**Location finder app using ReactJS**

**Creating a React application and installing some npm packages:**

**Step 1:**Create a react application by typing the following command in the terminal:

npx create-react-app location-finder

**Step 2:** Now, go to the project folder i.e. location-finder by running the following command:

cd location-finder

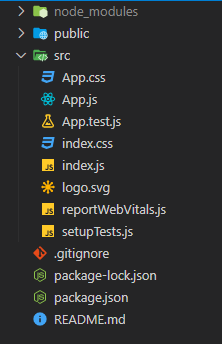
**Step 3:** Let’s install some npm packages required for this project:

npm install react-map-gl

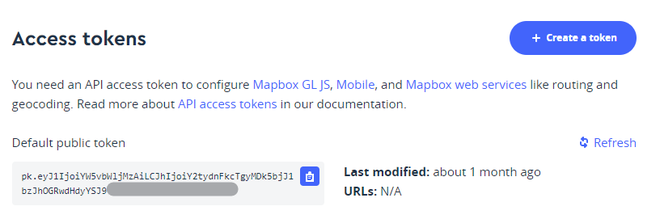
npm install axios

npm install react-icons

**Project Structure:**It will look like this:



**Example:**Let us grab the Mapbox API key required for this project. Follow the simple steps below:



* App.js

|  |
| --- |
| import { useEffect, useState } from "react";  import ReactMapGL, { Marker, FlyToInterpolator }        from "react-map-gl";  import Fly from "./Components/Fly";  import { ImLocation } from "react-icons/im";  import "./App.css";    function App() {      // Setting up the state for the latitude    // and longitude    const [lat, setLat] = useState(22.5726);    const [lon, setLon] = useState(88.3639);      // Setting up the state for the map    const [viewport, setViewport] = useState({      latitude: lat,      longitude: lon,      zoom: 12,      bearing: 0,      pitch: 0,      width: "100%",      height: "100vh",    });      // Map viewport updates whenever the    // latitude and longitude changes    useEffect(() => {      setViewport({        latitude: lat,        longitude: lon,        zoom: 12,        transitionInterpolator:          new FlyToInterpolator({ speed: 1.0 }),        transitionDuration: "auto",        width: "100%",        height: "100vh",      });    }, [lat, lon]);      return (      <ReactMapGL        mapboxApiAccessToken={"<YOUR\_API\_KEY>"}        {...viewport}        onViewportChange={(viewport) => setViewport(viewport)}      >        <Marker latitude={lat} longitude={lon}>          <ImLocation size="30px" />        </Marker>        <Fly setLat={setLat} setLon={setLon} />      </ReactMapGL>    );  }    export default App; |

In the App.js component, we have also imported our own custom component named “Fly” which is nothing but a simple box that takes user input, calls a forward geocoding API (https://docs.mapbox.com/api/search/geocoding/) provided by Mapbox itself, and sets the coordinates accordingly. Let us create that component.

Create a folder under the **src** folder named “**Components**” and under that folder create a file named “**Fly.jsx**“

**Filename: Fly.jsx** Now write down the following code in the Fly.jsx file.

* Fly.jsx

|  |
| --- |
| import React, { useState } from "react";  import Axios from "axios";    const API\_KEY = "<YOUR\_API\_KEY>";    const Fly = ({ setLat, setLon }) => {      // Setting up the state variable to store user input    const [city, setCity] = useState("Kolkata");      // Function to call the API and set the    // coordinates accordingly    function getCoordinates() {      Axios.get(        `https://api.mapbox.com/geocoding/v5/          mapbox.places/${city}.json?access\_token=${API\_KEY}`      ).then((res) => {          // Longitude        setLon(res.data.features[0].geometry.coordinates[0]);          // Latitude        setLat(res.data.features[0].geometry.coordinates[1]);      });    }      return (      <div className="fly">        <h2>Enter a city name</h2>        <div className="inp-box">          <input            type="text"            onChange={(e) => {              setCity(e.target.value);            }}          />          <button onClick={() => getCoordinates()}>Go</button>        </div>      </div>    );  };    export default Fly; |

**Remember** to replace <YOUR\_API\_KEY> with your own Mapbox public access token.

**Filename: App.css** Now let’s edit the file named **App.css**to style our app.

* App.css

|  |
| --- |
| .fly {    display: flex;    align-items: center;    justify-content: center;    width: 300px;    height: 100px;    color: white;    background-color: #3061e7;    margin-top: 1%;    margin-left: 40%;    padding: 10px;    border-radius: 5px;  }  .fly input {    padding-left: 5px;    font-size: 18px;    height: 30px;  }  .fly button {    cursor: pointer;    width: 50px;  } |

**Step to Run Application:** Run the application using the following command from the root directory of the project.

npm start

**Output:** Now open your browser and go to **http://localhost:3000/**, you will see the following output:

