**Anotações durante a NLW/Together – ReactJS**

**AULA-02**

**React-router-dom:** ferramenta para configurar routes no React, para instalar no projeto:

**$$ yarn add react-router-dom  
Observação: Isso no caso de utilizar JS puro  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\*\*\* Utilizando TypeScript, é necessário rodar:  
$$ yarn add @types/react-router-dom -D**

• Por padrão do React, todos os **Hooks** devem estar dentro dos componentes.

**Contextos** no React **-** são formas de compartilhar informações entre dois ou mais componentes da aplicação. (textos, objetos, array etc...)

**Hook**: hooks são todas as funcionalidades com prefixo ‘use’.

• **useState** – **Estado**.

• **useEffect** – Hook para disparo de efeitos colaterais  
 • utilizar quando disparar uma função sempre quando algo acontecer.  
 • esse algo acontecer, pode ser uma informação que mudou ou, quando um componente é mostrado em tela, etc...  
 • toda vez que declaramos um Event Listener, no React dentro do useEffect, é recomendado que salvamos um Event Listener em uma variável e essa variável é para desligarmos ele.  
 • e **SEMPRE** faça o retorno do useEffect de uma função que me descadastre de todos os Event Listeners que foram cadastrados.  
  
**\*\*\* Anotações sobre a aula:** Como o AuthContext.Provider está por volta das duas routes da aplicação, as routes irão conseguir enxergar o valor do contexto. As routes Home e NewRoom irão enxergar o valor do contexto.

**\*\*\*** O **exact** é uma propriedade *boolean* - ou seja, por padrão ela vem true (por isso não setamos o valor true nela e, utilizamos apenas atribuindo ‘exact’).  
 • A função do exact é dizer ao react-router-dom que o path é exatamente ‘/’ e não, algo que comece com ‘/’, sem o exact estaria mostrando as duas telas ao mesmo tempo, por que a page NewRoom também começa com ‘/’.

**\*\*\* Resumo da aula:**

- Começamos com a estilização, estrutura das pages.  
 • criamos todo o HTML para depois usar o CSS.

- React por padrão importação de imagens, CSS, tudo dentro do próprio **Componente.  
 • Sempre importação.**

- Utilizamos SCSS – pré processador CSS.

- Aprendemos utilizar routes e navegação.

- Autenticação com Firebase

- Função Hook começa com “use” e hooks são funções utilizadas apenas no contexto do Componente.

**• [NLW/Together - ReactJS] Stage 3 – Encontro ao-vivo [Twitch]**

**Componente:**

1. Aquilo se repete;
2. Consigo isolar o funcionamento de restante;

**Link** – é um componente para você redirecionar um usuário de uma página a outra.  
**Redirect** – o user não precisa clicar, quando o componente aparecer em tela, ele já é redirecionado para outra página.

**useHistory** – quando precisa fazer uma navegação para o usuário, que não venha de um clique, por exemplo de um Link, caso uma ação der certo, cadastro deu certo, ai vamos redirecionar o user, usando o useHistory.

► Caso queira implementar um button Sign Out:

async function signOut() {  
 await auth.signOut();  
}

• e fazer o button chamar a função signOut()

**\*\*\* Sintaxe do useEffect:**

• o primeiro parâmetro do useEffect, é a função que será executada e, o segundo parâmetro é a informação que ele estará monitorando. Lembrando que o useEffect sempre dispara uma função na mudança de uma informação.  
 • caso o segundo parâmetro esteja vazio ( [] ), ele irá executar o useEffect UMA vez em que o componente aparecer em tela.   
 • **CUIDADO:** caso o segundo parâmetro não for declarado, sua aplicação entrará em loop, bugando e crashando.

**+++**

import { useEffect } from 'react';

function App() {

  const newInput = '';

  useEffect( () => {

    // event listener

  return () => {

    // remove os event listeners

      }

    }, [newInput])

  }

**• Generics <> (TypeScript) –** um parâmetro do TypeScript. Uma forma de parametrizar uma tipagem.

import { useState } from 'react';

// Pela comunidade => a parametrização para string é com T

type *User*<*T* = *string*> = {

  name: *T*;

}

const user: *User*<*number*> = {

  name: 1,

}

function App() {

  // Parâmetro do TypeScript para dizer que o valor name é do tipo string.

  const [ name, setName ] = useState<*string*>();

  }