PRÁTICAS AVANÇADAS EM DESENVOLVIMENTO WEB

Davi Schneid – davi.Schneid@gmail.com

05/08/2024

Agenda

- Salvar o upload de arquivos no backend
  - Salvar arquivos em um diretório temporário
  - Salvar no banco de dados
  - Buscar arquivos

- Upload de arquivos na APIWEB
  - Criar componente para efetuar upload de arquivos
  - Criar componente para puscar arquivos na APIWEB

APIROUTER

- Salvar a imagem no backend
  - **Instalar npm install multer**

```
PS C:\Users\SenacRs\PraticasAvancadasDesenvolvimentoWeb\ApiRouter> npm install multer

added 18 packages, and audited 166 packages in 4s

17 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

  - Criar pasta chamada Multer para a configuração do Multer
  - Criar arquivo de configuração multer.js dentro da pasta Multer
  - Criar arquivo UploadArquivosRota.js com a nova rota de upload dentro da pasta rotas
  - Definir duas rota /uploadarquivo e /uploadarquivo/:id
  - Criar o modelo Upload para salvar os arquivos no banco de dados
  - Criar o controller uploadArquivosController.js dentro da pasta controller
  - Criar uma pasta uploads na raiz do projeto para salvar os arquivos de forma temporária

APIROUTER

- Criar a pasta Multer na raiz do projeto backend
  - Criar o arquivo multer.js , escrever o código abaixo:

Multer > JS multer.js > storage

```
1
2   //Middleware para lidar com uploads de arquivos.
3   const multer = require('multer');
4
5   const path = require('path');
6
7   // Configuração do Multer
8   const storage = multer.diskStorage({
9       destination: (req, file, cb) => {
10        cb(null, path.join(__dirname, '../uploads/'));  // Diretório onde os arquivos serão armazenados
11      },
12      filename: (req, file, cb) => {
13        cb(null, Date.now()+path.extname(file.originalname));  // Nome do arquivo
14      },
15      limits: { fileSize: 50 * 1024 * 1024 }  // Limite de tamanho de arquivo de 50MB, por exemplo
16    });
17
18    const upload = multer({ storage });
19
20    module.exports = upload;
```

APIROUTER

- Criar o arquivo uploadArquivoRotas.js dentro da pasta rotas escrever o código abaixo:

```
//Importa o modulo Express
const express = require('express');
const uploadarquivocontroller  = require('../controllers/uploadArquivosController');

const upload = require('../Multer/multer');  // Importa a configuração do multer

//Cria o objeto rotas
const router = express.Router();

//criar a rota para salvar o upload de arquivos
/**
 * @swagger
 * /uploadarquivo:
 *   post:
 *     summary: Salvar o upload de arquivo
 *     tags: [Upload]
 *     requestBody:
 *       required: true
 *       content:
 *         multipart/form-data:
 *           schema:
 *             type: object
 *             properties:
 *               image:
 *                 type: string
 *                 format: binary
 *     responses:
 *       201:
 *         description: Upload de imagem efetuado com sucesso
 *       500:
 *         description: Erro ao efetuar o upload de imagem
 */
router.post('/uploadarquivo', upload.single('image'), uploadarquivocontroller.uploadarquivo);

```

APIROUTER

- Continuação do código do uploadArquivoRotas.js

```
35
36  /**
37   * @swagger
38   * /uploadarquivo/{id}:
39   *   get:
40   *     summary: Buscar um arquivo pelo ID
41   *     tags: [Upload]
42   *     parameters:
43   *       - in: path
44   *         name: id
45   *         required: true
46   *         schema:
47   *           type: integer
48   *         description: ID do arquivo a ser buscado
49   *     responses:
50   *       200:
51   *         description: Arquivo buscado com sucesso
52   *       404:
53   *         description: Arquivo não encontrado
54   *       500:
55   *         description: Erro ao buscar o arquivo
56   */
57  router.get('/uploadarquivo/:id', uploadarquivocontroller.getArquivo);
58
59  //exporta as rotas criadas
60  module.exports = router;
```

APIROUTER

- Criar o arquivo Upload.js dentro da pasta modelo, escrever o código:

```
const Sequelize = require('sequelize');
const database = require('../data_base/db');

const { DataTypes } = require('sequelize');

const Upload = database.define('upload', {
  id: {
    type: Sequelize.INTEGER,
    autoIncrement: true,
    allowNull: false,
    primaryKey: true
  },
  nomeArquivo: {
    type: Sequelize.STRING,
    allowNull: false
  },
  dados: {
    type: DataTypes.BLOB('long'), // Especifica um LONGBLOB
    allowNull: false
  }
}, {

  // Configurações do modelo
  timestamps: true, // Habilita createdAt e updatedAt
  hooks: {
    beforeCreate: (upload, options) => {
      const now = new Date();
      const threeHoursLater = new Date(now.getTime() - 3 * 60 * 60 * 1000);
      upload.createdAt = threeHoursLater;
      upload.updatedAt = threeHoursLater;
    },
    beforeUpdate: (upload, options) => {
      const now = new Date();
      const threeHoursLater = new Date(now.getTime() - 3 * 60 * 60 * 1000);
      upload.updatedAt = threeHoursLater;
    }
  }

})

module.exports = Upload;
```

APIROUTER

- Criar o arquivo uploadArquivosController.js dentro da pasta controller, escrever o código:

controllers > JS uploadArquivosController.js > getArquivo > getArquivo

```
1   const fs = require('fs').promises;
2
3   //Importa o objeto usuario
4   const Upload = require('../modelo/Upload');
5
6
7   // Salvar o upload de arquivo
8   exports.uploadarquivo = async (req, res) => {
9       console.log('uploadarquivo');
10
11      // Lendo o arquivo temporário
12      const diretorioArquivo = req.file.path;
13      const nomeArquivo = req.file.filename;
14      console.log('uploadarquivo.diretorioArquivo:'+diretorioArquivo);
15      console.log('uploadarquivo.nomeArquivo:'+nomeArquivo);
16
17
18      try {
19        const dados = await fs.readFile(diretorioArquivo);
20        console.log('uploadarquivo.Conteudo do Arquivo:'+dados);
21
22          // Certifique-se de que dadosArquivo não é null
23          if (!dados) {
24            throw new Error('dadosArquivo é null');
25          }
26
27        const novoUpload = await Upload.create({ nomeArquivo, dados});
28        res.status(201).json({ message: 'Upload realizado com sucesso', nomeArquivo });
29      } catch (err) {
30        console.log("Erro ao efetuar upload"+err);
31        res.status(500).json({ message: 'Erro ao efetuar upload', nomeArquivo });
32      }finally{
33        console.log("Finally Apaga o arquivo temporario");
34          // Apaga o arquivo temporario
35        await fs.unlink(diretorioArquivo);
36      }
37    };
```

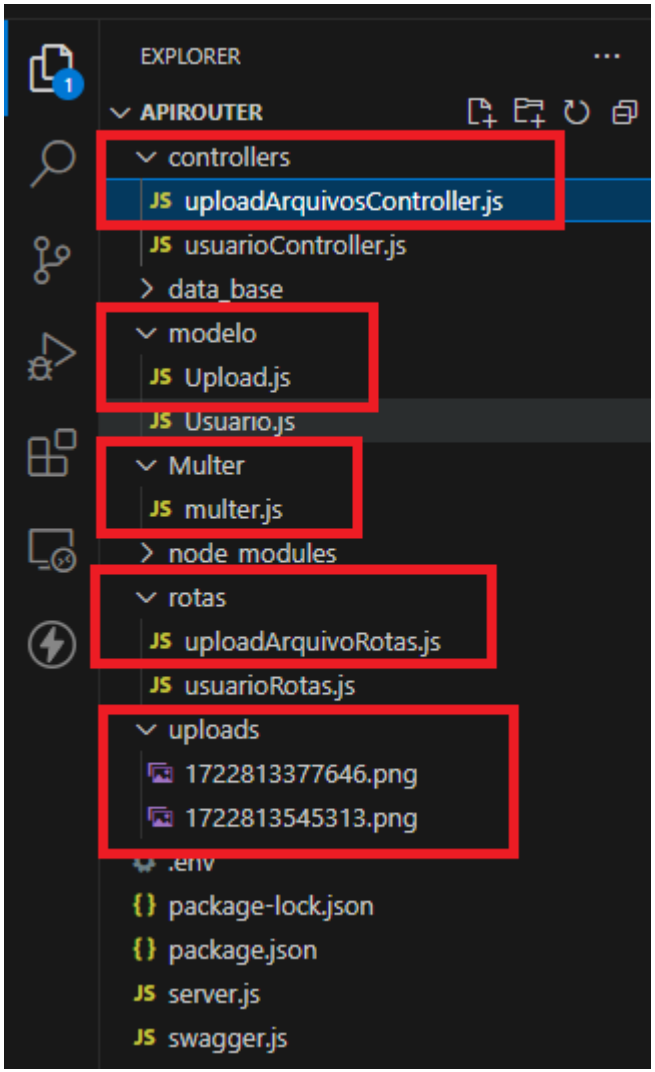
APIROUTER

- Continua o código do arquivo uploadArquivosController.js

```
40  //buscar os arquivos do banco
41  exports.getArquivo = async (req, res) => {
42    try {
43      const id = req.params.id;
44      const upload = await Upload.findByPk(id);
45
46      if (!upload) {
47        return res.status(404).json({ message: 'Arquivo não encontrado' });
48      }
49
50      console.log('Nome do arquivo:', upload.nomeArquivo);
51      console.log('Tamanho do arquivo encontrado:'+upload.dados.byteLength);
52      console.log('Conteúdo do arquivo:', upload.dados);
53
54      // Definindo o tipo de conteúdo apropriado com base na extensão do arquivo
55      let contentType;
56      if (upload.nomeArquivo.endsWith('.jpg') || upload.nomeArquivo.endsWith('.jpeg')) {
57        console.log('image/jpeg');
58        contentType = 'image/jpeg';
59
60      } else if (upload.nomeArquivo.endsWith('.png')) {
61        console.log('image/png');
62        contentType = 'image/png';
63      }if (upload.nomeArquivo.endsWith('.mp3')) {
64        console.log('audio/mpeg');
65          contentType = 'audio/mpeg';
66      } else {
67        console.log('application/octet-stream');
68        contentType = 'application/octet-stream'; // Tipo de conteúdo genérico
69      }
70
71
72      res.set('Content-Type', contentType);
73      //res.set('Content-Disposition', `attachment; filename="${upload.nomeArquivo}"`);
74
75      res.set('Content-Disposition', `inline; filename="${upload.nomeArquivo}"`);
76      res.status(200).end(upload.dados); // Envie os dados binários diretamente
77
78    } catch (err) {
79      console.log("Erro ao buscar o arquivo: " + err);
80      res.status(500).json({ error: 'Erro ao buscar o arquivo' });
81    }
82  };
```

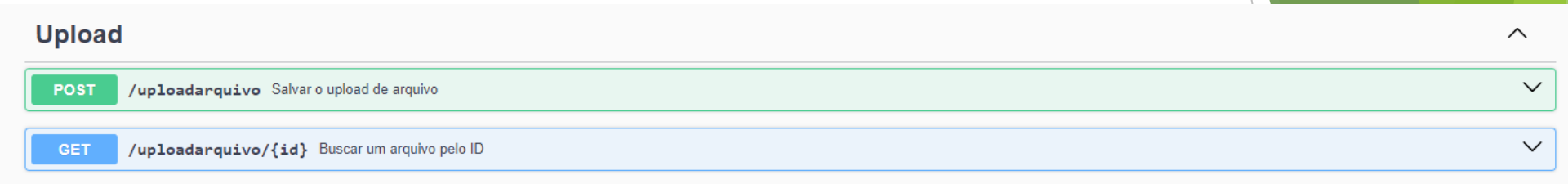
APIROUTER

- Criar a pasta uploads na raiz do projeto



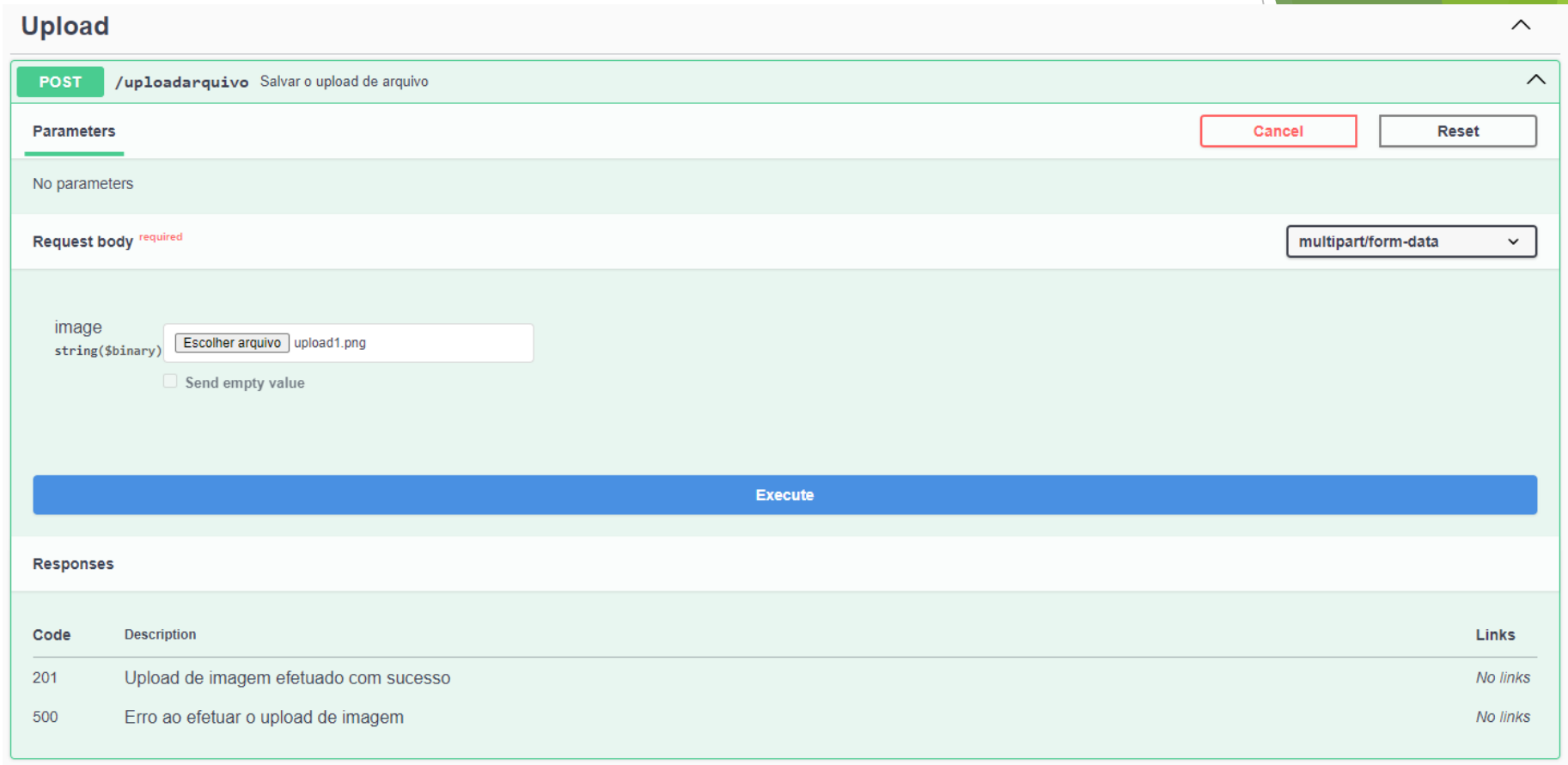
APIROUTER

- Iniciar a aplicação router: npm start
- Validar as novas rotas pelo Swagger



APIROUTER

- Efetuando o upload de arquivo



APIROUTER

- Response do upload

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:3001/api/uploadarquivo' \
  -H 'accept: */*' \
  -H 'Content-Type: multipart/form-data' \
  -F 'image=@upload1.png;type=image/png'
```

Request URL

```
http://localhost:3001/api/uploadarquivo
```

Server response

Code	Details
201	Response body

```
{
  "message": "Upload realizado com sucesso",
  "nomeArquivo": "1722856309727.png"
}
```

Download

Response headers

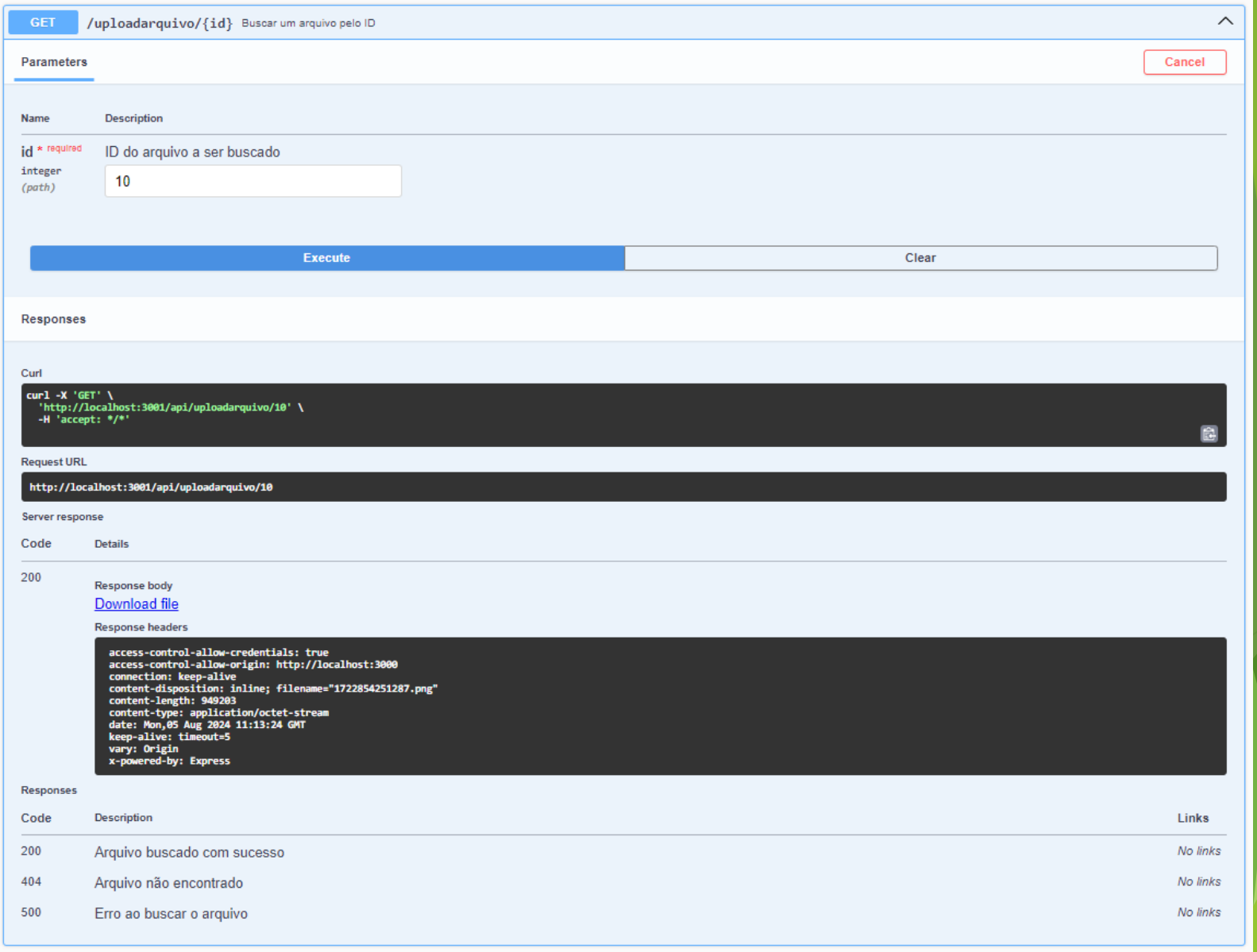
```
access-control-allow-credentials: true
access-control-allow-origin: http://localhost:3000
connection: keep-alive
content-length: 76
content-type: application/json; charset=utf-8
date: Mon,05 Aug 2024 11:11:49 GMT
etag: W/"4c-ZcBCaxeUuFggcPAVqX8KPCZKsoA"
keep-alive: timeout=5
vary: Origin
x-powered-by: Express
```

Responses

Code	Description	Links
201	Upload de imagem efetuado com sucesso	*No links*
500	Erro ao efetuar o upload de imagem	*No links*

APIROUTER

- Buscar o arquivo pelo ID



API-WEB

- Criar o componente de upload na aplicação web
  - Criar o arquivo Upload.js dentro da pasta componentes
  - Escrever o código abaixo

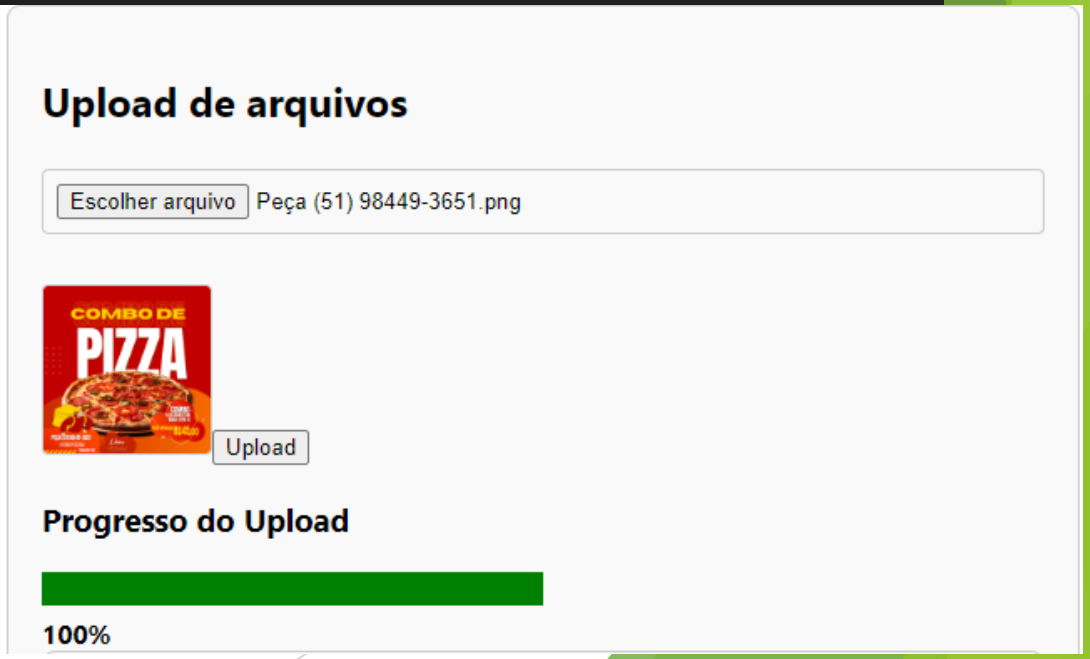
src > componentes > JS Upload.js > default

```jsx
import React, { useState } from 'react';
import axios from 'axios';

import '../App.css';
import BuscarArquivo from './BuscarArquivo';

const Upload = () => {

  //criar o estado do arquivo selecionado
  const [arquivoSelecionado, setArquivoSelecionado] = useState(null);

  //cria o estada da funcao de previsuar
  const [preVisualizacao, setPreVisualizacao] = useState(null);

  //cria o estado do progresso do upload
  const [uploadProgress, setProgressoUpload] = useState(0);

  const handleFileChange = (event) => {
    const file = event.target.files[0];
    setArquivoSelecionado(file);
    setPreVisualizacao(URL.createObjectURL(file));
  };

  const handleFormSubmit = async (event) => {
    event.preventDefault();
    const formData = new FormData();
    formData.append('image', arquivoSelecionado);

    try {
      const response = await axios.post('http://localhost:3001/api/uploadarquivo', formData, {
        headers: {
          'Content-Type': 'multipart/form-data',
        },
        onUploadProgress: (progressEvent) => {
          const progress = Math.round((progressEvent.loaded * 100) / progressEvent.total);
          setProgressoUpload(progress);
        },
      });
      console.log('Upload sucesso:', response.data);
    } catch (error) {
      console.error('Error no uploading da imagem:', error);
    }
  };
```

API-WEB

- Continuação do código:

```
45    return (
46      <div className="form-container">
47        <h2>Upload de arquivos</h2>
48        <form onSubmit={handleFormSubmit}>
49          <input type="file" onChange={handleFileChange} />
50          {preVisualizacao && <img src={preVisualizacao} alt="preVisualizacao" style={{ width: '20px' }} className="preview-image"/>}
51          <button type="submit" className="submit-button" >Upload</button>
52        </form>
53        {uploadProgress > 0 && (
54          <div className="upload-progress">
55            <h3>Progresso do Upload</h3>
56            <progress value={uploadProgress} max="50" />
57            <span>{uploadProgress}%</span>
58          </div>
59        )}
60
61          <BuscarArquivo />
62      </div>
63    );
64  };
65
66  export default Upload;
```



API-WEB

- Adicionar nova rota /upload no arquivo Rotas.js

```
import React from 'react';

//importa 3 objetos da lib
import { Route, Routes, BrowserRouter } from 'react-router-dom';

//Importa a página Home
import Home from '../paginas/Home';

//Importa a página Cadastro
import Cadastro from '../paginas/Cadastro';
import CadastroCEP from '../paginas/CadastroCEP';
import ListaRegistros from '../paginas/ListaRegistros';
import EditarRegistro from '../paginas/EditarRegistro';
import Upload from '../componentes/Upload';

function Rotas() {
    return (
        <BrowserRouter>
            <Routes>
                <Route element={<Home />} path="/" exact component={Home}/>
                <Route element={<Cadastro />} path="/cadastro" component={Cadastro} />
                <Route element={<CadastroCEP />} path="/cadastrocep" component={CadastroCEP} />
                <Route element={<ListaRegistros />} path="/lista" component={ListaRegistros} />
                <Route element={<EditarRegistro />} path="/editar/:id" component={EditarRegistro} />
                <Route element={<Upload />} path="/upload" component={Upload} />
            </Routes>
        </BrowserRouter>
    )
}

export default Rotas;
```

API-WEB

- Criar o componente de buscar arquivo na aplicação web
  - Criar o arquivo BuscarArquivo.js dentro da pasta componentes
  - Escrever o código abaixo

src > componentes > JS BuscarArquivo.js > BuscarArquivo

```
import React, { useState } from 'react';
import './BuscarArquivo.css';


const BuscarArquivo = () => {
  const [fileUrl, setFileUrl] = useState(null);
  const [error, setError] = useState(null);
  const [id, setId] = useState('');
  const [fileType, setFileType] = useState('');


  const encontrarArquivo = async () => {
    try {
      const response = await fetch(`http://localhost:3001/api/uploadarquivo/${id}`);
      if (!response.ok) {
        throw new Error('Erro ao buscar o arquivo');
      }
      const blob = await response.blob();
      const url = URL.createObjectURL(blob);

      console.log('Blob URL:', url);

      setFileUrl(url);
      setError(null);  // Limpa o erro ao buscar com sucesso

      const contentType = response.headers.get("Content-Type");

      console.log("ContentType:",contentType);


      if (contentType.startsWith('image/') || contentType.startsWith('application/octet-stream')) {
        console.log("IMAGEM");
        setFileType('imagem');
      } else if (contentType === 'audio/mpeg') {
        console.log("AUDIO");
        setFileType('audio');
      } else {
        console.log("NAO ENCONTRADO");
        setFileType('outro');
      }
```

API-WEB

- Continuação do código:

```
42          console.log("Tipo de arquivo:",fileType);
43
44        } catch (err) {
45          console.log("Erro ao buscar arquivo");
46          setError(err.message);
47        }
48      };
49
50      return (
51        <div className="buscar-arquivo-container">
52          <h2>Buscar Arquivo</h2>
53          <div className="input-group">
54            <input
55              type="text"
56              placeholder="Informe o ID do arquivo"
57              value={id}
58              onChange={(e) => setId(e.target.value)}
59            />
60            <button onClick={encontrarArquivo}>Buscar</button>
61          </div>
62          {error && <p className="error">{error}</p>}
63
64          {fileType === 'audio' && (
65
66           <div>
67
68           <div>
69             <a href={fileUrl} >Download</a>
70           </div>
71         </div>
72          )}
73
74          {fileType === 'imagem' && (
75            <img src={fileUrl} alt="Arquivo encontrado" className="file-preview" />
76          )}
77
78        </div>
79      );
80    };
81
82    export default BuscarArquivo;
```

API-WEB

- Criar o Buscar.css dentro da pasta componentes.
- Escrever o código abaixo:

src > componentes > # BuscarArquivo.css > .buscar-arquivo-container

```css
.buscar-arquivo-container {
  max-width: 600px;
  margin: 0 auto;
  padding: 20px;
  border: 1px solid #ccc;
  border-radius: 8px;
  background-color: #f9f9f9;
  text-align: center;
}

.buscar-arquivo-container h2 {
  margin-bottom: 20px;
  color: #333;
}

.input-group {
  display: flex;
  justify-content: center;
  align-items: center;
  margin-bottom: 20px;
}

.input-group input {
  width: 60%;
  padding: 10px;
  border: 1px solid #ccc;
  border-radius: 4px;
  margin-right: 10px;
}

.input-group button {
  padding: 10px 20px;
  border: none;
  border-radius: 4px;
  background-color: #007bff;
  color: white;
  cursor: pointer;
  font-size: 1em;
}

.input-group button:hover {
  background-color: #0056b3;
}

.error {
  color: red;
  margin-top: 10px;
}

.file-preview {
  max-width: 100%;
  height: auto;
  margin-top: 20px;
}
```

API-WEB

- Adicionar o link para a página de Upload na Home.js.
- Escrever o código abaixo:

```
import '../App.css';

//Importa o componente Header
import Header from '../Header';

//Importa o componente Header
import Footer from '../Footer';

//Importa o recurso para criar link do react
import {Link} from 'react-router-dom';


function Home() {
  return (
    <div className="App">
      {/* Importamos o componente Header criado como HTML */}
      <Header title="Programa 3 Mil Talentos" />

      <header className="App-header">

        <p>Praticas avançadas em Desenvolvimento Web.</p>

        <Link to="/cadastro">Acessar cadastro</Link>
        <Link to="/lista">Listagem de cadastro</Link>
        <Link to="/upload">Upload de arquivos</Link>

      </header>

      {/* Importamos o componente Footer  criado como HTML */}
      <Footer/>

    </div>
  );
}

export default Home;
```

APIWEB

- Conferir as rotas de upload e buscar arquivo por ID
- Iniciar o projeto Web para validar o upload de arquivos.