PRIMER PILOTO

Mapa Interactivo, Departamento del Cauca

Herramientas que hemos usado:

VS Code: Visual Studio Code es un editor de código fuente desarrollado por Microsoft. Tiene una excelente integración con tecnologías web, como JavaScript, React y Node.js, además de una gran cantidad de extensiones que nos ayudan a trabajar de manera más eficiente.

JavaScript: JavaScript es fundamental para cualquier desarrollo web interactivo. Permite la manipulación del DOM (Document Object Model) y hace posible la interacción dinámica con el mapa en la página web, como mover, hacer zoom, y agregar marcadores.

React: React nos permite construir la interfaz de usuario de nuestro mapa interactivo de una manera modular y eficiente. Su sistema de componentes facilita la creación y gestión de elementos del mapa, y su capacidad de manejar el estado hace que las actualizaciones del mapa sean rápidas y reactivas.

Node.js: se utiliza para ejecutar el entorno de desarrollo y gestionar dependencias mediante npm. También puede ser utilizado para desarrollar el backend de la aplicación si es necesario, el principal rol es como herramienta para manejar el entorno y construir la aplicación.

Leaflet: es la herramienta clave para crear el mapa interactivo. Es ligera, flexible y fácil de integrar con React. Leaflet te permite agregar y personalizar mapas, marcadores, pop-ups y otras funcionalidades interactivas directamente en la página web, lo que es esencial para el objetivo de tu proyecto.

Código del primer piloto:

archivo MapView.js

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

// Corrige los íconos rotos
delete L.Icon.Default.prototype._getIconUrl;

L.Icon.Default.mergeOptions({
    iconRetinaUrl: require('leaflet/dist/images/marker-icon-2x.png'),
    iconUrl: require('leaflet/dist/images/marker-icon.png'),
    shadowUrl: require('leaflet/dist/images/marker-shadow.png'),
});
```

```
const MapView = () => {
  return (
    <MapContainer center={[2.45, -76.61]} zoom={10} style={{ height: '100vh', width:
'100%' }}>
      <TileLayer
        url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png"
        attribution='&copy; <a
href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'
      />
      <Marker position={[2.45, -76.61]}>
        <Popup>
          ¡Aquí está el departamento del Cauca!
        </Popup>
      </Marker>
    </MapContainer>
  );
}

export default MapView;
```

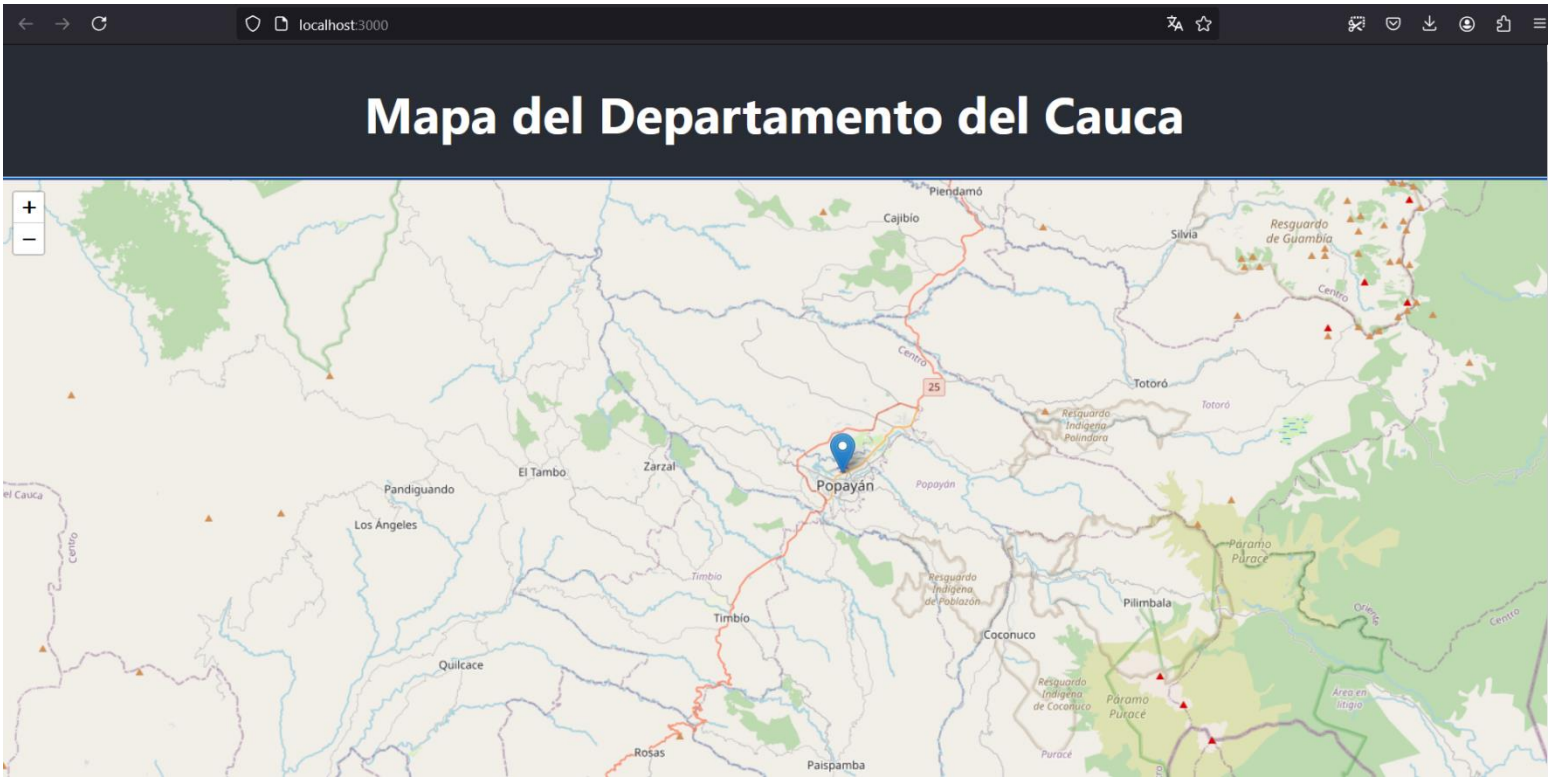
archivo app.js

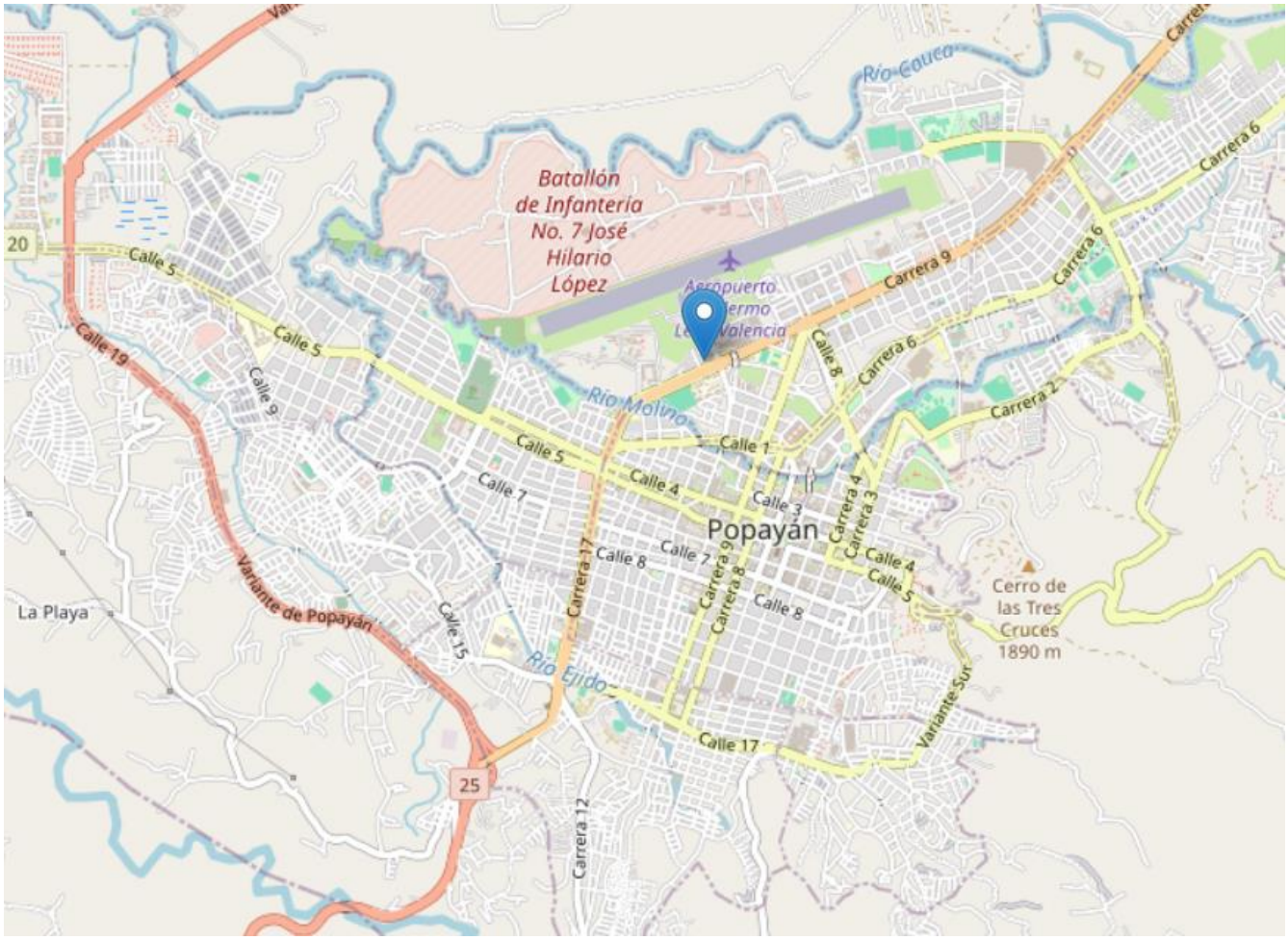
```
import logo from './logo.svg';
import './App.css';
import MapView from './MapView';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <h1>Mapa del Departamento del Cauca</h1>
        <MapView />
      </header>
    </div>
);
}

export default App;
```

El resultado es el siguiente:





Explicación del Código:

Archivo **MapView.js**

1. Corrección de íconos rotos en Leaflet:

```
// Corrige los íconos rotos
delete L.Icon.Default.prototype._getIconUrl;

L.Icon.Default.mergeOptions({
    iconRetinaUrl: require('leaflet/dist/images/marker-icon-2x.png'),
    iconUrl: require('leaflet/dist/images/marker-icon.png'),
    shadowUrl: require('leaflet/dist/images/marker-shadow.png'),
});
```

Este bloque de código corrige un problema común en Leaflet donde los íconos de los marcadores no se cargan correctamente en ciertas configuraciones. Se eliminan las referencias antiguas a las URLs de los íconos predeterminados y se establecen nuevas rutas para asegurar que los íconos de los marcadores se muestren correctamente en el mapa.

2. Componente MapView

```
const MapView = () => {
    return (
        <MapContainer center={[2.45, -76.61]} zoom={10} style={{ height: '100vh', width:
'100%' }}>
            <TileLayer
                url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png"
                attribution='&copy; <a
href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'
            />
            <Marker position={[2.45, -76.61]}>
                <Popup>
                    ¡Aquí está el departamento del Cauca!
                </Popup>
            </Marker>
        </MapContainer>
    );
}

export default MapView;
```

Este bloque define un componente de React llamado `MapView`. Este componente es responsable de mostrar el mapa interactivo en la página web.

Código del archivo **App.js**

Importaciones y configuración:

```
import logo from './logo.svg';
import './App.css';
import MapView from './MapView';
```

Este bloque importa recursos necesarios para la aplicación:

- **logo**: Importa una imagen de logo.
- **App.css**: Importa estilos CSS para dar diseño a la aplicación.
- **MapView**: Importa el componente MapView que definiste en el archivo anterior para mostrarlo en la página.

Componente App:

```
function App() {
  return (
    <div className="App">
      <header className="App-header">
        <h1>Mapa del Departamento del Cauca</h1>
        <MapView />
      </header>
    </div>
);
}

export default App;
```

Este bloque define el componente principal de la aplicación, App, que es el punto de entrada de tu aplicación React.

- ** **: Es un contenedor para toda la aplicación.
- **<header className="App-header">**: Es la cabecera de la aplicación donde se muestra un título y se incluye el componente MapView.
- **<h1>**: Muestra el título "Mapa del Departamento del Cauca" en la parte superior de la página.
- **<MapView />**: Inserta el mapa interactivo en la aplicación.

Exportación: Finalmente, App se exporta para que pueda ser utilizado en otros lugares de tu proyecto, como el punto de entrada principal.