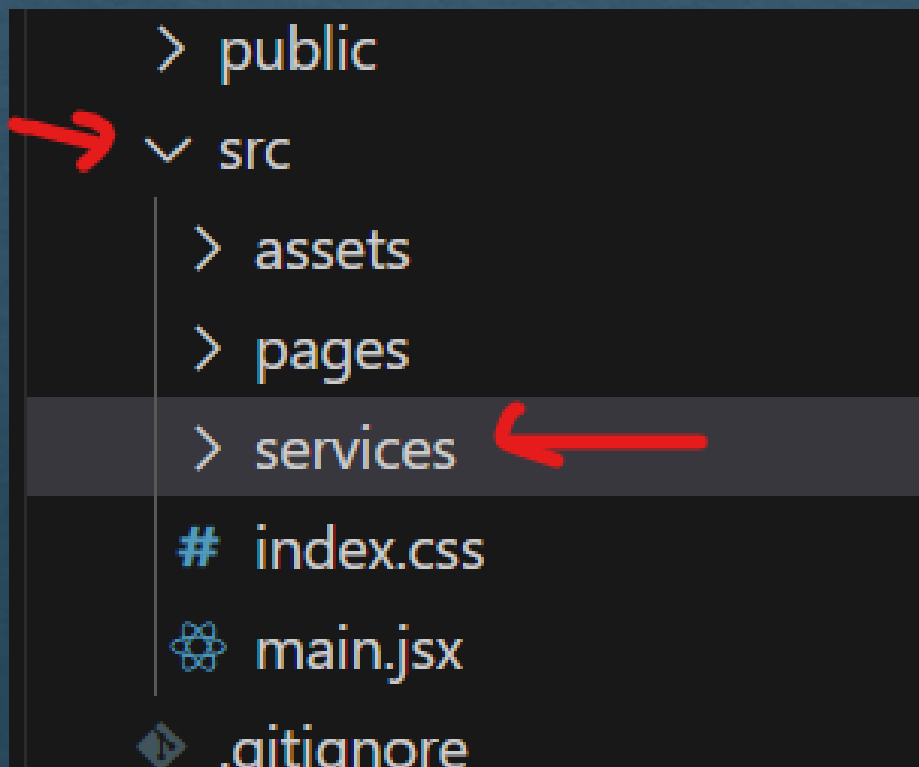


- Após a instalação volte a rodar seu Frontend

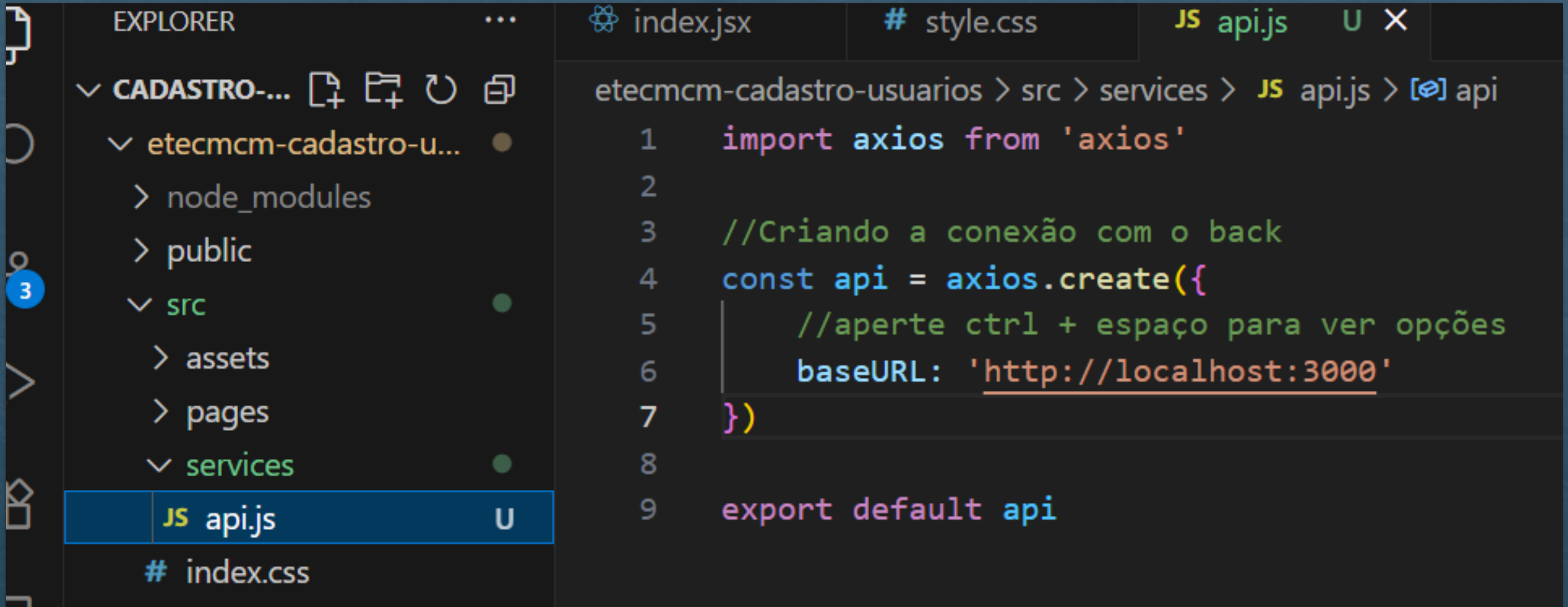
```
14:44:51 [vite] (client) Re-optimizing dependencies because lockfile has changed  
  
VITE v7.1.3 ready in 303 ms  
  
→ Local:   http://localhost:5173/  
→ Network: use --host to expose  
→ press h + enter to show help
```

# Organize seu frontend

- Dentro da pasta **src**, crie uma pasta chamada **services**
- Dentro da pasta **services**, crie um arquivo chamado **api.js**



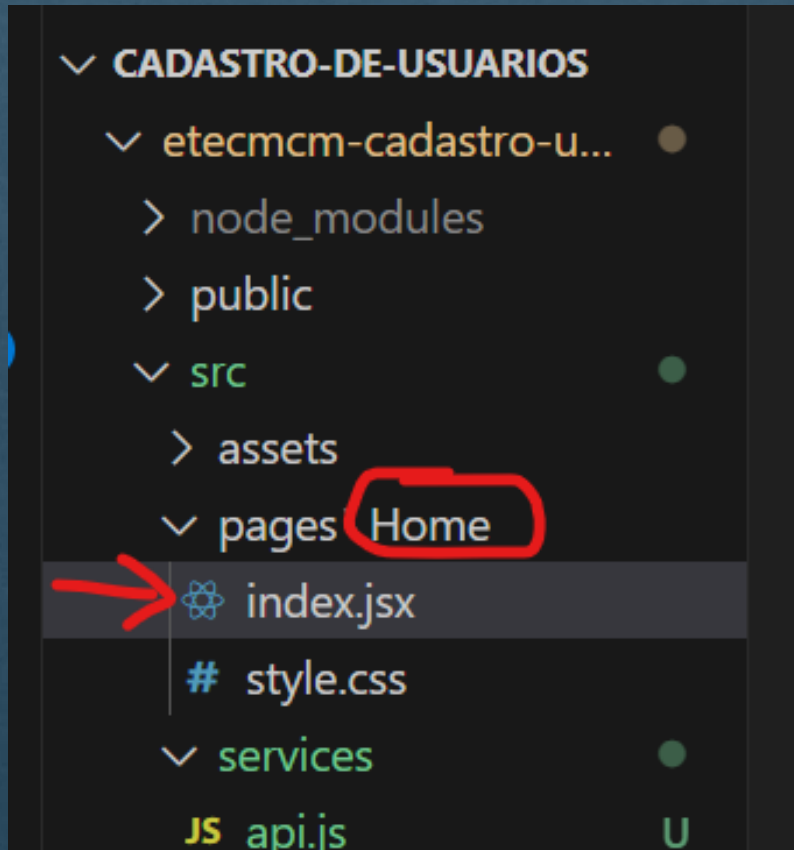
# Arquivo api.js



The image shows a screenshot of the Visual Studio Code editor. On the left, the Explorer sidebar displays a file tree for a project named 'etecmcm-cadastro-u...'. The tree structure includes folders like 'node\_modules', 'public', 'src', 'assets', 'pages', and 'services'. The 'services' folder is expanded, and the file 'api.js' is selected, highlighted in blue. The main editor area shows the content of 'api.js', which is a JavaScript file using the 'axios' library to create an API client. The code is as follows:

```
etecmcm-cadastro-usuarios > src > services > JS api.js > [api] api
1  import axios from 'axios'
2
3  //Criando a conexão com o back
4  const api = axios.create({
5      //aperte ctrl + espaço para ver opções
6      baseUrl: 'http://localhost:3000'
7  })
8
9  export default api
```

# Editar arquivo index.js da Home



```
index.jsx 1, M X  # style.css JS api.js U
etecmcm-cadastro-usuarios > src > pages > Home > index.jsx > Home
1  import './style.css'
2  import Lixeira from '../assets/lixeira_peq.svg'
3  import api from '../services/api'
4
5  function Home() {
6      let usuarios = []
7
8      async function getUsuarios() {
9          usuarios = await api.get('/cadastro')
10         console.log(usuarios)
11     }
12
13
14
15
16
17     return (
18         <div className='container'>
19             <form>
```





# Testando

## Cadastro de Usuários

Cadastrar

The screenshot shows the Chrome DevTools Network tab with the 'Fetch/XHR' filter selected. Two requests named 'cadastro' are listed, both with a status of 'CORS error' and type 'xhr'. A red circle highlights the 'Status' and 'Type' columns for these two requests. The 'Initiator' column shows 'index.jsx:10' for both. The 'Size' column shows '0.0 kB' for both. The top of the interface shows 6 errors and 2 warnings. The bottom of the interface shows '2 / 24 requests', '0.0 kB / 1.9 kB transferred', '0.0 kB / 1,424 kB resources', 'Finish: 141 ms', and 'DOMCon'.

Name	Status	Type	Initiator	Size
✖ cadastro	CORS error	xhr	index.jsx:10	0.0 kB
✖ cadastro	CORS error	xhr	index.jsx:10	0.0 kB

2 / 24 requests | 0.0 kB / 1.9 kB transferred | 0.0 kB / 1,424 kB resources | Finish: 141 ms | DOMCon

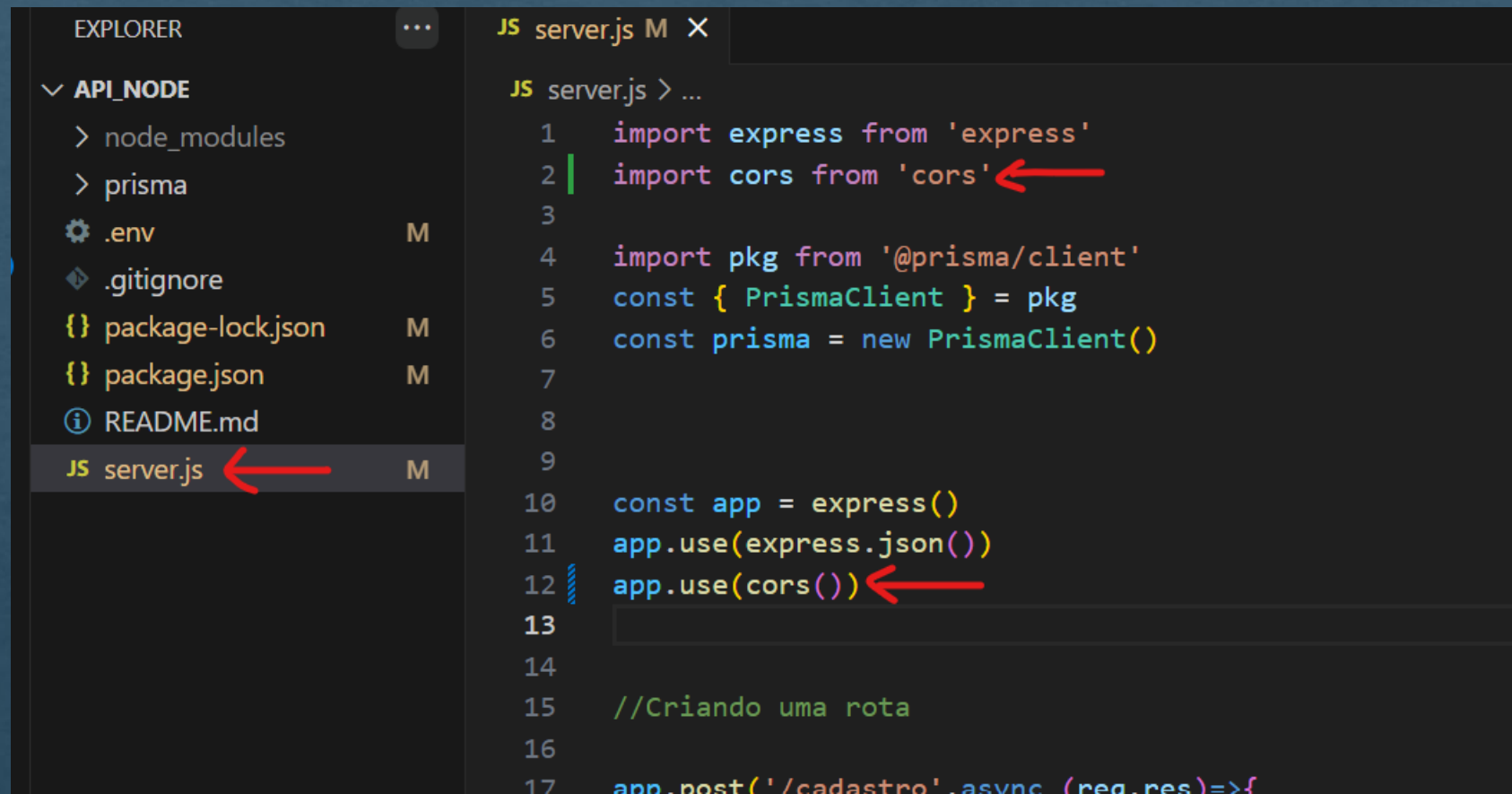
# Corrigindo erros de CORS

- Pare o servidor e instale a biblioteca:
- `npm i cors`

```
C:\Users\Anderson\Desktop\api_node>npm i cors ←  
  
added 2 packages, and audited 135 packages in 1s  
  
28 packages are looking for funding  
  run `npm fund` for details
```

# Edite o seu servidor para habilitar o cors

- Edite o arquivo **server.js** no seu servidor



```
JS server.js M X
JS server.js > ...
1  import express from 'express'
2  import cors from 'cors'
3
4  import pkg from '@prisma/client'
5  const { PrismaClient } = pkg
6  const prisma = new PrismaClient()
7
8
9
10 const app = express()
11 app.use(express.json())
12 app.use(cors())
13
14
15 //Criando uma rota
16
17 app.post('/cadastro', async (req, res) => {
```

- Coloque o servidor para rodar novamente.

# Volte a testar o FrontEnd

The screenshot shows a web browser at `localhost:5173` displaying a user registration form titled "Cadastro de Usuários". The form has three input fields: "Digite seu nome", "Digite sua idade", and "Digite seu email", followed by a "Cadastrar" button.

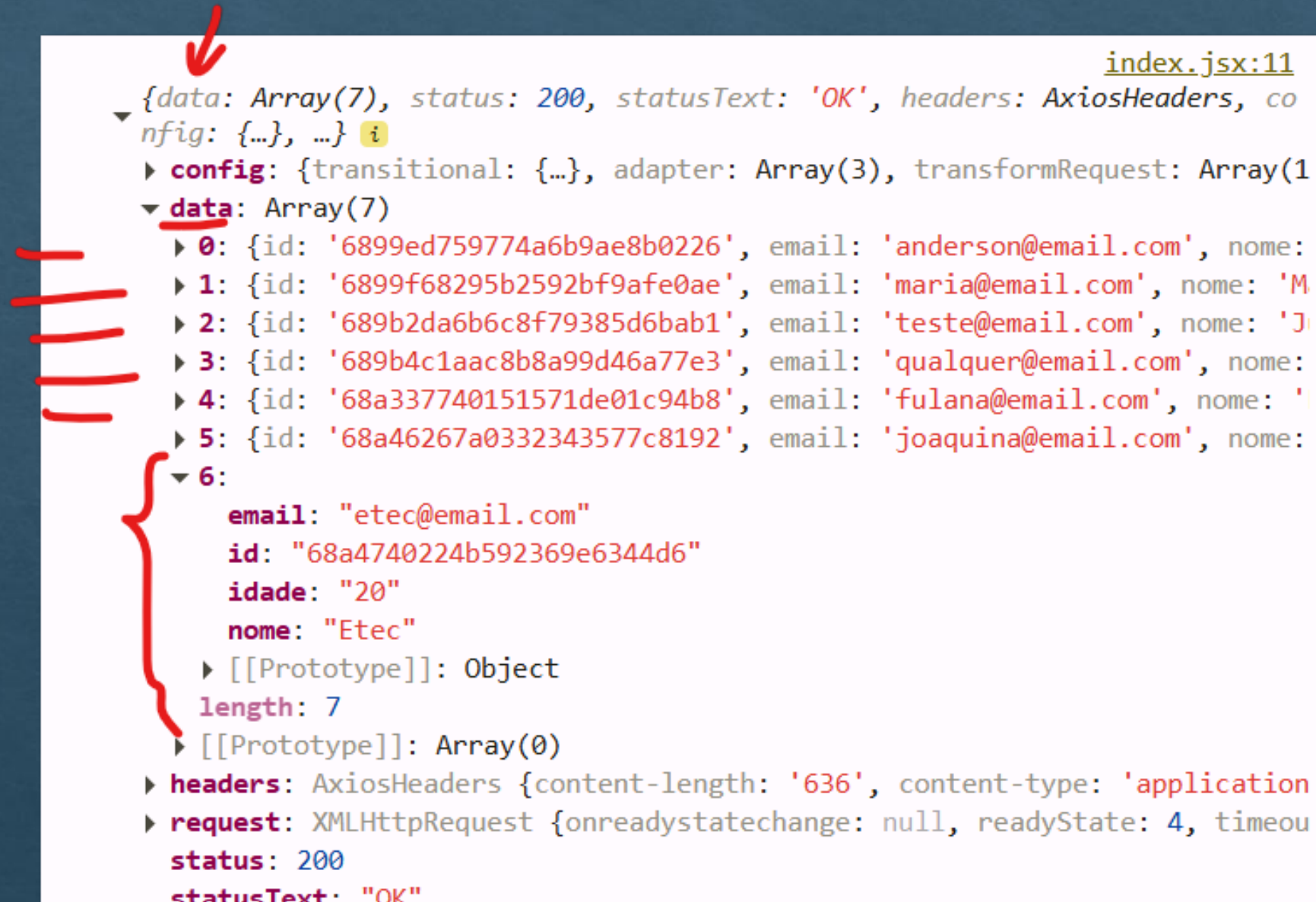
On the right, the Chrome DevTools Network tab is open, showing a list of network requests. The "Fetch/XHR" filter is selected. A red arrow points to the "Network" tab, and another red arrow points to the "Fetch/XHR" filter. A third red arrow points to the "Response" tab, which displays the JSON response for the selected "cadastro" request:

```
{
  "id": "6899ed759774a6b9ae8b0226",
  "email": "anderson@email.com",
  "nome": "Anderson",
  "idade": "51"
},
{
  "id": "6899f68295b2592bf9afe0ae",
  "email": "maria@email.com"
}
```

At the bottom of the DevTools panel, the Console tab is visible, showing "1 Issue".



# Veja no console.log o conteúdo da variável usuarios



```
index.jsx:11
{data: Array(7), status: 200, statusText: 'OK', headers: AxiosHeaders, config: {...}, ...} i
  ▶ config: {transitional: {...}, adapter: Array(3), transformRequest: Array(1)
  ▼ data: Array(7)
    ▶ 0: {id: '6899ed759774a6b9ae8b0226', email: 'anderson@email.com', nome:
    ▶ 1: {id: '6899f68295b2592bf9afe0ae', email: 'maria@email.com', nome: 'M
    ▶ 2: {id: '689b2da6b6c8f79385d6bab1', email: 'teste@email.com', nome: 'J
    ▶ 3: {id: '689b4c1aac8b8a99d46a77e3', email: 'qualquer@email.com', nome:
    ▶ 4: {id: '68a337740151571de01c94b8', email: 'fulana@email.com', nome: '
    ▶ 5: {id: '68a46267a0332343577c8192', email: 'joaquina@email.com', nome:
    ▼ 6:
      email: "etec@email.com"
      id: "68a4740224b592369e6344d6"
      idade: "20"
      nome: "Etec"
      ▶ [[Prototype]]: Object
      length: 7
      ▶ [[Prototype]]: Array(0)
    ▶ headers: AxiosHeaders {content-length: '636', content-type: 'application
    ▶ request: XMLHttpRequest {onreadystatechange: null, readyState: 4, timeou
      status: 200
      statusText: "OK"
```

- Queremos somente o que está dentro da chave chamada **data**
- Modifique o arquivo **index.jsx** de seu frontend

```
1 import { useEffect } from 'react'
2 import './style.css'
3 import Lixeira from '../assets/lixeira_peq.svg'
4 import api from '../services/api'
5
6 function Home() {
7   let usuarios = []
8
9   async function getUsuarios(){
10     const usuariosDaApi = await api.get('/cadastro')
11     usuarios = usuariosDaApi.data
12     console.log(usuarios)
13   }
14
15   useEffect(()=>{
16     getUsuarios()
17   },[])
18
19   return (
20     <div className='container'>
21       <form>
```

```
index.jsx:12
(7) [{id: '6899ed759774a6b9ae8b0226', email: 'anderson@email.com', nome: 'Anderson'}, {id: '6899f68295b2592bf9afe0ae', email: 'maria@email.com', nome: 'Maria'}, {id: '689b2da6b6c8f79385d6bab1', email: 'teste@email.com', nome: 'Jub'}, {id: '689b4c1aac8b8a99d46a77e3', email: 'qualquer@email.com', nome: 'Qualquer'}, {id: '68a337740151571de01c94b8', email: 'fulana@email.com', nome: 'Fulana'}, {id: '68a46267a0332343577c8192', email: 'joaquina@email.com', nome: 'Joaquina'}, {id: '68a4740224b592369e6344d6', email: 'etec@email.com', nome: 'Etec'}]
[[Prototype]]: Array(7)

index.jsx:12
(7) [{id: '6899ed759774a6b9ae8b0226', email: 'anderson@email.com', nome: 'Anderson'}, {id: '6899f68295b2592bf9afe0ae', email: 'maria@email.com', nome: 'Maria'}, {id: '689b2da6b6c8f79385d6bab1', email: 'teste@email.com', nome: 'Jub'}, {id: '689b4c1aac8b8a99d46a77e3', email: 'qualquer@email.com', nome: 'Qualquer'}, {id: '68a337740151571de01c94b8', email: 'fulana@email.com', nome: 'Fulana'}, {id: '68a46267a0332343577c8192', email: 'joaquina@email.com', nome: 'Joaquina'}, {id: '68a4740224b592369e6344d6', email: 'etec@email.com', nome: 'Etec'}]
[[Prototype]]: Array(7)
```

- Para que a tela seja atualizada com estes novos dados dentro da variável **usuarios**, vamos precisar utilizar outro Hook do React chamado **useState**.
- Altere o código do arquivo **index.jsx** para utilizar este novo **hook**.

```
EXPLORER
▼ CADASTRO-DE-USUARIOS
  ▼ etecmcm-cadastro-u...
    > node_modules
    > public
    ▼ src
      > assets
      ▼ pages\Home
        index.jsx 1, M
        # style.css
      ▼ services
        JS api.js U
        # index.css
        # main.jsx
        .gitignore
        eslint.config.js
        index.html
        package-lock.json M
        package.json M
        README.md
        vite.config.js

index.jsx 1, M X
etecmcm-cadastro-usuarios > src > pages > Home > index.jsx > Home > useEffect() callback
1 | import { useEffect, useState } from 'react'
2 | import './style.css'
3 | import Lixeira from '../../assets/lixeira_peq.svg'
4 | import api from '../../services/api'
5 |
6 | function Home() {
7 |   const [usuarios, setUsuarios] = useState([])
8 |   //let usuarios = []
9 |
10 |   async function getUsers(){
11 |     const usuariosDaApi = await api.get('/cadastro')
12 |     //setUsuarios = usuariosDaApi.data
13 |     setUsuarios(usuariosDaApi.data)
14 |     console.log(usuarios)
15 |   }
16 |
17 |   useEffect(()=>{
18 |     getUsers()
19 |   },[])
20 |
21 |
22 |   return (
23 |     <div className='container'>
24 |       <form>
25 |         <h1>Cadastro de Usuários</h1>
```

## Usuários

Cadastrar

**Nome:** Anderson  
**Idade:** 51  
**Email:** anderson@email.com



**Nome:** Maria  
**Idade:** 35  
**Email:** maria@email.com



**Nome:** Jubileu  
**Idade:**  
**Email:** teste@email.com





Na próxima aula vamos alterar este  
frontend para cadastrar um novo usuário...