

# PRÁTICAS AVANÇADAS EM DESENVOLVIMENTO WEB

Davi Schneid - [davi.Schneid@gmail.com](mailto:davi.Schneid@gmail.com)

12/08/2024

# Agenda

- ▶ Criar a feature de Esqueci minha senha
- ▶ Enviar e-mail para resetar a senha
- ▶ Gerar token para resetar senha
- ▶ Validar se o token é valido
- ▶ Registrar a solicitação de resete de senha
- ▶ Criar o formulário
  - ▶ Esqueci minha senha
  - ▶ Resetar senha

# APIWeb

- ▶ Criar a página EsqueciMinhaSenha.js dentro da pasta páginas
- ▶ Escrever o código

```
src > paginas > JS EsqueciMinhaSenha.js > [E] EsqueciMinhaSenha

1
2 import React, { useState } from 'react';
3 import axios from 'axios';
4 import '../CSS/login.css';
5 import BotaoVoltar from '../componentes/BotaoVoltar';
6 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
7 import { faUser } from '@fortawesome/free-solid-svg-icons';
8
9 const EsqueciMinhaSenha = () => {
10   const [email, setEmail] = useState('');
11   const [mensagem, setMensagem] = useState('');
12   const handlePasswordReset = async () => {
13
14     if (!email) {
15       setMensagem('Informar o e-mail');
16     } else {
17       try {
18         const response = await axios.post('http://localhost:3001/api/esqueci-minha-senha', { email });
19         setMensagem('Um link de recuperação de senha foi enviado para o seu e-mail.');
```

# APIWeb

- ▶ Atualizar a página Rotas.js na pasta Rotas
- ▶ Escrever o código, adicionando a nova rota para esqueci minha senha.

```
<Route path="/cadastro" element={<Cadastro />} />  
<Route path="/login" element={<Login />} />  
<Route path="/esqueci-minha-senha" element={<EsqueciMinhaSenha />} />
```

- ▶ Atualizar a página de Login.js, adicionando o link para a página esqueci minha senha


```
<br></br>  
<Link to="/esqueci-minha-senha">Esqueceu a senha?</Link>  
<div className="button-container">  
  <button onClick={login} className="button">Login</button>  
</div>  
<br></br>  
<Link to="/cadastro">Acessar cadastro</Link>  
</div>
```

- ▶ A página de `esqueciMinhaSenha.js` deverá ficar assim.



The image shows a mockup of a web form for forgetting a password. It is centered on a light gray background. At the top, the title 'Esqueci Minha Senha' is displayed in a bold, black, sans-serif font. Below the title is a white input field with a thin gray border. Inside the input field, on the left, is a small gray user icon, followed by the placeholder text 'Digite seu e-mail' in a gray font. Below the input field is a prominent blue rectangular button with the word 'Enviar' in white, bold, sans-serif font. At the bottom of the form area is a link labeled 'Voltar' in a gray font.

**Esqueci Minha Senha**

 Digite seu e-mail

**Enviar**

[Voltar](#)

# APIRouter

- ▶ Criar uma nova rota para esqueci minha senha no usuarioRotas.js
- ▶ Escrever o código

```
const EsqueciMinhaSenha = require('../modelo/EsqueciMinhaSenha');
```

```
249
250 //cria a rota de esqueci minha senha
251 /**
252  * @swagger
253  * /esqueci-minha-senha:
254  *   post:
255  *     summary: Recuperar senha do usuário
256  *     tags: [Recuperar senha]
257  *     requestBody:
258  *       required: true
259  *       content:
260  *         application/json:
261  *           schema:
262  *             type: object
263  *             properties:
264  *               email:
265  *                 type: string
266  *                 description: O email do usuário
267  *     responses:
268  *       200:
269  *         description: Dados encontrados
270  *         content:
271  *           application/json:
272  *             schema:
273  *               type: object
274  *               properties:
275  *                 token:
276  *                   type: string
277  *       400:
278  *         description: Dados não encontrados ou e-mail inválido
279  *       500:
280  *         description: Erro ao recuperar a senha
281  */
282 router.post('/esqueci-minha-senha', usuarioController.esqueciMinhaSenha);
283
```

# APIRouter

- ▶ Criar a função esqueciMinhaSenha no UsuarioController.js
- ▶ Escrever o código

```
182
183 // buscar por ID do usuário
184 exports.esqueciMinhaSenha = async (req, res) => {
185   console.log('esqueciMinhaSenha');
186   const { email } = req.body;
187   try {
188     const usuario = await Usuario.findOne({ where: { email } });
189
190     if (usuario) {
191       const token = gerarToken(usuario, '10m');
192       const usuarioId = usuario.id;
193       const esqueciMinhaSenha = await EsqueciMinhaSenha.create({ usuarioId, email, token });
194       // Enviar e-mail com o link de recuperação
195       const assunto = 'Recuperar senha'
196       const resetUrl = `http://localhost:3000/resetar-senha/${esqueciMinhaSenha.token}`;
197       const mensagem = `
198         <h1>Você solicitou a redefinição de senha</h1>
199         <p>Clique no link abaixo para redefinir sua senha:</p>
200         <a href="${resetUrl}" clicktracking=off>${resetUrl}</a>
201       `;
202
203       const response = await axios.post('http://localhost:3001/api/enviaremail', {
204         destinatario: email,
205         assunto: assunto,
206         mensagem: mensagem
207       });
208
209       if (response.status === 200) {
210         console.log('E-mail de recuperação enviado com sucesso.');
```

# APIRouter

- Atualizar o arquivo enviarEmailService.js na pasta service
- Escrever o código

```
service > JS enviarEmailService.js > enviarEmailService > mailOptions > html
1
2
3  const nodemailer = require('nodemailer');
4
5  // Configura o transporte de email
6  const transporter = nodemailer.createTransport({
7    host: 'smtp.gmail.com',
8    port: 465,
9    secure: true, // true para usar SSL/TLS
10   service: 'gmail', // ou outro serviço como Yahoo, Outlook, etc.
11   auth: {
12     user: process.env.USER_EMAIL, // Seu email
13     pass: process.env.PWD_EMAIL_GMAIL_APP // Sua senha de email ou senha de app
14   }
15 });
16
17 // Função para enviar email
18 function enviarEmailService(destinatario, assunto, mensagem) {
19   console.log('EnviarEmailService.destinatario',destinatario);
20   const mailOptions = {
21     from: process.env.USER_EMAIL, // Remetente
22     to: destinatario, // Destinatário
23     subject: assunto, // Assunto do email
24     html: mensagem // Conteúdo do email
25   };
26
27   transporter.sendMail(mailOptions, (error, info) => {
28     if (error) {
29       console.log('Erro ao enviar email:', error);
30     } else {
31       console.log('Email enviado com sucesso:', info.response);
32     }
33   });
34 }
35
36 module.exports = { enviarEmailService };
```



# APIRouter

- Atualizar o arquivo enviarMensagemRotas.js na pasta service
- Escrever o código

```
91
92 //cria a rota de enviar Email
93 /**
94  * @swagger
95  * /enviaremail:
96  *   post:
97  *     summary: Enviar mensagem via E-mail
98  *     tags: [Envio de mensagens]
99  *     requestBody:
100  *       required: true
101  *       content:
102  *         application/json:
103  *           schema:
104  *             type: object
105  *             properties:
106  *               destinatario:
107  *                 type: string
108  *                 description: Destinatario que recebera o e-mail
109  *               assunto:
110  *                 type: string
111  *                 description: Assunto do e-mail
112  *               mensagem:
113  *                 type: string
114  *                 description: Mensagem do e-mail
115  *     responses:
116  *       200:
117  *         description: E-mail enviada com sucesso
118  *         content:
119  *           application/json:
120  *             schema:
121  *               type: object
122  *               properties:
123  *                 retorno:
124  *                   type: string
125  *       400:
126  *         description: Erro ao enviar o e-mail ou e-mail inválido
127  *       500:
128  *         description: Erro ao enviar a e-mail
129  *   /
130 router.post('/enviaremail', enviarMensagemController.enviarEmail);
```

# APIRouter

- ▶ Criar o arquivo EsqueciMinhaSenha.js na pasta modelo
- ▶ Escrever o código

```
modelo > JS EsqueciMinhaSenha.js > ...
1
2 const Sequelize = require('sequelize');
3 const database = require('../data_base/db');
4 const Usuario = require('../Usuario');
5
6 const EsqueciMinhaSenha = database.define('esqueci_minha_senha', {
7   id: {
8     type: Sequelize.INTEGER,
9     autoIncrement: true,
10    allowNull: false,
11    primaryKey: true
12  },
13  usuarioId: {
14    type: Sequelize.INTEGER,
15    allowNull: false,
16    references: {
17      model: Usuario, // Model ao qual se refere
18      key: 'id' // Chave da tabela User que está sendo referenciada
19    },
20    onUpdate: 'CASCADE', // Ação ao atualizar o id do usuário
21    onDelete: 'CASCADE', // Ação ao deletar o usuário
22  },
23  email: {
24    type: Sequelize.STRING,
25    allowNull: false
26  },
27  token: {
28    type: Sequelize.STRING,
29    allowNull: false
30  }
31 }, {
32   tableName: 'esqueci_minha_senha', // Nome da tabela no banco de dados
33   // Configurações do modelo
34   timestamps: true, // Habilita createdAt e updatedAt
35   hooks: {
36     beforeCreate: (esqueciMinhaSenha, options) => {
37       const now = new Date();
38       const threeHoursLater = new Date(now.getTime() - 3 * 60 * 60 * 1000);
39       esqueciMinhaSenha.createdAt = threeHoursLater;
40       esqueciMinhaSenha.updatedAt = threeHoursLater;
41     },
42     beforeUpdate: (esqueciMinhaSenha, options) => {
43       const now = new Date();
44       const threeHoursLater = new Date(now.getTime() - 3 * 60 * 60 * 1000);
45       esqueciMinhaSenha.updatedAt = threeHoursLater;
46     }
47   }
48 });
49
50
51 // Definir a associação entre Usuario e EsqueciMinhaSenha
52 Usuario.hasMany(EsqueciMinhaSenha, { foreignKey: 'usuarioId' });
53 EsqueciMinhaSenha.belongsTo(Usuario, { foreignKey: 'usuarioId' });
54
55 module.exports = EsqueciMinhaSenha;
```

# APIRouter

- ▶ Testar a api esqueci-minha-senha no Swagger

## Recuperar senha

POST

/esqueci-minha-senha

Recuperar senha do usuário

Parameters

No parameters

Request body

required

application/json

```
{  "email": "string"}|
```

+

G

Execute

▶ Vai receber e-mail para recuperar a senha.



turmamiltalentos@gmail.com

para mim ▼

 Traduza para o português

**Você solicitou a redefinição de senha**

Clique no link abaixo para redefinir sua senha:

[http://localhost:3000/resetar-senha/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c3VhcmliSWQIQiMsImhhdCI6MTcyMzU1OTEzNiwiZXhwIjoxNzIzNTU5NzY2fQ.3gdPIQx6kBbmhyULeeDTup3dJfY\\_bAP1g734-vCcEJY](http://localhost:3000/resetar-senha/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c3VhcmliSWQIQiMsImhhdCI6MTcyMzU1OTEzNiwiZXhwIjoxNzIzNTU5NzY2fQ.3gdPIQx6kBbmhyULeeDTup3dJfY_bAP1g734-vCcEJY)

← Responder

→ Encaminhar



# APIWeb

- ▶ Criar a página ResetarSenha.js dentro da pasta páginas
- ▶ Escrever o código

src > paginas > JS ResetarSenha.js > [🔍] ResetarSenha

```
1  import React, { useState } from 'react';
2  import axios from 'axios';
3  import { useParams } from 'react-router-dom';
4  import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
5  import { faLock } from '@fortawesome/free-solid-svg-icons';
6  import { Link } from 'react-router-dom';
7
8  const ResetarSenha = () => {
9
10     const [senha, setSenha] = useState('');
11     const [confirmaSenha, setConfirmaSenha] = useState('');
12     const [mensagem, setMensagem] = useState('');
13     const { token } = useParams();
14
15     const handleSubmit = async (event) => {
16         event.preventDefault();
17         if (senha !== confirmaSenha) {
18             setMensagem('As senhas não coincidem');
19             return;
20         }
21
22         try {
23             const response = await axios.post(`http://localhost:3001/api/resetarsenha`, {
24                 senha,
25                 token
26             });
27             console.log('response', response.status);
28
29             if (response.status === 200) {
30                 console.log('200');
31                 setMensagem('Senha redefinida com sucesso');
32             }
33
34         } catch (error) {
35             setMensagem('Token invalido!', error);
36         }
37     };
38 }
```

# APIWeb

## ► Continuação do código da página ResetarSenha.js


```
38
39   return (
40     <div className="login-container">
41       <div className="login-box">
42         <h2>Redefinir Senha</h2>
43         <form onSubmit={handleSubmit}>
44
45           <div className="input-container">
46             <FontAwesomeIcon icon={faLock} className="input-icon" />
47             <input
48               type="password"
49               placeholder="Senha"
50               value={senha}
51               onChange={(e) => setSenha(e.target.value)}
52               required
53             />
54           </div>
55           <br></br>
56           <div className="input-container">
57             <FontAwesomeIcon icon={faLock} className="input-icon" />
58             <input
59               type="password"
60               placeholder="Confirmar senha"
61               value={confirmaSenha}
62               onChange={(e) => setConfirmaSenha(e.target.value)}
63               required
64             />
65           </div>
66           <br></br>
67           <button type="submit" className="button">Redefinir Senha</button>
68         </form>
69         {mensagem && <p style={{ color: 'red' }}>{mensagem}</p>}
70         <br></br>
71         <Link to="/login">Login</Link>
72       </div>
73     </div>
74   );
75 };
76 export default ResetarSenha;
```


- ▶ A página resetar senha ficara assim:



The image shows a web form for resetting a password. It is titled "Redefinir Senha" in bold black text. Below the title are two input fields, each with a lock icon on the left. The first field is labeled "Senha" and the second is labeled "Confirmar senha". Below these fields is a large blue button with the text "Redefinir Senha" in white. At the bottom of the form is a link labeled "Login" in purple text.

**Redefinir Senha**

 Senha

 Confirmar senha

**Redefinir Senha**

[Login](#)

# APIWeb

- ▶ Atualizar a página Rotas.js na pasta Rotas
- ▶ Escrever o código, adicionando a nova rota para resetar senha.

```
<Route path="/cadastro" element={<Cadastro />} />
<Route path="/login" element={<Login />} />
<Route path="/esqueci-minha-senha" element={<EsqueciMinhaSenha />} />
<Route path="/resetar-senha/:token" element={<ResetarSenha />} />
```

# APIRouter

- ▶ Criar uma nova rota para resetar a senha no usuarioRotas.js
- ▶ Escrever o código

```
285
286 //cria a rota para resetar a senha
287 /**
288  * @swagger
289  * /resetarsenha:
290  *   post:
291  *     summary: Redefinir senha do usuário
292  *     description: Redefine a senha do usuário usando um token de recuperação.
293  *     tags:
294  *       - Recuperar senha
295  *     requestBody:
296  *       required: true
297  *       content:
298  *         application/json:
299  *           schema:
300  *             type: object
301  *             required:
302  *               - token
303  *               - senha
304  *             properties:
305  *               token:
306  *                 type: string
307  *                 description: Token de recuperação de senha
308  *               senha:
309  *                 type: string
310  *                 description: Nova senha do usuário
311  *     responses:
312  *       200:
313  *         description: Senha redefinida com sucesso
314  *       401:
315  *         description: Token inválido ou expirado
316  *       500:
317  *         description: Erro ao resetar a senha
318  */
319 router.post('/resetarsenha', usuarioController.resetarSenha);
320
321 //exporta as rotas criadas
322 module.exports = router;
```



# APIRouter

- ▶ Criar a função resetarSenha no UsuarioController.js
- ▶ Escrever o código

```
231
232 // Resetar senha do usuário
233 exports.resetarSenha = async (req, res) => {
234
235   const { senha } = req.body;
236   const { token } = req.body;
237
238   try {
239     const esqueciMinhaSenha = await EsqueciMinhaSenha.findOne({ where: { token } });
240
241     if (esqueciMinhaSenha) {
242
243       try {
244         const response = await axios.post('http://localhost:3001/api/validarToken', { token: token });
245
246         if (response.status === 200) {
247           console.log('Token valido.....');
248           const usuario = await Usuario.findByPk(esqueciMinhaSenha.usuarioId);
249           if (usuario) {
250             const hashedPassword = getHashedPassword(senha);
251             usuario.updatedAt = new Date();
252             usuario.senha = hashedPassword;
253             await usuario.save();
254             res.status(200).json('Senha alterada com sucesso');
255           }
256         }
257         else {
258           console.log('Token invalido');
259           res.status(500).json({ error: 'Token invalido' });
260         }
261       } catch (erro) {
262         res.status(401).json({ error: 'Token invalido' });
263       }
264     } else {
265       console.log('Dados não encontrados');
266       res.status(401).json({ error: 'Dados não encontrados' });
267     }
268   } catch (err) {
269     console.log('Erro ao resetar a senha', err);
270     res.status(500).json({ error: 'Erro ao resetar a senha' });
271   }
272 }
273 ;
```

# APIRouter

- ▶ Testar a api resetar-senha no Swagger
- ▶ Usar o token gerado no esqueci-minha-senha
- ▶ Informar a nova senha

The image shows a Swagger UI interface for a REST API endpoint. The endpoint is a POST request to `/resetarsenha` with the description "Redefinir senha do usuário". The description text says "Redefine a senha do usuário usando um token de recuperação." There is a "Parameters" section which is currently empty, with a "Cancel" button to its right. Below that is the "Request body" section, which is marked as "required" and has a dropdown menu set to "application/json". The request body is a JSON object: 

```
{  "token": "string",  "senha": "string"}
```

 At the bottom of the interface is a large blue button labeled "Execute".

**POST** `/resetarsenha` Redefinir senha do usuário

Redefine a senha do usuário usando um token de recuperação.

**Parameters** Cancel

No parameters

**Request body** *required* application/json

```
{  "token": "string",  "senha": "string"}
```

**Execute**

# Resumo

- ▶ Usuario solicita o resete de senha
- ▶ Recebe um e-mail para resetar a senha, com token gerado para expirar em X minutos
- ▶ O usuário abre o e-mail informado, utiliza o link para resetar a senha.
- ▶ O link abre a página para resetar a senha.
- ▶ Após resete, o usuário deverá utilizar a nova senha.