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

import {

  View,

  Text,

  StyleSheet,

  Image,

  TouchableOpacity,

  Alert,

  ActivityIndicator,

  SafeAreaView, // Importa SafeAreaView

} from "react-native";

import MapView, { Marker, PROVIDER\_GOOGLE } from "react-native-maps";

import \* as Location from "expo-location";

import { eventsService } from "../services/api";

import { CONFIG } from "../constants/config";

export default function EventsMapScreen({ navigation }) {

  const [region, setRegion] = useState(CONFIG.MAP\_CONFIG.INITIAL\_REGION);

  const [events, setEvents] = useState([]);

  const [loading, setLoading] = useState(true);

  const [userLocation, setUserLocation] = useState(null);

  const [showMyReports, setShowMyReports] = useState(false);

  useEffect(() => {

    requestLocationPermission();

    loadEvents();

  }, []);

  const requestLocationPermission = async () => {

    try {

      const { status } = await Location.requestForegroundPermissionsAsync();

      if (status === "granted") {

        const location = await Location.getCurrentPositionAsync({});

        setUserLocation({

          latitude: location.coords.latitude,

          longitude: location.coords.longitude,

        });

        if (isInCalabria(location.coords.latitude, location.coords.longitude)) {

          setRegion({

            latitude: location.coords.latitude,

            longitude: location.coords.longitude,

            latitudeDelta: 0.1,

            longitudeDelta: 0.1,

          });

        }

      }

    } catch (error) {

      console.error("Errore geolocalizzazione:", error);

    }

  };

  const isInCalabria = (latitude, longitude) => {

    return (

      latitude >= 37.9 &&

      latitude <= 40.2 &&

      longitude >= 15.6 &&

      longitude <= 17.2

    );

  };

  const loadEvents = async () => {

    setLoading(true);

    try {

      // const response = await eventsService.findEvents({ status: "open", isPublic: true });

      const mockEvents = [

        {

          id: "1",

          title: "Allagamento Strada Provinciale",

          description: "Segnalazione di allagamento sulla SP 123",

          latitude: 39.3081,

          longitude: 16.2539,

          type: "flood",

          status: "open",

          createdAt: new Date().toISOString(),

        },

        {

          id: "2",

          title: "Frana Località Monte Alto",

          description: "Frana che blocca la viabilità",

          latitude: 39.2081,

          longitude: 16.3539,

          type: "landslide",

          status: "open",

          createdAt: new Date().toISOString(),

        },

        {

          id: "3",

          title: "Incendio Boschivo",

          description: "Principio di incendio in zona boschiva",

          latitude: 39.4081,

          longitude: 16.1539,

          type: "fire",

          status: "open",

          createdAt: new Date().toISOString(),

        },

      ];

      setEvents(mockEvents);

    } catch (error) {

      console.error("Errore caricamento Eventi:", error);

      Alert.alert("Errore", "Impossibile caricare gli Eventi");

    } finally {

      setLoading(false);

    }

  };

  const getMarkerColor = (eventType) => {

    switch (eventType) {

      case "flood":

        return "#000000ff";

      case "landslide":

        return "#FF9500";

      case "fire":

        return "#FF3B30";

      default:

        return "#FF6B35";

    }

  };

  const handleMarkerPress = (event) => {

    Alert.alert(event.title, event.description, [

      { text: "Chiudi", style: "cancel" },

      { text: "Dettagli", onPress: () => showEventDetails(event) },

    ]);

  };

  const showEventDetails = (event) => {

    console.log("Mostra dettagli evento:", event);

  };

  const handleSegnala = () => {

    navigation.navigate("Segnala");

  };

  return (

    <SafeAreaView style={styles.container}>

      {/\* Header Section \*/}

      <View style={styles.header2}>

        <Text style={styles.headerTitle2}>

          Protezione Civile | Regione Calabria

        </Text>

        <Image

          source={require("../components/Logo.png")}

          style={styles.reportButtonImage}

        />

      </View>

      <View style={styles.header}>

        <Text style={styles.headerTitle}>ProCiv Calabria</Text>

        <TouchableOpacity

          onPress={handleSegnala}

          activeOpacity={1}

          style={styles.reportButton}

        >

          <Text style={styles.reportButtonText}>Segnala</Text>

        </TouchableOpacity>

      </View>

      <View style={styles.tabContainer}>

        <TouchableOpacity

          style={styles.tab}

          onPress={() => navigation.navigate("Eventi")}

        >

          <Text style={[styles.tabText, styles.activeTabText]}>Mappa</Text>

        </TouchableOpacity>

        <TouchableOpacity

          style={styles.tab}

          onPress={() => navigation.navigate("EventsList")}

        >

          <Text style={styles.tabText}>Lista</Text>

        </TouchableOpacity>

      </View>

      <MapView

        provider={PROVIDER\_GOOGLE}

        style={styles.map}

        region={region}

        onRegionChangeComplete={setRegion}

        showsUserLocation={true}

        showsMyLocationButton={true}

      >

        {events.map((event) => (

          <Marker

            key={event.id}

            coordinate={{

              latitude: event.latitude,

              longitude: event.longitude,

            }}

            title={event.title}

            description={event.description}

            pinColor={getMarkerColor(event.type)}

            onPress={() => handleMarkerPress(event)}

          />

        ))}

      </MapView>

      {loading && (

        <View style={styles.loadingOverlay}>

          <ActivityIndicator size="large" color="#FF6B35" />

          <Text style={styles.loadingText}>Caricamento Eventi...</Text>

        </View>

      )}

    </SafeAreaView>

  );

}

const styles = StyleSheet.create({

  container: {

    flex: 1,

    marginTop: 20,

  },

  header: {

    flexDirection: "row",

    justifyContent: "space-between",

    alignItems: "center",

    backgroundColor: "#0091D6",

    paddingHorizontal: 20,

    paddingVertical: 10,

  },

  header2: {

    flexDirection: "row",

    justifyContent: "space-between",

    alignItems: "center",

    backgroundColor: "#0091D6",

    paddingHorizontal: 20,

    paddingVertical: 10,

  },

  headerTitle2: {

    fontSize: 16,

    color: "white",

  },

  headerTitle: {

    fontSize: 20,

    color: "white",

    fontWeight: "bold",

  },

  reportButton: {

    backgroundColor: "#FF6B35",

    paddingVertical: 8,

    paddingHorizontal: 16,

    borderRadius: 5,

    borderColor: "white",

    borderWidth: 3,

  },

  reportButtonText: {

    color: "#FFFF",

    fontSize: 14,

  },

  reportButtonImage: {

    width: 50, // Imposta la larghezza dell'immagine

    height: 50, // Imposta l'altezza dell'immagine

    resizeMode: "contain", // Aggiusta l'immagine per non deformarla

  },

  tabContainer: {

    flexDirection: "row",

    borderBottomWidth: 1,

    borderBottomColor: "#ddd",

  },

  tab: {

    flex: 1,

    flexDirection: "row",

    justifyContent: "center",

    alignItems: "center",

    paddingVertical: 10,

  },

  tabText: {

    marginLeft: 8,

    color: "#666",

  },

  activeTabText: {

    color: "blue",

    fontWeight: "bold",

  },

  map: {

    flex: 1,

  },

  loadingOverlay: {

    position: "absolute",

    top: 0,

    left: 0,

    right: 0,

    bottom: 0,

    backgroundColor: "rgba(255, 255, 255, 0.8)",

    justifyContent: "center",

    alignItems: "center",

  },

  loadingText: {

    marginTop: 10,

    fontSize: 16,

    color: "#333",

  },

});