import React, { useState, useEffect } from 'react';

import { GoogleMap, LoadScript, MarkerF, InfoWindow } from '@react-google-maps/api';

import droneImageBase64 from './droneImage';

import './DroneMap.css';

const center = {

  lat: 0,

  lng: 0

};

const createCustomMarker = (label) => {

  return `

    <svg width="100" height="100" xmlns="http://www.w3.org/2000/svg">

      <image x="20" y="20" width="60" height="60" href="${droneImageBase64}" />

      <text x="50" y="15" font-size="12" font-family="Arial" font-weight="bold" text-anchor="middle" fill="black">${label}</text>

    </svg>

  `;

};

const DroneMap = () => {

  const [droneData, setDroneData] = useState([]);

  const [currentPosition, setCurrentPosition] = useState(center);

  const [currentSpeed, setCurrentSpeed] = useState(0);

  const [currentDroneId, setCurrentDroneId] = useState('');

  const [mapsLoaded, setMapsLoaded] = useState(false);

  const [selectedCenter, setSelectedCenter] = useState(null);

  useEffect(() => {

    const fetchData = async () => {

      try {

        const response = await fetch('http://localhost:8000/stream\_data');

        const data = await response.json();

        console.log('Fetched data:', data); // Log fetched data to inspect its structure

        if (!data.error) {

          setDroneData(prevData => [...prevData, data]);

          setCurrentPosition({

            lat: parseFloat(data.Position.Latitude),

            lng: parseFloat(data.Position.Longitude)

          });

          // Simulating speed and id as they are not part of the response

          setCurrentSpeed(0.066363276); // Replace with actual speed if available

          setCurrentDroneId("No.1"); // Replace with actual ID if available

        } else {

          console.error("Error in data:", data.error);

        }

      } catch (error) {

        console.error("Error fetching drone data:", error);

      }

    };

    const intervalId = setInterval(fetchData, 1000);

    return () => clearInterval(intervalId);

  }, []);

  return (

    <div>

      <div className="nav-bar">

        <h2>Drone Tracker</h2>

      </div>

      <div className="content">

        <h3>Overall Drone Information</h3>

        <p><strong>Current Speed:</strong> {currentSpeed} m/s</p>

      </div>

      <div className="map-container">

        <LoadScript

          googleMapsApiKey="AIzaSyAjgPRiedbwBEiiLSw4DC2YFDqAHo8Kh4g"

          onLoad={() => setMapsLoaded(true)}

        >

          <GoogleMap

            mapContainerClassName="map"

            center={currentPosition}

            zoom={15}

          >

            {mapsLoaded && (

              <MarkerF

                position={currentPosition}

                icon={{

                  url: `data:image/svg+xml;charset=UTF8,

${encodeURIComponent(createCustomMarker(currentDroneId))}`,

                  scaledSize: new window.google.maps.Size(100, 100)

                }}

                onClick={() => {

                  setSelectedCenter(currentPosition);

                }}

              />

            )}

            {selectedCenter && (

              <InfoWindow

                position={currentPosition}

                onCloseClick={() => {

                  setSelectedCenter(null);

                }}

              >

                <div>

                  <h3>Drone Information</h3>

                  <p><strong>Speed:</strong> {currentSpeed} m/s</p>

                  <p><strong>Latitude:</strong> {currentPosition.lat}</p>

                  <p><strong>Longitude:</strong> {currentPosition.lng}</p>

                  <p><strong>ID:</strong> {currentDroneId}</p>

                </div>

              </InfoWindow>

            )}

          </GoogleMap>

        </LoadScript>

      </div>

    </div>

  );

};

export default DroneMap;