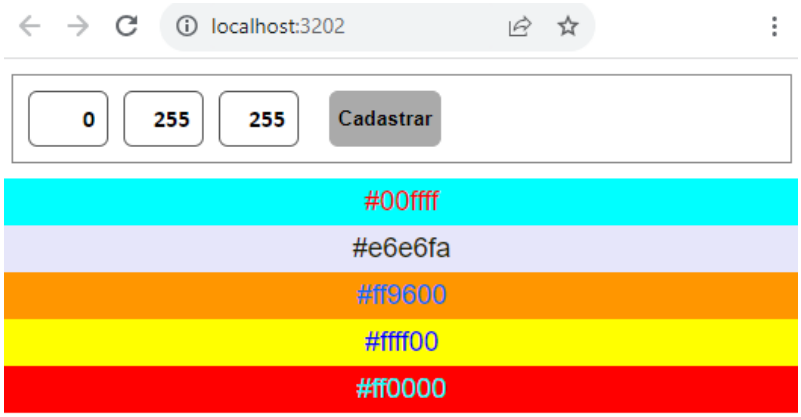
Prova III – Tipo A - Desenvolvimento Web II – DSM - Prof. Arley - 27/11/2023

Instruções:

- A prova é individual e com consulta ao próprio material e internet. Não é permitido consultar material de outra pessoa da sala;
- Fazer a aplicação no VS Code;
- A prova encerra-se às 22h00. Ao terminar, você deverá chamar o professor para apresentar a aplicação.

Objetivo: Fazer uma aplicação front-end, usando React TS, para cadastrar cores no formato RGB. A aplicação deverá usar o back-end disponível em https://github.com/arleysouza/prova-a-back.



Requisitos funcionais:

1. (1 pt.) As cores salvas no servidor devem ser carregadas na tela ao iniciar a aplicação;
2. (1 pt.) Cada registro de cor deverá ser exibido numa linha, a cor deve ser carregada no formato #rrggbb;
3. (1 pt.) Cada registro de cor deverá ser exibido numa linha, com a cor de fundo igual ao valor da cor;
4. (1 pt.) Cada registro de cor deverá ser exibido numa linha, com a cor do texto igual ao inverso da cor;
5. (1 pt.) Cada registro de cor deverá ser exibido numa linha, ao clicar sobre a linha o registro deverá ser removido no servidor e a tela deverá ser atualizada;
6. (1 pt.) Ao clicar no botão cadastrar um novo registro de cor deverá ser criado no servidor e a tela deverá ser atualizada.

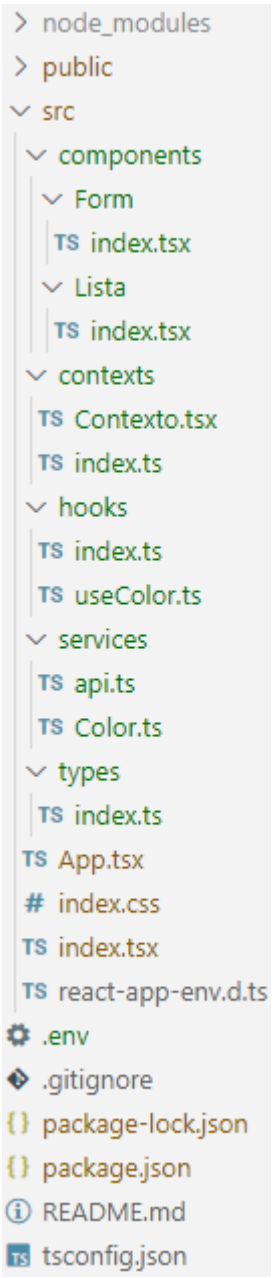
Requisitos não funcionais:

1. A aplicação servidora não poderá ser alterada;
2. (1 pt.) As requisições devem estar num pacote chamado service;
3. (2 pts.) A comunicação entre os componentes deverá utilizar Context e Hooks;
4. (1 pt.) Os componentes e estilos devem ser definidos no pacote components.

Dicas:

Prova III – Tipo A - Desenvolvimento Web II – DSM - Prof. Arley - 27/11/2023

a) Ao lado tem-se uma sugestão de estrutura de pastas e arquivos;



b) No arquivo index.css defina a fonte a ser utilizada na página:

```css
body {
  margin: 0;
  font-family: Arial;
}
```

c) O servidor responde nas seguintes rotas:

- Listar os registros: http://localhost:3201
- Inserir um registro com os valores R:0, G:255 e B:200: http://localhost:3201/0/255/200
- Remover o registro que possui id:1: http://localhost:3201/1

d) Dica de parte do código para o arquivo services/index.ts:

```ts
export interface ColorProps {
  id?: number;
  red: number;
  green: number;
  blue: number;
}

export interface ResultProps {
  count: number;
}
```

e) Dica de código para o arquivo services/api.ts:

```ts
import axios, { AxiosInstance } from "axios";

const api:AxiosInstance = axios.create({
    baseURL: "http://localhost:3201",
    headers: {
        "Content-Type":"application/json"
    }
});

export default api;
```

f) Dica de código para o arquivo services/Color.ts:

```ts
import { ColorProps, ResultProps } from "../types";
import api from "./api";

class Color {
    async list(): Promise<ColorProps[]> {
        const {data} = await api.get("/");
        return data;
    }
```

```
    async create(props:ColorProps): Promise<ColorProps> {
        const {data} = await api.get(`/${props.red}/${props.green}/${props.blue}`);
        return data;
    }

    async remove(id:number): Promise<ResultProps> {
        const {data} = await api.get(`/${id}`);
        return data;
    }
}

const color = new Color();
export default color;
```

g) Dica de código para gerar o hexadecimal da cor e para obter o inverso da cor:

```
function rgbHexadecimal(color: ColorProps) {
  const r =
    color.red.toString(16).length < 2
      ? "0" + color.red.toString(16)
      : color.red.toString(16);
  const g =
    color.green.toString(16).length < 2
      ? "0" + color.green.toString(16)
      : color.green.toString(16);
  const b =
    color.blue.toString(16).length < 2
      ? "0" + color.blue.toString(16)
      : color.blue.toString(16);
  return `#${r}${g}${b}`;
}

function inverseColor(color: ColorProps) {
  const rinverse = 255 - color.red, ginverse = 255 - color.green,
    binverse = 255 - color.blue;
  const r =
    rinverse.toString(16).length < 2
      ? "0" + rinverse.toString(16)
      : rinverse.toString(16);
  const g =
    ginverse.toString(16).length < 2
      ? "0" + ginverse.toString(16)
      : ginverse.toString(16);
  const b =
    binverse.toString(16).length < 2
      ? "0" + binverse.toString(16)
      : binverse.toString(16);
  return `#${r}${g}${b}`;
}
```