

Roteiro

React

Antes de iniciar este roteiro, teremos feito um roteiro base com jsx, babel e webpack.

Ordenação:

Criar projeto

Home

TiposEventos

React Router

Component Rodapé

TiposEventos – Listar da API

TiposEventos – Cadastrar chamando a API

Login

Eventos

Instalar NodeJS

npm install -g create-react-app

create-react-app senai-svigufo-ui

cd senai-svigufo-ui

npm start

Links úteis

<https://medium.com/tableless/o-guia-completo-do-react-e-o-seu-ecossistema-b31a10ecd84f>

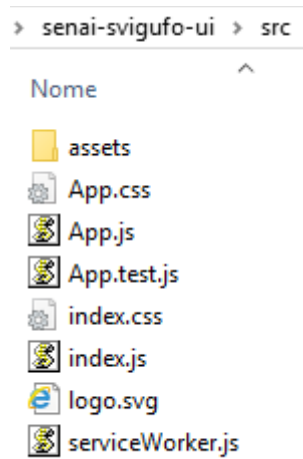
<https://medium.com/tableless/organizando-uma-aplica%C3%A7%C3%A3o-com-react-5b8ea9075596>

<https://medium.com/tableless/webpack-para-react-o-guia-final-cb8a95b369ed>

Criar projeto

Classes

Copiar a pasta assets definida no projeto base e colar dentro de src do projeto do React.



Abrir o App.js e colar o conteúdo dentro de body no render() -> return do conteúdo da home.html do conteúdo base.

Alterar tudo o que for class para className.

Realizar um npm start para verificar se o conteúdo foi gerado.

```
import React, { Component } from "react";
// import logo from "../Logo.svg";
// import "../App.css";

// realizar o import dos estilos e logo
import './assets/css/flexbox.css';
import './assets/css/reset.css';
import './assets/css/style.css';

class App extends Component {
  render() {
    return (
      <div>
        <header className="cabecalhoPrincipal">
          <div className="container">
            

            <nav className="cabecalhoPrincipal-nav">
              <a>Home</a>
              <a>Eventos</a>
              <a>Contato</a>
              <a className="cabecalhoPrincipal-nav-login" href="login.html">
                Login
              </a>
            </nav>
          </div>
        </header>
      </div>
    );
  }
}
```

Importando o ícone e o incluindo na tela.

```
import React, { Component } from "react";
// import logo from "../Logo.svg";
// import "../App.css";

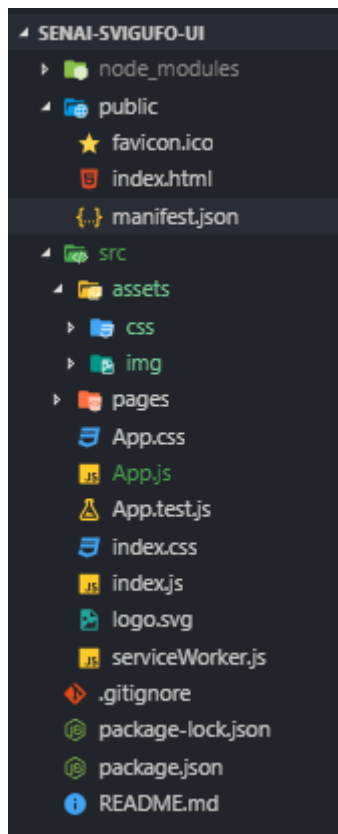
import logo from '../assets/img/icon-login.png';

// realizar o import dos estilos e logo
import './assets/css/flexbox.css';
import './assets/css/reset.css';
import './assets/css/style.css';

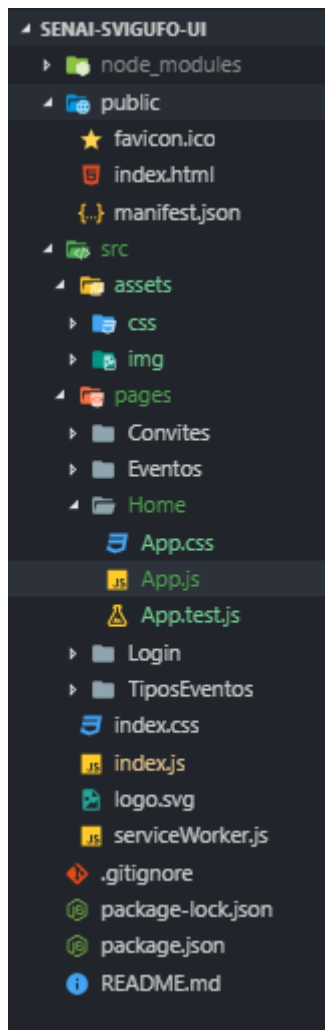
class App extends Component {
  render() {
    return (
      <div>
        <header className="cabecalhoPrincipal">
          <div className="container">
            <img src={logo} alt="Svigufo - ícone de login"/>
          </div>
        </header>
      </div>
    );
  }
}
```

Após termos colocado a home, vamos realizar um cadastro de tipo de evento para verificar que não há diferença entre eles.

Para melhor organização, vamos criar uma pasta chamada pages.



E aproveitar, criar subpastas para organizar os nossos códigos.



Alterar o index.js para encontrar a página de Home

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './pages/Home/App';
import * as serviceWorker from './serviceWorker';

ReactDOM.render(<App />, document.getElementById('root'));

// If you want your app to work offline and load faster, you can change
// unregister() to register() below. Note this comes with some pitfalls.
// Learn more about service workers: https://bit.ly/CRA-PWA
serviceWorker.unregister();
```

App.js

```
import React, { Component } from "react";
// import logo from "../Logo.svg";
// import "../App.css";

import logo from '../../assets/img/icon-login.png';

// realizar o import dos estilos e logo
import '../../assets/css/flexbox.css';
import '../../assets/css/reset.css';
import '../../assets/css/style.css';
```

Criar TiposEventos.js dentro da pasta de pages/TiposEventos com o conteúdo abaixo.

```
import React, { Component } from "react";

class TiposEventos extends Component {
  render() {
    return (
      <div>
        TiposEventos
      </div>
    );
  }
}

export default TiposEventos;
```

Para que essa página possa ser acessada, precisamos instalar o react-router.

<https://medium.com/collabcode/roteamento-no-react-com-os-poderes-do-react-router-v4-fbc191b9937d>

Parar o projeto e executar o seguinte comando. Para dar suporte à navegação do projeto.

```
npm install --save react-router-dom
```

index.js

```
import React from "react";
import ReactDOM from "react-dom";
import { Route, Link, BrowserRouter as Router } from "react-router-dom";
import "./index.css";
import App from "./pages/Home/App";
import TiposEventos from "./pages/TiposEventos/TiposEventos";
import * as serviceWorker from "./serviceWorker";

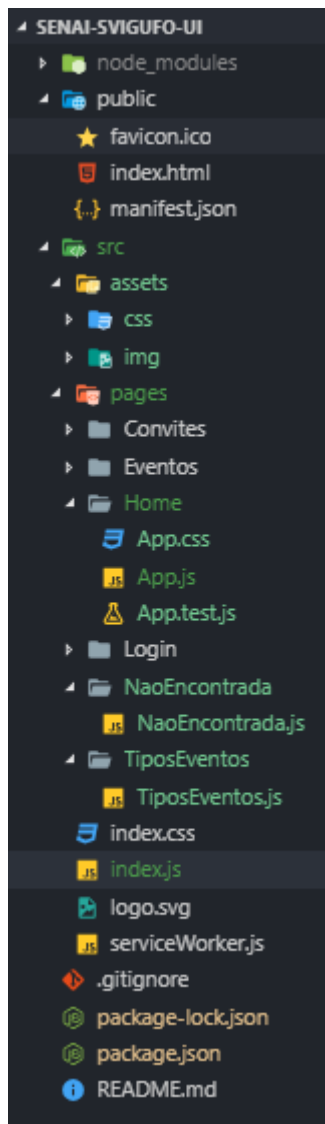
const routing = (
  <Router>
    <div>
      <Route exact path="/" component={App} />
      <Route path="/tiposeventos" component={TiposEventos} />
    </div>
  </Router>
);

// ReactDOM.render(<App />, document.getElementById("root"));
ReactDOM.render(routing, document.getElementById("root"));

// If you want your app to work offline and load faster, you can change
// unregister() to register() below. Note this comes with some pitfalls.
// Learn more about service workers: https://bit.ly/CRA-PWA
serviceWorker.unregister();
```

Sugestão de Página Não Encontrada

Criar uma pasta chamada NaoEncontrada bem como uma página NaoEncontrada.js com o seguinte conteúdo.



```
import React, { Component } from "react";

class NaoEncontrada extends Component {
  render() {
    return (
      <div>
        Erro 404 - Página Não Encontrada
      </div>
    );
  }
}

export default NaoEncontrada;
```



```

import React from "react";
import ReactDOM from "react-dom";
import { Route, Link, BrowserRouter as Router, Switch } from "react-router-dom";
import "./index.css";
import App from "../pages/Home/App";
import TiposEventos from "../pages/TiposEventos/TiposEventos";
import NaoEncontrada from "../pages/NaoEncontrada/NaoEncontrada";
import * as serviceWorker from "../serviceWorker";

const routing = (
  <Router>
    <div>
      <Switch>
        <Route exact path="/" component={App} />
        <Route path="/tiposeventos" component={TiposEventos} />
        <Route component={NaoEncontrada} />
      </Switch>
    </div>
  </Router>
);

// ReactDOM.render(<App />, document.getElementById("root"));
ReactDOM.render(routing, document.getElementById("root"));

// If you want your app to work offline and load faster, you can change
// unregister() to register() below. Note this comes with some pitfalls.
// Learn more about service workers: https://bit.ly/CRA-PWA
serviceWorker.unregister();

```

Criando uma rota de exemplo

```

import { Link } from 'react-router-dom';

import Rodape from "../../components/Rodape/Rodape";

class App extends Component {
  render() {
    return (
      <div>
        <header className="cabecalhoPrincipal">
          <div className="container">
            <img src={logo} alt="Svigufo - ícone de login" />

            <nav className="cabecalhoPrincipal-nav">
              <a>Home</a>
              <Link to="/tiposeventos">Tipos Eventos</Link>
              <a>Contato</a>
            </nav>
          </div>
        </header>
      </div>
    );
  }
}

```

TiposEventos.js

Copiar todo o conteúdo para a tela de TiposEventos.

```

import React, { Component } from "react";

import logo from "../../assets/img/icon-login.png";

import "../../assets/css/flexbox.css";
import "../../assets/css/reset.css";
import "../../assets/css/style.css";

class TiposEventos extends Component {
  render() {
    return (
      <div>
        <header class="cabecalhoPrincipal">
          <div class="container">
            <img src={logo} alt="Svigufo - ícone de login" />

            <nav class="cabecalhoPrincipal-nav">Administrador</nav>
          </div>
        </header>

        <main class="conteudoPrincipal">
          <section class="conteudoPrincipal-cadastro">
            <h1 class="conteudoPrincipal-cadastro-titulo">Tipos de Eventos</h1>
            <div class="container" id="conteudoPrincipal-lista">
              <table id="tabela-lista">
                <thead>
                  <tr>
                    <th>#</th>
                    <th>Título</th>
                    <th>Ação</th>
                  </tr>
                </thead>

                <tbody id="tabela-lista-corpo" />
              </table>
            </div>

            <div class="container" id="conteudoPrincipal-cadastro">
              <h2 class="conteudoPrincipal-cadastro-titulo">
                Cadastrar Tipo de Evento
              </h2>
              <form>
                <div class="container">
                  <input
                    type="text"
                    id="nome-tipo-evento"
                    placeholder="tipo do evento"
                  />
                  <button class="conteudoPrincipal-btn conteudoPrincipal-btn-cadastro">
                    Cadastrar
                  </button>
                </div>
              </form>
            </div>
          </section>
        </main>
      </div>
    );
  }
}

```

Lembrar de trocar o que for class para className.

Criando apenas um rodapé.

Criar uma pasta chamada componentes com uma pasta chamada Rodape e um arquivo chamado Rodape.js.

```
import React from "react";

function Rodape() {
  return (
    <footer class="rodapePrincipal">
      <section class="rodapePrincipal-patrocinadores">
        <div class="container">
          <p>Escola SENAI de Informática - 2019</p>
        </div>
      </section>
    </footer>
  );
}

export default Rodape;
```

Lembrar de trocar o que for class para className.

App.js

```
import Rodape from "../../components/Rodape/Rodape";

{/* <footer className="rodapePrincipal">
  <section className="rodapePrincipal-patrocinadores">
    <div className="container">
      <p>Escola SENAI de Informática - 2019</p>
    </div>
  </section>
</footer> */}

<Rodape />
```

Realizar o mesmo no TiposEventos.js

```
import Rodape from "../../components/Rodape/Rodape"

<Rodape />
```

TiposEventos.js

Mostrar os tipos de eventos da API

```

class TiposEventos extends Component {
  constructor() {
    super();
    this.state = {
      lista: [
        {
          id: 7,
          nome: "Desenvolvimento de Sistemas"
        },
        {
          id: 2,
          nome: "DESIGN"
        },
        {
          id: 4,
          nome: "LINKEDIN"
        },
        {
          id: 3,
          nome: "MARKETING"
        },
        {
          id: 1,
          nome: "TECNOLOGIA"
        }
      ]
    };
  }
}

```

Lembrar de deixar sem o key. Depois incluir.

```

<tbody>
  {this.state.lista.map(function(tipoEvento) {
    return (
      <tr key={tipoEvento.id}>
        <td>{tipoEvento.id}</td>
        <td>{tipoEvento.nome}</td>
      </tr>
    );
  })}
</tbody>

```

Ciclo de Vida

[https://medium.com/@edmo\\_/m%C3%A9todos-do-ciclo-de-vida-de-componentes-reactjs-um-mergulho-profundo-332ed7b3b782](https://medium.com/@edmo_/m%C3%A9todos-do-ciclo-de-vida-de-componentes-reactjs-um-mergulho-profundo-332ed7b3b782)

**Uso:** `componentWillMount` é usado para inicializar os estados ou adereços, há um grande debate para fundir isso com o construtor.

**Onde você deve fazer as chamadas API?**

*As chamadas da API devem ser feitas sempre no método `componentDidMount`.*

```

class TiposEventos extends Component {
  constructor() {
    super();
    this.state = { lista: [] }
    // lista: [
    //   {
    //     id: 7,
    //     nome: "Desenvolvimento de Sistemas"
    //   },
    //   {
    //     id: 2,
    //     nome: "DESIGN"
    //   },
    //   {
    //     id: 4,
    //     nome: "LINKEDIN"
    //   },
    //   {
    //     id: 3,
    //     nome: "MARKETING"
    //   },
    //   {
    //     id: 1,
    //     nome: "TECNOLOGIA"
    //   }
    // ]
  };
}

componentDidMount() {
  fetch("http://localhost:5000/api/tiposeventos")
    .then(response => response.json())
    .then(data => this.setState({ lista: data }));
}

```

<https://www.robinwieruch.de/react-fetching-data/>

There is another lifecycle method that is a perfect match to fetch data: `componentDidMount()`. When this method runs, the component was already rendered once with the `render()` method, but it would render again when the fetched data would be stored in the local state of the component with `setState()`. Afterward, the local state could be used in the `render()` method to display it or to pass it down as props.

```

buscarTiposEventos() {
  fetch("http://localhost:5000/api/tiposeventos")
    .then(response => response.json())
    .then(data => this.setState({ lista: data }));
}

componentDidMount() {
  this.buscarTiposEventos();
}

```

Cadastrar um tipo de evento

```
  cadastrarTipoEvento(evento) {
    evento.preventDefault();
    console.log('Enviando formulário');
  }

  buscarTiposEventos() {
    fetch("http://localhost:5000/api/tiposeventos")
      .then(response => response.json())
      .then(data => this.setState({ lista: data }));
  }

  componentDidMount() {
    this.buscarTiposEventos();
  }

  render() {
    return (
      <div>
        <header className="cabecalhoPrincipal">
          <div className="container">
            <img src={logo} alt="Svigufo - ícone de login" />

            <nav className="cabecalhoPrincipal-nav">Administrador</nav>
          </div>
        </header>

        <main className="conteudoPrincipal">
          <section className="conteudoPrincipal-cadastro">
            <h1 className="conteudoPrincipal-cadastro-titulo">...
            <div className="container" id="conteudoPrincipal-lista">...

            <div className="container" id="conteudoPrincipal-cadastro">
              <h2 className="conteudoPrincipal-cadastro-titulo">
                Cadastrar Tipo de Evento
              </h2>
              <form onSubmit={this.cadastrarTipoEvento}>
                <div className="container">
                  <input
                    type="text"
                    id="nome-tipo-evento"
                    placeholder="tipo do evento"
                  />
                </div>
              </form>
            </div>
          </section>
        </main>
      </div>
    );
  }
}
```

<https://medium.com/@rossbulat/an-introduction-to-using-form-elements-in-react-3778042ff334>

<https://medium.com/capital-one-tech/how-to-work-with-forms-inputs-and-events-in-react-c337171b923b>

```
class TiposEventos extends Component {
  constructor() {
    super();
    this.state = {
      lista: [],
      nome: ""

      // lista: [
      //   {
      //     id: 7,
      //     nome: "Desenvolvimento de Sistemas"
      //   },
      //   {
      //     id: 2,
      //     nome: "DESIGN"
      //   },
      //   {
      //     id: 4,
      //     nome: "LINKEDIN"
      //   },
      //   {
      //     id: 3,
      //     nome: "MARKETING"
      //   },
      //   {
      //     id: 1,
      //     nome: "TECNOLOGIA"
      //   }
      // ]
    };

    this.atualizadoEstadoNome = this.atualizadoEstadoNome.bind(this);
    this.cadastrarTipoEvento = this.cadastrarTipoEvento.bind(this);
  }
}
```

LEMBRETE: colocar somente response no formulário, para que ele não forneça nenhum erro e atualize a lista da tela.

```
// handleSubmit
cadastrarTipoEvento(evento) {
  evento.preventDefault();

  fetch("http://localhost:5000/api/tiposeventos", {
    method: "POST",
    // tem como melhorar essa parte?
    body: JSON.stringify({nome: this.state.nome}),
    headers: {
      "Content-Type": "application/json"
    }
  })
  .then(response => response.json())
  // como fazer ele dar um refresh?
  .then(this.buscarTiposEventos())
  .catch(error => console.log(error));
}

atualizadoEstadoNome(event) {
  this.setState({ nome: event.target.value });
}
```



```

render() {
  return (
    <div>
      <header className="cabecalhoPrincipal">
        <div className="container">
          <img src={logo} alt="Svigufo - ícone de login" />

          <nav className="cabecalhoPrincipal-nav">Administrador</nav>
        </div>
      </header>

      <main className="conteudoPrincipal">
        <section className="conteudoPrincipal-cadastro">
          <h1 className="conteudoPrincipal-cadastro-titulo">...
          <div className="container" id="conteudoPrincipal-lista">...

          <div className="container" id="conteudoPrincipal-cadastro">
            <h2 className="conteudoPrincipal-cadastro-titulo">
              Cadastrar Tipo de Evento
            </h2>
            <form onSubmit={this.cadastrarTipoEvento}>
              <div className="container">
                <input
                  type="text"
                  id="nome-tipo-evento"
                  value={this.state.nome}
                  onChange={this.atualizadoEstadoNome}
                  placeholder="tipo do evento"
                />
                <button className="conteudoPrincipal-btn conteudoPrincipal-btn-cadastro">
                  Cadastrar
                </button>
              </div>
            </form>
          </div>
        </section>
      </main>

      <Rodape />
    </div>
  );
}

```

## Login

<https://blog.rocketseat.com.br/reactjs-autenticacao/>

Criar uma pasta chamada Login e um arquivo dentro chamado Login.js com o conteúdo a seguir (lembrando de trocar o className)

```
1  import React from "react";
2
3  import logo from "../../assets/img/icon-login.png";
4
5  import "../../assets/css/login.css";
6
7  class Login extends React.Component {
8    render() {
9      return (
10       <section className="container flex">
11         <div className="img_login">
12           <div className="img_overlay" />
13         </div>
14
15         <div className="item_login">
16           <div className="row">
17             <div className="item">
18               <img
19                 src={logo}
20                 className="icone_login"
21                 alt="Svigufo - ícone de login"
22               />
23             </div>
24             <div className="item" id="item_title">
25               <p className="text_login" id="item_description">
26                 Bem-vindo! Faça login para acessar sua conta.
27               </p>
28             </div>
29             <form>
30               <div className="item">
31                 <input
32                   className="input_login"
33                   placeholder="username"
34                   type="text"
35                   name="username"
36                   id="login_email"
37                 />
38               </div>
39               <div className="item">
40                 <input
41                   className="input_login"
42                   placeholder="password"
43                   type="password"
44                   name="password"
45                   id="login_password"
46                 />
47               </div>
48               <div className="item">
49                 <button className="btn btn_login" id="btn_login">
50                   Login
51                 </button>
52               </div>
53             </form>
54           </div>
55         </div>
56       </section>
57     );
58   }
59 }
60
61 export default Login;
```

Comentar sobre escopos de css globais

Alterando o link para redirecionar para o login

```
1  import React, { Component } from "react";
2  import { Link } from "react-router-dom";
3
4  import logo from "../../assets/img/icon-login.png";
5
6  import "../../assets/css/flexbox.css";
7  import "../../assets/css/reset.css";
8  import "../../assets/css/style.css";
9
10 import Rodape from '../../components/Rodape/Rodape';
11
12 class App extends Component {
13   render() {
14     return (
15       <div>
16         <header className="cabecalhoPrincipal">
17           <div className="container">
18             <img src={logo} />
19
20             <nav className="cabecalhoPrincipal-nav">
21               <a>Home</a>
22               <a>Eventos</a>
23               <a>Contato</a>
24               <Link className="cabecalhoPrincipal-nav-login" to="/login">
25                 Login
26               </Link>
27             </nav>
28           </div>
29         </header>
30       </div>
    )
  }
}
```

Alterar o arquivo index.js

```
JS Login.js JS index.js x
1 import React from "react";
2 import ReactDOM from "react-dom";
3 import { Route, BrowserRouter as Router, Switch } from "react-router-dom";
4 import "./index.css";
5 import App from "../pages/Home/App";
6 import TiposEventos from "../pages/TiposEventos/TiposEventos";
7 import NaoEncontrada from "../pages/NaoEncontrada/NaoEncontrada";
8 import Login from "../pages/Login/Login";
9 import * as serviceWorker from "../serviceWorker";
10
11 const routing = (
12   <Router>
13     <div>
14       <Switch>
15         <Route exact path="/" component={App} />
16         <Route path="/tiposeventos" component={TiposEventos} />
17         <Route path="/login" component={Login} />
18         <Route component={NaoEncontrada} />
19       </Switch>
20     </div>
21   </Router>
22 );
23
24 ReactDOM.render(routing, document.getElementById("root"));
25
26 // If you want your app to work offline and load faster, you can change
27 // unregister() to register() below. Note this comes with some pitfalls.
28 // Learn more about service workers: https://bit.ly/CRA-PWA
29 serviceWorker.unregister();
30
```

Criar o construtor com seus respectivos valores

```
class Login extends React.Component {
  constructor() {
    super();
    this.state = {
      email: "",
      senha: ""
    };
  }

  atualizaEstadoEmail(event) {
    this.setState({ email: event.target.value });
  }

  atualizaEstadoSenha(event) {
    this.setState({ senha: event.target.value });
  }
}
```

Criar os valores do onchange e dos valores para o React

```
<div className="item">
  <input
    className="input__login"
    placeholder="username"
    type="text"
    name="username"
    id="login_email"
    value={this.state.email}
    onChange={this.atualizaEstadoEmail.bind(this)}
  />
</div>
<div className="item">
  <input
    className="input__login"
    placeholder="password"
    type="password"
    name="password"
    id="login_password"
    value={this.state.senha}
    onChange={this.atualizaEstadoSenha.bind(this)}
  />
</div>
<div className="item">
  <button className="btn btn_login" id="btn_login">
    Login
  </button>
</div>
```

npm install --save axios

Criar a função para realizar o login

```
efetuaLogin(event) {
  event.preventDefault();

  axios
    .post("http://localhost:5000/api/login", {
      email: this.state.email,
      senha: this.state.senha
    })
    .then(data => {
      if (data.status === 200) {
        console.log(data.data.token);
        localStorage.setItem("usuario-svigufu", data.data.token);
        // this.props.history.push("/tiposeventos");
      } else {
        // this.setState({ erro: "Usuário ou senha inválidos." });
        console.log("erro");
      }
    })
    .catch(erro => {
      // this.setState({ erro: "Usuário ou senha inválidos." });
      console.log(erro);
    });
}
```

Chamar a função no formulário

```
<form onSubmit={this.efetuaLogin.bind(this)}>
  <div className="login">
```

Mostrando uma mensagem de erro caso o login seja inválido

```
class Login extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      email: "",
      senha: "",
      erro: ""
    };
  }

  atualizaEstadoEmail(event) {
    this.setState({ email: event.target.value });
  }

  atualizaEstadoSenha(event) {
    this.setState({ senha: event.target.value });
  }

  efetuaLogin(event) {
    event.preventDefault();

    axios
      .post("http://localhost:5000/api/login", {
        email: this.state.email,
        senha: this.state.senha
      })
      .then(data => {
        if (data.status === 200) {
          console.log(data.data.token);
          localStorage.setItem("usuario-svigufu", data.data.token);
          // this.props.history.push("/tiposeventos");
        } else {
          // this.setState({ erro: "Usuário ou senha inválidos." });
          console.log("erro");
        }
      })
      .catch(erro => {
        this.setState({ erro: "Usuário ou senha inválidos." });
        console.log(erro);
      });
  }
}
```

Mostrar o erro

```
<p className="text_login" style={{ color: 'red', textAlign: 'center' }}>{this.state.erro}</p>
<div className="item">
  <button className="btn btn_login" id="btn_login">
    Login
  </button>
</div>
```

Mostrando uma mensagem de erro e redirecionando caso seja feito com sucesso

```
efetuaLogin(event) {  
  event.preventDefault();  
  
  axios  
    .post("http://localhost:5000/api/login", {  
      email: this.state.email,  
      senha: this.state.senha  
    })  
    .then(data => {  
      if (data.status === 200) {  
        console.log(data.data.token);  
        localStorage.setItem("usuario-svigufo", data.data.token);  
        this.setState({ erro: null });  
        this.props.history.push("/tiposeventos");  
      } else {  
        this.setState({ erro: "Usuário ou senha inválidos." });  
        console.log("erro");  
      }  
    })  
    .catch(erro => {  
      this.setState({ erro: "Usuário ou senha inválidos." });  
      console.log(erro);  
    });  
}
```



No arquivo index.js, criar uma rota privada para que apenas usuários autenticados consigam realizar o login.

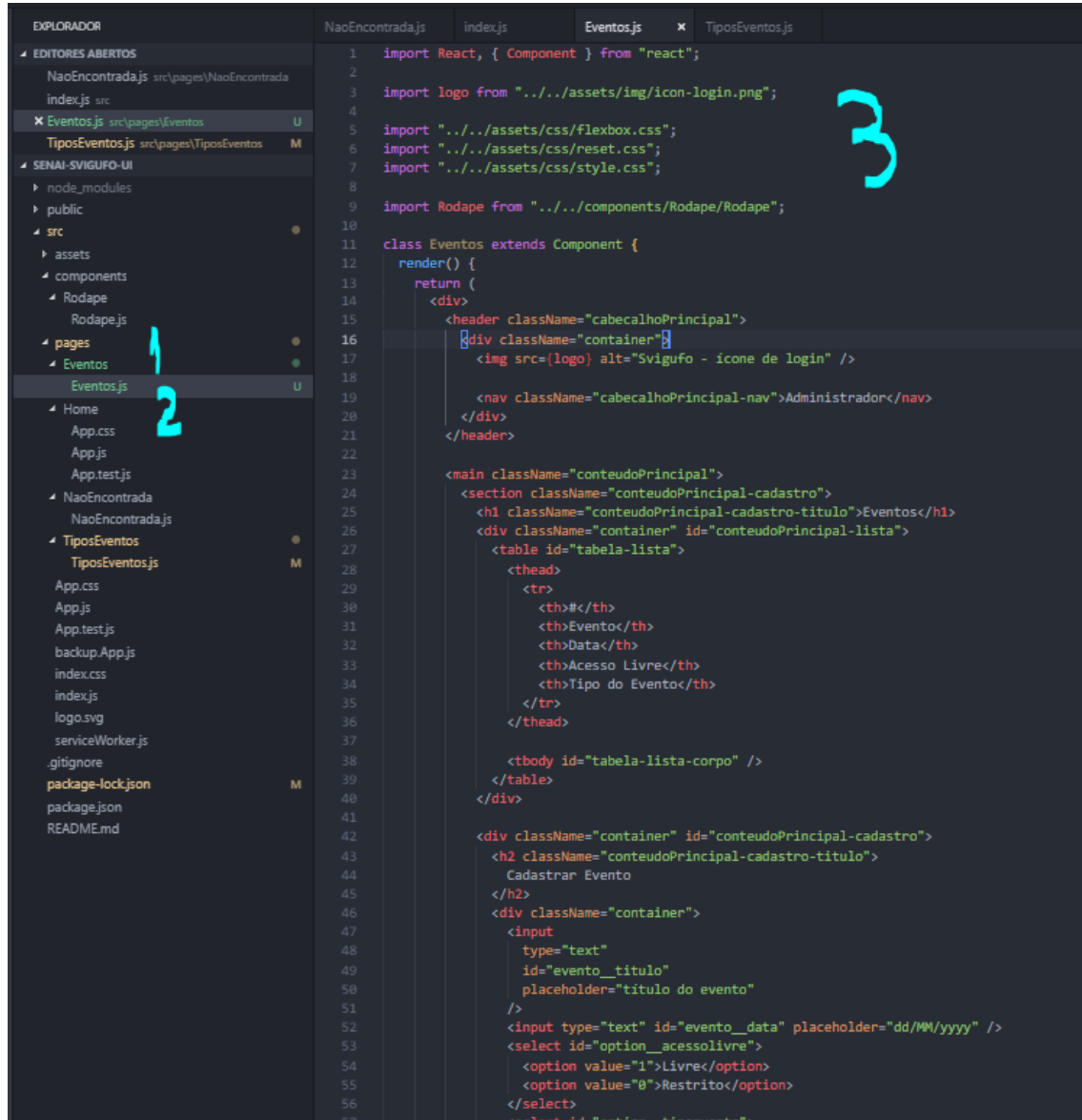
```
1  import React from "react";
2  import ReactDOM from "react-dom";
3  import {
4    Route,
5    BrowserRouter as Router,
6    Switch,
7    Redirect
8  } from "react-router-dom";
9  import "./index.css";
10 import App from "../pages/Home/App";
11 import TiposEventos from "../pages/TiposEventos/TiposEventos";
12 import NaoEncontrada from "../pages/NaoEncontrada/NaoEncontrada";
13 import Login from "../pages/Login/Login";
14 import * as serviceWorker from "../serviceWorker";
15
16 const PrivateRoute = ({ component: Component, ...rest }) => (
17   <Route
18     {...rest}
19     render={props =>
20       localStorage.getItem("usuario-svigifo") !== null ? (
21         <Component {...props} />
22       ) : (
23         <Redirect
24           to={{ pathname: "/login", state: { from: props.location } }}
25         />
26       )
27     />
28 );
29
30
31 const routing = (
32   <Router>
33     <div>
34       <Switch>
35         <Route exact path="/" component={App} />
36         { /* <Route path="/tiposeventos" component={TiposEventos} /> */ }
37         <PrivateRoute path="/tiposeventos" component={TiposEventos} />
38         <Route path="/login" component={Login} />
39         <Route component={NaoEncontrada} />
40       </Switch>
41     </div>
42   </Router>
43 );
44
```

<https://stackoverflow.com/questions/43484302/what-does-it-mean-rest-in-react-jsx>

Criar uma pasta chamada Eventos

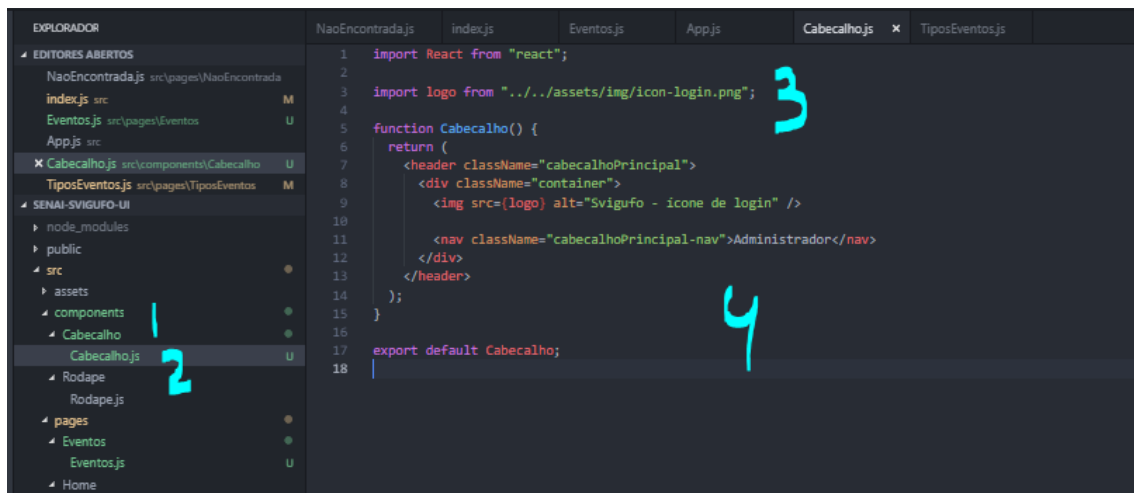
Criar um arquivo chamado Eventos.js

Realizar os devidos imports e copiar o conteúdo de eventos.html (da pasta base). Lembrar de fechar as tags de inputs e trocar as 'class' para 'className'.



```
1 import React, { Component } from "react";
2
3 import logo from "../../assets/img/icon-login.png";
4
5 import "../../assets/css/flexbox.css";
6 import "../../assets/css/reset.css";
7 import "../../assets/css/style.css";
8
9 import Rodape from "../../components/Rodape/Rodape";
10
11 class Eventos extends Component {
12   render() {
13     return (
14       <div>
15         <header className="cabecalhoPrincipal">
16           <div className="container">
17             <img src={logo} alt="Svigufo - icone de login" />
18
19             <nav className="cabecalhoPrincipal-nav">Administrador</nav>
20           </div>
21         </header>
22
23         <main className="conteudoPrincipal">
24           <section className="conteudoPrincipal-cadastro">
25             <h1 className="conteudoPrincipal-cadastro-titulo">Eventos</h1>
26             <div className="container" id="conteudoPrincipal-lista">
27               <table id="tabela-lista">
28                 <thead>
29                   <tr>
30                     <th>#</th>
31                     <th>Evento</th>
32                     <th>Data</th>
33                     <th>Acesso Livre</th>
34                     <th>Tipo do Evento</th>
35                   </tr>
36                 </thead>
37
38                 <tbody id="tabela-lista-corpo">
39                 </tbody>
40               </table>
41             </div>
42
43             <div className="container" id="conteudoPrincipal-cadastro">
44               <h2 className="conteudoPrincipal-cadastro-titulo">
45                 Cadastrar Evento
46               </h2>
47               <div className="container">
48                 <input
49                   type="text"
50                   id="evento_titulo"
51                   placeholder="titulo do evento"
52                 />
53                 <input type="text" id="evento_data" placeholder="dd/MM/yyyy" />
54                 <select id="option_acessolivre">
55                   <option value="1">Livre</option>
56                   <option value="0">Restrito</option>
57                 </select>
58                 <select id="option_tipoevento">
```

Criar um componente de cabeçalho



Importar este componente nas duas páginas (TiposEventos.js e Eventos.js)

```
import Rodape from "../../components/Rodape/Rodape";
import Cabecalho from "../../components/Cabecalho/Cabecalho";
```

```
render() {
  return (
    <div>
      <Cabecalho />

      <main className="conteudoPrincipal">
        <section className="conteudoPrincipal-ca
          <h1 className="conteudoPrincipal-cadas
```

Para verificar que a página evento foi criada, devemos incluí-la na rota.

```

Login.js  x  TiposEventos.js  index.js  x  Eventos.js  Cabecalho.js
1  import React from "react";
2  import ReactDOM from "react-dom";
3  import {
4    Route,
5    BrowserRouter as Router,
6    Switch,
7    Redirect
8  } from "react-router-dom";
9  import "../index.css";
10 import App from "../pages/Home/App";
11 import TiposEventos from "../pages/TiposEventos/TiposEventos";
12 import Eventos from "../pages/Eventos/Eventos";
13 import NaoEncontrada from "../pages/NaoEncontrada/NaoEncontrada";
14 import Login from "../pages/Login/Login";
15 import * as serviceWorker from "../serviceWorker";
16
17 const PrivateRoute = ({ component: Component, ...rest }) => (
18   <Route
19     {...rest}
20     render={props =>
21       localStorage.getItem("usuario-svigifo") !== null ? (
22         <Component {...props} />
23       ) : (
24         <Redirect
25           to={{ pathname: "/login", state: { from: props.location } }}
26         />
27       )
28     }
29   />
30 );
31
32 const routing = (
33   <Router>
34     <div>
35       <Switch>
36         <Route exact path="/" component={App} />
37         { /* <Route path="/tiposeventos" component={TiposEventos} /> */ }
38         <PrivateRoute path="/tiposeventos" component={TiposEventos} />
39         <Route path="/eventos" component={Eventos} />
40         <Route path="/login" component={Login} />
41         <Route component={NaoEncontrada} />
42       </Switch>
43     </div>
44   </Router>
45 );
46
47 ReactDOM.render(routing, document.getElementById("root"));
48
49 // If you want your app to work offline and load faster, you can change
50 // unregister() to register() below. Note this comes with some pitfalls.
51 // Learn more about service workers: https://bit.ly/CRA-PWA
52 serviceWorker.unregister();
53

```

Listar os eventos

```
import axios from "axios";  
  
class Eventos extends Component {  
  constructor() {  
    super();  
    this.state = {  
      listaEventos: []  
    };  
  }  
  
  buscarEventos() {  
    axios  
      .get("http://localhost:5000/api/eventos")  
      .then(data => this.setState({ listaEventos: data.data }));  
  }  
  
  componentDidMount() {  
    this.buscarEventos();  
  }  
}
```

```
<tbody>  
  {this.state.listaEventos.map(function(evento) {  
    return (  
      <tr key={evento.id}>  
        <td>{evento.id}</td>  
        <td>{evento.titulo}</td>  
        <td>{evento.dataEvento}</td>  
        {evento.acessoLivre === true ? (  
          <td>Sim</td>  
        ) : (  
          <td>Não</td>  
        )}  
        <td>{evento.tipoEvento.nome}</td>  
      </tr>  
    );  
  })}  
</tbody>
```

Mostrando o select com os tipos de eventos

```
class Eventos extends Component {
  constructor() {
    super();
    this.state = {
      listaEventos: [],
      listaTiposEventos: [],
      // tipoEvento: {
      //   tipoEventoId: null
      // }
    };

    // this.atualizaTipoEventoId = this.atualizaTipoEventoId.bind(this);
  }

  buscarEventos() {
    axios
      .get("http://localhost:5000/api/eventos")
      .then(data => this.setState({ listaEventos: data.data }));
  }

  buscarTiposEventos() {
    axios
      .get("http://localhost:5000/api/tiposeventos")
      .then(data => this.setState({ listaTiposEventos: data.data }));
  }

  componentDidMount() {
    this.buscarEventos();
    this.buscarTiposEventos();
  }
}
```

Colocando-os no select

```
<select
  value={this.state.tipoEventoId}
  name="tipoEventoId"
  id="tipoEventoId"
  onChange={this.atualizaTipoEventoId}
>
  <option value="">Selecione o Tipo de Evento</option>
  {this.state.listaTiposEventos.map(function(tipoEvento) {
    return (
      <option value={tipoEvento.id} key={tipoEvento.id}>
        {tipoEvento.nome}
      </option>
    );
  })}
</select>
```

```
class Eventos extends Component {
  constructor() {
    super();
    this.state = {
      listaEventos: [],
      listaTiposEventos: [],
      tipoEvento: {
        tipoEventoId: null
      }
    };
  }

  this.atualizaTipoEventoId = this.atualizaTipoEventoId.bind(this);
}

buscarEventos() {
  axios
    .get("http://localhost:5000/api/eventos")
    .then(data => this.setState({ listaEventos: data.data }));
}

buscarTiposEventos() {
  axios
    .get("http://localhost:5000/api/tiposeventos")
    .then(data => this.setState({ listaTiposEventos: data.data }));
}

componentDidMount() {
  this.buscarEventos();
  this.buscarTiposEventos();
}

atualizaTipoEventoId(event) {
  console.log(event.target.value);

  this.setState({
    tipoEvento: { tipoEventoId: event.target.value }
  });
}

render() {
```

## Props

Criar um componente de Titulo.js dentro da pasta de componentes.

```
index.js  TiposEventos.js  Titulo.js  x
1  import React from "react";
2
3  class Titulo extends React.Component {
4    render() {
5      return <h1 className="conteudoPrincipal-cadastro-titulo">{this.props.titulo}</h1>;
6    }
7  }
8
9  export default Titulo;
```

Importar o título dentro da tela de evento e tipos de eventos.

```
import React, { Component } from "react";
// import logo from "../../assets/img/icon-login.png";
import "../../assets/css/flexbox.css";
import "../../assets/css/reset.css";
import "../../assets/css/style.css";
import Rodape from "../../components/Rodape/Rodape";
import Cabecalho from "../../components/Cabecalho/Cabecalho";
import Titulo from "../../components/Titulo";
class TiposEventos extends Component {
  constructor() {
    super();
    this.state = {}
  }
  render() {
    return (
      <div>
        <Cabecalho />
        <main className="conteudoPrincipal">
          <section className="conteudoPrincipal-cadastro">
            <h1 className="conteudoPrincipal-cadastro-titulo">
              Tipos de Eventos
            </h1>
            <div className="container" id="conteudoPrincipal-lista">
              <table id="tabela-lista">
                <thead>
                  <tr>
                    <th>
                      Tipos de Eventos
                    </th>
                    <th>
                      Tipos de Eventos
                    </th>
                  </tr>
                </thead>
                <tbody>
                  <tr>
                    <td>
                      Tipos de Eventos
                    </td>
                    <td>
                      Tipos de Eventos
                    </td>
                  </tr>
                </tbody>
              </table>
            </div>
          </section>
        </main>
        <Rodape />
      </div>
    );
  }
}
```

Atualizar eventos e tipos de eventos com a props correspondente.

```
render() {
  return (
    <div>
      <Cabecalho />
      <main className="conteudoPrincipal">
        <section className="conteudoPrincipal-cadastro">
          <h1 className="conteudoPrincipal-cadastro-titulo">
            Tipos de Eventos
          </h1>
          <div className="container" id="conteudoPrincipal-lista">
            <table id="tabela-lista">
              <thead>
                <tr>
                  <th>
                    Tipos de Eventos
                  </th>
                  <th>
                    Tipos de Eventos
                  </th>
                </tr>
              </thead>
              <tbody>
                <tr>
                  <td>
                    Tipos de Eventos
                  </td>
                  <td>
                    Tipos de Eventos
                  </td>
                </tr>
              </tbody>
            </table>
          </div>
        </section>
      </main>
      <Rodape />
    </div>
  );
}
```



## Componentes personalizados

### Input.js

```
import React from "react";

const Input = props => {
  return (
    <input
      id={props.name}
      name={props.name}
      type={props.type}
      value={props.value}
      onChange={props.handleChange}
      placeholder={props.placeholder}
    />
  );
};

export default Input;
```

### TiposEventos.js

```
<Input
  type={"text"}
  title={"tipo do evento"}
  name={"tipoevento"}
  value={this.state.nome}
  placeholder={"tipo do evento"}
  handleChange={this.atualizadoEstadoNome}
/>
```

Melhorando as chamadas as API's

services

api.js

```
import axios from 'axios';

const URL = 'http://localhost:5000/api/';

export default {
  tiposEventos() {
    return {
      getAll: () => {
        return axios.get(URL + 'tiposeventos').then(data => { return data.data });
      }
    }
  }
}
```

TiposEventos.js

```
buscarTiposEventos() {
  // fetch("http://localhost:5000/api/tiposeventos")
  // fetch("http://192.168.4.112:5000/api/tiposeventos")
  // .then(response => response.json())
  // .then(data => this.setState({ lista: data }));
  api
    .tiposEventos()
    .getAll()
    .then(resultado => this.setState({ lista: resultado }));
}
```

## Cabecalho.js

```
import React from "react";

import logo from "../../assets/img/icon-login.png";

import jwt from "../../services/jwt";

function Cabecalho() {
  let usuario = jwt(localStorage.getItem("usuario-svigufo")).email;

  return (
    <header className="cabecalhoPrincipal">
      <div className="container">
        <img src={logo} alt="Svigufo - ícone de login" />

        <nav className="cabecalhoPrincipal-nav">{usuario}</nav>
      </div>
    </header>
  );
}

export default Cabecalho;
```