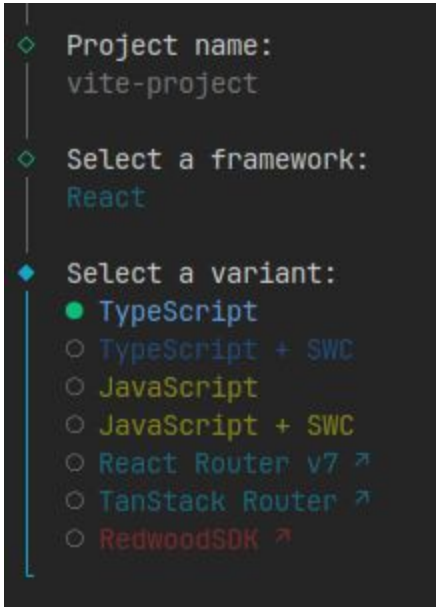
REACT

SAE COMP

Criando o projeto

Para criar o projeto se utiliza o vite com o comando npm create vite@latest.



Configurando o projeto

Também é necessário configurar o tailwind para funcionar no projeto.

```
cd vite-project
npm install
npm install tailwindcss @tailwindcss/vite
```

```
// vite.config.ts

import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'
import tailwindcss from '@tailwindcss/vite'


// https://vite.dev/config/
export default defineConfig({
  plugins: [react(), tailwindcss()],
})
```

Configurando o projeto

Também é necessário configurar o tailwind para funcionar no projeto.

```css
/* index.css */

@import "tailwindcss";

```

```tsx
// main.tsx

import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.tsx'

createRoot(document.getElementById('root')!).render(
  <StrictMode>
    <App />
  </StrictMode>,
)
```

Renderização dos componentes

O react é um biblioteca de JS que a partir de uma tag pré-definida no HTML, constrói componentes modularizados e reativos.

```html
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Vite + React + TS</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.tsx"></script>
  </body>
</html>
```

```tsx
// main.tsx

import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.tsx'

createRoot(document.getElementById('root')!).render(
  <StrictMode>
    <App />
  </StrictMode>,
)
```

Componentes

Assim como no JavaScript se separa os códigos em diversos arquivos, no react também se separa em diversos componentes. Todo componente tem que necessariamente exportar um HTML.

```tsx
import { useState } from 'react';
import Button from './components/Button';

function App() {
    const [count, setCount] = useState<number>(0);

    const showAlert = () => {
        alert('Valor do contador: ' + count);
    }

    return (
        <>
            <div className='flex justify-center items-center h-screen w-screen flex-col'>
                <p className='text-4xl font-extrabold mb-5' onMouseOver={showAlert}>
                    {count}
                </p>
                <Button
                    setCount={setCount}
                    count={count}
                />
            </div>
        </>
    )
}

export default App;
```

Componentes

Componentes também podem ter props, que são como se fosse argumentos de uma função só que passando como propriedade HTML.

```tsx

interface IButton {
    setCount: React.Dispatch<React.SetStateAction<number>>;
    count: number;
}

function Button({ setCount, count }: IButton) {
    return (
        <button
            className='h-10 w-32 bg-amber-300 rounded-2xl cursor-pointer'
            onClick={() => {
                setCount(count + 1);
            }}>
            botão
        </button>
    )
};

export default Button;
```

useEffect

O useEffect executa uma função toda vez que um dos states definidos em sua lista são modificados.

```
useEffect(() => {
    setImgList((curr) => {
        const url = imgGenerator + count;
        return curr.includes(url) ? curr : [...curr, imgGenerator + count];
    });
}, [count]);
```

map em componentes

Para fazer com que uma lista seja mostrada dinamicamente, fazemos map sobre ela e o retorno deve ser o componente HTML que vai ser mostrado para cada item da lista.

```
{imgList.map((img, index) => (
    <div className='flex flex-col justify-center items-center shrink-0' key={index}>
        <img
            key={index}
            src={img}
            alt={`Random ${index}`}
            className='h-32 w-32 m-2 rounded-lg'
        />
        <p>{index}</p>
    </div>
))}
```

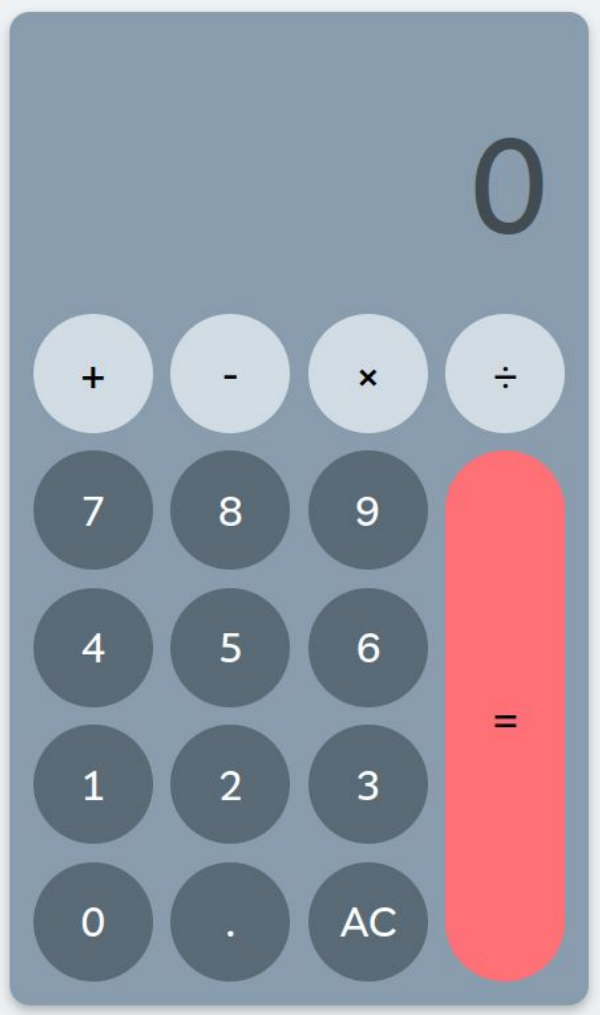
Projeto completo

Caso algo não funcione durante a criação do projeto. Baixe o projeto pronto aqui e rode npm install.

ATIVIDADES

Atividade 1

fazer a calculadora com react e tailwind utilizando componentes modulares e map para criar os botões



MUITO OBRIGADO


@saecomp.ec


saecomp@usp.br


saecomp.github.io


Prédio da Engenharia de Computação,
Campus 2, USP São Carlos

SAECOMP