USER INTERFACE

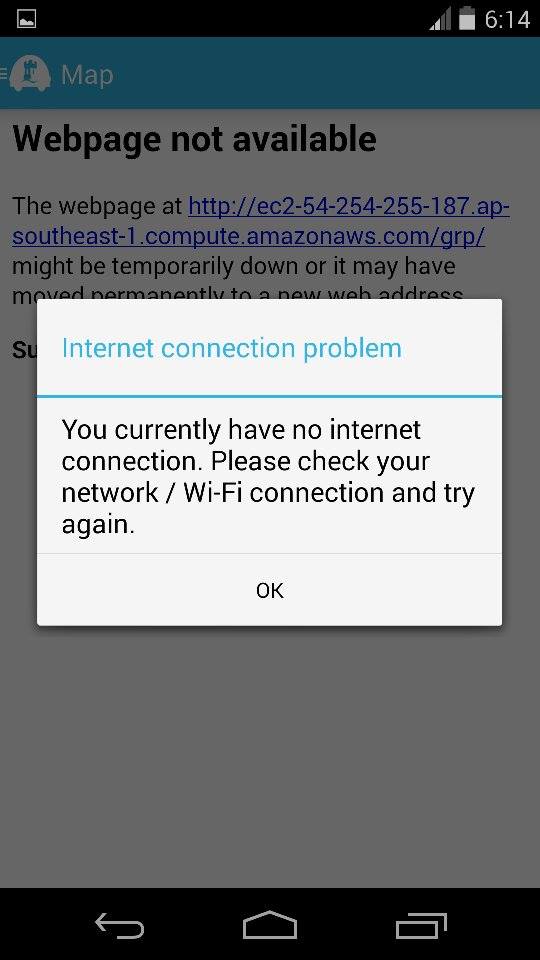
The final interface that was implemented had slight modifications from the initial prototypes that were developed, in various ways:

1. It was discovered that the method for navigation between the app’s pages were conflicting with the design guidelines suggested by Google, so the main navigation method decided on was through the use of navigation drawers.
2. A proto-type map was implemented, instead of the entire university parking.

**User Interface at Launch**

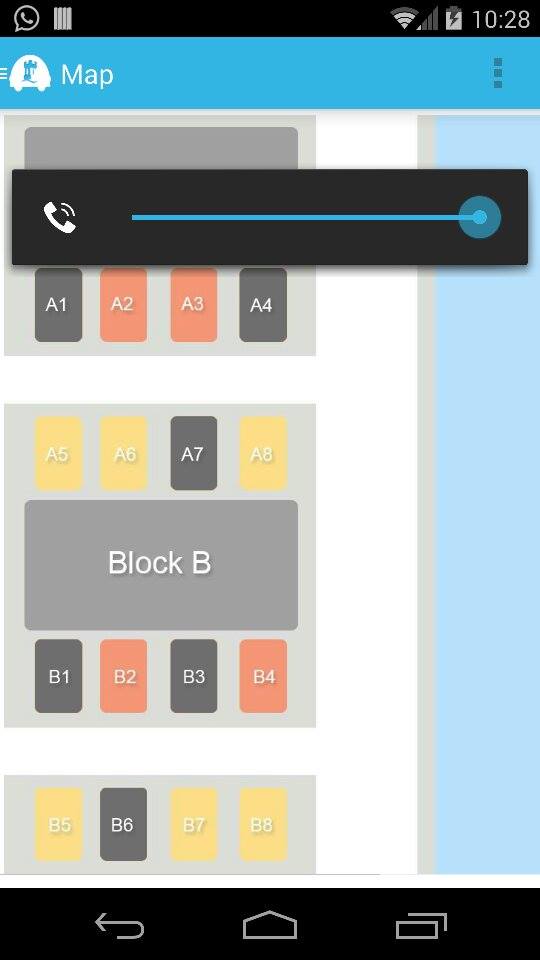
At launch, the application checks if the mobile device being used is connected to the internet.

In the event that it isn’t, the following prompt is displayed:



After dismissing the notification, the launch of the app is continued although the pages requiring an internet connection will be non-functional. It has been implemented in this way, as crucial information regarding the parking spot is saved locally.

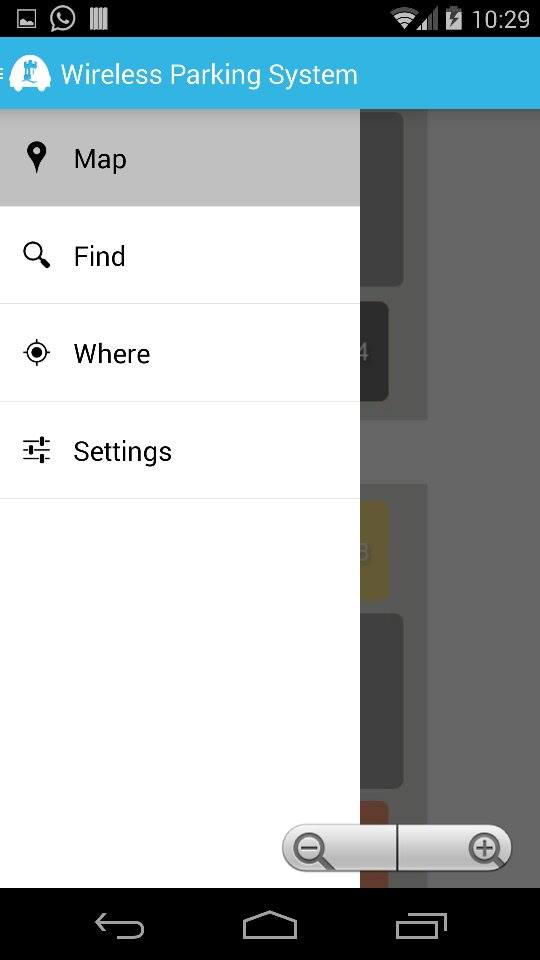
**Map page**



At launch, the first screen that the user sees is the map of the parking lot, along with their availability. A key is used to help for the first time users, with the following configuration:

* Red parking – The available parking spots usually reserved for the staff and lecturers on campus.
* Yellow – The available parking spots for both staff, lecturers and students
* Black – An occupied parking spot.

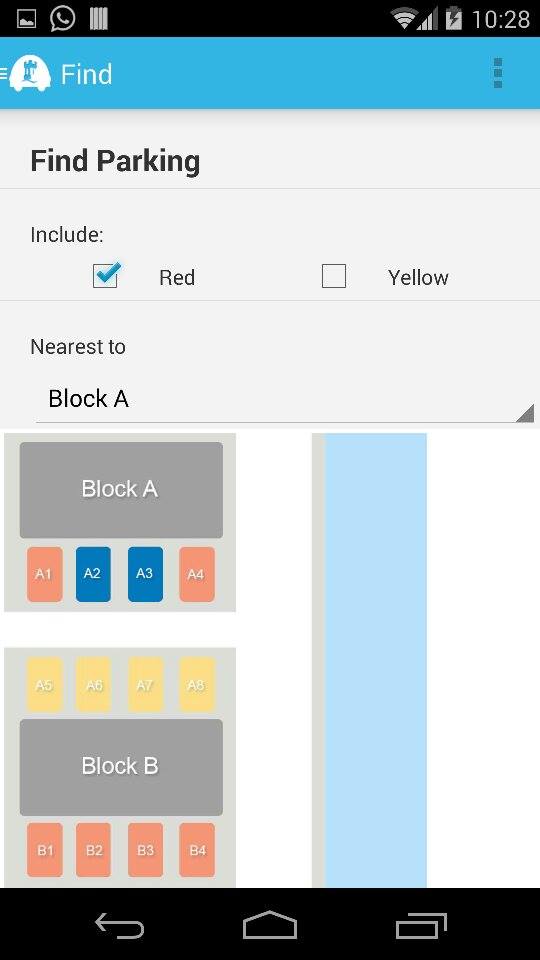
At the left of the top menu, a toggle is provided, which on touch opens the navigation drawer as illustrated below:



Other available pages within the app

Current page that the user is viewing

**Find page**



The find page within the app is used to search for a convenient parking location for the user, depending on 2 criteria:

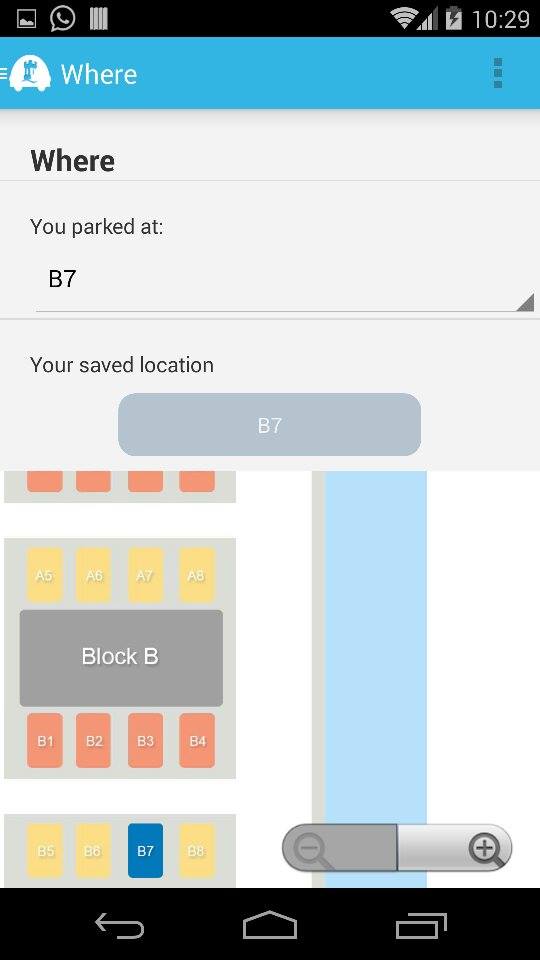
* The type of parking
* Proximity to building that they would like to go to

The results are then highlighted in a dark blue colour, which the user may go to.

**Where page**

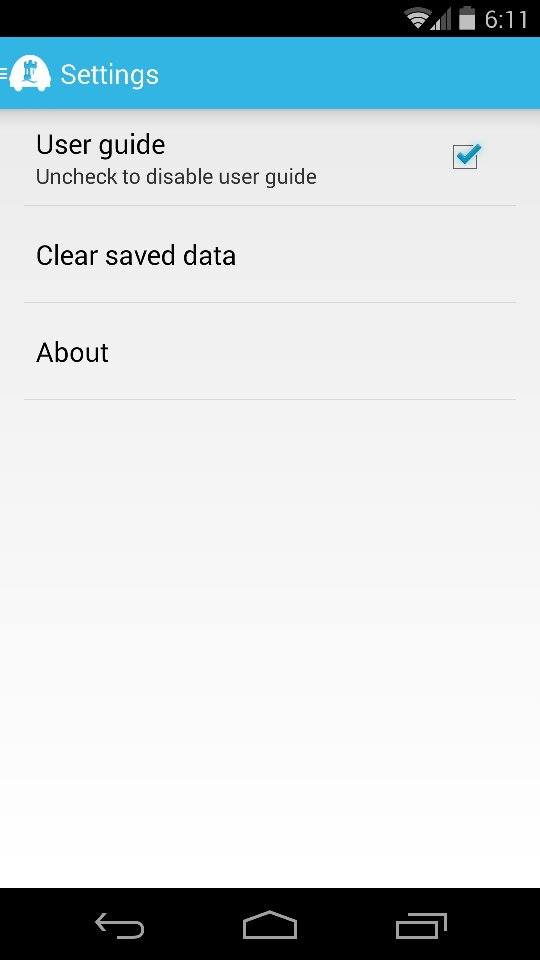
On this page, details of parking location that the user has parked in are saved. Once the user has parked, he selects the parking spot to save his location at. The parking is spot is then saved, listed and highlighted in a dark blue colour.

An example is as shown below:



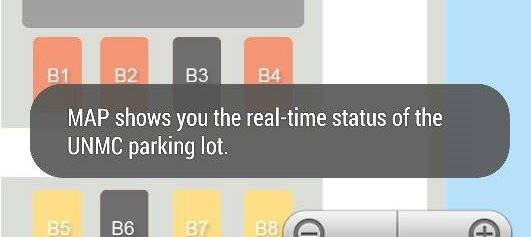
The parking location is saved, even if the user exits the application.

**Settings page**

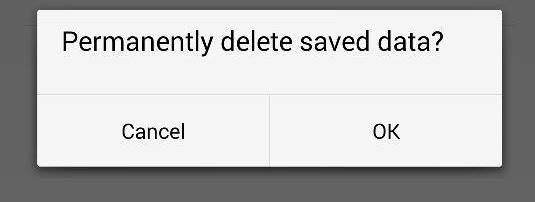
****

This is the settings page for the mobile app. It contains:

* A toggle for the user guide, which is usually visible during each start-up as notifications. It is implemented as shown below:

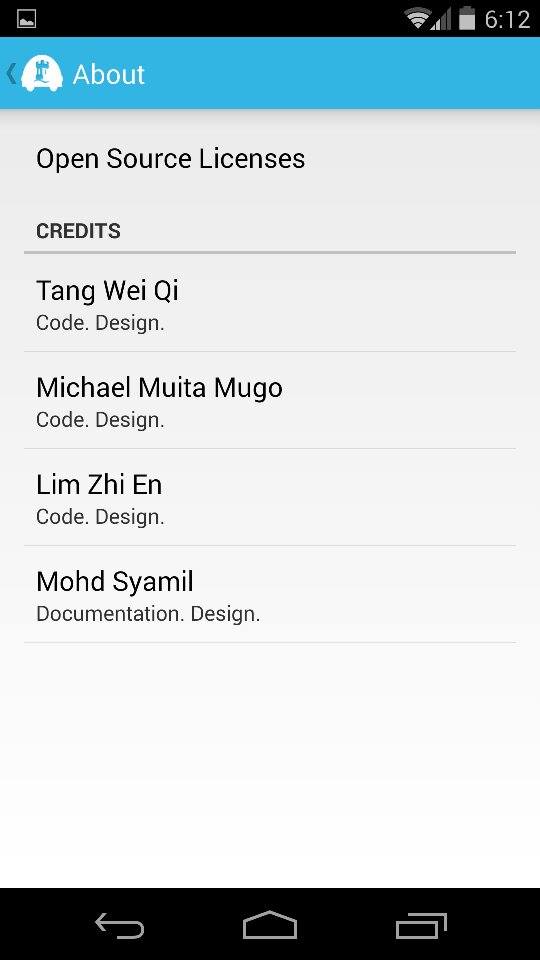


* A button to clear the locally saved information about the parking spot of the user. It is accompanied with the following prompt:



* And a button linking to an about page.

**About page**

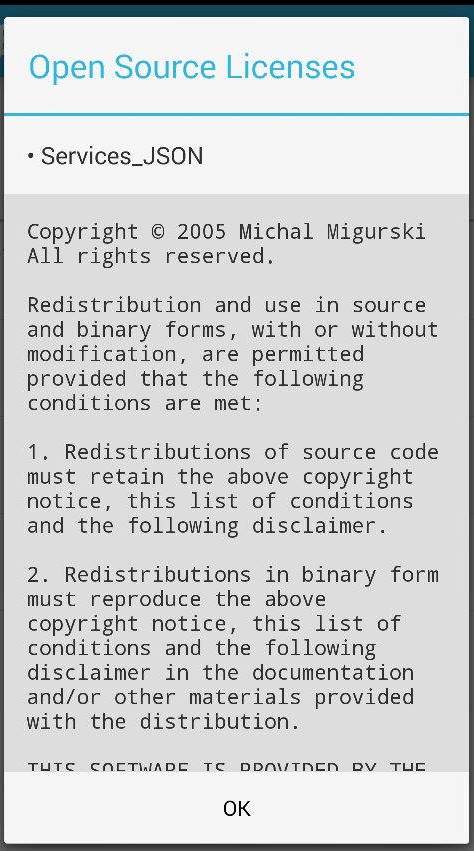


This is the About page, it details out:

* The credits for the mobile application
* The open source code implemented by the application.

Additionally, it has been set out to intentionally go back to the Settings page, instead of the drawer.

Upon selecting the Open Source Licenses button, the following notification is displayed:



**Implementation of the User Interface**

The application follows the fragmentation guidelines of Android. It contains 4 fragments, along with an adapter and model to manage the navigation drawer and a main activity to connect all the components together. Along with this, various styling information and resources were modified to implement the application in a single frame layout, to enable navigation through a drawer.

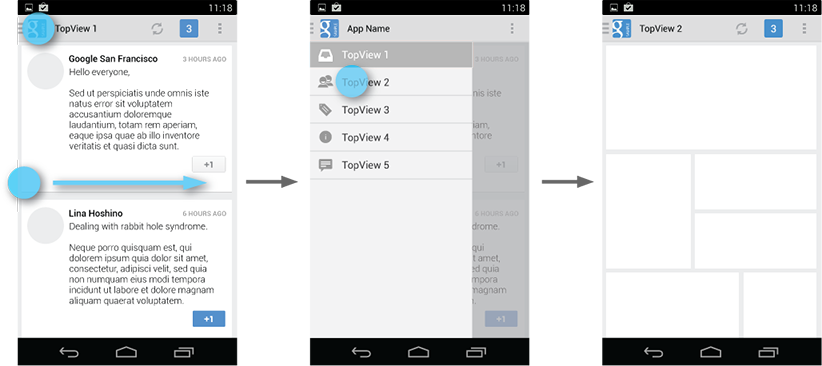
This was adapted from the following referenced:

Tamada, R (2013) Android Sliding Menu using Navigation Drawer. Source Code. <http://www.androidhive.info/2013/11/android-sliding-menu-using-navigation-drawer/>

**Background**

Following the upgrade in the Android ecosystem into the Jelly bean and future versions, Google implemented a new standard in navigation for all mobile applications. It entails the representation of the portions of a user interface in an activity[[1]](#footnote-1). These representations are known as fragments.

The following diagram illustrates the concept behind the use of fragments for navigation:

 **Source:** [**http://developer.android.com/design/media/navigation\_drawer\_overview.png**](http://developer.android.com/design/media/navigation_drawer_overview.png)

The user accesses the drawer by swiping from the left edge of the screen or by touching the application icon on the action bar. As it opens, it overlays the content but not the action bar.

The panel that displays app’s main navigation options on the left of the screen is implemented by declaring the main user interface component as a DrawerLayout object as the root. Inside it, two child views are used:

* FrameLayout to contain the main content (This is populated by the fragments at runtime)
* ListView to contain a list of the various pages available within the app.

At start-up, the navigation drawer’s list of items are populated, by the use of an adapter which basically provides a series of calls that connects all the various resources, such as strings, styling and images (stored in various sections of the app) together. The items are then associated with an action listener, which connects to changes the fragment viewed on the main activity. They are pre-assigned by default, and upon selection, the main activity’s fragments are changed.

**Layouts of individual pages**

Each Fragment has been assigned a unique xml layout, often containing multiple nested linear layouts, weights and alignments to ensure compatibility with various screen sizes.

1. This is a single, focused action that a user can perform [↑](#footnote-ref-1)