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

import { Canvas, useThree } from '@react-three/fiber';

import { OrbitControls, useGLTF } from '@react-three/drei';

import \* as THREE from 'three';  // Import THREE for vector operations

import Slider from 'react-slick';

import 'slick-carousel/slick/slick.css';

import 'slick-carousel/slick/slick-theme.css';

// Define GPS coordinates (latitude, longitude) for buildings

const buildingGPSCoordinates = {

  'B8': { lat: 31.781708309682333, lon: 76.99769895342719 }, // Replace with actual GPS coordinates

  'B12': { lat: 31.78133139951378, lon:  76.9982088531437 },

  'B19': { lat: 31.781258589264922,lon:  76.9972818147037  },

  'PINEMESS': { lat:31.780832969771087, lon: 76.99657709404012},

  'B12': { lat: 31.78133139951378, lon:  76.9982088531437 },

  'B12': { lat: 31.78133139951378, lon:  76.9982088531437 },

  'B12': { lat: 31.78133139951378, lon:  76.9982088531437 },

  'B12': { lat: 31.78133139951378, lon:  76.9982088531437 },

  'B12': { lat: 31.78133139951378, lon:  76.9982088531437 },

  // Add more buildings with their GPS coordinates here...

};

// Coordinates mapping for buildings

const building3DCoordinates = {

  'B8': [-10, 20, 45],

  'B12': [30, 20, 100],

  'B9': [30, 20, 40],

  'B11': [0, 35, 80],

  'B19': [-55, 25, 110],

  'B16': [-100, 25, 65],

  'B20': [-140, 25, 80],

  'B22': [-180, 25, 110],

  'B21': [-165, 25, 135],

  'B23': [-110, 25, 115],

  'B13': [-190, 25, 170],

  'PINEMESS': [-110, 35, 150],

  'B14': [-140, 25, 170],

  'B15': [-60, 25, 150],

  'B18': [-20, 25, 135],

  'B10': [65, 25, 75],

  'B17': [105, 25, 25],

  'B24': [150, 25, 5],

  'B26': [170, 30, 35],

  'B25': [130, 30, 60],

  'A19': [165, 30, 95],

  'A17': [105, 40, 110],

  'A18': [105, 50, 140],

  'A13': [-15, 35, 166],

  'AVL(GROUND-F\_A13)': [-15, 35, 166],

  'MNC-LAB(GROUND-F\_A13)': [-15, 35, 166],

  'A13-1A(GROUND-F\_A13)': [-15, 35, 166],

  'A13-2A(1ST-F\_A13)': [-15, 35, 166],

  'A13-2B(1ST-F\_A13)': [-15, 35, 166],

  'A13-2C(1ST-F\_A13)': [-15, 35, 166],

  'A13-2D(1ST-F\_A13)': [-15, 35, 166],

  'A13-3A(2ND-F\_A13)': [-15, 35, 166],

  'A13-L1(2ND-F\_A13)': [-15, 35, 166],

  'A13-L2(2ND-F\_A13)': [-30, 35, 206],

  'NKN\_CONFERENCE\_ROOM(3RD-F\_A13)': [-30, 35, 206],

  'A13-F1(3RD-F\_A13)': [-30, 35, 206],

  'A13-F2(3RD-F\_A13)': [-30, 35, 206],

  'A13-F7(3RD-F\_A13)': [-30, 35, 206],

  'A13-F4(3RD-F\_A13)': [-30, 35, 206],

  'A13-F3(3RD-F\_A13)': [-30, 35, 206],

  'A13-F5(3RD-F\_A13)': [-30, 35, 206],

  'A13-F6(3RD-F\_A13)': [-30, 35, 206],

  'A13-F9(3RD-F\_A13)': [-30, 35, 206],

  'A13-F11(3RD-F\_A13)': [-30, 35, 206],

  'A13-F12(3RD-F\_A13)': [-30, 35, 206],

  'A13-F13(3RD-F\_A13)': [-30, 35, 206],

  'A13-F14(3RD-F\_A13)': [-30, 35, 206],

  'A13-F15(3RD-F\_A13)': [-30, 35, 206],

  'SMSS\_CHAIRPERSON\_ROOM(3RD-F\_A13)': [-30, 35, 206],

  'SMSS\_\_OFFICE(3RD-F\_A13)': [-30, 35, 206],

  'A14': [-15, 35, 186],

  'BIOGEOCHEMISTRY\_LAB(1ST-F\_A14)': [-15, 35, 186],

  'DP\_LAB(1ST-F\_A14)': [-15, 35, 186],

  'TINKERING\_LAB(1ST-F\_A14)': [-15, 35, 186],

  'INNORVATION\_OFFICE(1ST-F\_A14)': [-15, 35, 186],

  'STEAM\_INNORVATION\_LAB(1ST-F\_A14)': [-15, 35, 186],

  'CAM\_LAB(GROUND-F\_A14)': [-15, 35, 186],

  'SHSS\_OFFICE(2ND-F\_A14)': [-15, 35, 186],

  'LANGUAGE\_LAB(2ND-F\_A14)': [-15, 35, 186],

  'CONFERENCE\_ROOM(2ND-F\_A14)': [-15, 35, 186],

  'HCI\_CENTER(3RD-F\_A14)': [-15, 35, 186],

  'MATERIAL\_SCIENCE\_LAB(3RD-F\_A14)': [-15, 35, 186],

  'QUANTUM\_TECH\_CENTRE(3RD-F\_A14)': [-15, 35, 186],

  'A11': [-130, 40, 240],

  'A10': [-150, 40, 215],

  'A9': [-220, 40, 265],

  'CENTRAL\_LIBRARY': [30, 35, 150], // Ensure correct key

  'TULSI\_MESS': [-95, 35, 193], // Ensure correct key

  'TRAGOPAN\_CANTEEN': [-95, 35, 193], // Ensure correct key

  'PEEPAL\_MESS': [195, 35, -20], // Ensure correct key

  'CHAAT\_JUNCTION\_CANTEEN ': [195, 35, -20], // Ensure correct key

  'OAK\_MESS': [65, 35, 40],

  'MONAL\_CANTEEN': [65, 35, 40],

  'ORIGIN': [0, 20, 0],

  'SPORTS\_COMPLEX': [-240, 20, 170],

  'HEALTH\_CENTRE': [-275, 20, 200],

  'GUEST\_HOUSE': [-255, 25, 100],

  'AUDITORIUM': [-295, 25, 150],

  'VILLAGE\_SQUARE': [-275, 10, 130],

  'ALDER\_MESS': [165, 30, 95],

  'KUKU\_CANTEEN': [165, 30, 95],

  'DRONGO\_CANTEEN': [-110, 35, 150],

  'CAFE O MOCHA': [-305, 15, 100],

  'SUPERMARKET': [-315, 10, 110],

  'ROBOTRONICS\_LAB(4TH-F-A18)': [105, 50, 140],

  'MANAS\_LAB(4TH-F-A18)': [105, 50, 140],

  'ACS\_LAB(4TH-F-A18)': [105, 50, 140],

  'A-18-2A(3RD-F-A18)': [105, 50, 140],

  'IKSHMA\_CLASSROOM(3RD-F-A18)': [105, 50, 140],

  'SCEE-INFO-LAB(3RD-F-A18)': [105, 50, 140],

  'SP\_COM\_LAB(2ND-F-A18)': [105, 50, 140],

  'VLSI\_LAB(2ND-F-A18)': [105, 50, 140],

  'A18-A1(1ST-F-A18)': [105, 50, 140],

  'SCEE\_CONF-ROOM(1ST-F-A18)': [105, 50, 140],

  'DATA\_SCIENCE\_LAB(1ST-F-A18)': [105, 50, 140],

  'CHEMISTRY\_LAB(1ST-F-A18)': [105, 50, 140],

  'SCEE\_ELECTRONIC\_LAB(GROUND\_F-A18)': [105, 50, 140],

  'A-17-1-A(GROUND-F-A17)': [105, 40, 110],

  'A-17-1-B(GROUND-F-A17)': [105, 40, 110],

  'A-17-1-D(GROUND-F-A17)': [105, 40, 110],

  'A-17-1-C(GROUND-F-A17)': [105, 40, 110],

  'A-17-1-E(GROUND-F-A17)': [105, 40, 110],

  'A-17-2-A(1ST-F-A17)': [105, 40, 110],

  'A-17-2-B(1ST-F-A17)': [105, 40, 110],

  'A-17-2-C(1ST-F-A17)': [105, 40, 110],

  'A-17-2-D(1ST-F-A17)': [105, 40, 110],

  'A-17-2-E(1ST-F-A17)': [105, 40, 110],

  'CSP\_LAB(2ND-F-A17)': [105, 40, 110],

  'SCEE\_OFFICE(2ND-F-A17)': [105, 40, 110],

  'SCEE\_CHAIRPERSON\_ROOM(2ND-F-A17)': [105, 40, 110],

  'MIC\_LAB(3RD-F-A17)': [105, 40, 110],

  'MIC\_LAB(3RD-F-A17)': [105, 40, 110],

  'PHOTONICS\_LAB(3RD-F-A17)': [105, 40, 110],

  'NSS(1ST\_F\_A19)': [165, 30, 95],

  'YANTRIK\_CLUB(1ST\_F\_A19)': [165, 30, 95],

  'ROBOTRONICS\_CLUB(1ST\_F\_A19)': [165, 30, 95],

  'E-CELL(1ST\_F\_A19)': [165, 30, 95],

  'STAC\_CLUB(1ST\_F\_A19)': [165, 30, 95],

  'KAMAND\_PROMPT\_CLUB(1ST\_F\_A19)': [165, 30, 95],

  'HNT\_CLUB(1ST\_F\_A19)': [165, 30, 95],

  'NIRMAAN\_CLUB(1ST\_F\_A19)': [165, 30, 95],

  'KAMAND\_BIO\_CLUB(1ST\_F\_A19)': [165, 30, 95],

  'TECHNICAL\_OFFICE(1ST\_F\_A19)': [165, 30, 95],

  'DESIGNAUTS\_CLUB(2ND\_F\_A19)': [165, 30, 95],

  'WRITING\_CLUB(2ND\_F\_A19)': [165, 30, 95],

  'ART\_GREEKS\_CLUB(2ND\_F\_A19)': [165, 30, 95],

  'DEBATING\_AND\_QUIZZING\_CLUB(2ND\_F\_A19)': [165, 30, 95],

  'GYMKHANA\_MEETING\_ROOM(2ND\_F\_A19)': [165, 30, 95],

  'PMC\_CLUB(2ND\_F\_A19)': [165, 30, 95],

  'MUSIC\_CLUB(2ND\_F\_A19)': [165, 30, 95],

  'SPICMACAY\_CLUB(2ND\_F\_A19)': [165, 30, 95],

  'DRAMA\_CLUB(3-F\_PEEPAL\_MESS)': [195, 35, -20],

  'DANCE\_CLUB(3-F\_PEEPAL\_MESS)': [195, 35, -20],

  'CULTURAL\_SOCIETY\_OFFICE(3-F\_PEEPAL\_MESS)': [195, 35, -20],

  'A-10\_1-A(GROUND-F\_A10)': [-150, 40, 215],

  'A-10\_1-B(GROUND-F\_A10)': [-150, 40, 215],

  'A-10\_1-C(GROUND-F\_A10)': [-150, 40, 215],

  'A-10\_1-D(GROUND-F\_A10)': [-150, 40, 215],

  'A-10\_202(COMPUTER\_LAB)(GROUND-F\_A10)': [-150, 40, 215],

  'COMMUNICATION-LAB(1ST-F\_A10)': [-150, 40, 215],

  'CONTROL\_SYSTEM-LAB(1ST-F\_A10)': [-150, 40, 215],

  'A-10\_2-A(1ST-F\_A10)': [-150, 40, 215],

  'A-10\_2-B(1ST-F\_A10)': [-150, 40, 215],

  'A-10\_2-C(1ST-F\_A10)': [-150, 40, 215],

  'GSC-ROOM(1ST-F\_A10)': [-150, 40, 215],

  'A-10\_3-A(2ND-F\_A10)': [-150, 40, 215],

  'A-10\_3-B(2ND-F\_A10)': [-150, 40, 215],

  'A-10\_3-C(2ND-F\_A10)': [-150, 40, 215],

  'SOM\_OFFICE(2ND-F\_A10)': [-150, 40, 215],

  'NKN\_CONFERENCE\_ROOM(2ND-F\_A10)': [-150, 40, 215],

  'FACULTY\_OFFICE(2ND-F\_A10)': [-150, 40, 215],

  '(2ND-F\_A10)': [-150, 40, 215],

  'FACULTY\_OFFICE(2ND-F\_A10)': [-150, 40, 215],

  'GEOTECHNICAL\_ENG\_LAB(2ND-F\_A11)': [-130, 40, 240],

  'CNC\_LAB(2ND-F\_A11)': [-130, 40, 240],

  'DESIGN\_LAB-1(2ND-F\_A11)': [-130, 40, 240],

  'THERMOFLUID\_LAB(2ND-F\_A11)': [-130, 40, 240],

  'A-11\_1-A(1ST-F\_A11)': [-130, 40, 240],

  'A-11\_1-B(1ST-F\_A11)': [-130, 40, 240],

  'CAIR\_LAB(1ST-F\_A11)': [-130, 40, 240],

  'DESIGN\_LAB-2(3RD-F\_A11)': [-130, 40, 240],

  'WATER\_RESIRE\_ENG\_LAB-2(3RD-F\_A11)': [-130, 40, 240],

  'EXPERIMENTAL\_THERMOFLUIDS\_LAB(3RD-F\_A11)': [-130, 40, 240],

  'A-11\_COMPUTER\_LAB(3RD-F\_A11)': [-130, 40, 240],

  'RHEOLOGY\_LAB(3RD-F\_A11)': [-130, 40, 240],

  'HYDROLOGY\_LAB(3RD-F\_A11)': [-130, 40, 240],

  'HYDROCLIMATOLOGY\_LAB(3RD-F\_A11)': [-130, 40, 240],

  'COMPOSITE\_DESIGN\_LAB(4TH-F\_A11)': [-130, 40, 240],

  'COMPUTATIONAL\_DESIGN\_LAB(4TH-F\_A11)': [-130, 40, 240],

  'SMEE\_MEETING\_ROOM(4TH-F\_A11)': [-130, 40, 240],

  'SCENE\_OFFICE(5TH-F\_A11)': [-130, 40, 240],

  'STUDENTS\_AFFAIR(GROUND-F\_A9)': [-220, 40, 265],

  'ACADEMICS\_SECTION(1ST-F\_A9)': [-220, 40, 265],

  'FACULTY\_AFFAIRS(1ST-F\_A9)': [-220, 40, 265],

  'REGISTRAR\_OFFICE(2ND-F\_A9)': [-220, 40, 265],

  'SATELLITE\_LIBRARY(3RD-F\_A9)': [-220, 40, 265],

  'FOUNTAIN\_AREA': [-330,15,200],

}

// Coordinate Marker Component

function CoordinateMarker({ position, color, isSelected }) {

  return (

    <mesh position={position}>

      <tetrahedronGeometry args={[isSelected ? 8 : 5, 4]} />

      <meshBasicMaterial color={isSelected ? 'red' : color} />

    </mesh>

  );

}

// Model Component

function Model() {

  const { scene } = useGLTF('s.glb');

  return <primitive object={scene} />;

}

// Camera Controls Component

function CameraControls({ targetPosition }) {

  const { camera } = useThree();

  const controls = useRef();

  useEffect(() => {

    camera.position.set(0, 150, -200);

    controls.current.update();

  }, [camera]);

  useEffect(() => {

    if (targetPosition) {

      controls.current.target.set(...targetPosition);

      const newCameraPosition = new THREE.Vector3(

        targetPosition[0] + 50,

        targetPosition[1] + 50,

        targetPosition[2] + 50

      );

      let frame = 0;

      const totalFrames = 60;

      const animateCamera = () => {

        frame++;

        const easedFrame = THREE.MathUtils.smoothstep(frame / totalFrames, 0, 1);

        camera.position.lerp(newCameraPosition, easedFrame);

        camera.lookAt(...targetPosition);

        if (frame < totalFrames) {

          requestAnimationFrame(animateCamera);

        } else {

          controls.current.update();

        }

      };

      animateCamera();

    }

  }, [targetPosition, camera]);

  return <OrbitControls ref={controls} />;

}

// Function to get the distance between two GPS coordinates (Haversine formula)

function getDistanceFromLatLonInKm(lat1, lon1, lat2, lon2) {

  const R = 6371; // Radius of the Earth in km

  const dLat = (lat2 - lat1) \* (Math.PI / 180);

  const dLon = (lon2 - lon1) \* (Math.PI / 180);

  const a =

    Math.sin(dLat / 2) \* Math.sin(dLat / 2) +

    Math.cos(lat1 \* (Math.PI / 180)) \* Math.cos(lat2 \* (Math.PI / 180)) \*

    Math.sin(dLon / 2) \* Math.sin(dLon / 2);

  const c = 2 \* Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));

  return R \* c; // Distance in km

}

// Function to find the nearest building based on user's GPS location

function findNearestBuilding(userLat, userLon) {

  let nearestBuilding = null;

  let minDistance = Infinity;

  Object.keys(buildingGPSCoordinates).forEach((building) => {

    const { lat, lon } = buildingGPSCoordinates[building];

    const distance = getDistanceFromLatLonInKm(userLat, userLon, lat, lon);

    if (distance < minDistance) {

      minDistance = distance;

      nearestBuilding = building;

    }

  });

  return nearestBuilding;

}

// Main Component

function ModelView() {

  const [fromBuilding, setFromBuilding] = useState('');

  const [toBuilding, setToBuilding] = useState('');

  const [fromPosition, setFromPosition] = useState(null);

  const [toPosition, setToPosition] = useState(null);

  const [targetPosition, setTargetPosition] = useState(null);

  const [selectedBuilding, setSelectedBuilding] = useState(null);

  const [livePosition, setLivePosition] = useState(null); // Live GPS location

  const [nearestBuilding, setNearestBuilding] = useState(null); // Nearest building based on GPS

  // Live location tracking

  useEffect(() => {

    if ("geolocation" in navigator) {

      navigator.geolocation.watchPosition(

        (position) => {

          const { latitude, longitude } = position.coords;

          const nearestBuilding = findNearestBuilding(latitude, longitude);

          // If the nearest building is detected

          if (nearestBuilding) {

            setNearestBuilding(nearestBuilding);

            setLivePosition(building3DCoordinates[nearestBuilding]);

            setTargetPosition(building3DCoordinates[nearestBuilding]);

          }

        },

        (error) => {

          console.error("Error getting live location: ", error);

        }

      );

    } else {

      console.log("Geolocation is not available in this browser.");

    }

  }, []);

  return (

    <div style={{ display: 'flex', height: '100vh' }}>

      <Canvas style={{ flex: 1 }}>

        <ambientLight intensity={0.5} />

        <pointLight position={[10, 10, 10]} />

        <Model />

        <CameraControls targetPosition={targetPosition} />

        {/\* Render the live location marker \*/}

        {livePosition && (

          <CoordinateMarker

            position={livePosition}

            color="orange"

            isSelected

          />

        )}

      </Canvas>

      <div style={{ width: '400px', padding: '20px' }}>

        <h3>Building Navigation</h3>

        <div>

          <h4>Live Location Nearest Building: {nearestBuilding}</h4>

        </div>

      </div>

    </div>

  );

}

// Main Application Component

function App() {

  return (

    <div style={{ height: '100vh' }}>

      <ModelView />

    </div>

  );

}

export default App;