Mobile Computing Report

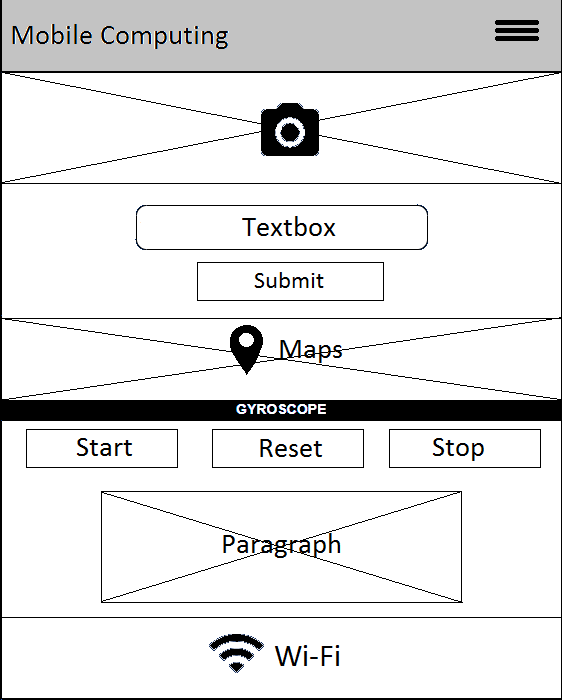
Mobile Computing App

Student Boanta Cristian

1. About the project

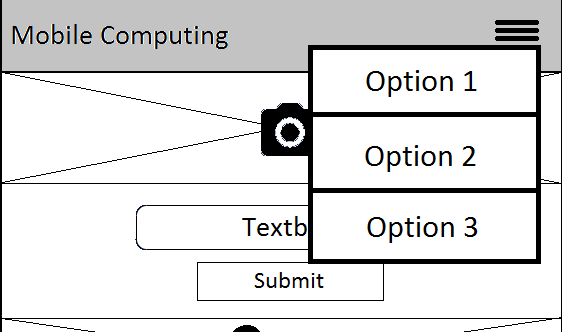
This mobile app consists in manipulating various functions of an android device. Some examples of functions that are accessed and used by this app are Wi-Fi listening, taking photos and viewing the picture after that, accelerometer, pedometer, maps, push notifications and notify.

1. Application wireframe

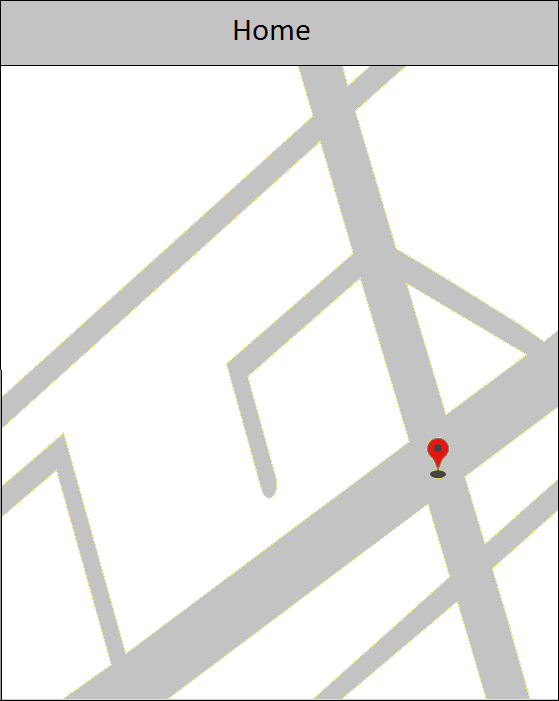


This first wireframe is the main which shows the template and an overview of the main screen of the app which consists in almost all functions that app can access, except the map.

There is a menu button with settings, information about the app and a “stop and exit” option.



Next wireframe is for the map.

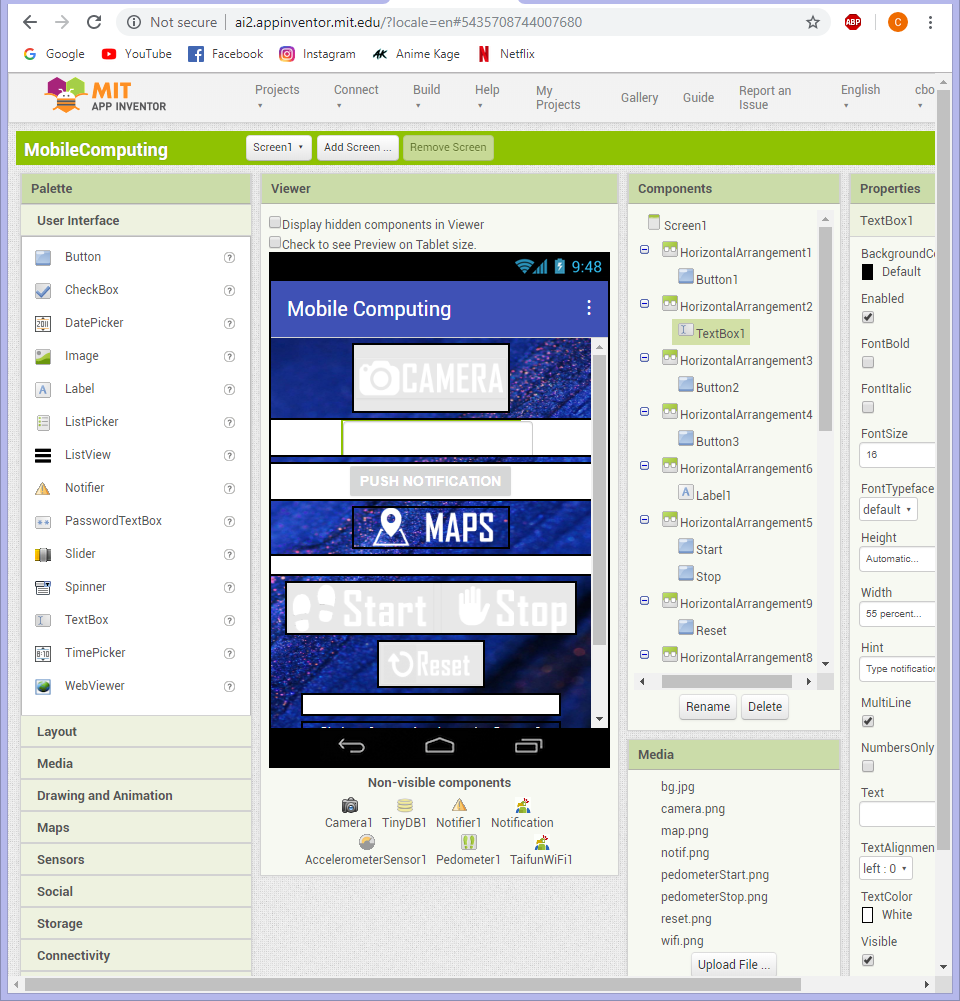


This wireframe is very simple and consists of a simple map with given coordinates that are given through the service and a Home button to return to the main screen.

You can scroll, pinch to zoom in and out and rotate the map.

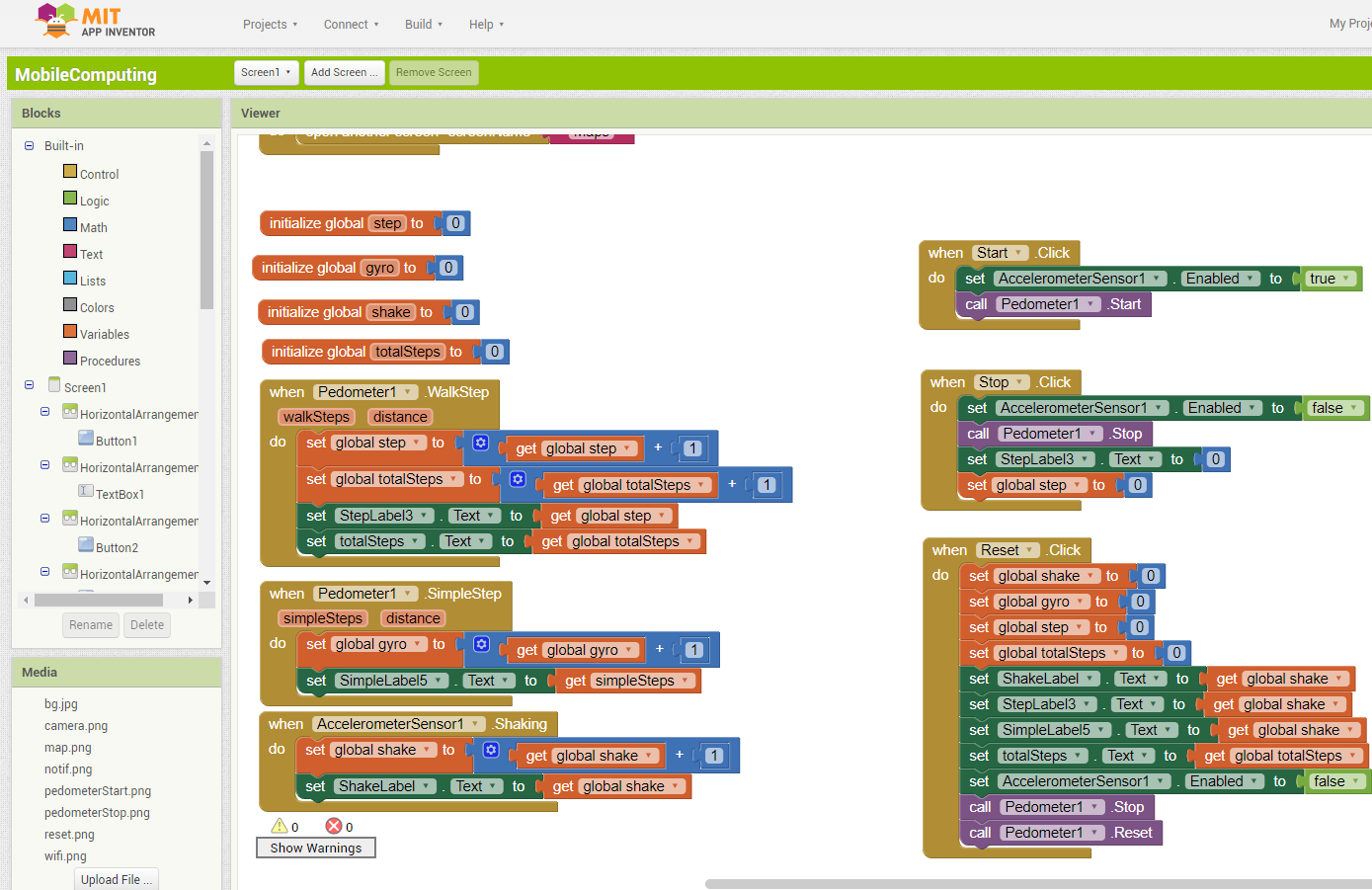
1. Server Side Functionality

For the services I used the recommended way on the classroom and I’ve connected on MIT App Inventor with the Google account and developed the app there. I’ve implemented the interface and the functionalities behind using this web service.



For transferring and viewing the app on an android device are more options:

* Build the app (.apk) and download it using a QR code scanner
* Build the app and save the .apk file to the computer
* Download from Play Store the AI Companion application and scan a QR code or type a code in the required field to launch the app directly on the android device
* Using an emulator that simulates the android device
* Connect android device to computer through USB port, allow USB data transfer and launch app



Blocks of code that do the logic behind the functionalities of the app.

1. Movement

For accelerometer are declared variables that store the information from accelerometer sensor: step (steps from pedometer), gyro (counts every movement of the phone), shake (counts just the hard moves of the phone) and totalSteps (store movements from pedometer and don’t erase them at the press of the stop button).

There are libraries used (drag and drop) for the movement of the phone, in fact for using accelerometer: Palette -> Sensors -> AccelerometerSensor/Pedometer.

**Buttons:**

**Start** - when clicked enable the accelerometer sensor by giving it the Boolean true value and start the pedometer.

**Stop** - when clicked disable the accelerometer sensor, stop the pedometer, set steps to 0 and render the value, but no not change the other values (totalSteps, shake and gyro).

**Reset** - when clicked disable the accelerometer sensor, stop and reset pedometer, update all variables to 0 and render them.

The buttons are labeled in the designer and called with the same name in the blocks section.

When the accelerometer detects movement it increments the global variables with 1 for every move of the device felt and render them in interface every time.

ex.: - sudden and powerful moves increments shake variable (and all others)

- axis (horizontal and vertical) movement increments the steps(pedometer) and gyro.

- soft movements in random directions increment just gyro( Accelerometer Gyro in interface)



1. Images

Images are captured with the camera feature called by the app after the camera button was clicked. For this I have used Camera library from Media tab.

When “CameraButton”.click do call Camera.TakePicture



7. Wi-fi Listener

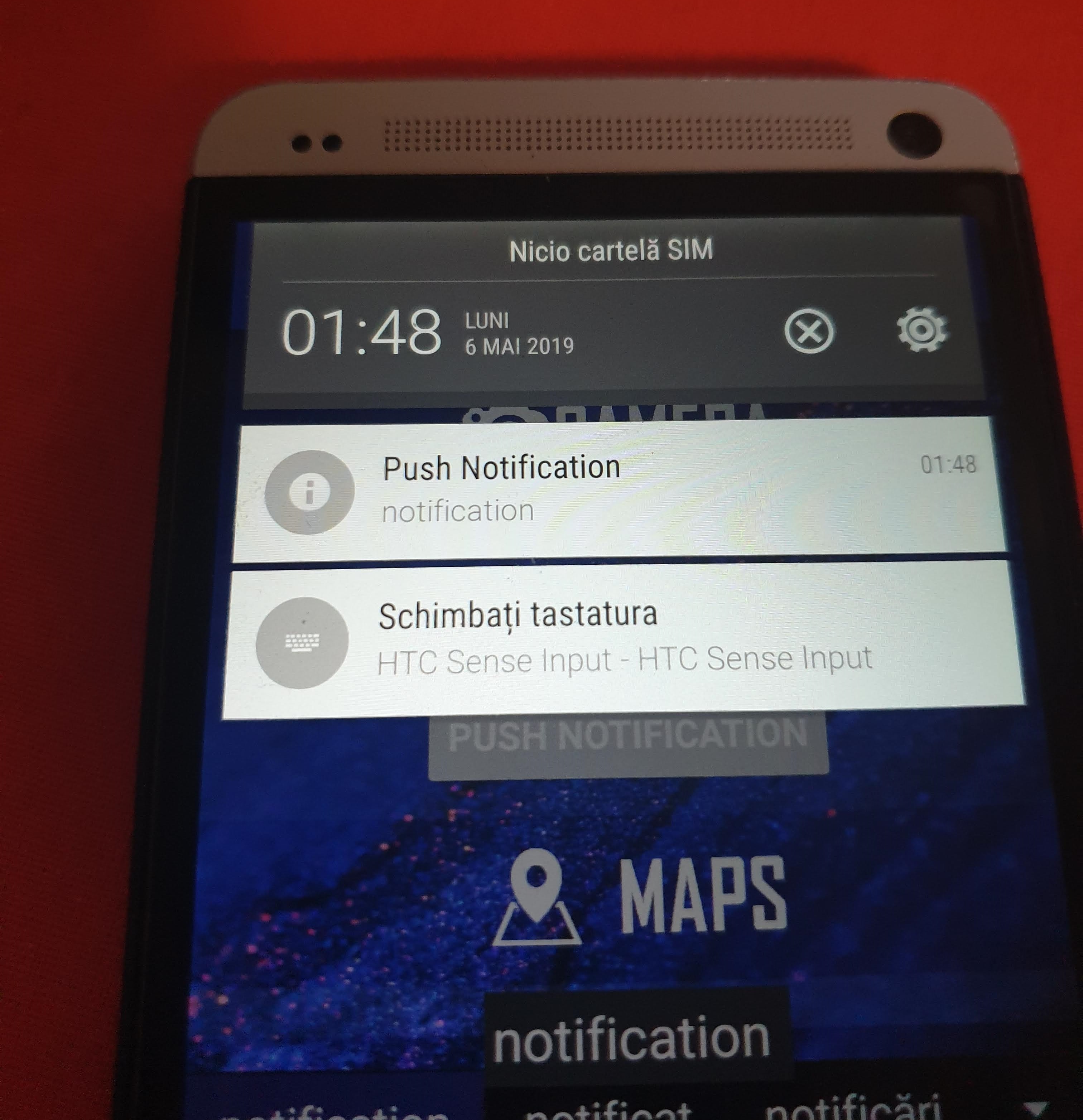
For Wi-Fi listener I have imported an extension named *TaifunWiFi* and after a button click is called a function from that library that checks if the wireless in on or off and display a notification in the middle of the screen with the specific text. The notification action is done using Notify library.



8. Push Notifications

Push Notifications function also uses an import a library: *TaifunNotification.*

The notification is pushed, with a message, after typing the message in an input box and click a button. The app notifies the user if the input box is empty. The notification has a title and the content consist of the message written in the input box.



**!** May not work with new phones that have better firewalls and more protection

9. Maps and Directions

The map function is the single one that opens in another screen after clicking a button.



After clicking the button, the interface switches to the MapScreen where is rendered the map with a home button in the top. The map is on the entire screen and the user can navigate on it. For rendering the map is used a map component (Maps -> Map) which is given the coordinates of a default location. Once home button is pressed the interface switches again to the main screen.

