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>

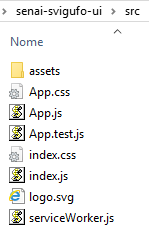
<https://medium.com/tableless/organizando-uma-aplica%C3%A7%C3%A3o-com-react-5b8ea9075596>

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

Criar projeto

Classes

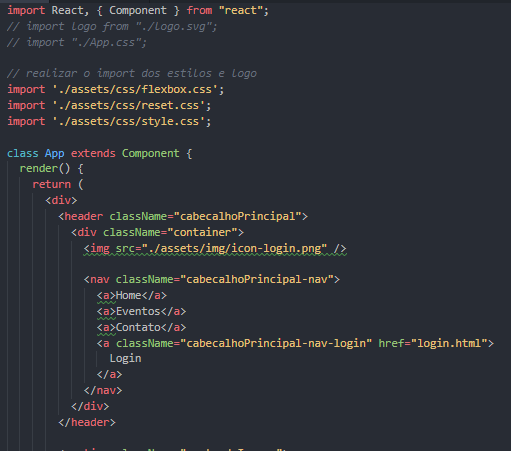
Copiar a pasta assets definida no projeto base e colar dentro de src do projeto do React.



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.

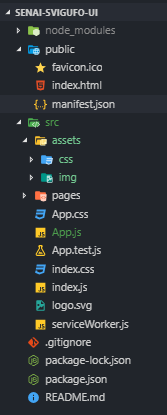


Importando o ícone e o incluindo na tela.

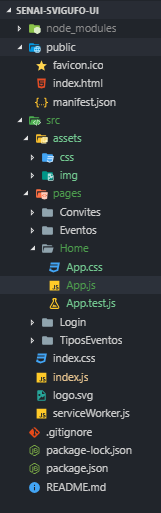


Após termos colocado a home, vamos realizar um cadastro de tipo de evento para verificar que não há diferença entre eles.

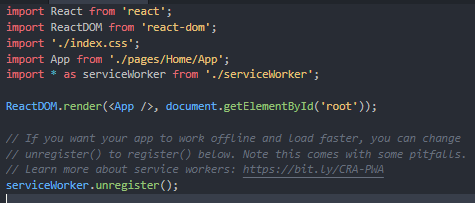
Para melhor organização, vamos criar uma pasta chamada pages.



E aproveitar, criar subpastas para organizar os nossos códigos.



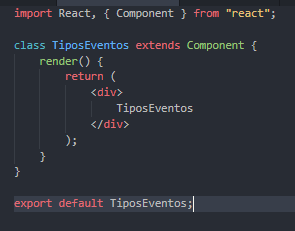
Alterar o index.js para encontrar a página de Home



App.js



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



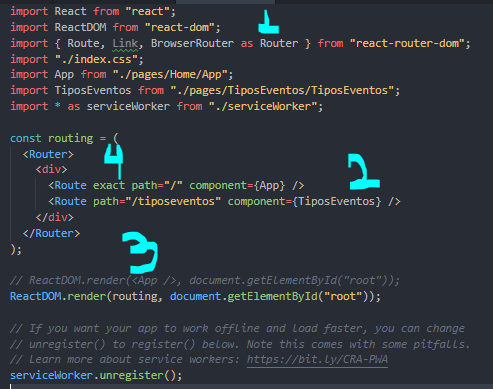
Para que essa página possa ser acessada, precisamos instalar o react-router.

<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.

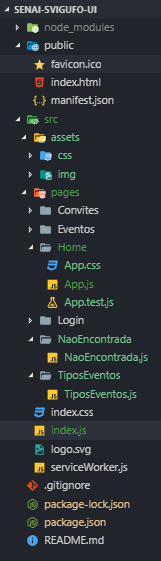
npm install --save react-router-dom

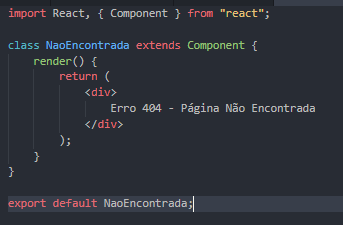
index.js



Sugestão de Página Não Encontrada

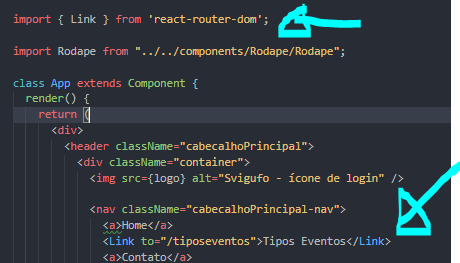
Criar uma pasta chamada NaoEncontrada bem como uma página NaoEncontrada.js com o seguinte conteúdo.





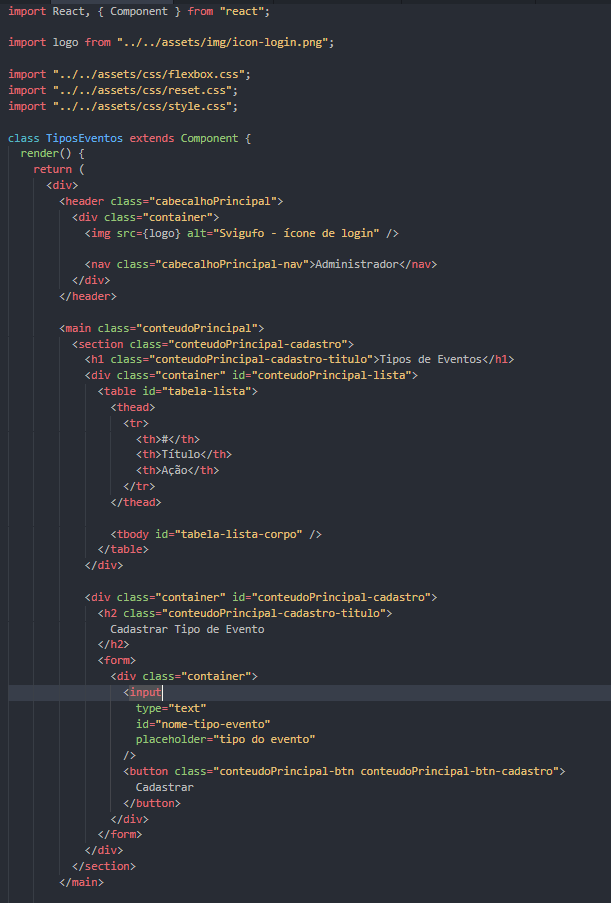


Criando uma rota de exemplo



TiposEventos.js

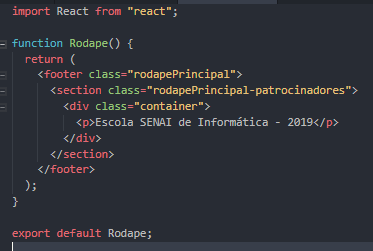
Copiar todo o conteúdo para a tela de TiposEventos.



Lembrar de trocar o que for class para className.

Criando apenas um rodapé.

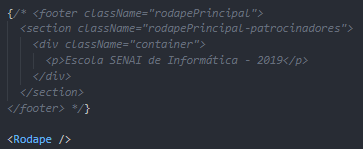
Criar uma pasta chamada componentes com uma pasta chamada Rodape e um arquivo chamado Rodape.js.



Lembrar de trocar o que for class para className.

App.js





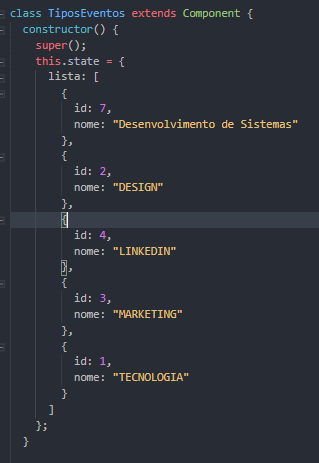
Realizar o mesmo no TiposEventos.js



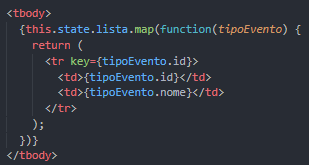


TiposEventos.js

Mostrar os tipos de eventos da API



Lembrar de deixar sem o key. Depois incluir.



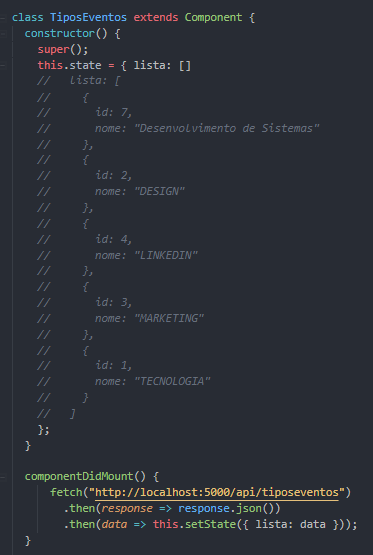
Ciclo de Vida

<https://medium.com/@edmo_/m%C3%A9todos-do-ciclo-de-vida-de-componentes-reactjs-um-mergulho-profundo-332ed7b3b782>

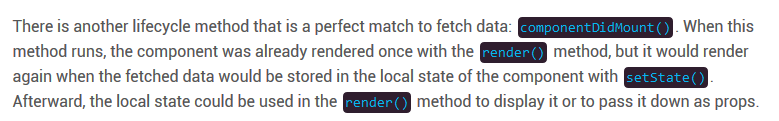
**Uso:** componentWillMount é usado para inicializar os estados ou adereços, há um grande debate para fundir isso com o construtor.

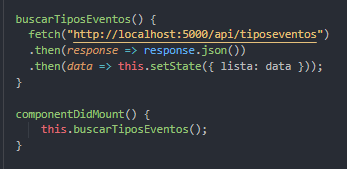
**Onde você deve fazer as chamadas API?**

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

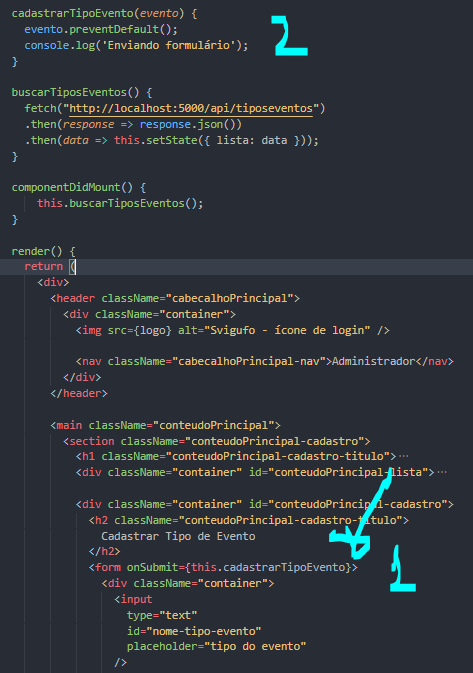


<https://www.robinwieruch.de/react-fetching-data/>





Cadastrar um tipo de evento



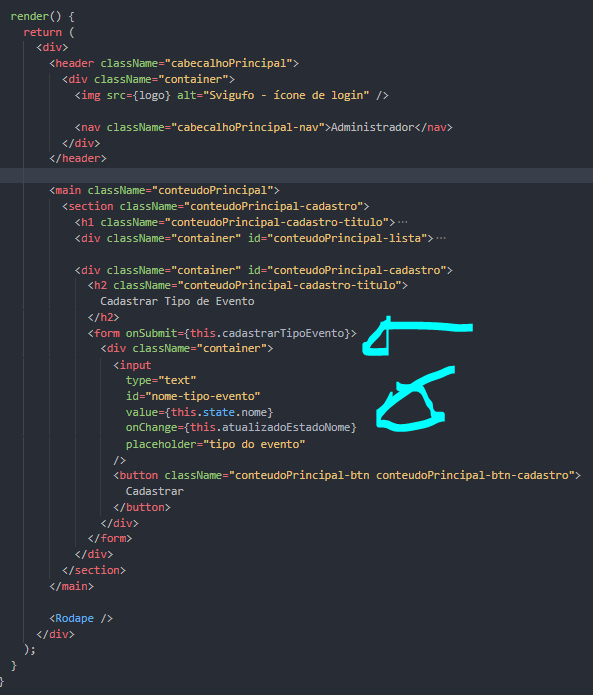
<https://medium.com/@rossbulat/an-introduction-to-using-form-elements-in-react-3778042ff334>

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



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





Login

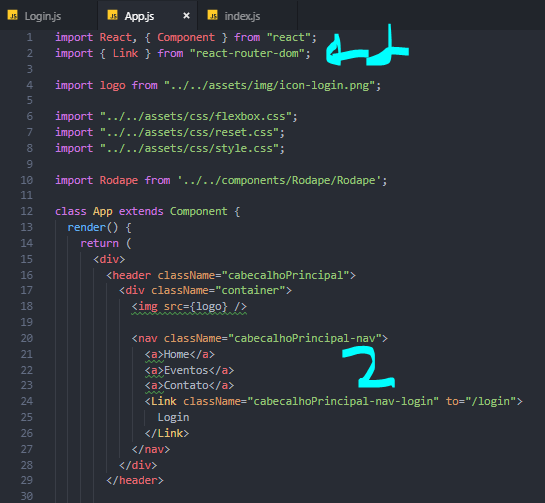
<https://blog.rocketseat.com.br/reactjs-autenticacao/>

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

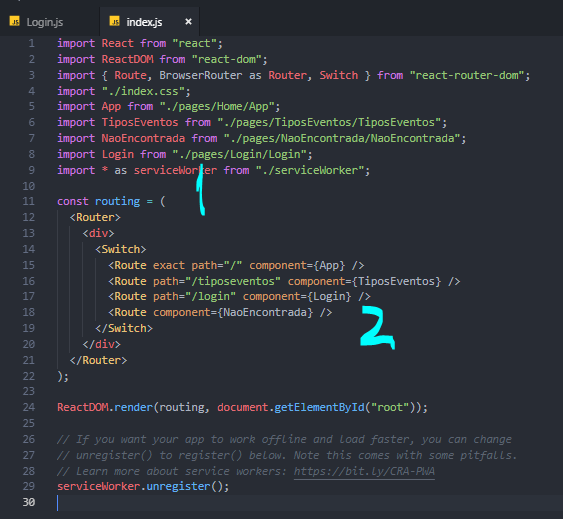


Comentar sobre escopos de css globais

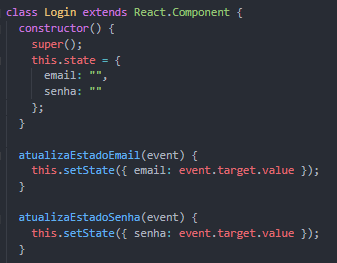
Alterando o link para redirecionar para o login



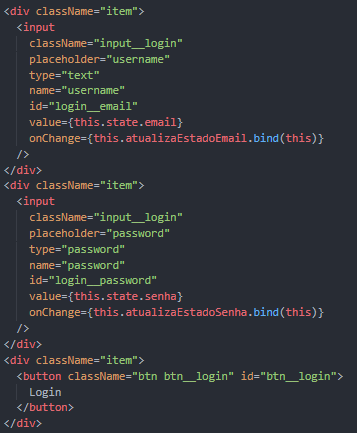
Alterar o arquivo index.js



Criar o construtor com seus respectivos valores



Criar os valores do onchange e dos valores para o React



npm install --save axios

Criar a função para realizar o login



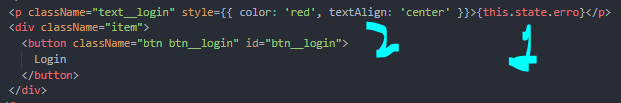
Chamar a função no formulário



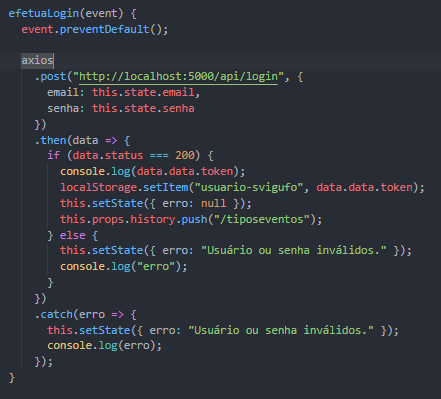
Mostrando uma mensagem de erro caso o login seja inválido



Mostrar o erro



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



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



<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’.



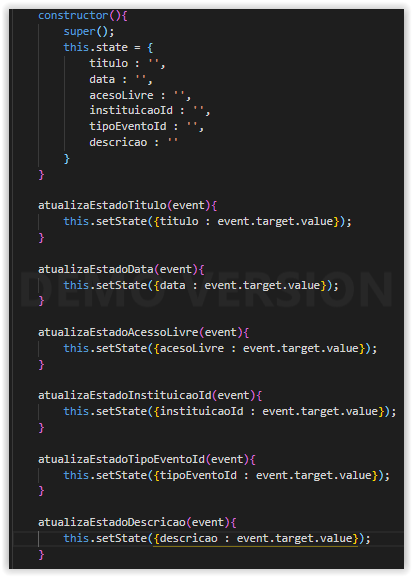
Altera o class para className

Criar rota em index.js, importar Eventos



Cadastrar um Evento

Criar o constructor com state com título, data, acessolivre, instituicaoId, tipoeventoid, descrição e Implementar funções para atualizar estado.



Implementa o value e onChange em todos os inputs e define o onSubmit, verificar o valor da instituição que está cadastrado assim como os tipos de eventos, fazer primeiro fixo.



Cria função para cadastrarEvento

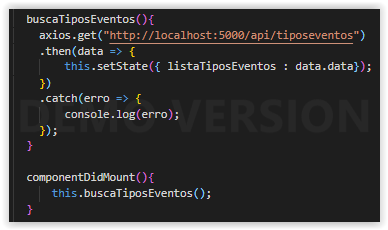
Import axios from ‘axios’



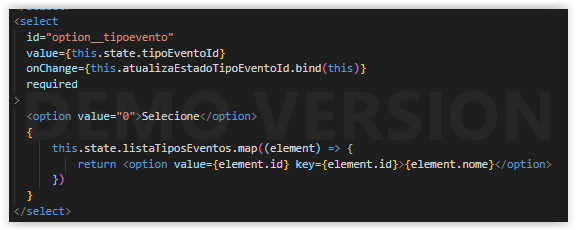
Listar os tiposEventos e colocar no select

Inclui no state listaTiposEventos : []

Cria function buscaTiposEventos



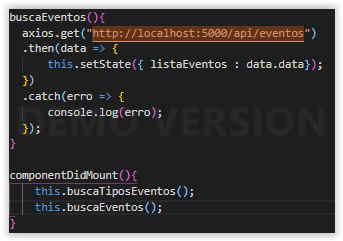
Inclui a lista nas opções



Lista os eventos na tabela

Inclui no state listaEventos : []

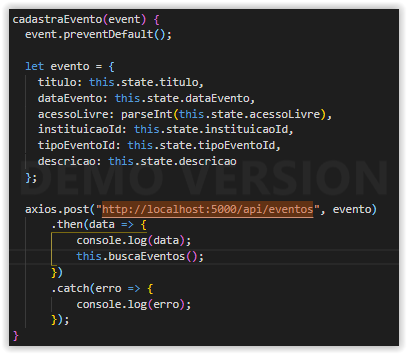
Cria function buscarEventos



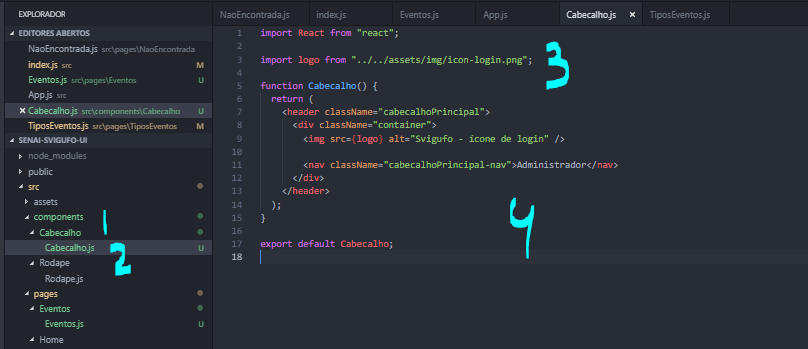
Implementar na tabela



Incluir para que depois de cadastra um evento listar

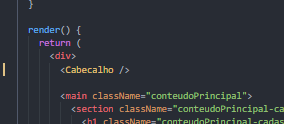


Criar um componente de cabeçalho



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

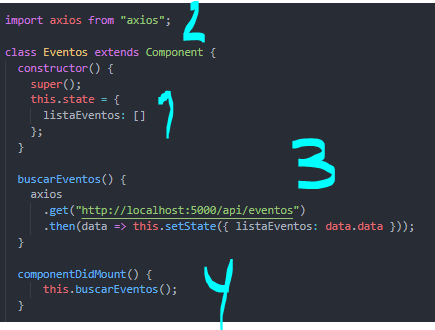




Para verificar que a página evento foi criada, devemos incluí-la na rota.



Listar os eventos

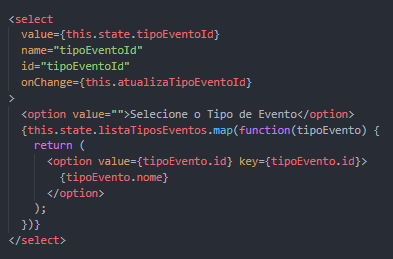


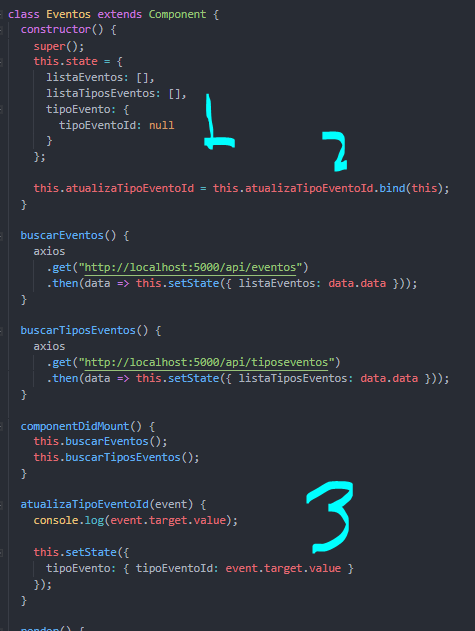


Mostrando o select com os tipos de eventos



Colocando-os no select



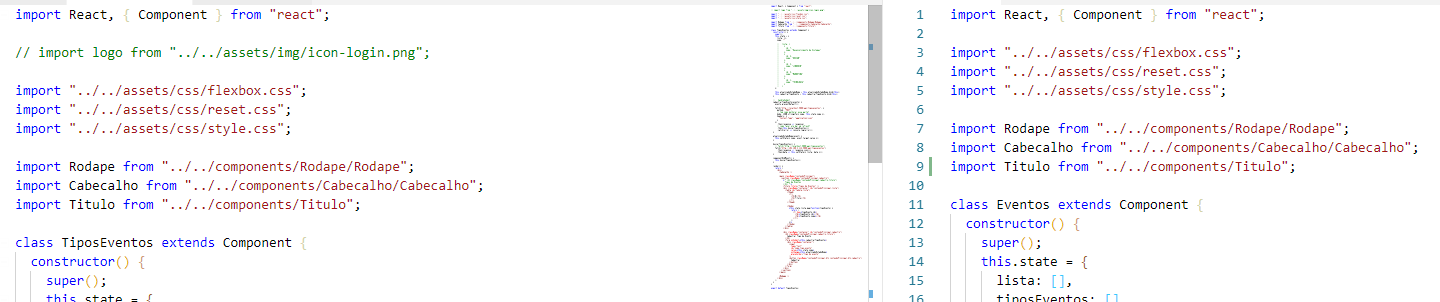


Props

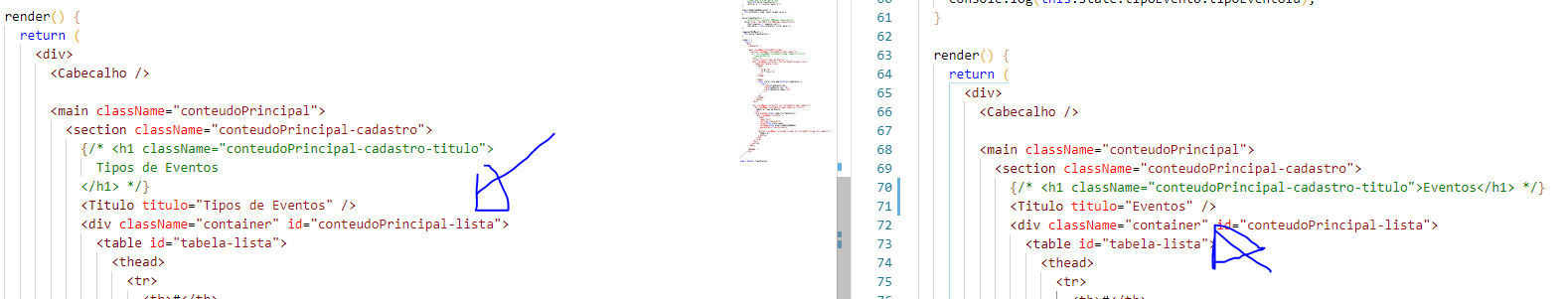
Criar um componente de Titulo.js dentro da pasta de componentes.



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

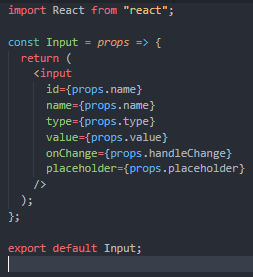


Atualizar eventos e tipos de eventos com a props correspondente.

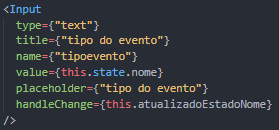


Componentes personalizados

Input.js



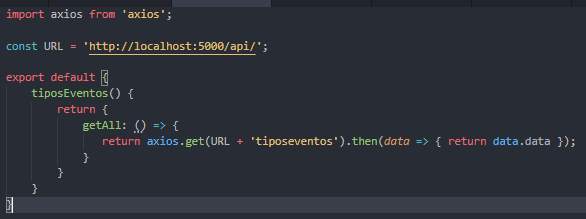
TiposEventos.js



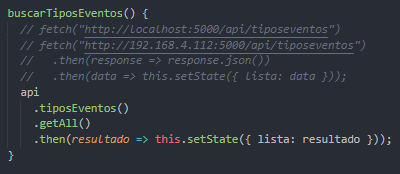
Melhorando as chamadas as API’s

services

api.js



TiposEventos.js



Cabecalho.js

