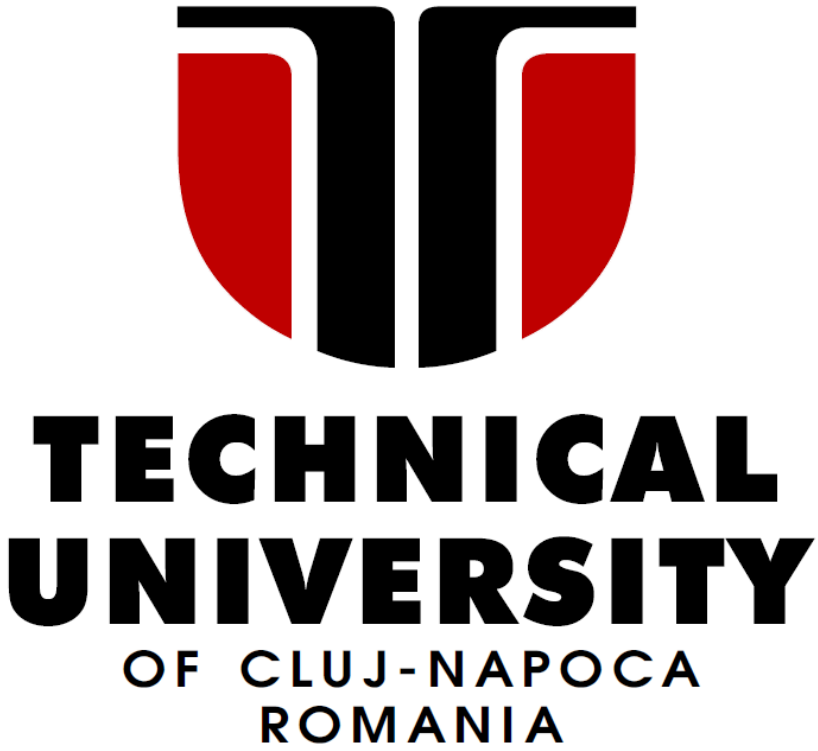
Location aware mobile application

NowWhat Mobile App



Student: Pop Robert Daniel

Group: 2341

Technical University of Cluj-Napoca

1. **What is a location aware mobile application?**

Is a mobile application that is using your GPS signal to locate you and show you where your phone location is. Mobile location came to be very accurate and can also track your current movements or determine how much time it would take you to go from point A to point B.

2. Why are we using them?

Locations based application are often used in real life. Some examples can be: tracking your package, seeing your friends locations or even your child. You can see your phones location if it have been stolen, when it connects to any network.

3. How do they work?

Most application are using **geolocation.**

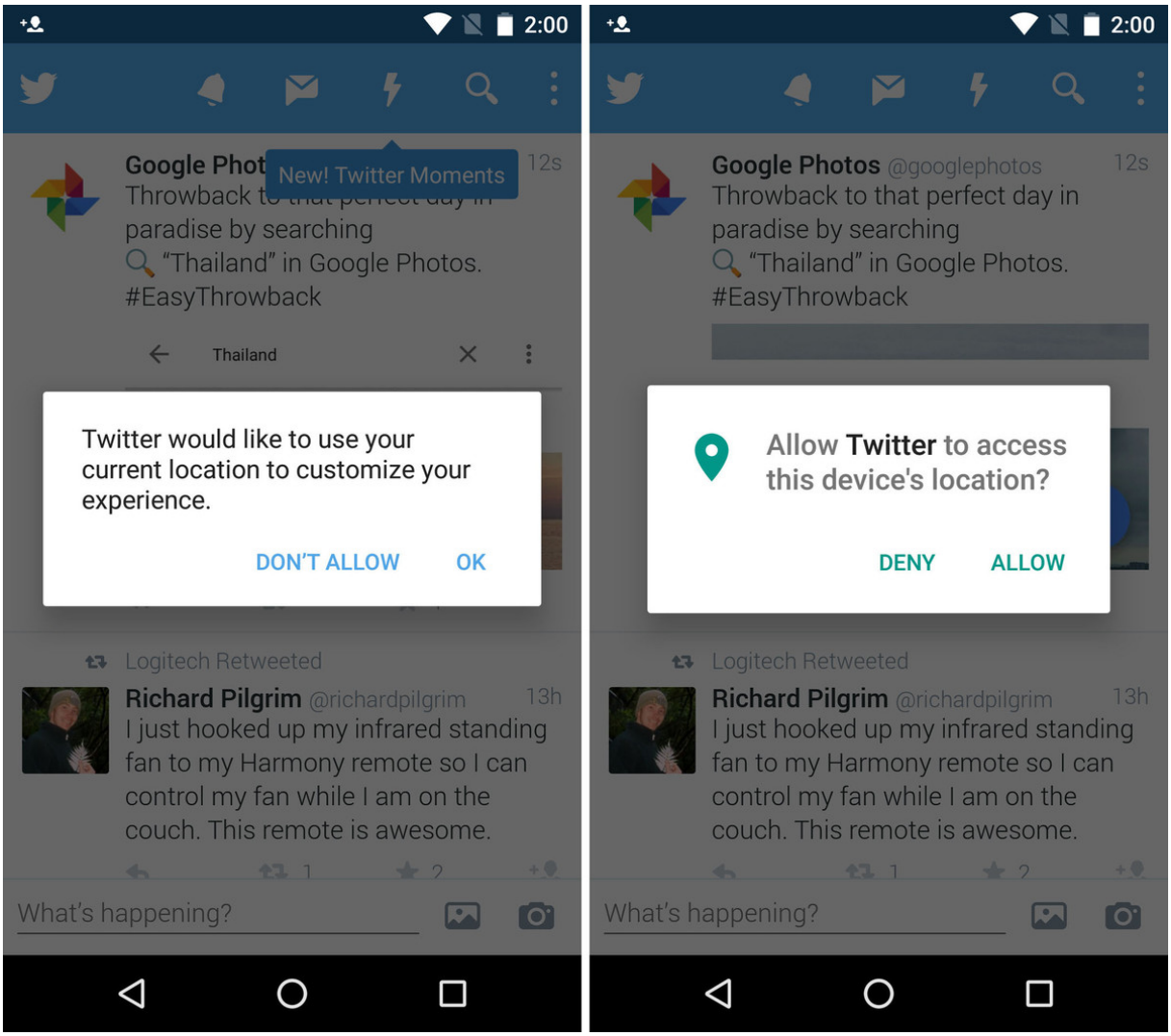
Geolocation is the identification or estimation of the real-world geographic location of an object, such as a radar source, mobile phone, or Internet-connected computer terminal.
For either geolocating or positioning, the locating engine often uses radio frequency (RF) location methods, for example Time Difference Of Arrival (TDOA) for precision. TDOA systems often use mapping displays or other geographic information system. When a GPS signal is unavailable, geolocation applications can use information from cell towers to triangulate the approximate position, a method that is not as accurate as GPS but has greatly improved in recent years. This is in contrast to earlier radiolocation technologies, for example Direction Finding where a line of bearing to a transmitter is achieved as part of the process.

Internet and computer geolocation can be performed by associating a geographic location with the Internet Protocol (IP) address, MAC address, RFID, hardware embedded article/production number, embedded software number (such as UUID, Exif/IPTC/XMP or modern steganography), invoice, Wi-Fi positioning system, device fingerprint, canvas fingerprinting or device GPS coordinates, or other, perhaps self-disclosed information. Geolocation usually works by automatically looking up an IP address on a WHOIS service and retrieving the registrant's physical address.

IP address location data can include information such as country, region, city, postal/zip code, latitude, longitude and time zone. Deeper data sets can determine other parameters such as domain name, connection speed, ISP, language, proxies, company name, US DMA/MSA, NAICS codes, and home/business.

Anyway any application before it uses the users geolocation it must first ask for its permission, after that you can show the user his location or even track his current location.

Example of a permission:



Implementation of NowWhat

Abstract

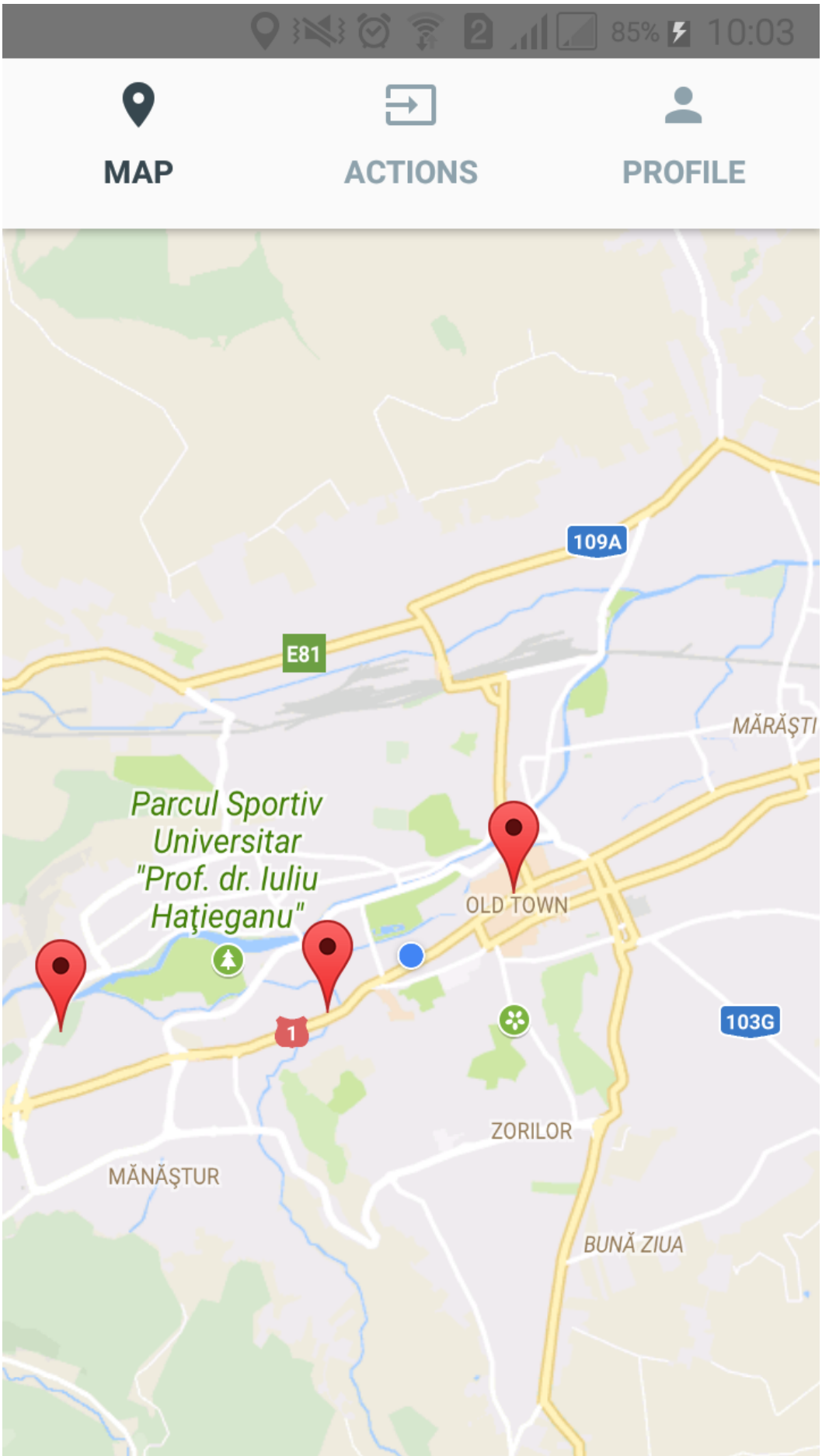
NowWhat shows you the opportunities that you have all around you without even knowing you have them. Now you can finally leave your cozy couch and participate to social events/party or what your preferences are. Instead of “scrolling” your life away you could engage into more activities.

1. Technologies used

For this project I chose React-Native which is a Javascript Library. With React Native, you don't build a "mobile web app", an "HTML5 app", or a "hybrid app". You build a real mobile app that's indistinguishable from an app built using Objective-C or Java. React Native uses the same fundamental UI building blocks as regular iOS and Android apps. You just put those building blocks together using JavaScript and React.
On top of it I used react-native-chunky, which is built upon react-native. Chunky includes all other libraries used for this project.
For the Database and login I used Firebase.

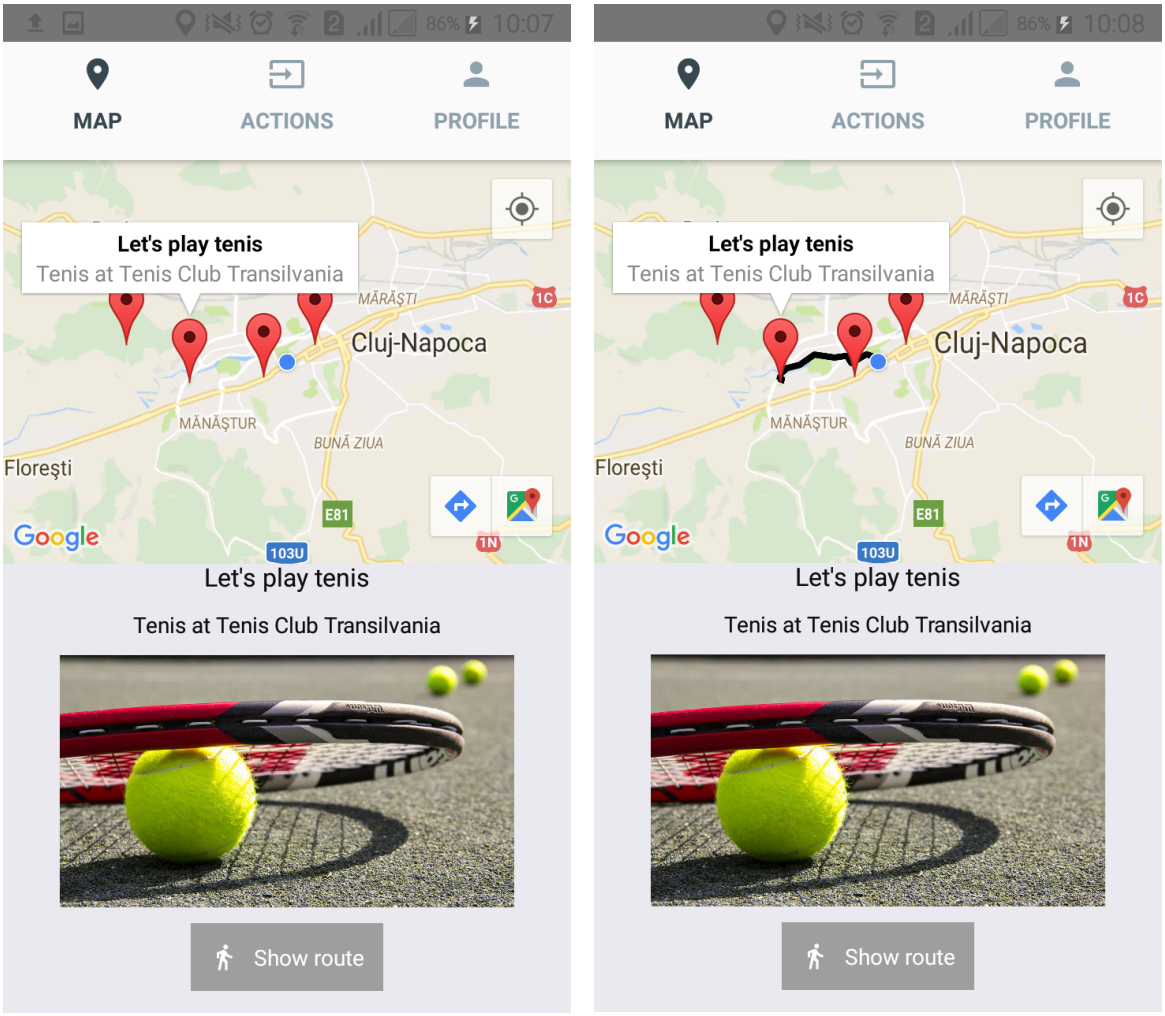
2. Layout

I wanted that the first thing for the users to see when they open up the app to be a map of events or places near them. So my layout would be 3 tabs representing: Map, Actions and Profile.


85% 10:03
MAP
ACTIONS
PROFILE
109A
E81
MĂRĂȘTI
Parcul Sportiv Universitar "Prof. dr. Iuliu Haţieganu"
OLD TOWN
103G
1
ZORILOR
MĂNĂȘTUR
BUNĂ ZIUA

3. How the app works

On the Map tab you see a map of your current location and pins showing the events or places near you. If you tap a pin you can see additional details of the selected pin. After you can press “Show route” button and the app will display a walking route from your location to the destination.

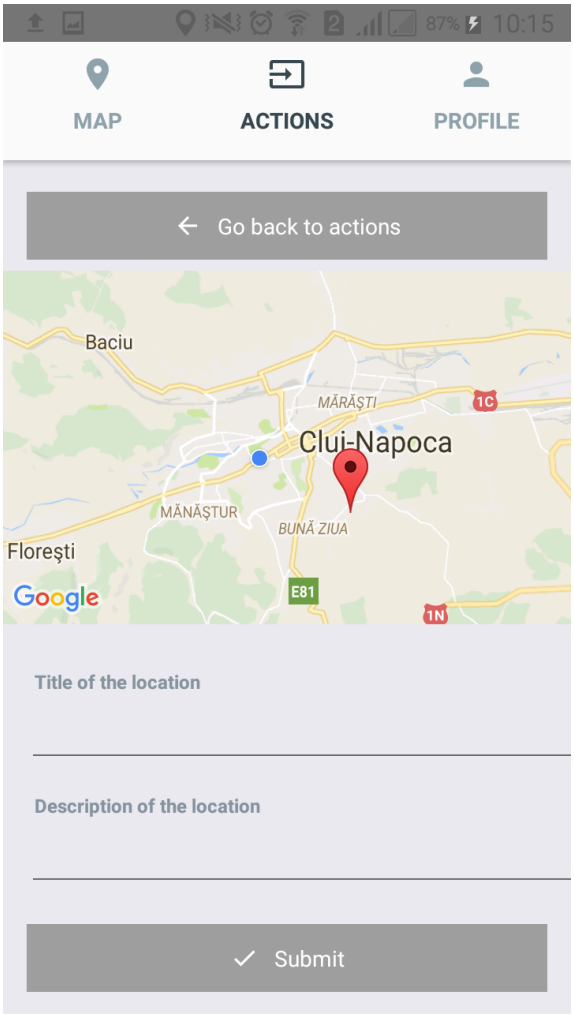


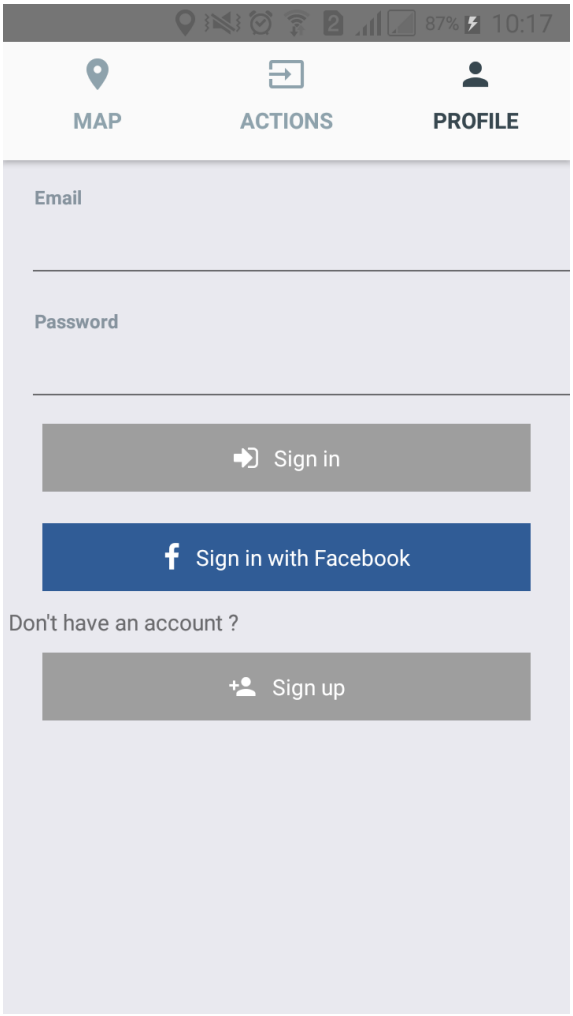
From the “Actions” tab you can add your own place, just press “Add new location” and a map will appear. When you tap on the map 2 inputs will show with “Title” and “Description” of your new location. After you submit it your new location will be available to be seen in the Map tab.

On “Profile” tab you can log into the app and change the filters. Or sign up if you don’t have an account.

All the locations are taken from Firebase Database and when you add a new location you saved it to the database.

For the route I used Google Directions API.





4. Configuration

Chunky had already react-native-maps included so I just needed to configure with my API KEY from Google and insert into my android project and xcode project for IOS.

5. Implementation

Main map screen:

```
import React from 'react'
import {
  StyleSheet,
  View,
  Text,
  Dimensions,
  ActivityIndicator,
  Image
} from 'react-native'
import { Screen } from 'react-native-chunky'

import { Button } from 'react-native-elements'

import firebase from '../../configs/firebase'

import MapView from 'react-native-maps'

import { decode } from '../../utils'

const { width, height } = Dimensions.get('window');

const ASPECT_RATIO = width / height;
const LATITUDE = 37.78825;
const LONGITUDE = -122.4324;
const LATITUDE_DELTA = 0.0922;
const LONGITUDE_DELTA = LATITUDE_DELTA * ASPECT_RATIO;
let markers = []

export default class MainIntroScreen extends Screen {

  constructor(props) {
    super(props)
    this.state = {
      latitude: null,
      longitude: null,
      error: null,
      isLoading: true,
      showDetails: false,
      mapContainerHeight: height
    }
  }

  componentDidMount() {
    super.componentDidMount()
    navigator.geolocation.getCurrentPosition(
      (position) => {
        this.setState({
          latitude: position.coords.latitude,
          longitude: position.coords.longitude,
          error: null
        })
        this.renderMarkers()
      },
      (error) => this.setState({ error: error.message }),
```

```
},
      (error) => this.setState({ error: error.message }),
      { enableHighAccuracy: false, timeout: 20000, maximumAge: 1000 },
    )
  }

  componentWillUnmount() {
    super.componentWillMount()
    navigator.geolocation.clearWatch(this.watchId)
  }

  renderDataError() {
    return this.renderData()
  }

  renderDataDefaults() {
    return this.renderData()
  }

  renderMarkers = () => {
    const self = this
    let data
    firebase.database().ref('/places/').on('value', function (snapshot) {
      data = snapshot.val()
      for (let key in data) {
        markers.push(data[ key ])
      }
      self.setState({
        isLoading: false
      })
    })
  }

  handlePressMarker = (marker) => {
    navigator.geolocation.clearWatch(this.watchId)
    const { title, description, picture, latlng } = marker
    this.setState({
      showDetails: true,
      title: title,
      description: description,
      pictureURL: picture,
      mapContainerHeight: 0.4 * height,
      destination: latlng,
      coords: ''
    })
  }
```

```
showRoute = () => {
    const mode = 'walking'
    const origin = `${this.state.latitude}, ${this.state.longitude}`
    const destination = `${this.state.destination.latitude},
${this.state.destination.longitude}`
    const APIKEY = 'AIzaSyCq1jTnrKJoFSK686je2-marmctSOd29cE'
    const url =
`https://maps.googleapis.com/maps/api/directions/json?origin=${origin}&destination=${destinatio
n}&key=${APIKEY}&mode=${mode}`

    fetch(url)
      .then(response => response.json())
      .then(responseJson => {
        if (responseJson.routes.length) {
          this.setState({
            coords: decode(responseJson.routes[ 0 ].overview_polyline.points) // definition
below
          })
          this.watchId = navigator.geolocation.watchPosition(
            (position) => {
              this.setState({
                latitude: position.coords.latitude,
                longitude: position.coords.longitude,
                error: null,
              })
            },
            (error) => this.setState({ error: error.message }),
            { enableHighAccuracy: true, timeout: 20000, maximumAge: 1000, distanceFilter: 10 },
          )
        }
      }).catch(e => { console.warn(e) });
  }

  render() {
    return (
      <View>
        { this.renderContent() }
      </View>
    )
  }

  renderContent() {
    if (this.state.isLoading) {
      return (
        <View style={{ flex: 1, paddingTop: 20 }}>
          <ActivityIndicator />
        </View>
      )
    }
    const { title, description, pictureURL, mapContainerHeight } = this.state
    const origin = {
      latitude: this.state.latitude !== null ? this.state.latitude : 46.769350,
      longitude: this.state.longitude !== null ? this.state.longitude : 23.589462
    }
```

```
const destination = this.state.destination ? this.state.destination : ''
    return (
      <View style={styles.container}>
        <View style={[styles.mapContainer, {height: mapContainerHeight}]}>
          <MapView
            style={styles.map}
            region={{
              latitude: this.state.latitude !== null ? this.state.latitude : 46.769350,
              longitude: this.state.longitude !== null ? this.state.longitude : 23.589462,
              latitudeDelta: LATITUDE_DELTA,
              longitudeDelta: LONGITUDE_DELTA
            }}
            showsUserLocation={true}
            onPress={() => this.setState({mapContainerHeight: height, showDetails: false})}
          >
          {
            this.state.coords ?
              <MapView.Polyline
                coordinates={this.state.coords}
                strokeWidth={4}
              />
            :
              null
          }
            {markers.map(marker => (
              <MapView.Marker
                coordinate={marker.latlng}
                title={marker.title}
                description={marker.description}
                onPress={() => this.handlePressMarker(marker)}
              />
            ))}
          </MapView>
        </View>
        {
          this.state.showDetails ?
            <View style={styles.detailsContainer}>
              <Text style={[styles.detailsText, {fontSize: 16}]}>{title}</Text>
              <Text style={[styles.detailsText, {fontSize: 14}]}>{description}</Text>
              <Image
                style={{ width: 0.8 * width, height: 0.25 * height, marginTop: 10,
marginBottom: 10 }}
                source={{ uri: pictureURL }}
              />
              <Button
                raised
                icon={{ name: 'directions-walk' }}
                title='Show route'
                onPress={this.showRoute} />
            </View>
          :
            null
        }
      </View>
    )
  }
```

```
const styles = StyleSheet.create({
  container: {
    flex: 1
  },
  mapContainer: {
    ...StyleSheet.absoluteFillObject,
    width: width,
    justifyContent: 'flex-end',
    alignItems: 'center'
  },
  map: {
    ...StyleSheet.absoluteFillObject
  },
  detailsContainer: {
    justifyContent: 'center',
    alignItems: 'center',
    marginTop: 0.32 * height + 20,
    height: 0.5 * height
  },
  detailsText: {
    color: '#0c0c0c',
    marginTop: 10,
  }
})
```

5. Conclusion

Location has become a part of our everyday life, we use it on a daily basis. For example when we want to eat out, we check the place and how to get there. Another example would be when you plan a vacation you check all the possible routes and how you can get there. Location evolved to the point that it can show any network connected terminal on the map. Nowadays, we can even use this function to find different items such as laptops/cars/phones.

6. Bibliography

https://facebook.github.io/react-native/

https://github.com/react-community/react-native-maps

http://www.chunky.io/

https://stackoverflow.com/

https://firebase.google.com/docs/

https://hackernoon.com/react-native-basics-geolocation-adf3c0d10112