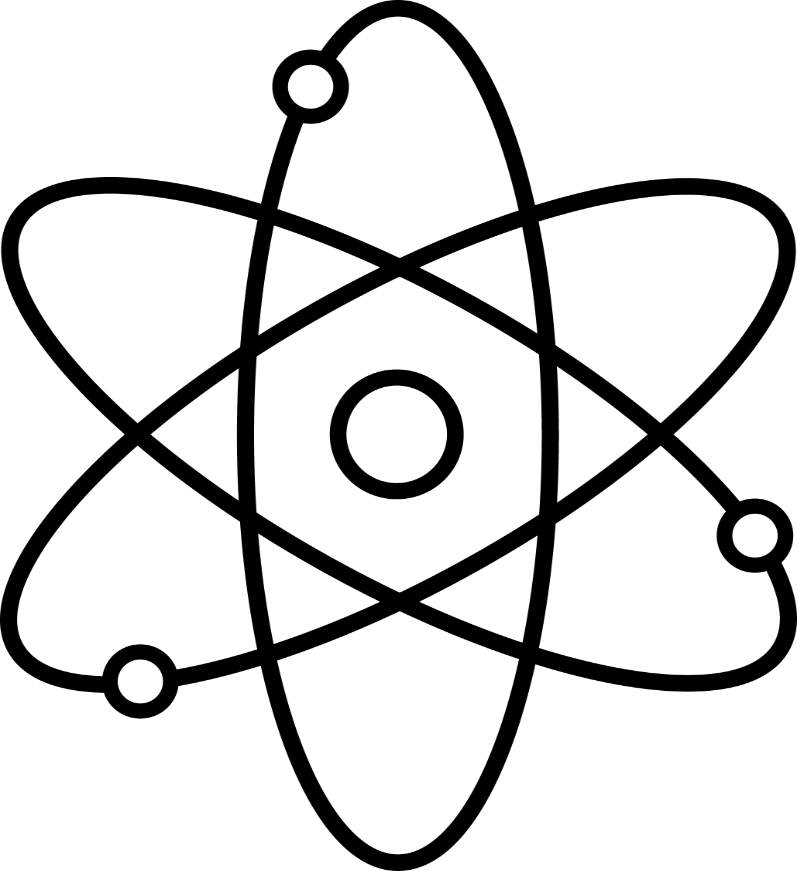


**CSC 7054  
Web and Mobile App Development**

Up and Atom:   
An App to Help You With Your Morning Routine!



Team Members:

Hannah Butler (40004276)  
Kelvin Harron (40042864) (Group Leader)  
 Peter McCaughey (14531046)  
 Conor Taggart (40164315)

Introduction

For this project, we decided to create an Android App using Android Studio. After discussing a number of options, we decided to develop an App that would set off an alarm when the user requested – however, to help those that aren’t “morning people”, the user would have to scan a preassigned barcode to get the alarm to stop. The idea behind this would be that the user would leave the **object with the** Barcode to be scanned out of reach **and ideally be something the user would use each morning such as a coffee jar**, therefore motivating them to get out of bed to make the alarm stop, thus helping along with their morning routine. While there are a few alarms with this feature available on the marketplace, we felt that our twist – geotagging – would make it a more interesting product.

Requirements  
  
a) Problem Statement  
The original problem faced by the potential customer is that many alarms are easily dismissed; people often do this in their sleep without realising and can often sleep through alarms meaning they are late for work, flights or important social engagements. The Play Store is heavily inundated with various alarms that request for a maths problem to be solved, a general knowledge question to be answered or a pattern to be drawn and while these engage with the brain, few actually get the user up and out of bed. While developing this idea we thought that it could develop it into a “Morning Routine” App – attach Barcodes to essential daily belongings also (wallet, keys) so that the user has to scan these items before they leave the house otherwise an alarm goes off and won’t stop until the correct Barcode is scanned.

b) System Requirements  
A requirement is defined as “a condition or capability to which a system must conform”. A number of requirements for the App were extracted from the Problem Statement and are listed below;

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Requirement** | **Brief Description** | **Priority** |
| 1. | User is able to view alarms | User is able to view and select different alarms that are already set. If there are no alarms, then a home screen is presented |  |
| 2. | User is able to set an alarm time | User is able to select any time within a 24 hour cycle, on any day of the week |  |
| 3. | User is able to set a home location | User is able to select a location as their “home” so the alarm won’t request a Barcode if they are outside of this location |  |
| 4. | User is able to pre-set Barcode | User is able to pre-set any Barcode to be scanned to deactivate alarm once activity is triggered |  |
| 5 | User is able to snooze alarm | User is able to snooze the alarm for a pre set amount of time rather than scan the barcode |  |
| 6. | User is able to override Barcode | User is able to enter an override code rather than scan the Barcode if they are outside their home location |  |
| 7 | User is able to review settings | User is able to view, edit, delete and update any information on the user settings at any time |  |
| 8. | App is easy to navigate | Usability is essential in this App; the process must be made clear to the user |  |
| 9. | App is clean and modern looking | A modern looking UI is essential to fit in with the Bootstrap aesthetic users are used to |  |

Functional Requirements Specification

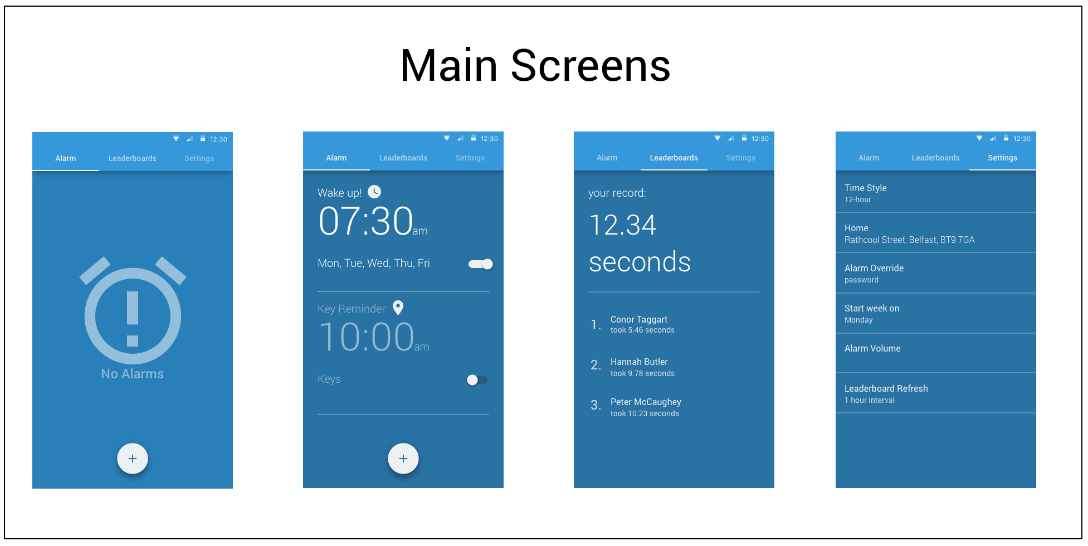
a) Actors and Goals

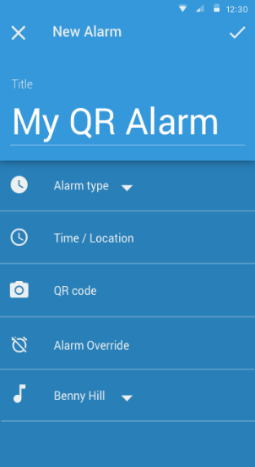
As the App will be available to the members of the general public they will be the main actors who will interact with this App. The goal of the user using the app will be to improve their morning routine. This is covered in the System Requirements which will be included in the finished product.

b) Use Cases

|  |  |  |
| --- | --- | --- |
| **No.** | **Casual Description** | **Related Requirement(s)** |
| 1 | User can view alarms | 1 |
| 2 | User can add, edit and delete an alarm | 2, 3, 4 |
| 3 | User can turn off an alarm by scanning a barcode/QR code | 2 |
| 4 | User can turn off an alarm using an override code | 2, 6 |
| 5 | User can disable alarming without deleting it | 1, 2 |
| 6 | User can set an override code | 6 |
| 7 | User can set their location | 3 |
| 8 | User can view settings | 7 |
| 9 | User can edit settings | 7 |

The Use Case Diagram can be found at Appendix A.

User Interface Specification  
  
a) Preliminary Design   
After deciding on the premise of the App, the initial design was mocked up on Sketch for Mac. Taking into consideration our ideas on how the main screens of the App would appear, what functions we would like that vary from other alarm Apps on the market, and ensuring it was intuitive and aesthetically pleasing to a user, the initial mockups are shown below:



*Fig. 5*

*Fig. 4*

*Fig. 3*

*Fig. 2*

*Fig. 1*

Initially, an extra feature we thought could be included with the app would be a  
leaderboard (Fig. 4) so that users could connect with friends and compete on weekly leaderboard, similar to apps such as MyFitnessPal or FitBit. This would be feature we would hope to further develop as the App userbase grew. We amended this in our final product to be a “How To” page for new users to make it explicitly clear how the App worked.

Design  
  
a) Features

Up and Atom allows a user to set up a large number of alarms, the user must scan a barcode or QR alarm to disable the alarm.

On starting the app the user is presented with a blank version of the main menu, as alarms are created, they are placed on a card **generated using** the recycler view that makes up the main menu.

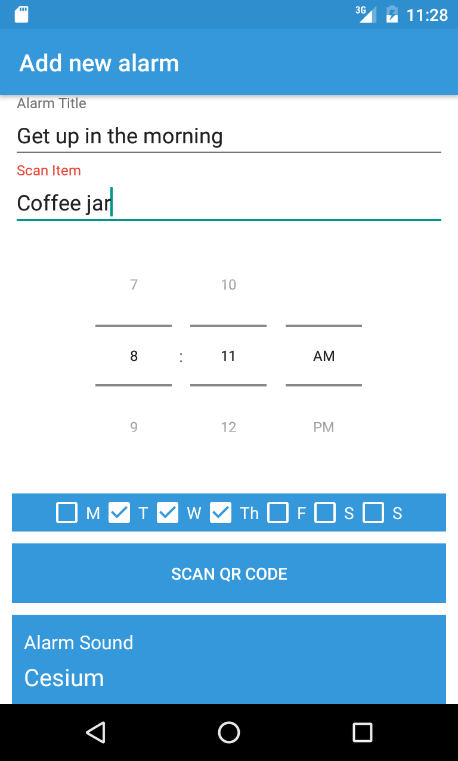


**Screenshot of empty “Main Menu”**

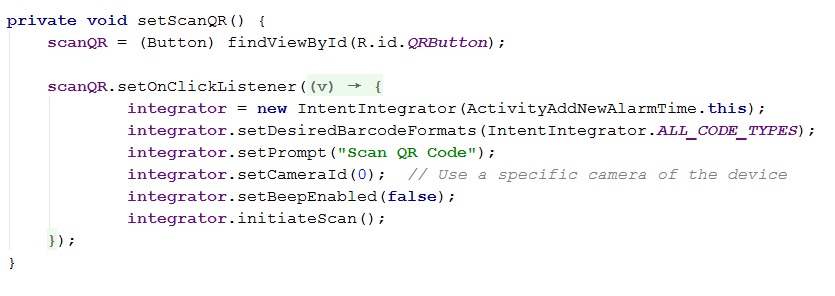
**XML code snippet show the main menu’s TabLayout**

The Main Menu also features a tab layout, swiping left from the main menu will show a brief tutorial on how to set a new alarm. A ‘Settings’ and ‘About’ screen can be accessed by tapping the overflow menu in the top right of the screen.

*Adding a New Alarm*

When the user presses the red floating action button on the main menu they will be presented with the ‘Add New Alarm’ screen, from here they can set the various attributes of their alarm – Name, a memo (for example, a reminder of the item they will scan), the time of the alarm, a barcode or QR code to associate with the alarm, the days of the week the alarm should repeat on and the ringtone the alarm should use. This page uses a ‘NestedScrollView’, which hides the toolbar at the top of the screen when the user scrolls down the page.

Screenshot of the ‘Add New Alarm’ Screen



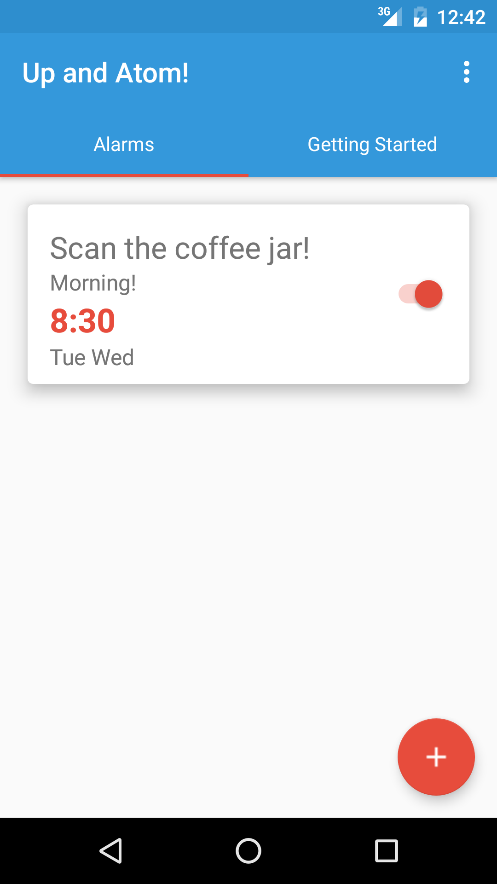
XML code showing the ‘Add New Alarm’ screen, note: TextInputEditLayout’s used in place of EditText’s

Java code showing the QR scanner launcher

*Alarm View*

In the image below, the newly created alarm details are placed into a CardView layout. As a user could have multiple alarms with different data we choose to implement the RecyclerView widget which allows us greater performance and flexibility to extend the feature if needed compared to the traditional ListView widget. We are taking one instance of the CardView and ‘recycling’ its views for multiple cards but keeping the data displayed by views unique. The RecyclerView gives us access to the powerful LayoutManager meaning we can specify different types of list layouts such as grid and staggered layouts however, we choose the LinearLayout as it would be most suitable for users of our app. The CardView has a very material aesthetic to it similar to how Google Now appears and as such, helps Up and Atom contribute to a consistent look and feel across the user’s device.

In this instance CardView essentially acts as template that is **used** when required by the RecyclerView on the main menu, this is a highly efficient method of showing lots of information, much more so than the older, but less complicated ListView. CardView also allows us to maintain the Material look and feel of ‘Up and Atom’.

**

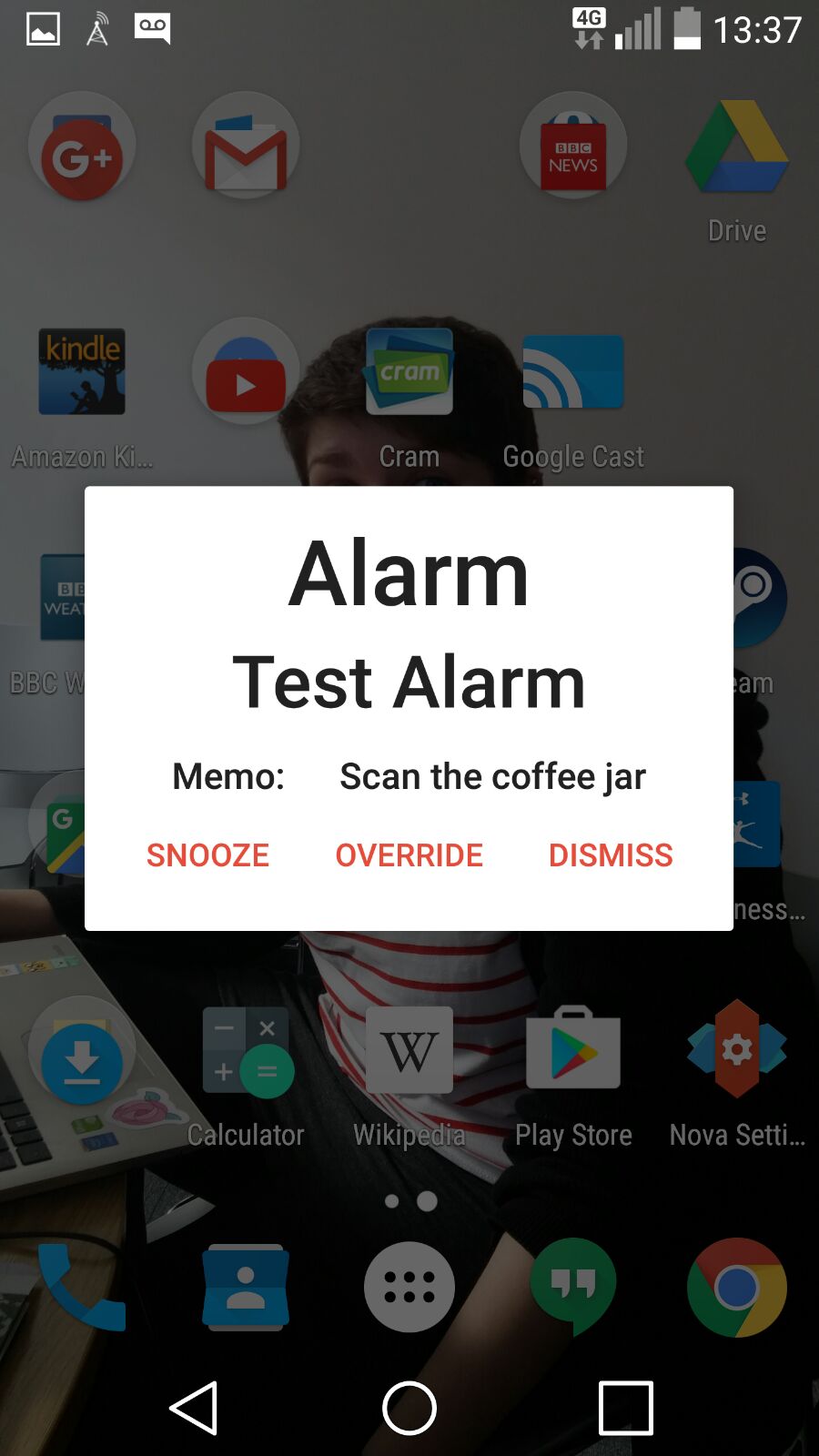


XML code showing CardView being implemented

Screenshot of the ‘Add New Alarm’ Screen

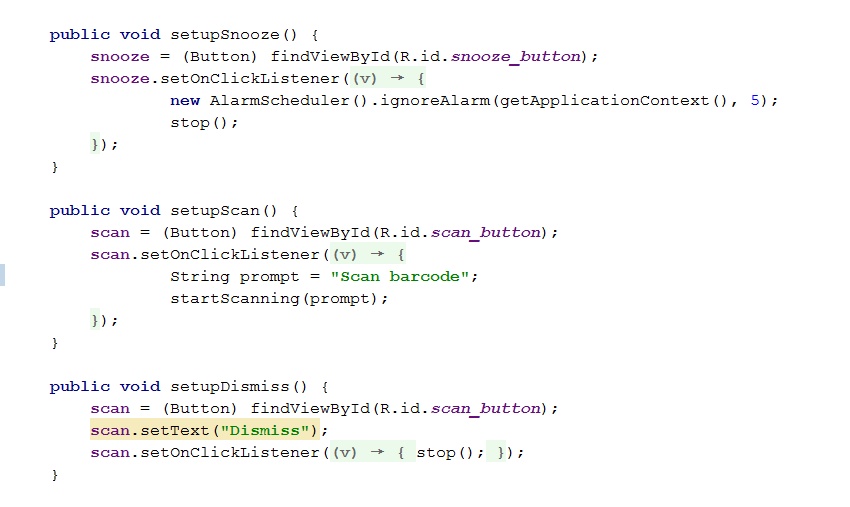
*Dismiss Alarm*

Once the alarm is activated as it is activated by the alarm scheduler. A transparent activity appears on screen giving the user the option to snooze, override or scan the pre-set barcode to make the ringtone/media cease. Due to the complexity of having the alarm preference change whenever your location is outside of your “home” setting, there are multiple methods within the ActivityDismissAlarm.java to ensure that validation is implemented and the correct option offered to the user whenever the alarm is activated.



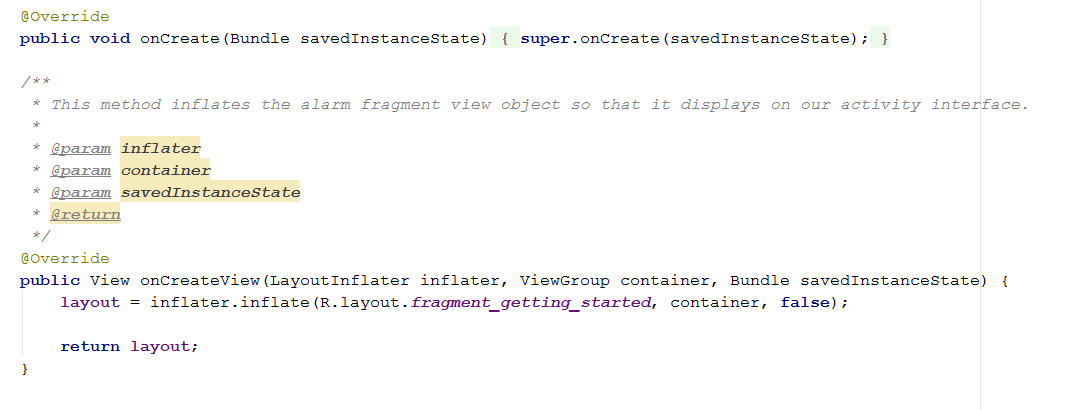
XML code showing the “Dismiss Alarm” Activity

Screenshot of ‘Dismiss Alarm’ Screen

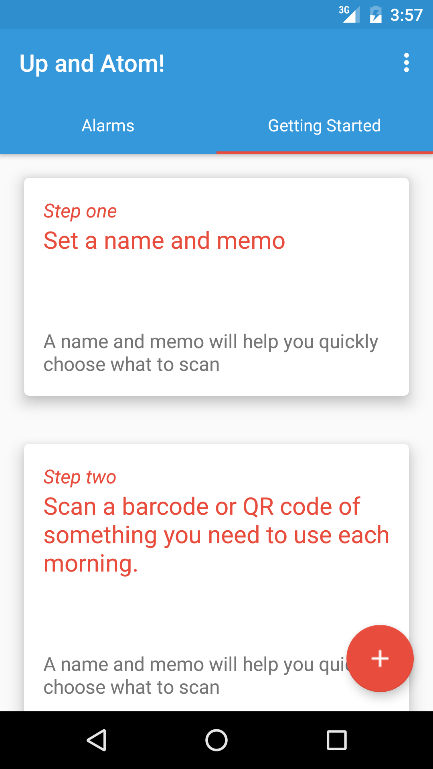


Java code showing the snooze, override and scan (or dismiss) button options

*Getting Starting tab*

As Up and Atom might be a new concept to some users, we designed a “Getting Started” tab so that the user can have a better understanding as to how the app works! This also uses CardView to maintain the material look throughout the rest of the App.

Javacode that creates the LayoutInflater





XML for individual CardView fragments

Screenshot of “Getting Started” tab

b) Database

A SQLite database was developed to persistently retain alarm objects. When a new alarm object is created it is stored, when the scheduled alarm activates the ActivityDismissAlarm.class, the alarm Id passed with the Bundle is used to read the relevant alarm object from the database. Similarly, alarms that are removed from the MainActivity.class, are removed from the database using the delete method. An update method was included to allow alarm objects to be edited. In future revisions of the application, the method would enable existing alarms to be changed, however, in its existing incarnation, new alarms can only be created, read and deleted.

c) Barcode scanner

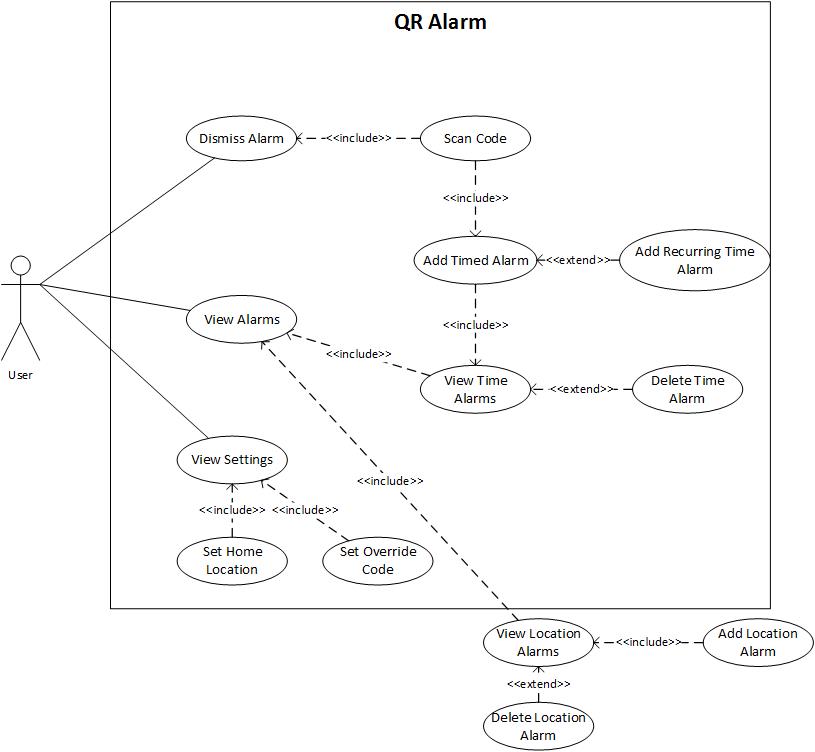
The Barcode was implemented by using the ZXing (Zebra Crossing) github library. This was interesting to implement because the library often requires you to install an external app alongside to use the results within your App. However, because we were only interested in receiving and analysing data from a Barcode, as opposed to finding out production information from the barcode, we were able to minimise the amount of the library required, embed it in the App, and have it run as a gradle.build as opposed to importing all the libraries of the official ZXing app.

Future Development

A feature we wanted to include in Up and Atom, but was not within the project timescale, was to include geofencing. There are some classes within our javacode that show the work that was started on this functionality. This feature would have given the user the option of a second alarm, based on location. Once the user leaves the set geofence, without scanned the pre-set barcode, an alarm would also go off. This would be used as a reminder app for such items like your keys or wallet.

To further enhance the look and feel of the app, in future we would be keen to introduce material based animations into the app. A popular example is when a user presses the floating action button, it draws an animation from the button that fills the screen and then inflates the new activity view. This would be achievable by following google developer tutorials but would take considerable time as it would need to be tweaked repeatedly to get the most ideal effect that is pleasing but doesn’t stall the user when they want to use the app.

As previously mentioned, we would also like to eventually incorporate a leaderboard so that friends can compete for fastest “morning routine” scores. **// need a rough idea on how to do it?**



**Appendix A**

*The use cases outside of the box represent stubs for future implementation*

**Appendix B**Gantt Chart

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Week 1 | | Week 2 | | Week 3 | | Week 4 | | Week 5 | |
| **Project Deliverables** |  | |  | |  | |  | |  | |
| Main Activity and XML |  |  |  |  |  |  |  |  |  |  |
| Alarm Screen Activity and XML |  |  |  |  |  |  |  |  |  |  |
| Setting Screen Activity and XML |  |  |  |  |  |  |  |  |  |  |
| Java code for Alarm Screen and Setting |  |  |  |  |  |  |  |  |  |  |
| Location Screen Activity and XML |  |  |  |  |  |  |  |  |  |  |
| SQL Lite Database |  |  |  |  |  |  |  |  |  |  |
| Geomaps Fragment and XML |  |  |  |  |  |  |  |  |  |  |
| Barcode Library and Implementation |  |  |  |  |  |  |  |  |  |  |
| Java code for Alarm Class, Broadcaster and Receiver, and Dialog Fragment |  |  |  |  |  |  |  |  |  |  |