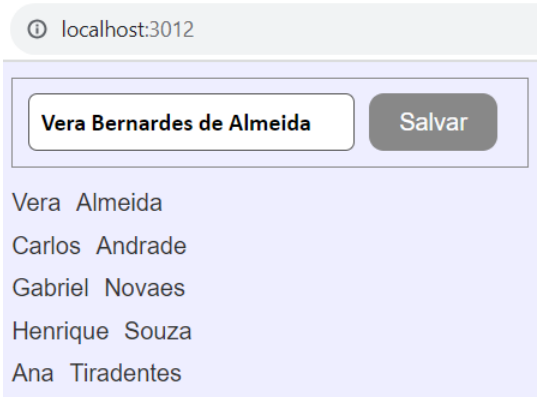
Prova III – Tipo A - Desenvolvimento Web II – DSM - Prof. Arley - 18/06/2024

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 21h00. Ao terminar, você deverá chamar o professor para apresentar a aplicação.

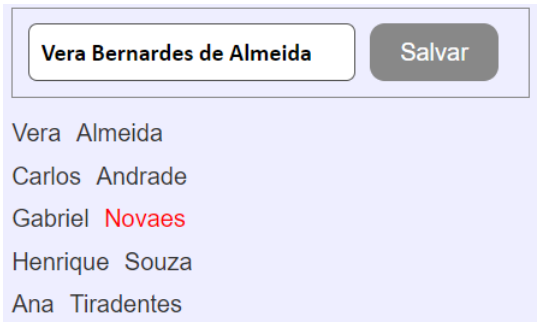
Objetivo: Fazer uma aplicação front-end, usando React TS, para persistir o 1º e o último nome de uma pessoa. As operações devem ser persistidas no back-end disponível em https://github.com/arleysouza/prova-a-back.



Requisitos funcionais:

1. (1 pt.) Ao clicar no botão Salvar deverá ser extraído o 1º e o último nome do campo de entrada e persistido no servidor. Após salvar no servidor a lista de nomes deverá ser atualizada na tela;

2. (1 pt.) Ao posicionar o cursor sobre o nome ou sobrenome o texto deverá ter cor de texto alterado para vermelho. No exemplo ao lado o cursor está sobre o texto Novaes;



Prova III – Tipo A - Desenvolvimento Web II – DSM - Prof. Arley - 18/06/2024

3. (1 pt.) Ao clicar com o botão esquerdo do mouse sobre algum nome deverá ser feita uma consulta no servidor para obter os dados cadastrados e ordenados pelo campo nome e ao clicar com o botão esquerdo sobre algum sobrenome deverá ser feita uma consulta no servidor para obter os dados cadastrados e ordenados pelo campo sobrenome;

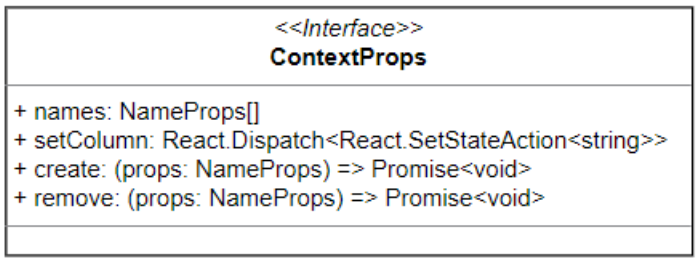
4. (1 pt.) Ao clicar com o botão direito sobre o nome ou sobrenome deverá ser feita uma consulta no servidor para remover o registro. Após remover no servidor a lista de nomes deverá ser atualizada na tela;

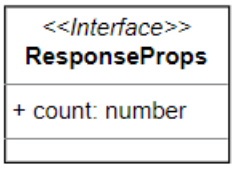
5. (1 pt.) Ao iniciar a aplicação deverá ser feita uma consulta no servidor e os nomes cadastrados devem ser carregados na tela. A lista de nomes cadastrados deverá ser ordenada pelo 1º nome.

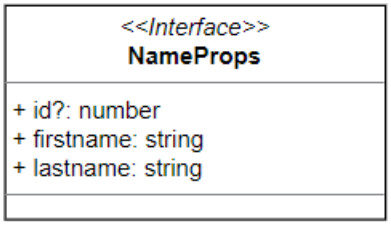
Requisitos não funcionais:

Prova III – Tipo A - Desenvolvimento Web II – DSM - Prof. Arley - 18/06/2024

1. A aplicação servidora não poderá ser alterada;
2. (1 pt.) Codificar no arquivo src/types/index.ts as interfaces NameProps, ResponseProps e ContextProps;







3. (1 pt.) Codificar no arquivo src/services/index.ts o corpo dos métodos create, remove e list da classe Service:

```
import axios, { AxiosInstance } from "axios";
import { NameProps, ResponseProps } from "../types";

class Service {
    private api:AxiosInstance = axios.create({
        baseURL: "http://localhost:3011",
        headers: {
            "Content-Type":"application/json"
        }
    });

    async list(column:string): Promise<NameProps[]> {  }

    async create(props:NameProps): Promise<NameProps> {  }

    async remove(props:NameProps): Promise<ResponseProps> {  }
}

const service = new Service();
export default service;
```

4. (1 pt.) Codificar no arquivo src/contexts/Contexto.ts o corpo das funções create, remove, list e useEffect:

```
import { createContext, useEffect, useState } from "react";
import { ContextProps, NameProps } from "../types";
import service from "../services";
```

```
export const Contexto = createContext({} as ContextProps);

export function Provider({ children }: any) {
  const [names, setNames] = useState([] as NameProps[]);
  const [column, setColumn] = useState("firstname");

  const list = async () => {  };

  const create = async (name: NameProps) => {  };

  const remove = async (name: NameProps) => {  };

  useEffect(() => {  }, []);

  return (
    <Contexto.Provider value={{ names, setColumn, create, remove }}>
      {children}
    </Contexto.Provider>
  );
}
```

5. (0,5 pt.) As propriedades `{ names, setColumn, create, remove }` do contexto devem ser acessadas nos componentes através do hooks definido na pasta src/hooks;
6. (1 pt.) A tela deverá ser formada pelos componentes Form e List. No componente Form deverá estar o campo de entrada e o botão Salvar. No componente List deverão estar os dados cadastrados;
7. (0,5 pt.) Os estilos dos componentes Form e List devem usar Styled components.