




CRUD JSON Server

Prof. Bruno Torres

Sistema de Anotações

Objetivos:

-  Cadastro e login de usuários
-  CRUD de anotações (data, título, descrição)
-  Persistência com JSON Server

Configuração Inicial

Estrutura do Projeto

```
projeto/  
├─ db.json      # Banco de dados  
├─ index.html   # Página inicial  
├─ style.css    # Estilos  
└─ script.js    # Lógica principal
```

Instalando o JSON Server

1. Instalar o JSON Server:

```
npm install -g json-server
```

2. Iniciar o servidor:

```
json-server --watch db.json --port 3000
```

Resolvendo Erro de Execução no Windows

Se ocorrer um erro de permissão ao iniciar o JSON Server no Windows, siga este procedimento:

1. Abrir o PowerShell como Administrador.
2. Executar o seguinte comando para permitir a execução de scripts:

```
Set-ExecutionPolicy RemoteSigned -Scope CurrentUser
```

Isso permite a execução de scripts confiáveis no PowerShell.

3. Tente iniciar o JSON Server novamente com o comando:

```
json-server --watch db.json --port 3000
```

Banco de Dados (db.json)

```
{
  "users": [{
    "id": 1,
    "name": "João",
    "login": "joao123",
    "password": "senha123"
  }],
  "notes": [{
    "id": 1,
    "userId": 1,
    "date": "2023-11-20",
    "title": "Reunião",
    "description": "Com cliente às 14h"
  }]
}
```

IDs são gerados automaticamente!

Cadastro de Usuário

HTML:

```
<input type="text" id="name" placeholder="Nome">  
<input type="text" id="login" placeholder="Login">  
<input type="password" id="password" placeholder="Senha">  
<button onclick="register()">Cadastrar</button>
```

Cadastro de Usuário

JavaScript:

```
async function register() {  
  const user = {  
    name: document.getElementById("name").value,  
    login: document.getElementById("login").value,  
    password: document.getElementById("password").value  
  };  
  await fetch("http://localhost:3000/users", {  
    method: "POST",  
    headers: { "Content-Type": "application/json" },  
    body: JSON.stringify(user)  
  });  
}
```


Login Simples

JavaScript:

```
async function login() {  
  const login = document.getElementById("login").value;  
  const password = document.getElementById("password").value;  
  
  const res = await fetch(`http://localhost:3000/users?login=${login}&password=${password}`);  
  const users = await res.json();  
  
  if (users.length > 0) {  
    localStorage.setItem("userId", users[0].id);  
    window.location.href = "notes.html"; // Redireciona  
  } else {  
    alert("Login falhou!");  
  }  
}
```

Listagem de Notas: HTML

```
<table id="notesTable">
  <thead>
    <tr>
      <th>Data</th>
      <th>Título</th>
      <th>Descrição</th>
      <th>Ações</th>
    </tr>
  </thead>
  <tbody id="tableBody"></tbody>
</table>
```

Listagem de Notas : JavaScript

```
async function loadNotes() {  
  const userId = localStorage.getItem("userId");  
  
  const res = await fetch(`http://localhost:3000/notes?userId=${userId}`);  
  
  const notes = await res.json();  
  
  const tableBody = document.getElementById("tableBody");  
  
  tableBody.innerHTML = notes.map(note => `  
    <tr>  
      <td>${note.date}</td>  
      <td>${note.title}</td>  
      <td>${note.description}</td>  
      <td>  
        <button onclick="deleteNote(${note.id})">Excluir</button>  
        <button onclick="editNote(${note.id})">Editar</button>  
      </td>  
    </tr>  
  `).join("");  
}
```

CRUD Completo

1. Adicionar Nota

```
async function addNote() {  
  const note = {  
    userId: localStorage.getItem("userId"),  
    date: document.getElementById("date").value,  
    title: document.getElementById("title").value,  
    description: document.getElementById("description").value  
  };  
  await fetch("http://localhost:3000/notes", {  
    method: "POST",  
    headers: { "Content-Type": "application/json" },  
    body: JSON.stringify(note)  
  });  
  loadNotes();  
}
```

Conclusão

- Criamos um sistema funcional com **cadastro, login e gerenciamento de notas**.
- Utilizamos **JSON Server** para simular um backend.
- Implementamos um **CRUD completo** para as anotações.

Agora, é possível expandir esse sistema com novas funcionalidades, como autenticação JWT e melhorias na interface!