

Declaration Cover Sheet for Project Submission

SECTION 1

Name: Kieran Shine
Student ID: x12113239
Supervisor: Manuel Tova Izquierdo

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: _____ Date: _____

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

National College of Ireland
BSc in Computing
2015/2016

Kieran Shine
12113239
shinekieran@gmail.com

StarGaze
Technical Report



Contents

1	Executive Summary	5
2	Introduction	6
2.1	Background	6
2.2	Aims	7
2.3	Technologies	7
2.4	Research	8
3	System	9
3.1	Requirements	9
3.1.1	Functional requirements	9
3.1.2	User requirements	12
3.1.3	Environment Requirements	13
3.2	Design and Architecture	13
3.3	Implementation	14
3.3.1	Component Building	14
3.3.2	Integrating the Open Weather Map API	16
3.3.3	Integrating Google Maps API	17
3.3.4	Integrating Astronomy RSS Feeds	18
3.3.5	Connecting the SQLite Database	19
3.3.6	Connecting Screens	20
3.3.7	Compass	20
3.3.8	Other Functions	21
3.4	Testing	21
3.4.1	Unit Testing	21
3.4.2	System Testing	23
3.4.3	User Acceptance Testing	24
3.5	Graphical User Interface (GUI) Layout	29
4	Obstacles Encountered	34
5	Future Improvements	35
6	Conclusions	36

7	References	37
8	Appendix.....	38
8.1	Project Proposal	38
8.2	Project Plan	39
8.3	Requirements Specification	41
8.3.1	Document Control	41
8.3.2	Distribution List.....	41
8.3.3	Introduction	42
8.3.4	Definitions, Acronyms, and Abbreviations	42
8.3.5	User Requirements Definition	42
8.3.6	Requirements Specification.....	42
8.3.7	Non-Functional Requirements.....	50
8.4	Monthly Journals.....	52

1 Executive Summary

The recent explosion of the smartphone market has opened up a world of technological possibilities which hitherto would have required significant investment in hardware from the end-user. StarGaze aims to integrate several of these technologies in order to make the night sky more accessible to all.

The StarGaze mobile application is about providing amateur astronomy enthusiasts with a low cost, mobile, effective tool which will enable them to leverage existing technologies from their handset device to greatly increase the efficiency of their astronomical efforts. In addition to current astronomy enthusiasts it is my hope that the intuitive and attractive interface will attract newcomers to the hobby.

It is my intention to develop this application for use on Google's Android operating system. If completed successfully, I believe it will address a gap that currently exists in the market and would attract significant interest from a cohort of users who are willing to pay for services which they are confident will be of a high quality.

2 Introduction

2.1 Background

Having been interested in Astronomy for many years I have always found it difficult to get timely and accurate information which would aid me in planning viewing sessions. Knowing several fellow enthusiasts with the same problem, I envisioned the concept of this application. Added to this there is an abundance of applications in the market place which while popular serve no other purpose than to entertain the user. While entertainment of course has its place in mobile development I believe the potential of existing technologies are not being realised and certainly not being experienced by a significant percentage of smartphone users. This may have a lot to do with poor interface design which makes many applications seem inaccessible and difficult to use.

The resources available to those interested in astronomy and science in general have never been more abundant and easily accessible. This has been made even more so by the advent of the smartphone. By using these resources in new and innovative ways it is possible to open up access to huge swathes of information which can be applied to multiple pursuits.

There are numerous weather apps currently on the market and almost all smartphone devices now come with a built in mapping application. Added to this there are multiple online resources where accurate and timely astronomy information is available. However there is no one application which brings all three together in a manner that is intuitive and user friendly, and all three are essential to the effective planning of an evening's astronomy viewing.

On my own smartphone device, I currently have four applications which I use to aid me to plan a viewing and also in finding objects during a viewing. While all four are effective and useful it is inconvenient to have to constantly switch between applications. Added to this the process of finding an application that suits a user's requirements can be an arduous one, considering the amount of applications which are available, all purporting to meet your needs. Multiplying this process by four makes the chances of a successful outcome all the more remote. By integrating these services in one application the end users experience will be much more fruitful.

If successfully completed, I believe the application would find a valuable niche in the market as astronomy enthusiasts are generally willing to invest in tools which they believe will be useful to them and can enhance their viewing experience.

2.2 Aims

The primary aim of this project is to create a mobile application which will act as an aide to amateur astronomers wishing to plan their viewing sessions in advance.

By matching weather data with a mapping API, users will be able to receive information advising them of the best options available within a certain timeframe. The following features will be available:

A history of previous viewing sessions so users will be able to record information about their sessions. This information can include data such as the date, time and location along with what they observed etc.

When selecting a location for observing sessions the app will provide directions on how to get there including route, distance and time.

The application will also advise users of the weather at these locations.

The application should be able to provide information on what objects may be observed on a given night and what times are optimal for viewing particular objects. This will be achieved through a combination of hardcoded information and the integration of a third party resource.

2.3 Technologies

The application has been developed in the Android Studio IDE using the Java programming language. A version of the Model, View, Controller (MVC) pattern is used to structure the code. Application Programming Interfaces (API's) used were Google maps, Google Directions and Open Weather Map. In order to use the Google API's, the Google Play Services library was implemented. The Open Weather Map API's were retrieved as JSON files and parsed using JSON parsers in order to manipulate the data into readable, relevant information.

For the astronomy related data two external RSS feeds were implemented from sciencedailynews.com and spaceandtelescope.com respectively. These are displayed on the user interface using the RecyclerView functionality of Android Studio.

2.4 Research

In order to begin this project various forms of research have been necessary, from online tutorials, forums and examples of code to examining what kind of API's are available to me. These can be viewed in the references section at the end of this document.

To ensure that the application I designed would meet the needs of amateur astronomers, while also being built efficiently, I established three different groupings who I regularly approached for feedback.

I work in the IT Section of the Department of Public Expenditure and Reform (OGCIO) and have found several of my work colleagues were invaluable sources of advice in moulding my approach to this project. As well as being interested in the progress I made, the group I frequently consulted with came from various IT backgrounds including systems development, IT project management and network support. They gave me innumerable tips and pieces of advice on how to structure the project and on which areas I needed to focus on firstly and which could wait until later in the development lifecycle.

I also have various friends who like myself are astronomy enthusiasts and who are genuinely eager to see what kind of application can be produced and whether or not it will be useful to them. Unlike my work colleagues they are removed from the coding practicalities and focus only on the functionality they would like to see. Suggestions which they have put forward include an alert function which will tell a user of significant events in the near future (A comet pass or unusual alignment etc.). While this is something that in time I would consider adding in, it is not something I would consider in the initial scope of the project or a primary function. Nonetheless it is a good example of the useful feedback that such research yielded.

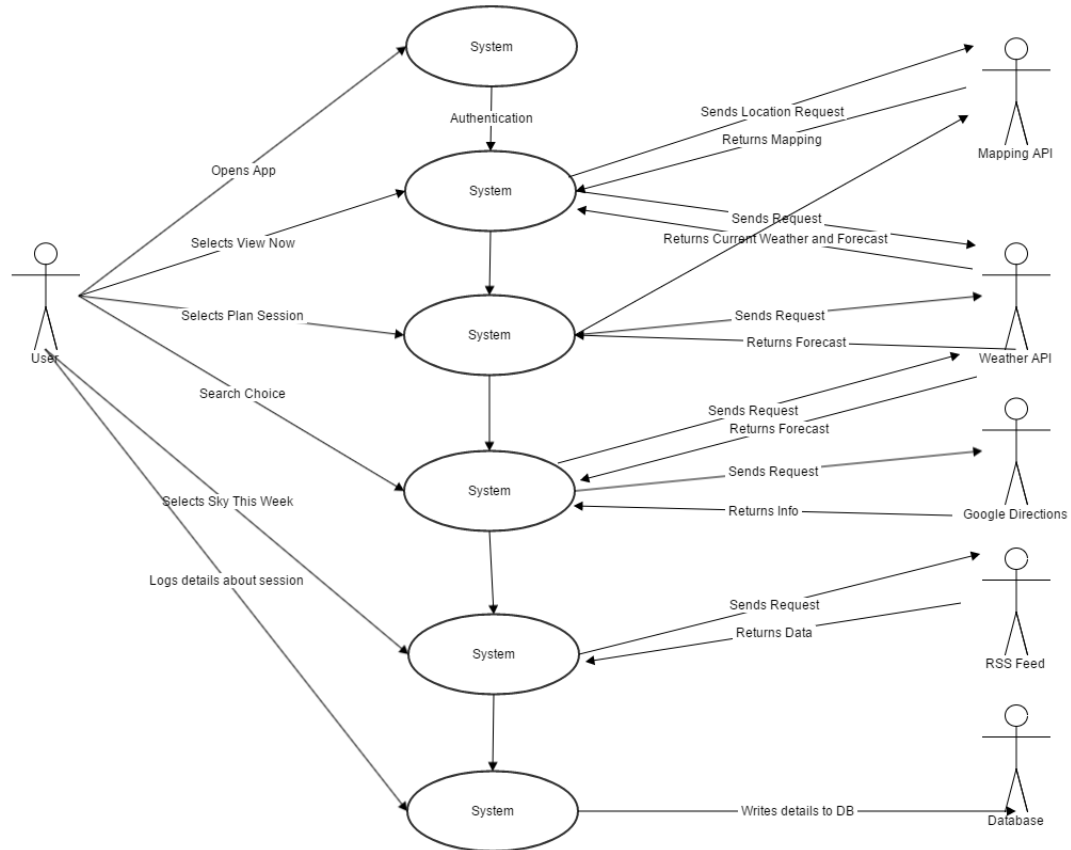
The third group I frequently consulted were non-IT, non-astronomy users. I found these were the best placed to say whether an interface was intuitive or not as they were not blinded by the potential of the application but saw only buttons which either worked or didn't. Especially in terms of developing the user interface, I found these discussions the most helpful.

3 System

3.1 Requirements

3.1.1 Functional requirements

The Use Case Diagram provides an overview of all functional requirements.



Functional Requirement 1 – User is Able to View Current Location Data

Description and Priority

This is required so that users can obtain relevant information relating to their current whereabouts. It is required for both of the main API's with which the application integrates, namely mapping and weather. This is a fundamental requirement to the functioning of the application. Priority: HIGH.

Requirement Activation

There are a number of scenarios where this requirement is used. When the user wishes to view their current location and weather now, if they want to plan a future

session and interact with Google's Directions API or if they wish to view whether in nearby locations.

Technical Issues

The device being used must have internet access and GPS tracking enabled.

Risks

This function requires a fast internet connection, in the event of a slow connection there is the possibility it will display the last successful result, giving the user dated information.

Dependencies with Other Requirements

This is the foundational requirement of the application.

Functional Requirement 2 – System is able to communicate with Open Weather Map API

Description and Priority

In order to gain current and future weather data the system must be able to make API calls to this service. It would be impossible to plan sessions without it. Priority: HIGH

Requirement Activation

When user wishes to plan a session or view current weather conditions.

Technical Issues

User must have internet connection. API calls are coded to depend on current location.

Risks

There are inherent risks when depending on a third party source. The service chosen however is extremely well established.

Dependencies with Other Requirements

This requirement is dependent on the current location being retrieved. All API calls depend on the latitude and longitude coordinates being available.

Functional Requirement 3 – Interaction with Google Mapping API

Description and Priority

Essential in order to give context to weather information and so users can plan routes to chosen destinations. Also needed so users can connect to Google's directions service and satellite navigation. Priority: HIGH.

Requirement Activation

User must either select current location or specify other location. Directions option is activated by tapping the marker over the selected location.

Technical Issues

Internet access is required. Directions service diverts user from application to Google's own service.

Risks

Google's satellite navigation service is in Beta phase and is not yet fully documented, possibility of errors occurring as a result of this.

Dependencies with Other Requirements

Relies on current location requirement to operate. Requires weather API in order to give context. Without this it is just a mapping tool.

Functional Requirement 4 – Integration of RSS Feed

Description and Priority

There are two RSS feeds used in the application. Both are astronomy based and without these there is no automatically updated astronomy information. Priority: MEDIUM

Requirement Activation

On selecting the location and date for a viewing session, users can select the 'The Sky this Week' Option. This brings the user to the RSS Feed. In order to get detailed information, the user will then be taken to the provider's site.

Technical Issues

Internet access required so the feed is up to date. For information about what is in the sky this week the user will be diverted from the application.

Risks

Dependent on a third party to maintain the RSS feed.

Dependencies with Other Requirements

None.

Functional Requirement 5 – Writing and Reading from SQLite Database

Description and Priority

User should be able to log details about their viewing sessions so they can keep a record of their activities. PRIORITY: MEDIUM

Technical Issues

Database is stored locally on device. In exceptional circumstances this may limit the amount of sessions which can be stored or effect device memory. The latter being extremely unlikely due to the nature of data logged.

Risks

When device expires, the data logged will also expire and be lost.

Dependencies with Other Requirements

None.

3.1.2 User requirements

The user will need to have a device that uses Google's Android operating system.

The device is targeted at the Android 21 Lollipop release but is backwards compatible to Android15 Ice Cream Sandwich.

The user will need to have an internet capable device in order to use the service. The user will need to have geolocation enabled on their device to use weather and mapping services. Geolocation is not required for the astronomy services.

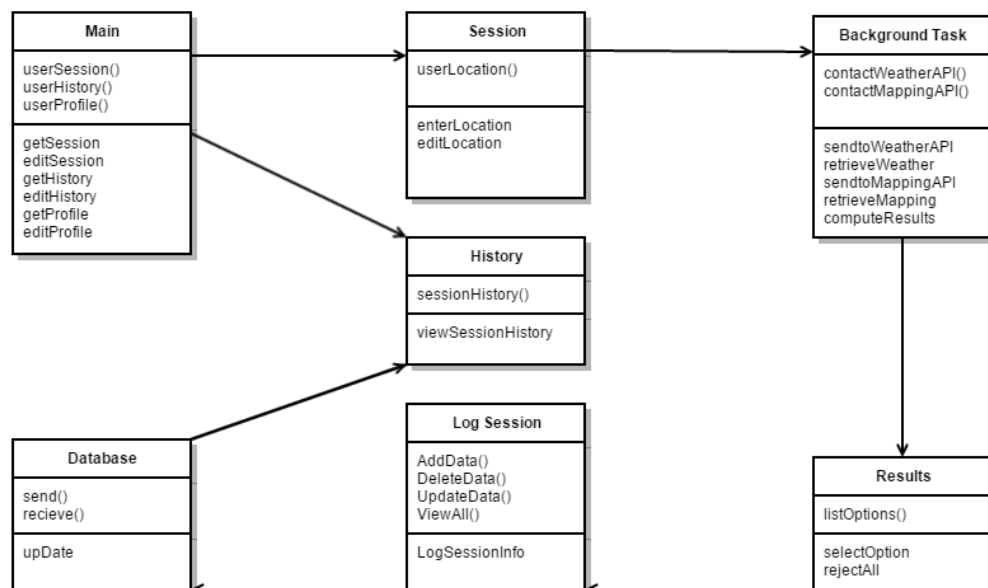
In order to use the compass function on the application the user's device must have an orientation sensor, this is built into the hardware of devices and cannot be downloaded if it is not included in the original manufacture.

3.1.3 Environment Requirements

In order to develop the application, an internet connected laptop or PC is required with sufficient memory to run Android Studio. I have used a Dell Inspiron 5000 series laptop. To test the application and its component parts, it was necessary to connect an android enabled device to the laptop, the device I use is a Huawei Y6. To test the application's adaptiveness to various devices, I have also tested on a Samsung Galaxy device.

3.2 Design and Architecture

From an architectural perspective it is important that the application is as lightweight as possible. In order to achieve this a version of the MVC design pattern has been utilised. The model is the part of the system which performs calculations and computes algorithms in order to meet user requests. The view is the interface that is presented to the user. It comprises of layouts, buttons, text, images and fragments. The controller determines how the model is presented to the user. It performs the function of connecting the model with the view.

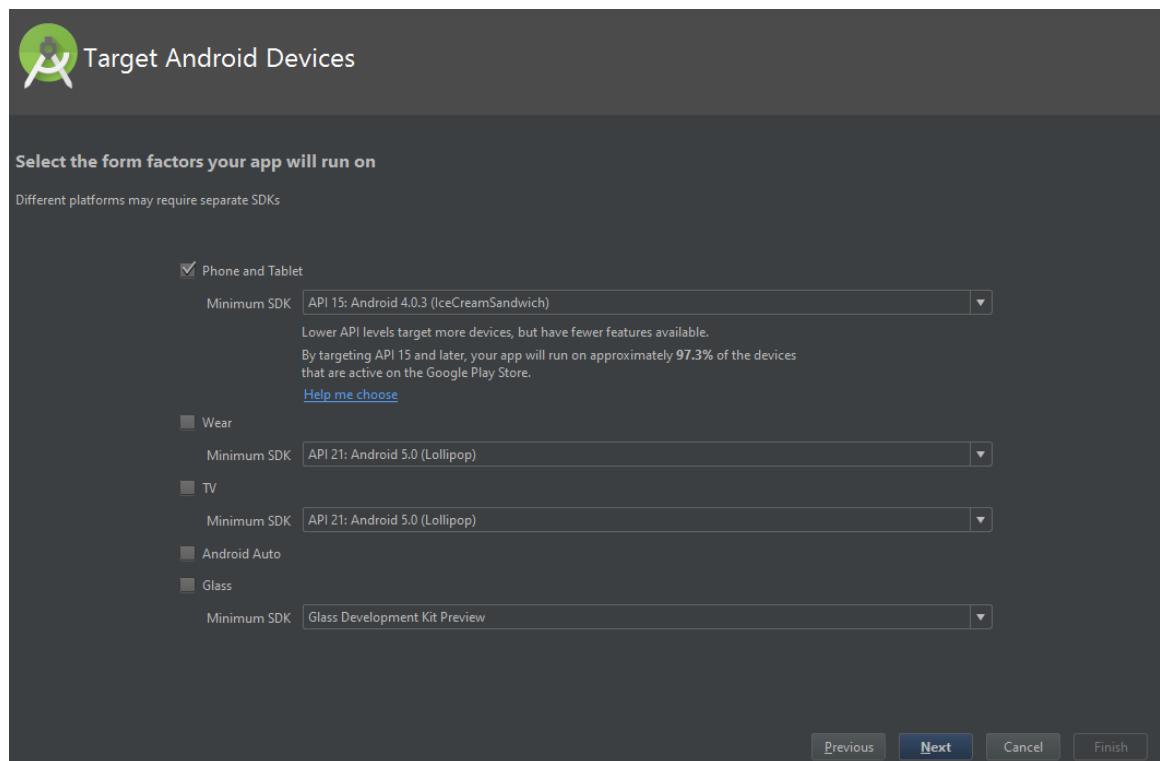


3.3 Implementation

In order to implement the application as described, each function was built and tested in isolation before being integrated and re-tested with the rest of the application. Below I will go through the process of developing one of these functions, the same process can loosely be applied to others. I will also go into detail on each of the key functions of the application.

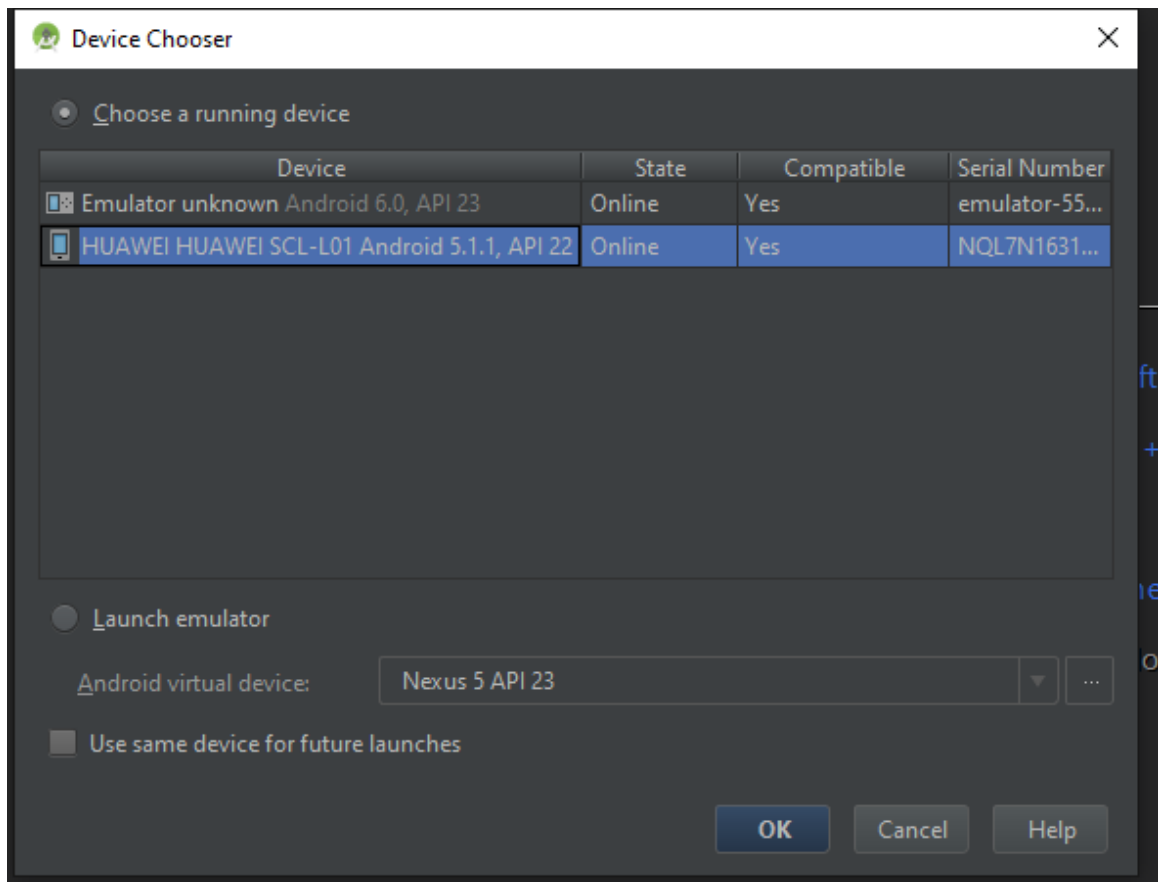
3.3.1 Component Building

To test a component, a new project first needed to be created in Android Studio. To ensure there were no compatibility issues with the rest of the project, I used the same target (21, Lollipop) and minimum SDK (15, Ice Cream Sandwich) in each instance.

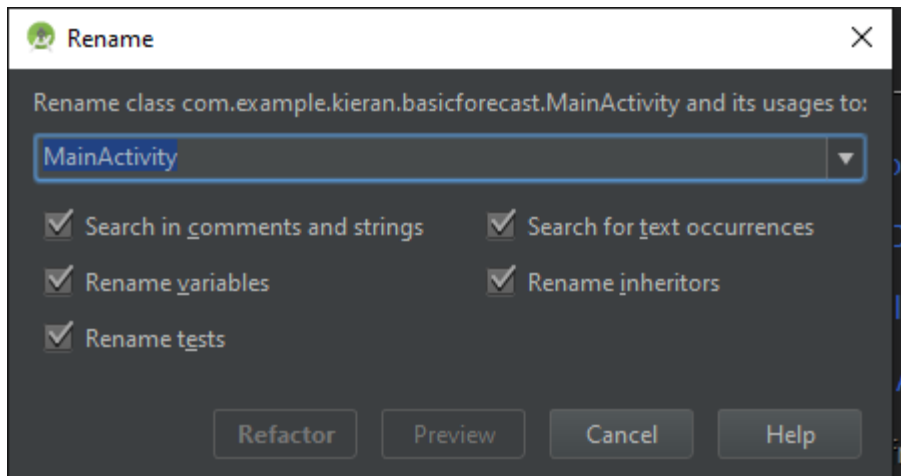


I would then select the type of activity. Generally this would be blank as this option provided me with a content and activity layout for each activity class making the user interface easier to design. For screens which included a google map the Google Maps Activity option was selected.

Once the code was completed, I ran the application on my Huawei Y6 device to check if functionality was as expected.



On confirming the successful operation of the component, I then set about integrating it into the main application. To do this I first ensured that the main project was backed up in the event that the new code caused an issue that was difficult to undo. I then created a new Activity in the main application and named it accordingly. I then copied the code directly from the component project and refactored class and object names where necessary. This ensures that all usages of the class or object are also changed.



3.3.2 Integrating the Open Weather Map API

In the final application three different API calls are used from the Open Weather Map service. They are the current weather for the current location, a five day weather forecast for the current location and the current weather for up to ten locations within a 1 degree of latitude and longitude block around the current location. As the code for the three is roughly similar I will only detail one here, namely the block around the current location.

In order to parse the JSON file, I created a public class which extended AsyncTask. By doing this the computation takes place in a background thread and does not hold up the onCreate() method which displays the user interface.

```
String finalJson = buffer.toString();
JSONObject parentObject = new JSONObject(finalJson);
JSONArray parentArray = parentObject.getJSONArray("list");

StringBuffer finalBufferedData = new StringBuffer();
for (int i = 0; i <= 10; i++) {
    JSONObject finalObject = parentArray.getJSONObject(i);
    String name = finalObject.getString("name");

    int date = Integer.parseInt(finalObject.getString("dt"));

    String format = "dd/MM: HH:mm";
    String fcTime = new java.text.SimpleDateFormat(format).format(new java.util.Date((long) (date * 1000L)));
```

The above code shows the passing of the list array into a JSONArray called parentArray. Doing this allowed me to iterate through the list array in the JSON file to retrieve the relevant data. As the code shows "name" and "dt" are children of the list array. As date "dt" was returned in Unix Epoch time it was necessary to apply the algorithm displayed in order to convert this into a readable date and time.

Not shown above is the retrieval of the current weather description i.e. Cloudy, Clear Skies etc. This was inside a nested array titled “main” and required a nested for loop to retrieve. All variables were then added to a string buffer so they could be displayed in the UI.

The API call itself required five parameters, two lines of latitude, two of longitude and the number of expected results. The code I wrote to achieve this is below.

```
final Double lat = location.getLatitude();
final Double lng = location.getLongitude();
final Double latNorth = lat +1;
final Double latSouth = lat -1;
final Double lngEast = lng +1;
final Double lngWest = lng -1;

new ForecastTask().execute("http://api.openweathermap.org/data/2.5/box/city?bbox=" +String.valueOf(lngWest)+"," +
    ""+String.valueOf(latSouth)+","+String.valueOf(lngEast)+","+String.valueOf(latNorth)+"," +
    "0&cluster=yes&appid=3afc9e467f02bfff96343d526af378c19");
```

In order to make this API call respond to the user’s current location I used the Location Manager. On retrieving the current latitude and longitude I then set about creating the four lines that would create the box around the current location. By creating four new variables from the current latitude and longitude and subtracting ‘1’ and adding ‘1’ to each as appropriate, then inserting these variables into the API call, I ensured that the data returned is always updated to match the users current location. For the purposes of conciseness I restricted the amount of results returned to less than ten. The results returned were then passed to the Views created in the layout files in order to be displayed to the user.

3.3.3 Integrating Google Maps API

Due to the highly advanced and user friendly nature of Google Maps API this was a relatively straightforward exercise. Though some innovative code was required to ensure that both the Google Maps and Open Weather Map API could make calls and return data based on the same user input.

There are two different formats of map used in the application. One is for the user’s current location and the other enables the user to search locations and divert to Google’s directions and satellite navigation service.

In the second of the two, the map is initialised to show a global view until the user enters a location they wish to see. The map is then created in the onSearch() function in order to display this.

```

public void onSearch(View view){

    EditText locationTxt = (EditText)findViewById(R.id.locationTxt);
    String location = locationTxt.getText().toString();
    List<Address> addressList = null;

    if (location != null || !location.equals("")){
        Geocoder geocoder = new Geocoder(this);
        try {
            addressList = geocoder.getFromLocationName(location, 1);
        } catch (IOException e) {
            e.printStackTrace();
        }

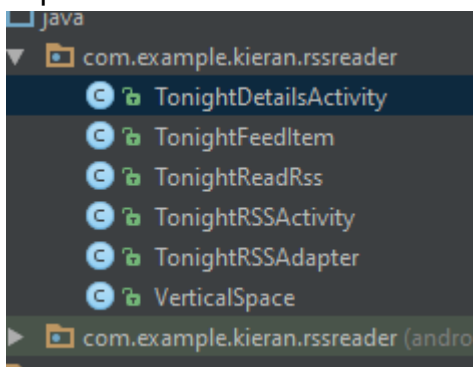
        Address address = addressList.get(0);
        LatLng latLng = new LatLng(address.getLatitude(), address.getLongitude());
        mMap.addMarker(new MarkerOptions().position(latLng).title("marker"));
        mMap.animateCamera(CameraUpdateFactory.newLatLng(latLng));
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(latLng, 11));
    }
    cityTextView = (TextView) findViewById(R.id.cityTextView);
}

```

As with the weather API calls the location manager is also utilised with google maps.

3.3.4 Integrating Astronomy RSS Feeds

Two RSS feeds were integrated into the application. As the process for both were roughly similar I will only describe one here. The RSS feed implementation utilises the MVC pattern more than most other functions due to the relatively large amount of background tasks being performed. Below is a list of the classes implemented.



The TonightRSSActivity loads the content view. The Feed Item is where the getters and setters for each tag that we wish to retrieve are set. The ReadRss class is where the feed is called. As shown below it is also where the RecyclerView is instantiated.

```
public class TonightReadRss extends AsyncTask<Void, Void, Void> {
    Context context;
    ProgressDialog progressDialog;
    String rssAddress = "http://www.skyandtelescope.com/observing/sky-at-a-glance/feed/";
    URL url;
    ArrayList<TonightFeedItem> tonightFeedItems;
    RecyclerView recyclerView;
```

The Adapter is the link between the actual data and the Views which it will populate. The VerticalSpace class controls how the feed will appear inside the app while the DetailsActivity implements the web view which enable users to connect to the RSS provider's website to read the full story or post.

To display the RSS feed in an aesthetically pleasing, user friendly manner the below libraries were included in the gradle build.

```
compile 'com.android.support:recyclerview-v7:23.2.1'
compile 'com.android.support:cardview-v7:23.2.1'
compile 'com.squareup.picasso:picasso:2.5.2'
```

Recycler View and Card View enable the user to scroll down a list of "cards" all following the same format. Picasso enabled me to include images and animations in these views.

3.3.5 Connecting the SQLite Database

The database was implemented using Android Studios built in SQLite database and as such data is saved locally to the host device. Two classes were created in order to implement the database. The main activity class contains the onCreate() method as well as all of the methods which need to be performed on the database such as AddData(), deleteData(), upDateData() and ViewAll(). The database helper class is where the database is created. Methods are created here, the results of which can then be passed to the methods outlined above. The code below shows the declaration of the tables and columns of the database used.

```
public class DatabaseHelper extends SQLiteOpenHelper {

    public static final String DATABASE_NAME = "observing.db";
    public static final String TABLE_NAME = "observing_table";
    public static final String COL_1 = "ID";
    public static final String COL_2 = "NAME";
    public static final String COL_3 = "NOTE";
    public static final String COL_4 = "LOCATION";
```

3.3.6 Connecting Screens

All screens are accessed via buttons and the app follows a distinctive hierarchy. Within each button an Intent is created which directs to the appropriate activity.

```
skyTonightButton.setOnClickListener((v) → {  
    Intent iSkyTonight = new Intent(PlanSessionActivity.this, TonightRSSActivity.class);  
    startActivity(iSkyTonight);  
});
```

3.3.7 Compass

The application contains a compass which turns the device's main screen into a standard compass with a rotating dial with pointers for north, east, south and west. Below is the code which uses SensorEventListener to determine what direction the device is facing and animates the image of the compass dial accordingly. It also displays the degree heading the user is facing via a text view.

```
@Override  
public void onSensorChanged(SensorEvent event) {  
  
    // get the angle around the z-axis rotated  
    float degree = Math.round(event.values[0]);  
  
    tvHeading.setText("Heading: " + Float.toString(degree) + " degrees");  
  
    // create a rotation animation (reverse turn degree degrees)  
    RotateAnimation ra = new RotateAnimation(  
        currentDegree,  
        -degree,  
        Animation.RELATIVE_TO_SELF, 0.5f,  
        Animation.RELATIVE_TO_SELF,  
        0.5f);  
  
    // how long the animation will take place  
    ra.setDuration(210);  
  
    // set the animation after the end of the reservation status  
    ra.setFillAfter(true);  
  
    // Start the animation  
    image.startAnimation(ra);  
    currentDegree = -degree;  
}
```

3.3.8 Other Functions

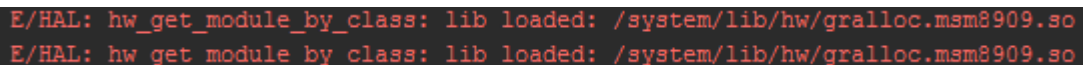
The sky chart in the application has pan and zoom functionality added. This was done using Android gesture recognition library. There are also static pages containing relatively constant astronomy related data such as planetary distances and a list of the brightest objects in the sky. These were created using Image and text views.

3.4 Testing

There were three distinct stages in the testing process. Each component or feature of the application was tested in isolation before being integrated into the main system. System stress testing was carried out using Android's Exerciser Monkey. The whole system was tested both by unscripted tests where testers could attempt to break the application by revealing thus far unseen bugs and scripted testing which checked the functioning of the entire system.

3.4.1 Unit Testing

At the end of writing the code, I ran each component on my Huawei Y6 device while connected to the laptop. This enabled me to monitor how the user interface responded to user actions while also seeing any errors or warnings being produced in the CatLog. One repeated warning which I received on running the application was regarding the gralloc files. See screenshot below:



```
E/HAL: hw_get_module_by_class: lib loaded: /system/lib/hw/gralloc.msm8909.so
E/HAL: hw_get_module_by_class: lib loaded: /system/lib/hw/gralloc.msm8909.so
```

Though I am unable to find the cause of this online research indicates that this is a known bug in Android Studio when working with particular SDK's. As it has no operational effect on the running of the application, it is not a cause for concern.

Below are some of the component tests which I carried out. I have not included them all as there were many where the component being tested did not make the final application, either due to failing the testing process or design decisions taken. Also, some of the components are relatively similar and so follow a similar testing script to others. Readers will notice that the scripts do not follow a pass-fail format, this was deliberate as I used these tests to observe how each component behaved so as to inform the development process.

StarGaze Component Testing	
Component: View Now Activity	
Dependencies: Google Maps and Open Weather Map API	
Tester Kieran Shine	
Action	Result
Open Activity	Layout displays in line with expectations
Time to execute:	< 2 seconds - Acceptable
Is map showing correct current location?	Map shows current user location to a high degree of accuracy
Is weather forecast showing correct dates?	Dates listed in forecast are correct.
Is weather forecast in line with forecasts from other services?	Weather checked against Yahoo weather API. Forecast is roughly similar, minor differences to be expected.
List warnings and errors from CatLog.	Gralloc

StarGaze Component Testing	
Component: Latest News Activity	
Dependencies: Science Daily News RSS Feed	
Tester: Kieran Shine	
Action	Result
Open Activity	Layout displays with Recycler View
Time to execute:	< 2 seconds - Acceptable
Are items clickable?	Clicking item diverts to providers website.
Is the list scrollable?	List scrolls through historic stories to at least three weeks of news. This amounts to approximately 40 stories available to read.
Does back button return user to application	Back button returns user to main app
Does RSS feed update to reflect content on providers website	Delay between content providers site and the RSS feed is approximately one hour. This is outside of the control of the application
List warnings and errors from CatLog.	Gralloc

StarGaze Component Testing	
Component: Database Testing	
Dependencies: SQLite	
Tester: Kieran Shine	
Action	Result
Open Activity	Layout displays as expected
Time to execute:	< 1 second - Excellent
Entering Data	All fields accept data as expected an in line with application needs.
Add Button	Adds Data Entered to the database and displays confirmation.
Update Button	Once session ID is entered that session is updated.
Delete Button	Once session ID is entered session is deleted.
View All Button	Displays list of all sessions logged.
Test Database Capacity	100 sessions logged. This is far above expected usage.
List warnings and errors from CatLog.	Gralloc

3.4.2 System Testing

System testing was carried out using Android's built in Exerciser Monkey. This carries out a stress test on the system by issuing pseudo-random streams of user events. The amount of commands input is specified by the developer, I instructed the Monkey to run one thousand commands. As well as seeing the commands executed in the command prompt I was able to watch the Monkey manipulate the UI to see what areas it was testing at a particular moment. This test was used at various intervals in the applications development. In the final test the application passed as the stress test did not cause the application to stop at any stage. Below I have included screenshots of the initial command to launch the Exerciser Monkey and the results of the test. The entire test was too large to print in its entirety here.

```

C:\Users\Kieran\AppData\Local\Android\sdk\platform-tools>adb shell monkey -p com.example.kieran.spacebook -v 1000
Monkey: seed=1463078916343 count=1000
:AllowPackage: com.example.kieran.spacebook
:IncludeCategory: android.intent.category.LAUNCHER
:IncludeCategory: android.intent.category.MONKEY
// Event percentages:
// 0: 15.0%
// 1: 10.0%
// 2: 2.0%
// 3: 15.0%
// 4: -0.0%
// 5: 25.0%
// 6: 15.0%
// 7: 2.0%
// 8: 2.0%
// 9: 1.0%
// 10: 13.0%
:Switch: #Intent;action=android.intent.action.MAIN;category=android.intent.category.LAUNCHER;launchFlags=0x10200000;component=com.example.kieran.spacebook/.MainActivity;end

:Sending Trackball (ACTION_MOVE): 0:(2.0,-1.0)
:Sending Touch (ACTION_DOWN): 0:(687.0,284.0)
:Sending Touch (ACTION_UP): 0:(720.0,229.71046)
// Rejecting start of Intent { act=android.intent.action.VIEW cat=[android.intent.category.BROWSABLE] dat=http:xxxxx
.xxxxxxxxxxxxxxxxxx.cmp=com.huawei.android.internal.app/.HwResolverActivity } in package com.huawei.android.internal.a
pp
// Rejecting start of Intent { act=android.intent.action.VIEW cat=[android.intent.category.BROWSABLE] dat=http:xxxxx
.xxxxxxxxxxxxxxxxxx.cmp=com.huawei.android.internal.app/.HwResolverActivity } in package com.huawei.android.internal.a
pp
:Sending Touch (ACTION_DOWN): 0:(67.0,432.0)
:Sending Touch (ACTION_UP): 0:(117.415085,482.47775)
:Sending Trackball (ACTION_MOVE): 0:(0.0,-1.0)
:Sending Trackball (ACTION_MOVE): 0:(4.0,0.0)
// activityResuming(com.example.kieran.spacebook)
Events injected: 1000
:Sending rotation degree=0, persist=false
:Dropped: keys=0 pointers=4 trackballs=0 flips=0 rotations=0
## Network stats: elapsed time=6358ms (0ms mobile, 0ms wifi, 6358ms not connected)
// Monkey finished

```

3.4.3 User Acceptance Testing

For the user acceptance testing, I created scripts based on a pass-fail structure and distributed these to several testers at each release of the application. Below are two such scripts. The second is after final release. Both tests were carried out by the end users.

Final StarGaze Version Full				
UAT Test		Date: 07/05/16		Tester: Orla McCarthy
Test Case Description	Expected Result	Actual Result	Passed Y/N	Comments
App installation & launch				
Install application on device	App launcher icon displays on app screen.		Y	
Launch App	App home screen appears.		Y	Nothing to suggest it's a Home Screen? Could there be a Home Screen at bottom of each page - rather than pressing back a few times to get to 1st page? View now isn't intuitive - had to go into the button to remember what was going to appear.
Plan Session Test				
Select Plan Session option	Screen opens.		Y	Not sure if "this week in the sky" fits here?? Or Weather Nearby as a location hasn't been selected yet. Perhaps Weather Nearby could go into view now instead of this week in the sky which could go into What's to see? "This week in the sky" and then when you go in wording is The Sky this week"...
Search for location	Map zooms to selected location and Weather data for this location displays below.			Test Kuala Lumpur.....not sure if what's happening is correct. Passed for Limerick but not for Norwood Park.
Select Marker	Google Maps and Directions options appear.		Y	
Select Google directions	App routes to Google directions app.		Y	Why does Cor Mariae appear when I click on the directions icon?
Select 'Sky This Week Button'	Sky this Week page opens.		Y	
Select Nearby Weather Option	Screen opens displaying map and weather forecast for 10 nearby locations.	Shows 9 locations. Map does not show current location but a general map.	N	"Weather Nearby" not Nearby Weather" - change test script?
Tap 'Weather in the Sky Tonight'	RSS Feed from Sky and Telescope opens.	Yes this feed opens but button appearing to press is "This week in the sky". I don't know the RSS Feed is from Sky and Telescope until I perform the next test.	N	
Tap a segment of RSS feed	App routes to provider website.		Y	

Test Script continued on next page.

View Now				
Select View Now	Screen opens. Displays current location on map and current weather data.		Y	Title View Now not displayed at top like the others (Nor is latest news but that may be because it's an RSS Feed?). A bit messy...6 today's forecasts and 5 day forecasts - not aligned.
Select Marker	Google Maps and Directions options appear.		Y	
Select Google directions	App diverts to Google directions app.		Y	Why does it choose the location for me?
Select 'Sky This Week Button'	Sky this Week page opens.		Y	Need to go back twice before getting to sky this week option.
What's To See				
Tap Button	Four options displayed. Solar System, Bright Stars, Constellations, Compass.		Y	Writing in Green should say "What's To See"
Tap Solar System	List of Planet Buttons displayed.		Y	Should be gap between Solar and System
Tap each planet.	Data relating to that planet is displayed.		Y	Average distance from earth - some state AU - Others don't show AU. What is AU? For MARS - Put a dash rather than comma after "two". What is the magnitude symbol? Re-look at punctuation e.g. full stops. Uranus - in late.
Tap Constellations	Instructions on how to use chart are displayed along with button to access chart.		Y	
Tap Star Chart Button	Star chart is displayed with Pan and Zoom functionality.		Y	
Tap compass	Responsive compass is displayed.		Y	
Log Session				
Tap Log Session Button	Log screen appears with fields for target, Notes and Location. Four buttons appear on screen - Add, Delete, Update and View all.		N	Session log or log session? Field for ID also. The 4 buttons are log session, delete, update and view. What is meant by "Target"?
Enter Session details and tap Add session	Message displays that data has been added.		Y	Says Data Inserted once I click Log Session.
Enter new session details but include ID of previous log click update	Message displays that data has been updated.		Y	
Enter ID and tap delete	Message displays that data has been deleted.		??	id 3124
Tap View All Button	View of logged sessions is displayed that reflects the previous actions.		N	I had updated session from Mars to Mars2 but not appearing.
Latest News				
Tap Latest news	Screen displays showing latest news RSS feed including thumbnail image and summary of each story.		Y	
Tap story	Diverts to provider website for full story.		Y	
Additional Observations				

Script 2

Final StarGaze Version Full				
UAT Test		Date: 08/05/2016		Tester: Jonathon Sweeney
Test Case Description	Expected Result	Actual Result	Passed Y/N	Comments
App installation & launch				
Install application on device	App launcher icon displays on app screen.	Correct	Y	
Launch App	App home screen appears.	Correct	Y	List of options
Plan Session Test				
Select Plan Session option	Screen opens.	Correct	Y	Opens immediately. Takes about a second for weather data to load
Search for location	Map zooms to selected location and Weather data for this location displays below.	Correct	Y	
Select Marker	Google Maps and Directions options appear.	Correct	Y	
Select Google directions	App routes to Google directions app.	Correct	Y	
Select 'The Sky This Week'	Sky this Week page opens.	Correct	Y	
Select Nearby Weather Option	Screen opens displaying map and weather forecast for 10 nearby locations.	Correct	Y	Takes approximately 2 Seconds for weather data to load
Tap 'Weather in the Sky Tonight'	RSS Feed from Sky and Telescope opens	Correct	Y	
Tap a segment of RSS feed	App routes to provider website.	Correct	Y	Have to scroll down to actual info

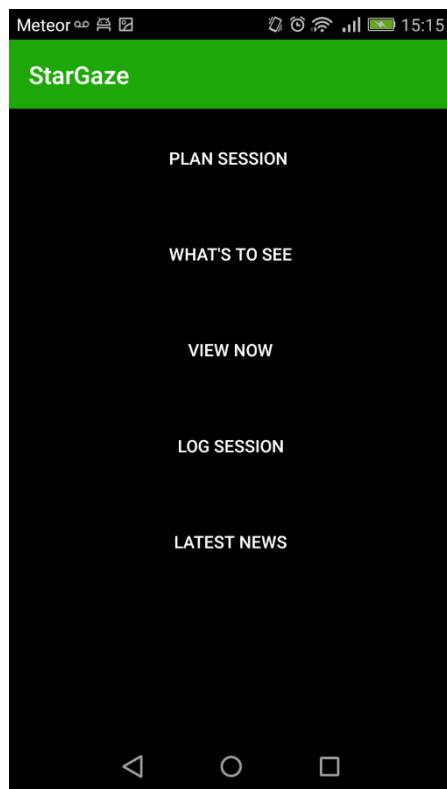
Script continued on next page.

View Now				
Select View Now	Screen opens. Displays current location on map and current weather data.	Correct	Y	Delay of up to 5 seconds in weather data loading
Select Marker	Google Maps and Directions options appear.	Correct	Y	
Select Google directions	App diverts to Google directions app.	Correct	Y	
Select 'The Sky This Week' Button	Sky this Week page opens.	Correct	Y	
What's To See				
Tap Button	Four options displayed. Solar System, Bright Stars, Constellations, Compass.	Correct	Y	
Tap Solar System	List of Planet Buttons displayed.	Correct	Y	
Tap each planet.	Data relating to that planet is displayed.	Correct	Y	
Tap Constellations	Instructions on how to use chart are displayed along with button to access chart.	Correct	Y	
Tap Star Chart Button	Star chart is displayed with Pan and Zoom functionality.	Correct	Y	
Tap compass	Responsive compass is displayed.	Correct	Y	
Log Session				
Tap Log Session Button	target, Notes and Location. Four buttons appear on screen - Log Session, Delete Session, Update Session and View all	Correct	Y	
Enter Session details and tap Add session	Message displays that data has been added.	Correct	Y	
Enter new session details but include Id of previous log click update	Message displays that session has been updated.	Correct	Y	
Enter ID and tap delete session	Message displays that session has been deleted	Correct	Y	
Tap View All Button	View of logged sessions is displayed that reflects the previous actions.	Correct	Y	
Latest News				
Tap Latest news	Screen displays showing latest news RSS feed including thumbnail image and summary of each story.	Correct	Y	
Tap story	Diverts to provider website for full story.	Correct	Y	
Additional Observations				

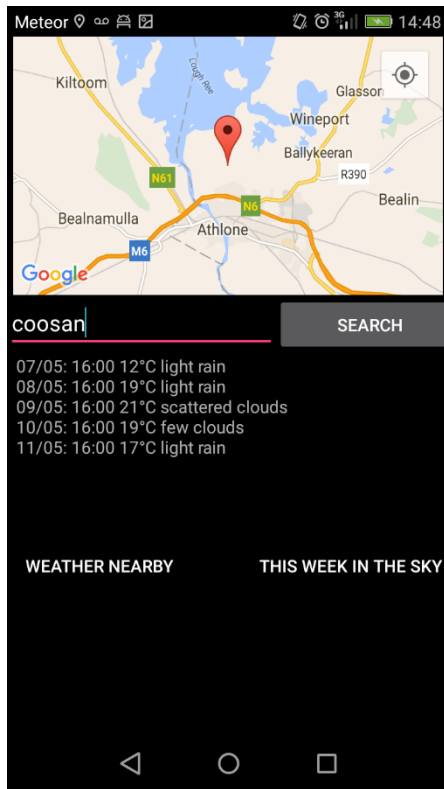
Many of the issues raised in the first test script were addressed before running the second script. Others were misinterpreted features of the system which I have attempted to make more intuitive as a result of the script. One issue raised in the first script related to the weather information not appearing for Kuala Lumpur. This is because Open Weather Map consider that term too vague to give a forecast. Searches for areas within Kuala Lumpur will return weather data. Another issue raised, concerned an area being identified on Google directions when the user requested directions to Limerick, converse to the issue with the Weather API, Google reduced the term Limerick to a very specific point within Limerick so that directions could be compiled. Both of these issues/features are outside of the scope of the application as they are ultimately controlled by third parties.

3.5 Graphical User Interface (GUI) Layout

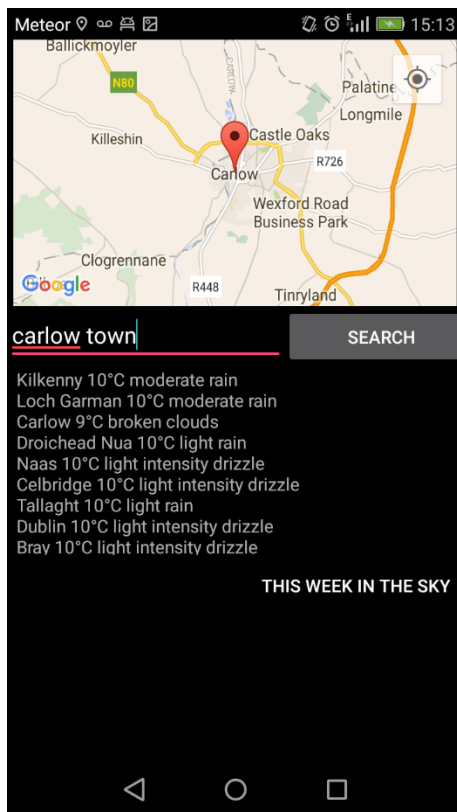
While designing the GUI, I attempted to make it as uncluttered and intuitive as possible. Many of the astronomy applications I have used in the past, while being very useful, were difficult to navigate and required the average user to have a far greater understanding of scientific terminology than I believe is realistic. By simplifying the interface, the application will have a much broader appeal.



Upon launching the application the user will be shown a screen with a list of five options. This can be considered the top of the “tree” under which all other activities are children and grandchildren.



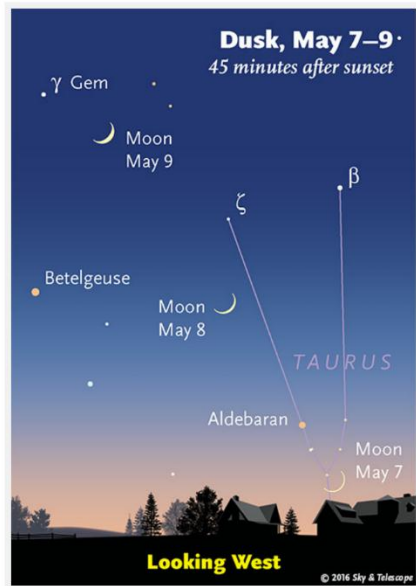
This screenshot is the Plan Session activity. Here the user is presented with a map of the world. It is only upon entering a location and tapping the search button that the map will zoom to that location, and a five day forecast for that location is shown below. By selecting the marker on the map the user then has the option to get directions to the location. On clicking the weather nearby button the below screen is initialised.



This is the Weather Nearby activity. This screen shows a list of 9 locations within approximately 50 - 100km of the user's current location, depending on their latitude. In the example shown Carlow has the most suitable conditions for viewing so I have then searched for Carlow on the map. The user can now proceed to get directions and satellite navigation if they will. The 'This Week in the Sky' button is also available here.

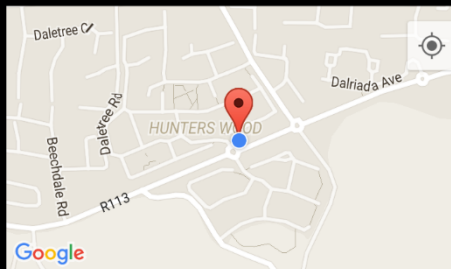
This Week's Sky at a Glance, May 6 – 14

By: Alan MacRobert | May 6, 2016



Can you catch the young Moon on Saturday evening the

This screen is from the Sky & Telescope web app that is linked via the RSS feed. As well as graphical representations of certain objects in the sky, users can scroll down to get specific information regarding each night of the current week. Previous weeks are also available via the RSS feed.



Woodtown 10°C light rain

Sunrise: 05:38:05

Sunset: 21:06:36

Today's Forecast

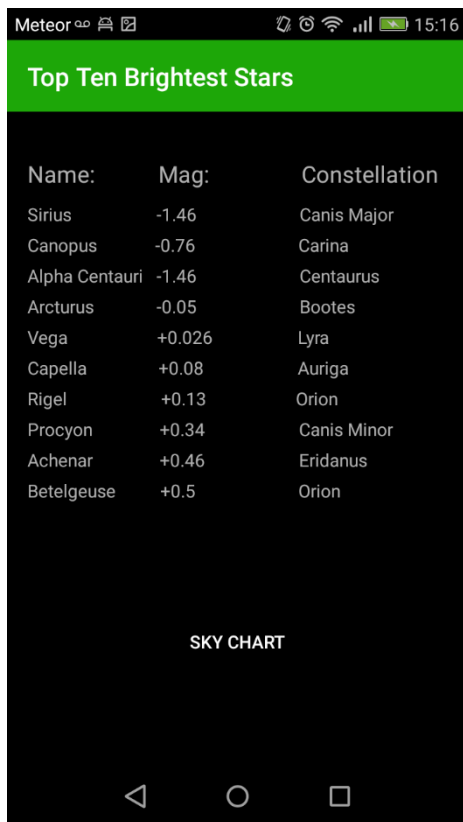
16:00 14°C light rain
19:00 13°C light rain
22:00 11°C light rain
01:00 10°C light rain
04:00 11°C light rain
07:00 11°C light rain

Five Day Forecast

07/05: 14°C light rain
08/05: 19°C scattered clouds
09/05: 20°C few clouds
10/05: 18°C light rain
11/05: 15°C light rain

THIS WEEK IN THE SKY

This screen shows the view now activity. It shows the user their current location along with text giving location, weather conditions, temperature, sunrise and sunset times. Below this there is a daily forecast which gives the weather for the next 18 hours. There is also five day forecast for the location alongside this. The 'This Week in the Sky' Button is also available here.



The 'What's To See' button gives users a list of options similar to the home screen. Because of this I have not shown a screenshot. The options presented here are Solar System, Bright Stars, Constellations and Compass.

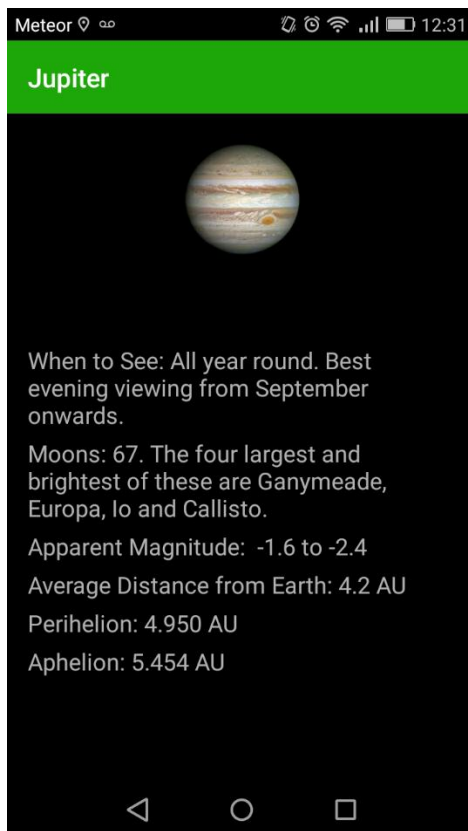
Here is the bright stars screen which shows users a list of the 10 brightest stars in the sky along with their apparent magnitude and the constellation which they can be found in. The 'Sky Chart' button below will display a sky chart so they can locate these constellations. This chart is also accessible via the constellations option.



This is a zoomed in image of the sky chart which depicts all of the major constellations in the night sky. The image is both zoom-able and pan-able and contains instructions upon opening regarding how to use the map in order to locate objects.



This is the compass activity which turns the screen into a responsive compass. The back drop is pink as this colour is much easier on the eyes. When viewing during an astronomy session it is important that your eyes remain adapted to the dark so that as much detail as possible can be seen in the sky.



This screen shows an example of the information available on each of the solar systems planets except Earth. The screen shown is Jupiter.

The information given about each planet is relevant for astronomy purposes when planning to view an object. It was important when creating these screens that the information given remained relevant and not stray into an 'interesting facts' page. All of the information given is useful to the would-be astronomer.

4 Obstacles Encountered

While initially the most significant obstacle was my own lack of technical knowledge of the various technologies required to build the application I would like to think this was in the most part overcome. However there were other unforeseen obstacles which proved difficult to overcome.

The primary difficulty I encountered was shortly after beginning the project when google deprecated their Sky API. It had been my intention to integrate that into the application as the primary source of astronomical information. While at first I was relatively unalarmed as I felt there was plenty of time to find a replacement I soon discovered that there were not any such replacements to be found. While the deprecated API was released as open source the lack of support provided for it meant that I could not depend on it in such a core role within my application. After trying numerous other alternatives I eventually settled on using a RSS feed from SkyandTelescope.com combined with my own hard coded astronomical information. While this meant that users would have to navigate away from the application for up to date specific astronomy information, Sky and Telescope is a very well established and respected source of astronomy information and I felt comfortable depending on it.

The next major obstacle I encountered relates directly to that mentioned above. While the Astronomy news RSS feed I implemented gives users a brief synopsis of each story before they decide if they want to navigate to the providers page, the 'The Sky this Week' RSS feed gives no summary and so users will have to navigate away from the site to read specifics.

5 Future Improvements

The application has many areas for potential improvement and it is my intention to implement many of these and further develop them in the summer months. Among these was one of the original features I had hoped to include in the application, namely a luminosity feature. Astronomy is a much more fruitful exercise when carried out away from bright lights and hence urban areas. I hoped to implement this by judging the approximate luminosity of an area based on its population. In practice however I could only find API's that gave the population of entire countries and did not break these down regionally. However population data is very accurate in the developed world and I feel that finding the correct source from which to build my own API may be the solution.

I would also like to implement a responsive sky map. One such map is available through the deprecated Google Sky API mentioned previously but a great deal of work is necessary to implement this and though I began this process I soon realised that I would not have sufficient time to implement it. Other less difficult improvements would include multiple languages and an option to select your hemisphere, the current application's astronomy features are tailored mainly towards the northern hemisphere.

6 Conclusions

In conclusion, I am proud of the application I have developed and I am confident that it will find a niche in the current app market place. I set out to provide astronomers with a tool which could be used to locate places where viewing conditions would be optimal, give directions on how to get there and information on what they can expect to see. The final product achieves all of these goals.

The functioning of the application depends on four external resources (Google Maps, Open Weather Map, Sky & Telescope, and Science Daily News). These are all well established and reliable services and I consider their integration a strength of the application.

The process of developing the application has been an extremely steep but equally enjoyable learning curve and the journey has been a valuable experience. Along with my own technical knowledge the application has evolved and grown over time.

As stated at the outset, it was important to me when developing the concept of this application that it would be an educational and useful tool which would help people to learn about the night sky and fuel their curiosity. I'm happy to have achieved this and believe the application will be an asset to those who wish to learn and discover. Though the process of improving the application will continue into the future, the current version meets all of the stated goals.

7 References

Angelov, Martin - How to use Geolocation and Yahoo's APIs to build a simple weather webapp. Retrieved February 3, 2016, from <http://tutorialzine.com/2012/05/weather-forecast-geolocation-jquery/>

Dalisay, M. Android Compass Code Example. Retrieved April 4, 2015, from <https://www.javacodegeeks.com/2013/09/android-compass-code-example.html>

Google Developers - Accessing Google APIs. Retrieved December 5, 2015, from <https://developers.google.com/android/guides/api-client>

Marcenelle, N. (2016). How to Make an Android Weather App. Retrieved January 13, 2016, from https://www.youtube.com/watch?v=oszmk0e_i18

Multi-Touch Panning & Pinch Zoom Image View in Android Using Android Studio. Retrieved April 5, 2016, from <http://www.c-sharpcorner.com/uploadfile/88b6e5/multi-touch-panning-pinch-zoom-image-view-in-android-using/>

Jones, Dave - Retrieved February 3, 2016, from <https://www.youtube.com/channel/ucdjn-70dpkdjem1ellssaew>
Ruby on Rails Tutorial Series

OpenWeatherMap Current Weather and Forecast. Retrieved November 3, 2015, from <http://openweathermap.org/Mapping-API>

Science Daily. Retrieved March 8, 2016, from <https://www.sciencedaily.com/newsfeeds.htm>

Sky & Telescope. Subscribing to Sky & Telescope's RSS Feeds - Sky & Telescope. Retrieved February 5, 2016, from <http://www.skyandtelescope.com/subscribing-sky-telescope-rss-feeds/>

Star Chart. Retrieved April 2, 2016, from <http://www.aurorahunter.com/starchart.php>

Tech Academy. (2015, May). Google Maps Tutorial. Retrieved December 2, 2015, from <https://www.youtube.com/watch?v=nhxa96-r8ty>

Yadav, A. Amar Yadav - Android. Retrieved February 10, 2016, from <http://www.amaryadav.in/android.html>

8 Appendix

8.1 Project Proposal

Project Proposal
StarGaze
Kieran Shine, 12113239, shinekieran@gmail.com
BSc (Hons) in Computing
Networking and Mobile Technologies
01/10/2015

1. Objectives

The primary objective of this project is to create a mobile application which will act as an aide to amateur astronomers wishing to plan their viewing sessions in advance.

By matching weather data with a mapping API users will be able to receive information advising them of the best options available within a time period specified. The following features will be available:

- A history of previous viewing sessions so users will be able to record information about their sessions. This information may include generated data such as the date, time and location along with user inputted data such as what they observed etc.
- When advising a location for observing sessions the app will provide directions on how to get there including route, distance and time.
- The application will also advise users of the weather at these locations.
- The application should be able to provide information on what objects may be observed on a given night and what times are optimal for viewing particular objects.

2. Background

I have been very interested in all things science but particularly in astronomy for many years. During this time I have always found it difficult to get accurate, astronomy specific forecasts regarding cloud cover and air temperature. Though weather forecasts are easy to come by they rarely focus on these factors which are of vital importance to astronomers. Forecasts such as the likelihood of rain are not much use for astronomy purposes as while this obviously implies cloud cover, if the rain is scattered the cloud cover may or may not be. In fact clear skies after a rain shower deliver close to optimal conditions for observing the night sky as the rain will have cleared the air of dust and molecules etc. However the likelihood of rain is the focus of most night time forecasts. Added to this is the problem of light pollution. Regardless of how clear the sky is if you are in a large urban centre the glare of the city lights block out all but the brightest objects in the sky. Therefore an app which could take into account the aforementioned weather conditions along with light pollution factors would be very useful. In the absence of an accurate method of determining light pollution of a location, a list of nearby locations may be sufficient to advise the user options available in less built up areas.

I believe that if successfully completed with an intuitive GUI this would be a very marketable niche app in a marketplace where customers are traditionally willing to pay for high quality, helpful tools.

3. Technical Approach

Having discussed the idea with various colleagues and fellow astronomy enthusiasts I have an outline of the basic requirements required. I intend however to extend these conversations into more formal brainstorming sessions so as to cover all necessary bases in terms of user requirements.

The application will be developed in Android Studio using the Java programming language. There will be a local database which uses Androids built in SQLite. There will be a weather API, a mapping API and two RSS feeds.

4. Special resources required

<https://developer.android.com/training/index.html>

<http://openweathermap.org/>

8.2 Project Plan

Maintain Journal of experience	14/09/2015 - 11/05/2016
Submit Proposal	2/10/2015
Upskill via technology tutorials	02/10/2015 - 14/10/2015
Assess Requirements	02/10/2015 - 11/11/2015
System Testing (Continuous)	06/10/2015 - 11/05/2016
Begin Developing Project Shell	06/10/2015 - 1/11/2015
Begin Developing Components	10/10/2015 - 1/11/2015
Compile Requirements Specification	1/11/2015
Update Requirements Continuously	1/11/2015 - 11/05/2016
Review design in line with requirements	1/11/2015 - 1/12/2015
Analysis and Design	1/12/2015
Complete Prototype Development	02/12/2015 - 01/02/2016
Present Prototype	5/12/2016
Develop Final Application	5/12/2016 - 11/5/2016
Prepare Final Report	02/02/2016 - 11/05/2016
Submit Final Report	11/5/2016

6. Technical Details

The application will be developed in Android Studio with SQLite used to connect with the local database. I have chosen Android Studio as I believe it is the best format through which to build an application which will be compatible with multiple devices.

7. Evaluation

From a testing perspective I will test each section of the application as it is added both in isolation and in conjunction with the rest of the program. I will build a continuous list of testing procedures aimed at breaking the application to find where its weak points are and addressing them as they arise. Added to this I will have colleagues test it along with people with a non-technical background. I intend to consult non-technical users throughout the process as I believe people who are entirely removed from the coding and infrastructure are best able to view the application simply as a tool with a purpose that either works or does not.

Signature of student and date

8.3 Requirements Specification

8.3.1 Document Control

Revision History

03/10/2015	1	Create	KS	X	X
12/10/2015	2	Update	KS	x	x
14/10/2015	3	Update	KS	x	x
16/10/2015	4	Update	KS	x	x
17/10/2015	5	Update	KS	x	x
20/10/2015	6	Update	KS	x	x
21/10/2015	7	Update	KS	x	x
24/10/2015	8	Update	KS	x	x
27/10/2015	9	Update	KS	x	x
28/10/2015	10	Update	KS	x	x
30/10/2015	11	Update	KS	x	x
03/10/2015	12	Update	KS	x	x

8.3.2 Distribution List

Eamon Nolan	Lecturer	
Manuel Tova Izquierdo	Supervisor	
Eugene McLoughlin	Lecturer	

8.3.3 Introduction

Purpose

The purpose of this document is to set out the requirements for the development of “StarGaze” which is the working title of the mobile application I intend to develop.

The intended customers are amateur astronomy aficionados who wish to have a useful tool to help them plan observation sessions

Project Scope

The scope of the project is to develop a mobile application which will integrate the resources of up to three separate API's to help users determine the location and times of good conditions for astronomy.

In order to explore all requirements of users feedback has been sought from various potential users. This has included amateur astronomers and non-astronomers who were able to give objective feedback without being blinded by an interest in the actual outcome and organisations.

8.3.4 Definitions, Acronyms, and Abbreviations

API Application Programming Interface

GUI Graphical User Interface

RDBMS Relational Database Management System.

8.3.5 User Requirements Definition

- Mobile tool for astronomy observation information.
- Get timescale and location for possible viewing.
- Enter their current or future location.
- Choose location from results returned.
- Save information relating to their observation sessions

8.3.6 Requirements Specification

All requirements should be verifiable. For example, experienced controllers shall be able to use all the system functions after a total of two hours training. After this training, the average number of errors made by experienced users shall not exceed two per day.

Functional requirements

1. Interaction with mapping API.
2. Interaction with weather API.
3. Matching of data received from the weather and mapping API's.
4. User log-on.
5. Interaction with astronomy API.

Use Case Diagram

The Use Case Diagram provides an overview of all functional requirements.



Requirement 1: Interaction with Mapping API

Description & Priority

It is vital that the system can identify where the user is so as to return relevant data. Without knowledge of where a user is there is no frame of reference and the main function of the application would be unachievable.

Use Case

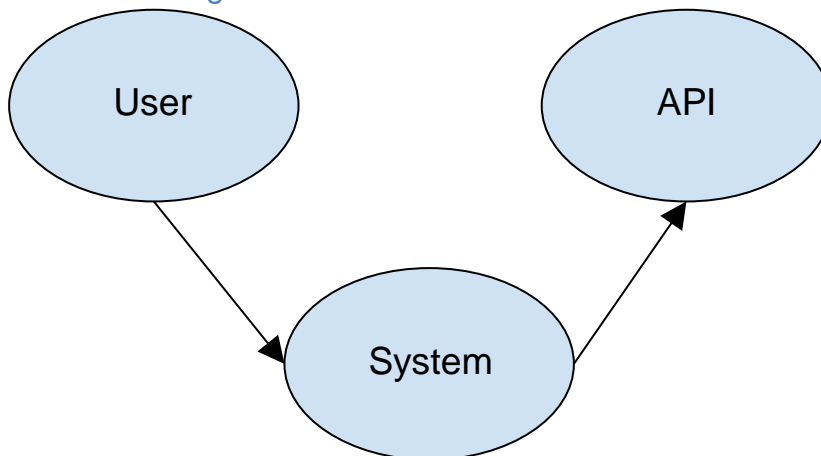
Scope

The scope of this use case is to outline how the system will send and retrieve information from the mapping API.

Description

This use case describes the process of interacting with the mapping API.

Use Case Diagram



Flow Description

Precondition

The user has downloaded the application to an android enabled smartphone. The application is open on the phone.

The phone has internet access.

Activation

This use case starts when a user enters their location either manually or through the application accessing the user's location.

Main flow

1. The system identifies the current location.
2. The system displays the current location.

Alternate flow

A1: Not current location

1. The system identifies the current location.
2. The user sees that location is incorrect and reloads screen
3. The use case continues at position 2 of the main flow

Exceptional flow

E1: No network connectivity

4. The system cannot identify the user's location.
5. The user awaits connectivity.
6. The System establishes network connection.
7. The use case continues at position 1 of the main flow

Termination

The system presents the information relevant to that location.

Post condition

The user receives data relevant to that location.

Requirement 2: Interaction with Weather API

Description & Priority

The system must be able to ascertain the current and future weather conditions of a location in order to carry out its primary function.

Use Case

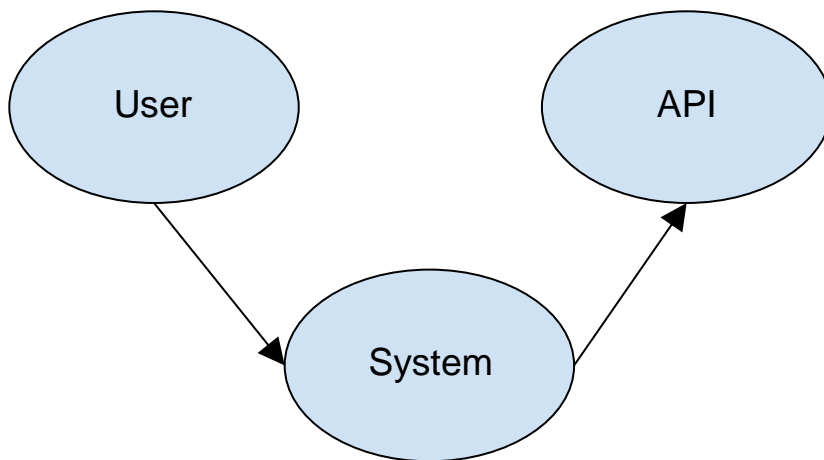
Scope

The scope of this use case is to send and retrieve data from a weather API.

Description

This use case describes the process of interacting with the weather API.

Use Case Diagram



Flow Description

Precondition

The system has an internet connection.

Activation

This use case starts when a user opens the View Now activity.

Main flow

1. The system identifies the current location.
2. The system returns weather data relating to the day current day at the current location
3. The user interprets this data.

Alternate flow

A1: Poor connectivity

1. The system cannot access the weather API due to poor connectivity.
2. The user must wait until connectivity is restored.
3. The use case continues at position 3 of the main flow

Exceptional flow

None.

Termination

The system compiles data received from both the weather API and the mapping API.

Post condition

The system goes into a wait state.

Requirement 3: Matching of Data from two API's

Description & Priority

The system must be able to combine the data received from the two API's in order to produce data that is meaningful to the user.

Use Case

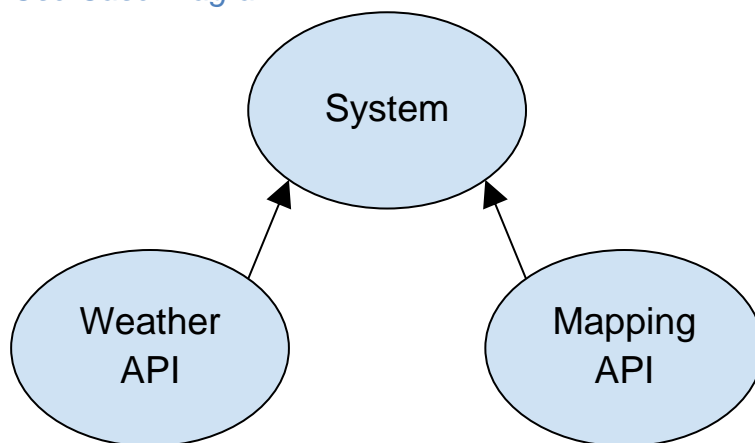
Scope

The scope of this use case is to compute the information returned from the mapping API and the weather API.

Description

This use case describes the process of combining the data from the two primary API's to locate nearby locations.

Use Case Diagram



Flow Description

Precondition

The system has sent and retrieved data from both API's.

Activation

This use case starts when a user has selected the nearby locations option.

Main flow

1. The system retrieves data from the weather API relating to weather conditions within 1 degree latitude and longitude (approximately 75km depending on latitude of user).
2. The system uses an algorithm to determine what results are returned from the weather API and displays the relevant results on the user interface.
3. The user identifies the most favourable option a searches for this location on the map displayed at the top of the screen.
4. The user selects the marker on the map to confirm that they wish to receive directions and or mapping information
5. The system presents buttons on the map for the google mapping service and google directions.
6. The user selects which one they require.
7. The system diverts them to the appropriate service. Mapping, directions or Satellite Navigation.

Alternate flow

A1: User unhappy with options (continued from position 2.)

1. The user must refresh screen.
2. The use case continues at position 2 of the main flow.

Exceptional flow

E1: User unable to find suitable result

1. User is unhappy with all results returned.
2. User must refresh page.
3. System returns updated list of options.
4. Resume position 2 of main flow and repeat until user is satisfied.

Termination

The user selects a location or exits screen.

Post condition

The system has performed its primary function and now goes into a wait state.

Requirement 4: Writing data to a secure database

Description & Priority

The system must be able to read and write user data to a secure database.

Use Case

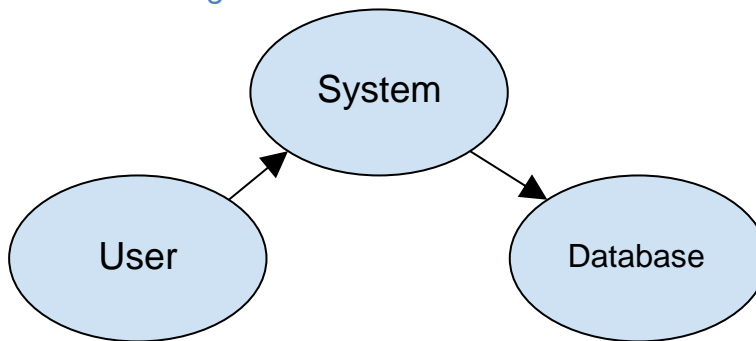
Scope

The scope of this use case is be able to read and write data to a secure database.

Description

This use case describes the process reading and writing data to a secure database.

Use Case Diagram



Flow Description

Precondition

The user wishes to log viewing data.

Activation

This use case starts when an <Actor> enters data.

Main flow

1. The user inputs data which the system
2. The system writes the data to the SQLite local database.
3. The user retrieves data from the database relating to history logged.
4. The user updates logged entry.
5. The system updates database as per user request.

Alternate flow

A1: User inputs data incorrectly

1. The user inputs data which does not meet the formatting requirements of the DB.

3. The System does not send data to the DB.
4. The System notifies the user of the anomaly.
5. The use case continues at position 1 of the main flow

Exceptional flow

E1: User inputs data matching other data currently in RDBMS.

1. User inputs data
2. System sends data to database
3. Database rejects data if unique identifier used (ID)
4. System warns user.
5. The use case continues at position 1 of main flow.

Termination

The system writes user data to the DB.

Post condition

The system goes into a wait state.

8.3.7 Non-Functional Requirements

Availability requirement

The application will be contained entirely on the user's devices with no dependency on external servers beyond the API calls and RSS feeds.

Recover requirement

The application will be backed up locally to ensure recoverability. Damage to the device could however result in the loss of data.

Robustness requirement

Because the application is entirely contained on the users device it's robustness depends on the device. However with an APK size of approximately 25MB this is not expected to be an issue.

Security requirement

No user data will be stored outside of the users device and any data stored relating to the application should relate only to viewing activities. No personal or sensitive data should be stored in relation to this application and to this end security is not of major concern.

Reliability requirement

The application should function successfully for the life of the device. Only established and well supported API's will be used in order to ensure maximum reliability of the information that users receive.

Maintainability requirement

The level of maintenance required should not be excessive however the site will be monitored and updated as necessary on a daily basis.

Portability requirement

The application will be accessible from any web-connected, android enabled smartphone.

Extendibility requirement

Though the application will be initially developed with an emphasis on the northern hemisphere, with only minor adjustments it could be released throughout the English speaking world immediately and the rest of the world once necessary translations have been performed. There is also large scope for additional features.

Reusability requirement

As the application will record user activity it should improve and become a richer resource with each visit from a particular user.

8.4 Monthly Journals

September

The story so far - 5th October

I settled on an idea pretty quickly for the project. Although I brainstormed a few others I was always leaning towards the one I eventually chose. I am going to develop an app which will act as an aide to amateur astronomers looking for good locations and weather for observation sessions. I worked on a similar project to this in last year's group project, the concept was mine and I always planned on bringing it a step further this year. While we achieved a lot last year I would like to make this an actual functioning application rather than just a collection of code that only works in limited circumstances. Before I decided on this project I felt I had to clear it with my group mates from last year. I will not be using any of the coding or user interfaces from that but still felt it better to talk to the others first to make sure no one felt I was taking a shortcut.

I have had an interest in astronomy for many years and am genuinely excited about the possibility of developing something which may be able to help me and others explore the night (and occasionally day) skies. The task however is extremely daunting and in putting together the project proposal I have realised how much work I need to do before even getting to the real work. My understanding and experience in almost all of the technologies I need is substandard and will have to improve dramatically before I can make any real inroads into the substance of the project.

So far my attempts at up-skilling have been frustrated by poor time management on my part and to a greater extent by an incredibly slow laptop. To that end I have ordered a new Dell 15 5000 Inspiron and taken a fair hit to my bank balance. I am very hopeful, perhaps naively so that my productivity will vastly increase once the new machine arrives.

So far I haven't been assigned a supervisor. I'm looking forward to hearing feedback on the concept. Although I completed a separate degree eleven years ago I have never worked on a yearlong project of this scale and am both excited and intimidated by the challenge.

October

It's Starting to Feel a lot Like Winter – 1st November

I have made a conscious decision to only log into this journal once a month. By doing this I hope that each time I write, actual progress will be reflected and by reading back over the diary I will have a true picture of what has been achieved in a given month. As mentioned in last month's post I have invested in a new laptop and I'm now in a much better position to work on the project without being slowed down by inferior equipment. Although there was a minor glitch when the laptop arrived (it had the wrong plug) it has made a big difference to the productivity of the time spent working on the project.

I met with my supervisor Manuel for about 40 minutes and had what I felt was quite a productive conversation. He focused mainly on the project management side of things but also gave me useful insight into what areas I should focus on in order to maximise marks etc. This is a good thing to keep in mind as it's easy to get blinkered to the whole scope of the project when you're over focused on certain areas.

I have been doing quite a lot of ruby on rails tutorials which I'm finding very useful and also tutorials on Android studio. So far I'm enjoying them and feel that they are bringing me closer to the level I need to be at. Of course none of this will be tested until I start to really bring the project together. The coding has taken a back seat in recent weeks though, as I've really focused in on the requirements specification that is due at the end of this week. I've made an honest attempt to discover and explain the key requirements of the application while not fluffing it out with nonsense. One difficulty I've found is that the requirements spec includes class diagrams and user interfaces which are not yet fully determined. That said it has been very useful for building a clearer picture of what these will look like and how the application will be structured as the exercise forces you to think about these things.

I'm hoping to meet with Manuel again in the next week or so, the conversation is likely to be more focused once the requirements specification is completed and there is a blueprint to work off.

November

The story so far - 1st December

Despite deciding to only update this journal once a month in my last entry, Manuel has persuaded me that it makes much more sense to update it a little a few times a week. This way the journal will better reflect my actual experience and also will take much less effort. So from now on that's the plan....starting tomorrow.

At the start of the month I spent about a week trying to get RubyMine correctly installed with the Rails SDK. After about 4 abortive attempts I eventually got it to work. I was far too happy about such a small milestone but at the time it was a big relief. It was by no means obvious why it wasn't working but after a lot of frustration and googling I eventually got it to work. At least now I am able to get some real coding done and am starting to understand the Controller, model, View structure a lot better. That said due to other projects and a wedding (not mine) the main project has taken something of a back seat in the last week or so. This will have to change however as I need to finalise the Analysis and Design document for Friday. I have another project due on Sunday but I'm confident that everything should be under control. In the next month and over Christmas I hope to start bringing different aspects of the project together to try and have the prototype taking shape by early January. I had hoped to be further along than I am at this point but that is probably due to me underestimating the size of the task ahead. Hopefully now I have a more realistic idea of what needs to be done. I met with Manuel last week and discussed various aspects of the project with a particular emphasis on project planning, something that is proving easy to overlook the busier I get but ironically becomes more important as otherwise a lot of time and effort can be wasted doing something unproductive. That's all for this instalment. Hopefully by the next one I will be telling you of my success.

December

The story so far - 19th December

Having submitted the first draft of the documentation required so far I have been able to spend the first part of this month trying to get my head around the code required for the application. While continuing to look at various online tutorials I have also started to get to grips with the google maps API which so far is going relatively well. As you would imagine it is a very popular API which luckily means there's plenty of resources and advice online to reference when various issues arise. So far I have managed not to let the Christmas season interfere with my study schedule too much but that is obviously going to get much tougher in the next two weeks.

22nd December

I noticed today that I had an email from Eamon informing us that the December journal was due for submission on the 18th of December, while I'm sure this was done with the best of intentions to give us less to worry about over Christmas it caught me off guard and so this installment will be a little late. Due to the college closing for Christmas I haven't met with Manuel this month. Hopefully this means when we do next meet up I will have some progress to show him. I have continued to work on the mapping API and now have a button which will load a user's current location and display a map of the locality centred on them. Basic enough I know, but it is a base which I am hoping to build around. I have also been looking into various weather API's to see which would be most suitable for the application, if anything the main issue I am having is that there are so many to choose from with each one appearing to fulfil most of my requirements. Whether they actually will or not is of course my worry as often the API is not in reality as good as is advertised. Nonetheless it is better than not having any to choose from. As predicted in my earlier submission Christmas has begun to encroach significantly on my study time though I do still hope to put aside a few hours most days over the holiday to make sure it keeps moving along.

January

January 9th

The months started slowly enough in terms of the project. I was home with my family until the 4th of January and while I did manage to get some work done during this period I wasn't able to spend hours at time working on the project. On my return to Dublin the Network Security exam took precedence as this was on the 8th of January. So all in all there was over a week of January gone before I was able to fully refocus on the project.

My initial focus on restating was choosing a weather API, I settled on openweathermap.org which is the API that google now recommend to use with their mapping API having deprecated their own weather API. Speaking of deprecating API's I was dealt a minor blow with the news that google have also decided to deprecate their Astronomy API which was by far the leading contender for my project. Nonetheless the search goes on. It took me quite a while to make a successful API call from Open Weather Map, mainly due to my own poor understanding of JQuery, Javascript and JSON files. While the learning curve was quite steep I took the decision to temporarily focus my attention on getting these technologies to work rather than learning a new environment. In practice this means that the application is being built in RubyMine and deploys as a Web App

rather than a native mobile app as it would be if coded in Android Studio. This is not necessarily a permanent change and will be reviewed continually.

January 26th

I now have a web app which opens to a home page. On selecting a “current location” option the user is brought to an interface where their current location is displayed on a map. A text box also declares the current location and another box holds data on the weather forecast for the next 9 hours, 3 forecasts in total. While this may seem simplistic enough it took me quite a while to get it all working as I wanted and I’m hoping to be able to kick on from here now and add much more functionality over the next few months as well as adding login capabilities and a database.

January 30th

I have added quite a bit of CSS in order to make the app look more user friendly and am now preparing for the mid-point presentation next Saturday. Hopefully I am in and around as far on in the development as I should be. My own opinion on this changes from day to day.

February

The month started off focused on the mid-point presentation which was on the 6th. I felt I did reasonably well and received some good feedback that I can implement into the project. One particular suggestion that came from it was to use third party authentication so as to avoid all the extra regulatory overheads that would come with storing user data myself. By using Facebook or Gmail authentication this may be circumvented. Despite this the mark I received for the presentation was only okay, any lower and I would have been very disappointed. Nonetheless the feedback was good so overall it was a positive experience.

Unfortunately the month since then has been almost completely taken over by the two other modules this semester. With one CA due last week, another due in two weeks, an in class CA tomorrow night and two other projects to be completed I have been unable to make significant further progress with the final year project. Hopefully however there will be time this weekend to get back into it as there is a temporary lull in the other two. I have not met Manuel since the presentation for the reasons outlined above. I hope to arrange a meeting in the coming weeks however.

One of the modules this semester is Mobile (Android) Development so I am hopeful that the technologies I am learning in this will be transferable into the

project and that appears to be the case. The work that I have completed this month on the project is the preparation of the 30 word blog for the brochure. I am also possibly the only member of the class who didn't have an existing LinkedIn account, probably a result of being in the public service and so not particularly active in looking for a job. Nonetheless I have one now.

March

This has been an extremely busy month between all of the CA's in my other two modules falling due and the continued work on StarGaze. I have implemented most of the technologies required for the application with varying degrees of success but I'm hopeful I'm not too far off with any of them. The trick then will be to integrate them all into a coherent, intuitive user experience. I am possibly changing my weather API from Open Weather Map to android studio. I haven't made a final decision on this yet and really it will come down to the kind of documentation and support material that I can find online for each. There doesn't appear to be a huge amount of difference between them. Google Maps API has proven very intuitive so far with loads of support material online etc. Both of the fundamental functions I require from this API have been implemented and it was a nice win to be able to do that with relative ease. One area I am having a little more difficulty with is the astronomy API. Replacing the now defunct Google Sky API has not proven as easy as anticipated and I am now looking at the possibility of using an astronomy RSS feed to obtain time specific information. I have been able to spend a few days specifically focused on the project over the last few weeks which was a relief though it is about to take a back seat for another week or so now as we have our final exams next week. After that it will be all out on the project and I am obviously hoping everything will fall into place.

April

This is my final journal entrance and not actually one that will be uploaded to Moodle except via this report. The last few months have been extremely intensive and the last month in particular there has literally been no respite from college demands. My final exams were in early-mid April and I am relatively confident that I performed well in these. Attention then turned solely to the final project. As discussed in last month's journal I had toyed with the idea of switching my Weather API to Yahoo! Weather. In the end however I stuck with Open Weather Map API and am extremely glad I did as once I got the hang of the format of their JSON responses I found I was able to manipulate the results to retrieve different responses as required. There is actually much more information which I could take

from these responses but in the time available I felt it was more sensible to stick to the plan rather than jeopardise the intuitive flow of the GUI by introducing too many options. The project has been an incredible learning experience and I have gone between despair and relief many times along the way. I am somewhere in between the two now. I have been in email contact with Manuel this month and hope to run my presentation by him before it is due to be given. His advice on how to format the presentation will be vital as it is not immediately obvious to me how best to do it.