**Troisième Partie : Maps et Géolocalisation**

1. **Maps**

Il est nécessaire d'installer cette dépendance.

**npm install react-native-webview**

Nous ajoutons un nouvel écran, nommé **MapScreen.js**

Nous devons modifier le **HomeStack** dans le **TabNavigator**.

Ajouter cet import :

import MapScreen from '../screens/MapScreen';

**const HomeStack = () => (**

**<Stack.Navigator>**

**<Stack.Screen name="Home" component={HomeScreen} options={{**

**headerShown: false**

**}} />**

**<Stack.Screen**

**name="Add Provider"**

**component={addProviderScreen}**

**options={({route}) => ({**

**title: route.params?.title,**

**headerStyle: {**

**backgroundColor: '#DFF8F7',**

**}**

**})}**

**/>**

**<Stack.Screen**

**name="Maps"**

**component={MapScreen}**

**options={({route}) => ({**

**title: route.params?.title,**

**headerStyle: {**

**backgroundColor: '#DFF8F7',**

**}**

**})}**

**/>**

**</Stack.Navigator>**

**);**

-On doit ajouter un lien vers MapsScreen dans homeScreen

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

import {

StyleSheet,

Text,

View,

Image,

FlatList,TouchableOpacity

} from 'react-native'

import axios from "axios";

import asyncStorage from "@react-native-async-storage/async-storage/src/AsyncStorage";

import Icon from 'react-native-vector-icons/FontAwesome'; // Exemple d'utilisation de FontAwesome

import { useNavigation } from '@react-navigation/native'

import \* as Device from 'expo-device';

import \* as Notifications from 'expo-notifications';

const HomeScreen = () => {

const [providers, setProviders] = useState([]);

const [ref,setRef] = useState("")

const navigation=useNavigation();

const fetchProviders = async () => {

const u= await asyncStorage.getItem("token");

axios.defaults.headers['Authorization'] = 'Bearer ' + u;

const a = await asyncStorage.getItem("id");

const res = await axios

.get("https://ams.smart-it-partner.com/api/providers")

.then(response=>response.data["hydra:member"])

setProviders(res);

console.log(res)

}

useEffect(() => {

fetchProviders();

}, []);

const addProvider = () => {

navigation.navigate('Add Provider');

};

const deleteProvider = async (idprovider) => {

const u= await asyncStorage.getItem("token");

axios.defaults.headers['Authorization'] = 'Bearer ' + u;

await axios.delete( "https://ams.smart-it-partner.com/api/providers/"+idprovider)

.then( response => {

response.data;

})

navigation.reset({

index: 0,

routes: [{ name: 'Home', params: { refresh: true } }],

});

}

async function registerForPushNotificationsAsync() {

let token;

if (Device.isDevice) {

const { status: existingStatus } = await Notifications.getPermissionsAsync();

let finalStatus = existingStatus;

if (existingStatus !== 'granted') {

const { status } = await Notifications.requestPermissionsAsync();

finalStatus = status;

}

if (finalStatus !== 'granted') {

alert('Failed to get push token for push notification!');

return;

}

token = (await Notifications.getExpoPushTokenAsync()).data;

setRef(token)

const id = await asyncStorage.getItem("id");

axios.post(

"https://ams.smart-it-partner.com/api/push\_notives",

{ token: token, user: id },

{ headers: { 'Content-Type': 'application/ld+json' } }

);

} else {

alert('Must use physical device for Push Notifications');

}

if (Platform.OS === 'android') {

Notifications.setNotificationChannelAsync('default', {

name: 'default',

importance: Notifications.AndroidImportance.MAX,

vibrationPattern: [0, 250, 250, 250],

lightColor: '#FF231F7C',

});

}

return token;

}

useEffect(() => {

registerForPushNotificationsAsync();

Notifications.setNotificationHandler({

handleNotification: async () => ({

shouldShowAlert: true,

shouldSetBadge: true,

}),

});

}, []);

return (

<View style={{ flex: 1 }}>

<TouchableOpacity style={styles.addButton} onPress={addProvider}>

<Icon name="plus" size={20} color="white" />

<Text style={styles.addButtonText}>Add Provider</Text>

</TouchableOpacity>

<FlatList

enableEmptySections={true}

data={providers}

keyExtractor={item => item.id}

renderItem={({ item }) => {

return (

<View style={styles.box}>

<Image style={styles.image} source={{ uri:'https://ams.smart-it-partner.com/uploads/provider/'+ item.photo }} />

<View style={styles.boxContent}>

<Text style={styles.title}>{item.name}</Text>

<Text style={styles.description}>{item.adress}</Text>

<Text style={styles.description}>{item.email}</Text>

<View style={styles.buttons}>

<TouchableOpacity

style={[styles.button, styles.view]}

onPress={() => deleteProvider(item.id)}

>

<Image

style={styles.icon}

source={{ uri: 'https://cdn-icons-png.flaticon.com/512/6861/6861362.png' }}

/>

</TouchableOpacity>

<TouchableOpacity

style={[styles.button, styles.view]}

onPress={() => navigation.navigate('Maps', {

id: item.id,

})}

>

<Image

style={styles.icon}

source={{ uri: 'https://upload.wikimedia.org/wikipedia/commons/thumb/3/39/Google\_Maps\_icon\_%282015-2020%29.svg/512px-Google\_Maps\_icon\_%282015-2020%29.svg.png' }}

/>

</TouchableOpacity>

</View>

</View>

</View>

)

}}

/></View>

)

}

const styles = StyleSheet.create({

image: {

width: 100,

height: 100,

},

box: {

padding: 20,

marginTop: 5,

marginBottom: 5,

backgroundColor: 'white',

flexDirection: 'row',

},

boxContent: {

flex: 1,

flexDirection: 'column',

alignItems: 'flex-start',

marginLeft: 10,

},

title: {

fontSize: 18,

color: '#151515',

},

description: {

fontSize: 15,

color: '#646464',

},

view: {

backgroundColor: '#eee',

},

addButton: {

backgroundColor: '#07B0A8',

paddingHorizontal: 20,

paddingVertical: 10,

flexDirection: 'row',

alignItems: 'center',

justifyContent: 'center',

borderRadius: 5,

marginTop: 10, // Marge en haut

marginLeft: 200, // Marge à gauche

marginRight:10

},

addButtonText: {

color: 'white',

marginLeft: 10,

}, imageContainer: {

alignItems: 'center',marginLeft:30,

marginVertical: 10,width:'80%'

},

buttons: {

flexDirection: 'row',

},

button: {

height: 35,

flexDirection: 'row',

justifyContent: 'center',

alignItems: 'center',

borderRadius: 10,

width: 50,

marginRight: 5,

marginTop: 5,

},

icon: {

width: 20,

height: 20,

},

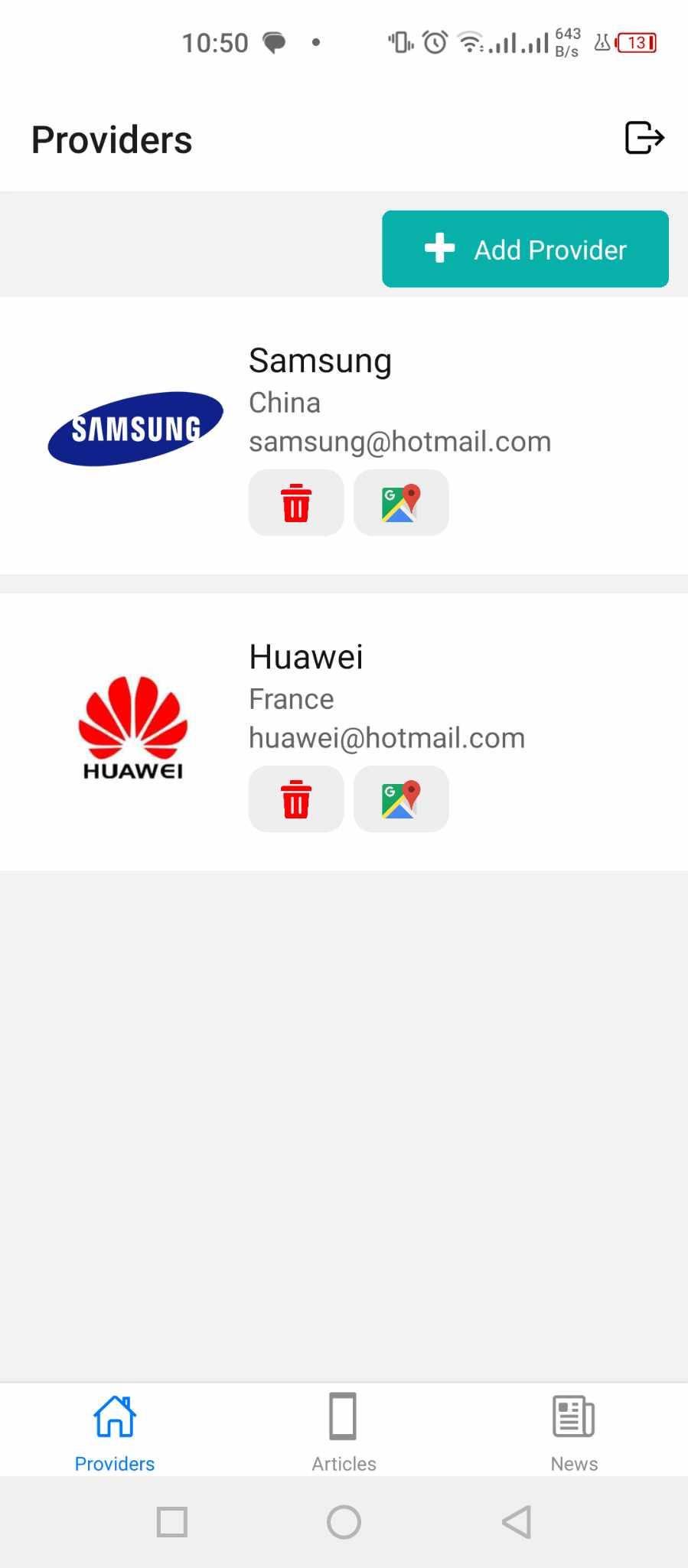
view: {

backgroundColor: '#eee',

},

})

export default HomeScreen;



**Pour l’accès à mes paramètres de localisation il est nécessaire d'installer les deux dépendances**

npm install expo-permissions

npm install expo-location

**MapScreen.js**

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

import { View, Text, Image, Modal, StyleSheet, TouchableOpacity,Linking } from 'react-native';

import { WebView } from 'react-native-webview';

import { useRoute, useNavigation } from '@react-navigation/native';

import asyncStorage from "@react-native-async-storage/async-storage/src/AsyncStorage";

import axios from "axios";

import \* as Permissions from 'expo-permissions';

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

const MapScreen = () => {

const route = useRoute();

const [nom, setNom] = useState('');

const [email, setEmail] = useState('');

const [latitude, setLatitude] = useState('');

const [longitude, setLongitude ]= useState('');

const [image, setImage] = useState('');

const idprovider=route.params?.id;

const getDataProvider= async () => {

const u= await asyncStorage.getItem("token");

axios.defaults.headers['Authorization'] = 'Bearer ' + u;

const responses = await axios

.get("https://ams.smart-it-partner.com/api/providers/"+idprovider)

.then(response=>response.data)

setNom(responses.name)

setEmail(responses.email)

setLatitude(parseFloat(responses.latitude));

setLongitude(parseFloat(responses.longitude));

setImage(responses.photo)

console.log(responses.latitude)

}

const description = email;

const imageUrl = 'https://ams.smart-it-partner.com/uploads/provider/'+image;

useEffect( () => {

getDataProvider();

}, []);

const openGoogleMapsDirections = async (destinationLat, destinationLon) => {

const { status } = await Permissions.askAsync(Permissions.LOCATION);

if (status !== 'granted') {

console.error('Permission to access location was denied');

return;

}

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

const originLat = location.coords.latitude;

const originLon = location.coords.longitude;

const url = `https://www.google.com/maps/dir/?api=1&origin=${originLat},${originLon}&destination=${destinationLat},${destinationLon}`;

Linking.openURL(url).catch(err => console.error("Couldn't open Google Maps", err));

};

const mapHTML = `

<!DOCTYPE html>

<html>

<head>

<title>Simple Map with Marker, Popup and Image</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>

<style>

#map { height: 100vh; width: 100%; }

.leaflet-popup-content { width: auto !important; }

</style>

</head>

<body>

<div id="map"></div>

<script>

var map = L.map('map').setView([${latitude}, ${longitude}], 13);

L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {

maxZoom: 19,

attribution: '© OpenStreetMap contributors'

}).addTo(map);

var marker = L.marker([${latitude}, ${longitude}]).addTo(map);

marker.bindPopup('<div><b>${nom}</b><br><img src="${imageUrl}" alt="image" style="width:100px;"><br>${description}</div>').openPopup();

</script>

</body>

</html>

`;

return (

<View style={styles.container}>

<WebView

originWhitelist={['\*']}

source={{ html: mapHTML }}

style={{ flex: 1 }}

/>

<TouchableOpacity

style={styles.button}

onPress={() => openGoogleMapsDirections(latitude, longitude)}>

<Text style={styles.buttonText}>Voir l'itinéraire</Text>

</TouchableOpacity>

</View>

);

};

const styles = StyleSheet.create({

container: {

flex: 1,

justifyContent: 'flex-end',

},

button: {

backgroundColor: '#007bff',

padding: 15,

margin: 20,

borderRadius: 5,

alignItems: 'center',

},

buttonText: {

color: 'white',

fontSize: 16,

fontWeight: 'bold',

},

});

export default MapScreen;



**Faites le builds**



***Bon travail !***