Documentación técnica – Aplicación de mapa con React Leaflet

github link : https://github.com/OthmanDouiri/react-map.git

1. Descripción general

Esta aplicación es una herramienta interactiva basada en mapas, desarrollada con React y TypeScript, que permite al usuario:

Permite a los usuarios buscar ciudades, añadirlas al mapa, mostrar información meteorológica de las ubicaciones y cambiar el estilo visual del mapa.

El objetivo principal del proyecto es demostrar el uso combinado de librerías como **react-leaflet** con múltiples APIs externas, en este caso:

OpenCage Geocoder : para obtener coordenadas a partir del nombre de la ciudad)
OpenWeatherMap : para obtener información del clima), implementando una funcionalidad completa y útil para el usuario.

---

2. Especificaciones técnicas

- **Lenguaje:** TypeScript
- **Framework:** React 18 (Vite)
- **Librerías principales:**
    - react-leaflet : Para renderizar y gestionar el mapa interactivo.
    - Leaflet : Motor de mapas.
    - OpenCage Geocoder API : Para convertir nombres de ciudades en coordenadas geográficas.
    - OpenWeatherMap API : Para obtener datos meteorológicos.
- **Requisitos del sistema:**

- Navegador moderno (Chrome, Firefox, Edge).
- Conexión a internet (para acceder a las APIs y tiles del mapa).

---

3. Estructura del código

Componentes clave:

- MapView.tsx: Componente principal que gestiona el mapa, la lógica de búsqueda, la geolocalización, los marcadores y el cambio de estilo del mapa.

Estructura del código:

Importaciones:

```
import {
  MapContainer,
  TileLayer,
  Marker,
  Popup,
} from 'react-leaflet';
import L from 'leaflet';
import 'leaflet/dist/leaflet.css';
import { useRef, useState } from 'react';
```

Se importan componentes específicos de React Leaflet para construir el mapa
Se importa la biblioteca Leaflet y sus estilos CSS
Se importan hooks de React (**useRef** y **useState**) para gestionar el estado y referencias

Configuración de Iconos:

```
delete (L.Icon.Default.prototype as any)._getIconUrl;
L.Icon.Default.mergeOptions({
  iconRetinaUrl: 'https://unpkg.com/leaflet@1.9.4/dist/images/marker-icon-2x.png',
  iconUrl: 'https://unpkg.com/leaflet@1.9.4/dist/images/marker-icon.png',
  shadowUrl: 'https://unpkg.com/leaflet@1.9.4/dist/images/marker-shadow.png',
});
```

Soluciona un problema con los iconos de marcadores en Leaflet.

Definición de tipos :

```
type City = {
  id: number;
  name: string;
  lat: number;
  lng: number;
  weather: string | null;
};
```

Define la estructura de datos para las ciudades.
Cada ciudad tiene un ID único, nombre, coordenadas (latitud y longitud) e información del clima.

Estados y Referencias:

```
const MapView: React.FC = () => {
  const [cities, setCities] = useState<City[]>([]);
  const [query, setQuery] = useState('');
  const [mapStyle, setMapStyle] = useState<'streets' | 'terrain'>('streets');
  const mapRef = useRef<any>(null);
```

cities: Almacena la lista de ciudades añadidas al mapa
query: Gestiona el texto de búsqueda ingresado por el usuario
mapStyle: Controla el estilo de visualización del mapa (calles o terreno)
mapRef: Mantiene una referencia al objeto del mapa para manipularlo programáticamente

Funciones principales:

- **addCity()**: Busca una ciudad por nombre (usando OpenCage), obtiene sus coordenadas y clima (OpenWeather) y añade un marcador al mapa.
- **removeCity(id)**: Elimina una ciudad del array y por tanto su marcador.
- **locateMe()**: Usa la API de geolocalización del navegador para centrar el mapa en la posición actual del usuario.
- **getTileLayer():** Devuelve la URL del estilo de mapa seleccionado (calles o terreno).

Gestión de APIs

- Ambas APIs se consumen mediante fetch, y las claves están protegidas con variables de entorno (import.meta.env.VITE_*).

---

4. Uso, ejemplos y pruebas

Buscar una ciudad:

1. Introduce una ciudad en el input (por ejemplo, Paris).
2. Pulsa el botón "Agregar".
3. Se coloca un marcador en el mapa, mostrando su nombre y clima actual.
4. En el popup puedes eliminar la ciudad.

Ubicación actual:

- Pulsa el botón “Mi ubicación” y se centra el mapa automáticamente sobre tu posición.

Cambiar estilo de mapa:

- Pulsa el botón "Calles"/"Terreno" para alternar entre los dos estilos.

Pruebas realizadas:

- Búsqueda de ciudades conocidas y desconocidas.
- Eliminación de ciudades.
- Pruebas con conexión lenta/desconectada.
- Pruebas con y sin permisos de geolocalización.

Posibles mejoras o extensiones

- Guardar las ciudades en localStorage para mantenerlas entre sesiones.
- Autocompletado del input al escribir ciudades (usando por ejemplo la API de Mapbox o Algolia Places).

- Mostrar más datos del clima.
- Historial de ciudades buscadas.
- Modo oscuro.