1) Criar o projeto e configurar Tailwind

npm create vite@latest frontend --template react-ts

cd frontend

npm install

npm i -D tailwindcss@3 postcss autoprefixer

2) Criar tailwind.config.ts

/\*\* @type {import('tailwindcss').Config} \*/

export default {

  content: ["./index.html", "./src/\*\*/\*.{js,ts,jsx,tsx,ts,tsx}"],

  theme: { extend: {} },

  plugins: [],

};

3) Criar postcss.config.js

export default {

  plugins: {

    tailwindcss: {},

    autoprefixer: {},

  },

};

4) Alterar src/index.css

@tailwind base;

@tailwind components;

@tailwind utilities;

body {

  margin: 0;

  display: flex;

  justify-content: center; /\* centraliza na horizontal \*/

  align-items: flex-start; /\* mantém alinhado no topo (se não quiser no meio da tela) \*/

  min-width: 320px;

  min-height: 100vh;

}

5) Criar .env

VITE\_API\_URL=http://localhost:3000

6) Criar src/types.ts

export type TipoDeficiencia = {

  id: number;

  nome: string;

  createdAt?: string;

  updatedAt?: string;

};

7) Criar src/lib/api.ts

import type { TipoDeficiencia } from "../types";

const BASE\_URL = import.meta.env.VITE\_API\_URL ?? "http://localhost:3000";

async function http<T>(path: string, init?: RequestInit): Promise<T> {

  const res = await fetch(`${BASE\_URL}${path}`, {

    headers: { "Content-Type": "application/json", ...(init?.headers || {}) },

    ...init,

  });

  if (!res.ok) {

    const text = await res.text().catch(() => "");

    let msg = text || res.statusText || "Erro na requisição";

    try {

      const j = JSON.parse(text);

      msg = j.error || msg;

    } catch {}

    throw new Error(msg);

  }

  return res.json() as Promise<T>;

}

export const api = {

  listarTipos() {

    console.log(http<TipoDeficiencia[]>("/tipos"))

    return http<TipoDeficiencia[]>("/tipos");

  },

  criarTipo(nome: string) {

    return http<TipoDeficiencia>("/tipos", {

      method: "POST",

      body: JSON.stringify({ nome }),

    });

  },

};

8) Criar src/components/TipoForm.tsx

import { useState } from "react";

import { api } from "../lib/api";

type Props = {

  onCreated: () => void; // para recarregar a lista ao criar

};

export default function TipoForm({ onCreated }: Props) {

  const [nome, setNome] = useState("");

  const [loading, setLoading] = useState(false);

  const [erro, setErro] = useState<string | null>(null);

  async function handleSubmit(e: React.FormEvent) {

    e.preventDefault();

    setErro(null);

    const trimmed = nome.trim();

    if (!trimmed) {

      setErro("Informe um nome.");

      return;

    }

    setLoading(true);

    try {

      await api.criarTipo(trimmed);

      setNome("");

      onCreated();

    } catch (err: any) {

      setErro(err?.message ?? "Erro ao criar tipo");

    } finally {

      setLoading(false);

    }

  }

  return (

    <form onSubmit={handleSubmit} className="card space-y-3">

      <div>

        <label className="label">Nome do tipo</label>

        <input

          className="input"

          placeholder="ex.: Deficiência Cognitiva"

          value={nome}

          onChange={(e) => setNome(e.target.value)}

          disabled={loading}

        />

        {erro && <p className="error">{erro}</p>}

      </div>

      <div className="flex justify-end">

        <button disabled={loading} className="btn btn-primary">

          {loading ? "Salvando..." : "Criar tipo"}

        </button>

      </div>

    </form>

  );

}

9) Criar src/components/TipoList.tsx

import type { TipoDeficiencia } from "../types";

type Props = {

  tipos: TipoDeficiencia[];

};

export default function TipoList({ tipos }: Props) {

  if (!tipos.length) {

    return <div className="card text-gray-500">Nenhum tipo cadastrado ainda.</div>;

  }

  return (

    <div className="card">

      <h3 className="text-lg font-semibold mb-3">Tipos cadastrados</h3>

      <ul className="divide-y">

        {tipos.map((t) => (

          <li key={t.id} className="py-2 flex items-center justify-between">

            <div>

              <span className="font-medium">{t.nome}</span>

            </div>

            <span className="text-xs text-gray-400">#{t.id}</span>

          </li>

        ))}

      </ul>

    </div>

  );

}

10) Criar src/pages/TiposPage.tsx

import { useEffect, useState } from "react";

import TipoForm from "../components/TipoForm";

import TipoList from "../components/TipoList";

import { api } from "../lib/api";

import type { TipoDeficiencia } from "../types";

export default function TiposPage() {

  const [tipos, setTipos] = useState<TipoDeficiencia[]>([]);

  const [loading, setLoading] = useState(true);

  const [erro, setErro] = useState<string | null>(null);

  async function carregar() {

    setErro(null);

    setLoading(true);

    try {

      const data = await api.listarTipos();

      setTipos(data);

    } catch (err: any) {

      setErro(err?.message ?? "Erro ao carregar tipos");

    } finally {

      setLoading(false);

    }

  }

  useEffect(() => {

    carregar();

  }, []);

  return (

   <div className="container-page space-y-6 py-8">

            <header>

                <h1 className="text-2xl font-bold">Tipos de Deficiência</h1>

                <p className="text-gray-600">Crie e consulte os tipos cadastrados na API.</p>

            </header>

        <TipoForm onCreated={carregar} />

      {loading ? (

        <div className="card">Carregando...</div>

      ) : erro ? (

        <div className="card text-red-600">{erro}</div>

      ) : (

        <TipoList tipos={tipos} />

      )}

        </div>

  );

}

11) Alterar src/App.tsx

import TiposPage from "./pages/TiposPage";

export default function App() {

  return (

    <div className="min-h-screen bg-gray-50">

      <TiposPage />

    </div>

  );

}

12) executar npm run dev