User Friendly Calendar Application

2015

Isaac Kingsley Tyson-Seale

Supervisor: Rabia Khan

Git Repository: https://github.com/IsaacKingsley/ProjectHandIn

Declaration

No part of this project has been submitted in support of an application for any other qualification or degree at this, or any other learning institute. Apart from the parts of the project that have been cited as the work of another. This project is my own, unaided work.

Abstract

This report provides an analysis & evaluation of the usability of a selection of current existing calendar applications available on the Android Marketplace and uses those findings to produce a new, calendar application that has a high level of usability and user friendliness.

Methods of analysis and evaluation include Heuristics evaluation, user feedback via questionnaires and Usability Testing. All results and feedback can be found in the Electronic Appendices.

After the implementation and testing of the new application, the same analysis and evaluation techniques were applied to the new application and those results were compared with those from the current existing applications. The results showed that although the new application has high usability and user friendly scores, it was still not on the level as the current existing applications.

The conclusion of the project was a success, as a user friendly calendar application was successfully designed and all core features were implemented, it does however still have much room for future improvements, as due to time constraints, not all of the desired features were fully implemented. The addition of these future improvements would hopefully bring the new application to the same level as the existing applications, or even surpass them in terms of usability and user friendliness.

Table of Contents

List of Figures

List of tables

Chapter One – Introduction

1.1 Project Overview

The project consists of building a user-friendly calendar application for a mobile device. As a mobile application that depends on user input and persistent data to store the users input, this project will include the following components:

- A database
- A data access object
- A user-friendly Calendar Interface
- An Event entry form
- Multiple methods of viewing and managing entries (All, filtered)

The end user of the project are students, the source code of the application will be made available to those who are new to mobile development and will allow them to see how data is stored and retrieved within a mobile application.

As well as being a learning tool, the application is also functional. Students can use the application to set alarms and reminders for exams or other important University events and also view them in an uncluttered way.

1.2 Aims and Objectives

After the Terms of Reference were created, there were some changes to the original aims of the project.

1.2.1 Aims

The original aim of the project was:
To create a user-friendly calendar app that has Social Media functionality. That allows the user to add their own entries as well as allowing the user to sync information (Birthdays) from a social media network (Facebook).

This however was revised slightly. The new aim was to produce a User Friendly Calendar application that was open-source that would allow a new student to see how data management between an Android device and an internal database worked, although the aim of trying to implement social integration was still present.

1.2.2 Objectives

After the revision of the aims, some new objectives were added. Achieving the objectives of the project would allow for the aims to be successfully achieved as well. These objectives were:

1.2.2.1 Research

- To use Surveys to identify the features in the calendar apps that are highly rated by users, and improve on the bad features within my own app.
- Using SUS Analysis to measure the usability of my App once a prototype has been developed.
- Research Data Access Objects for Android

1.2.2.2 Design

- Design and implement an application to allow a user to create, view, edit and delete entries on a calendar.

1.2.2.3 Implementation

- Implement linking with Social Media websites (Such as Facebook) to allow users to sync their friend's birthdays to the Calendar.
- Creating a prototype app and having testers fill in a SUS (System Usability Scale) questionnaire to measure the testers experiences in order to improve the app.

1.2.2.4 Testing and Evaluation

- Evaluate the calendar that I have created by comparing the final version to the prototype and comparing it against similar applications.

1.2.2.5 Documentation

- Write a final report that details the entire project and collating all the documentation produced throughout the project.
- To create a detailed tutorial that will allow users to learn how to use the app quickly and efficiently.

Chapter Two – Literature Review

2.1 Introduction

The purpose of this chapter is to provide a detailed background into the processes and the logic behind designing a system that is user-friendly as well as the different methodologies used when designing a user interface and the available technical solutions that can be used to produce the Calendar Application.

The literature survey will focus on finding the functions and features of the current existing calendar applications that users like, and those that they do not like. As well as finding usability ratings for the overall design of each applications interface. These findings, along with the user requirements will be used in the design and implementation stages of the new application.

2.2 User Experience

User Experience, known as UX, has no universal definition and many professionals have their own idea of what UX is. One possible definition is that it is meeting all of the needs of an end-user with any service or product that is designed. **Thomas Tullis** says that User Experience consists of three main defining characteristics.

The **first** characteristic is that a *User* is involved. The **second** characteristic is that the user must be interacting with a product or system and the **third** characteristic is that the users experience must be of interest, must be observable and must be measurable. (**Tullis, T. and Albert, W. 2014**). The term UX is used when making all manner of decisions, including **Design**, **Content** and **Development**.

2.2.1 Examples of Good User Experience

Apple.com have good User Experience, their website has a simple layout and to the point information, navigation around the website is easy and consistent between pages which increases efficiency of the user. It is also engaging to users as it contains interactive (clickable) images that can assist the user to get to their desired destination quicker.

Amazon.co.uk is another example for good User Experience, although the layout is not as simplistic as Apples, navigation on the website is still fluid and easy. Like Apple, the website contains interactive images and but it includes recommendations to the user that change depending on what the user has previously searched for.

2.2.2 Examples of Bad User Experience

Direct.gov.uk is an example of bad User Experience. Navigation around the website requires lots of clicks to get to the desired location, and once you reach that location you have to fill in information on multiple pages.

2.2.3 Why is User Experience Important?

User Experience is important for any product to survive. If completing a task is complicated or finding the right content takes too long, then a user will leave a website or uninstall a product and find an alternative solution, which means they'll be looking at your competitors' product or website.

2.3 User Experience Design (UXD)

UXD is a process used to enhance a customer or users satisfaction by improving a system's usability and ease of use. The process uses traditional Human Computer Interaction (HCI) design but further extends on these methods. After more than 20 years, User Experience still has no common agreement and it is also unclear as to who is responsible for UXD (**Kuniavsky, 2010**)

2.3.1 Human Computer Interaction (HCI)

HCI is studying how humans interact with computers; it first began in the early 1980s as a specialty area of Computer Science that looked into the cognitive science and human factors. **(Carroll, 2013).** Since then, HCI has rapidly expanded, attracting professionals from a wide range of disciplines.

Personal and Home computers are very common today, but in 1955 there were only 250 computers being used in the world. These computers were only used by IT professionals and a small selection of die-hard, dedicated hobbyists.

This did not change much until the late 1970s when personal computing was introduced. This introduction brought software such as text editors, spreadsheets and interactive games into the eye of the general public as well as introducing new platforms such a programming languages, operating systems and a verity of different hardware. This introduction helped highlight the issues with regards to usability for those that wished to use a computer as a tool. (**The Personal Computer, 2014**)

2.3.2 Benefits of User Experience Design

One of the main benefits of User Experience Design is that it helps to improve the usability and overall customer satisfaction of a system, which in turn means better acceptance from the users that it is targeted at. If a customer has a good experience, they are more likely to recommend the product or service to friends and family which can also lead to an increase of sales of a product or increase the number of visitors to a website as well as lead to good user feedback. A well-designed product also increases productivity and usability for a user.

Another benefit to UXD is reduced ownership costs and support. A well-produced, easy to learn and use product or system will mean reduced requests for support from customers and users. Also, a well-designed product would require a reduced amount of documentation, as it would have been designed with ease of use and learnability in mind. (**Quora.com, 2014**).

2.3.3 User Experience Design Criticism

User Experience Design is still in early stages of development and it has no universally recognized standards or definitions. Critics say it is just a Buzzword existing practices that brings no real Return on Investment (*Return on Investment*) to the field of Computing. A users experience cannot be fully measured, which again questions the Return on Investment of User Experience Design.

2.3.4 Smart Phone Design Patterns

There are a number of techniques that developers use to make usability and navigation of a mobile application fluid and easy to learn, these include: -

Coach marks: If a developer wishes for an application to be successful then the app needs to be useful. Some tools that are useful also end up being very complicated. Coach marks are hints and tips that are presented to a user, usually when an application is opened for the first time, that teaches them what the features and functions do and how to use them properly.

Content –Based Navigation is a method used to merge the transition from complex and detailed views with less detailed views, for example, when a user searches for another users profile using the Facebook App, they are presented with a List overview of people and groups that match the name that was searched for, tapping the profile they wish to view brings up a more detailed view.

Slide/Swipe is mainly used for setting changes, for example in the Android Operating System, the user can change the brightness of the screen using a slider, or lower/raise the volume in media players. Apple devices allow the user to swipe the screen to unlock their device. Sliding can also be used to navigate between views of an application as they give a seamless feel to features and navigation.

Popovers, also known as 'Pop-ups', allow additional information, options or notifications to be shown to a user. For example, on the Spotify application the user can prompt additional options by tapping or long pressing on a song. Another example would be logging out of Facebook, a pop-up appears asking if the user is sure they would like to log out.

Slideouts/Sidebars/Drawers are used to reduce the amount of content and clutter being displayed to the user at one time, for example hiding extra menus or alternate views within an app. *(UXPin, 2014)*

2.4 User Experience Content Strategy

Content Strategies focus on creating content that is not only meaningful but also engaging to a user as well as having content that is sustainable

Having a good content strategy helps you to identify what content already exists, what content needs to be created as well as helping you identify why that content needs to be created. (**Halvorson, K. and Rach, M. 2014**).

2.4.1 Content Lifecycle

In order to produce sustainable content, you must first understand and follow the content Lifecycle. A lifecycle is a series of changes, and in order for content to stay relevant then it also has to change throughout its lifecycle. If content does not change, then it can miss out important information and become inaccurate.

There are many different variations of the Content Lifecycle, but they are mostly all very similar to each other. A general overview of the Content lifecycle is: -

Analysis: The analysis stage focuses on the strategic aspects of the content and can be split into four sub-categories. These are *Requirements*, *User Research*, *Governance* and *Content Analysis*.

Firstly, the current content needs to be analysed, if there is any, for technical gaps to determine the contents readiness. A requirements matrix is then used to organize the business and technical requirements of the content into a summary form. Process models are then developed; these dictate the sequence of activities for future steps.

User Research is identifying who the consumer is and what content they would like to see as well as how and when the content is used.

Governance helps to establish who is responsible for content and processes throughout the content lifecycle. It also helps with establishing a budget and assigning budgets to the different areas, for example technology upgrades and implementation or localisation of content.

Content Analysis helps determine what content is already available and leads to auditing of such content for quality, it can also help to identify content that is missing or that is inaccurate.

Collection: In the collection stage, the content is gathered using the structures set in the analysis stage. Content can be collected by either be newly developed, editing and updating existing content, ingesting data from external sources, content can also be collaborated to generate new content.

Standards for content delivery and re-usability are also laid out in this stage as well as development of layout templates to determine where and how content will be displayed on a display.

Management: The management stage focuses on how efficiently and effectively the content is used. For larger organisations this includes Content Management Systems, as there is too much content to manage manually so systems are used to help automate functions and processes that can be automated. For smaller organisations with less content, it can be possible for this to be done manually.

Publication: The publication stage is the final step in a single lifecycle, but that does not mean it is the final step. Publication leads back to Analysis stage so that the process can begin again to help deliver even better content. This stage deals with delivering the content to users. (**Johnnyholland.org, 2010**).

2.5 Systems Development Lifecycle

The System Development Life Cycle is the process for Information System Deployment Projects, it is used to ensure a product meets the user requirements and functional requirements. (**Webopedia.com, 2014**). The model used when developing a system or piece of software can have a high impact on how testing is carried out in later stages.

2.5.1 The Waterfall Model

There are a many variations of the development lifecycle; the Waterfall Model is one such model.

The Waterfall Model has 6 stages, *Requirement Gathering* and *Analysis*, *System Design*, *Implementation*, *Testing*, *Deployment of the System* and *Maintenance*. In order to progress onto the next stage, the previous stage must be fully completed; it is typically used for small projects that have no uncertain requirements. At the end of each phase a review of the project takes places to ensure that the project is progressing properly.

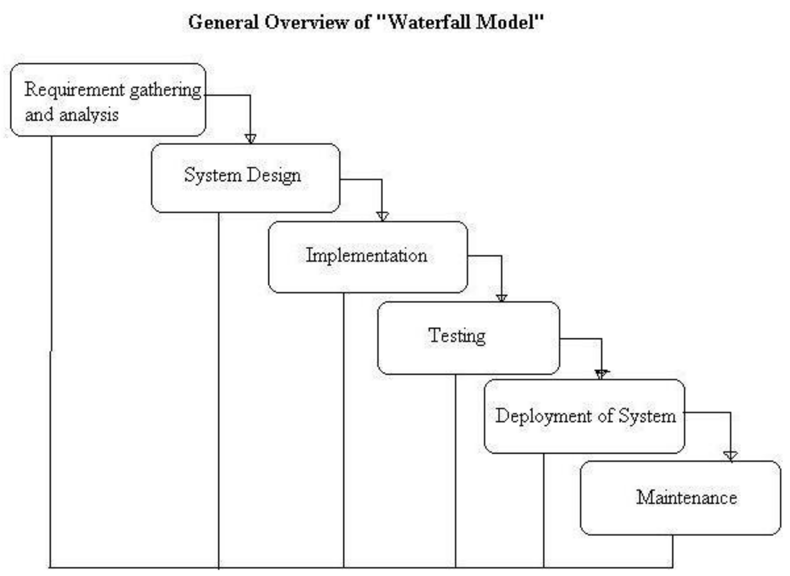

General Overview of "Waterfall Model"
Requirement gathering and analysis
System Design
Implementation
Testing
Deployment of System
Maintenance

Figure 1 - Waterfall Method

2.5.1.1 Advantages
The Waterfall Model is that it is simple and easy to understand, it is easy to manage, as each phase must be completed before the next phase can begin.

2.5.1.2 Disadvantages
Once the testing stage has begun, it is very difficult to change something from prior stages. It is also not a good model for projects that are object-oriented, complex, projects that are long or projects that are a risk of having requirements that could change. (**Istqbexamcertification.com, 2014**).

2.5.2 Iterative Waterfall Model
The Iterative waterfall model allows for development to start without a full list of requirements for a system. Small sections of the system or rough versions can be designed, implemented and then analysed before moving onto the next iteration and starting the process again. Each time the process is repeated new versions that are more refined are produced.

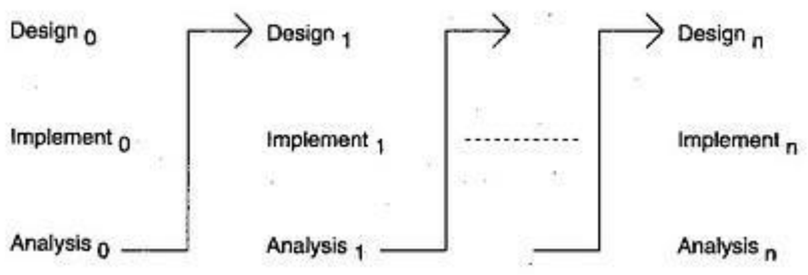


Figure 2 - Iterative Waterfall Model

2.5.2.1 Advantages
As the system is being developed and improved step-by-step, it means that problems can be detected early rather than in later stages. This helps to prevent a spiral of defects being detected in the final stages that would cause the entire system to need re-designing or re-working.

Users can be asked to test individual parts of a system as it is produced leading to higher quality feedback and suggestions.

2.5.2.2 Disadvantages

1. Development can take longer, as there is no overlapping for each part of the system. One section gets full attention at one time.
2. Ground-breaking architecture and design issues can arise as the full requirements for the system are not clear before development begins. This can be costly to a company as time will be wasted with development if the design needs to be re-done.
(**Istqbexamcertification.com, 2014**).

This is the method that will be used throughout the development stage of this project.

2.5.3 User-Centric Content

Content Strategy is the practice of planning the creation, delivery and governance of useful, usable content (**Kristina Halvorson, 2014**). A good content strategy takes into account the goals of the user, the goals of the organisation and the content that is already available.

For user-centred strategies, the users have to first be identified; the users of a system may not be the people you first assumed would use the system. Users can be reviewed and identified in a number of ways including databases, registration data, questionnaires, interviews, focus groups and various other analytic techniques.

After that the needs of the user must be identified, this data can be gathered using various techniques including, questionnaires, interviews, focus groups as well as other methods.

To go even further beyond, is to identify the wants of the user. This can be done in a number of ways, one example would be examining search results. If users regularly use search boxes to find the address of the company, then this content should be made more readily available on the homepage.

2.6 Usability

Usability is making products and systems easy for a user to use and making sure that the user's needs are fulfilled by the system. It is about Effectiveness, Efficiency and Satisfaction for the user meaning that users should be able to complete tasks or achieve goals with minimal effort and little frustration.

These factors can be affected by a number of things, the first being the experience of the user. An experienced user will be far more effective than a novice user when performing tasks, especially if the product is not designed for novices. The second factor is the goals of the user, if a user is trying to perform a task that is not supported by the product or system, for example Microsoft Excel can be used for balance sheets, but it may not support a complicated equation or may require multiple simpler steps to get the same result as a piece of software designed specifically for mathematics. The third factor is the context of usage, for example a user may be trying to render a 3D image on a laptop. Although the laptop can perform the task using the software, it will be slower than using a higher-powered machine built specifically for image rendering. (Nielson. J, 2014), (*UsabilityNet.org, 2014*).

2.6.1 Usability Metrics

Metrics are a method of evaluating something, for example the heights and weights of people. Once data has been collected it can be analysed and compared in order to come to a conclusion but in order to do this correctly the process requires an agreement on how things are going to be measured.

Usability metrics are tests used by designers to assess the usability of a product. There are some general (10) metrics that can be used and these are:

1. Completion Rates: Completion rates are very simple way to measure the usability of a product, and are usually recorded as a Binary Metric where the Result is either 1 for Success or 0 for Failure. If a goal cannot be accomplished then nothing else really matters.

2. Usability Problems: This usually measuring the severity of problems encountered with the UI of a product. It involves descriptions from a user about the problem they encountered. Developers can use this to determine how many users encountered a specific problem as well as determining which users specifically encountered the problem.

3. Task Time: This is a simple measurement of efficiency and productivity. The time it takes for a user to complete a task is recorded.

4. Task Level Satisfaction: To collect this type of Data, a user must perform a task and then be asked some questions about how difficult or not they found the task. This test will immediately flag when a task is difficult, especially if there is a database of tasks to compare the results with.

Errors: This simple test is used to record the unintended actions and mistakes that a user makes whilst attempting a task. For example, if a user enters their first name into the last name field of a registration form. A more complex test is adding severity ratings to different errors or classifying them into set categories.
Errors can help diagnose UI problems but can be very time consuming, as somebody must review the collected data.

Expectations: Users generally have their own expectations of how easy or difficult a task should be. This type of data is collected by asking a user "How difficult do you expect the task to be", and then comparing the response to the actual difficulty rating that the task was given upon completion collect this type of data.

Views/Clicks: When analysing websites or applications, Views and Clicks may be the only basic information accessible without setting up time for users to test. The number of clicks has been shown to have high correlation with Time on Task, which is a good way of measuring efficiency.

Conversion: Conversion rates are type of Completion Rate and are essential to e-Commerce. For example they measure whether a user has the ability to sign in or purchase products from a website. The results are Binary Metrics and are measured from the start of a user's interaction (viewing the page) to finalising the payment details. Often it is a combination of multiple other tests that lead to having low conversion rates.

Single Usability Metric (SUM): Single Usability Metrics is combining the scores of multiple metrics into a single score.

Measuringu.com, (2014), Travis, D. (2014).

2.6.2 Nielson's Goals

Jakob Nielson defines five components to Usability.
Learnability, denotes how easy it is for a user to complete simple tasks when they first encounter them.

Efficiency: Is how fast can an end-user perform tasks once they have had time to learn the interface design.
Memorability: If a user returns after a not using the interface, how easy is it for them to regain proficiency with the interface.
Errors (Low error rate), denotes how many errors are made by the user, how severe are the errors that are made and how easy is it for the user to recover from those errors.
Satisfaction: Is the design appealing to the user? (**Nielson, 2012**), **(Nielson 2013, Mobile Usability).**

2.6.3 Why is Usability Important

According to the Usability expert, Jakob Nielson, Usability is key for survival, whether it is a website or mobile application. If an application is difficult to use then users will leave and search for alternate solutions, likewise with websites, if information is not clearly stated or difficult to find then the user will leave. For e-Commerce, this can mean a loss in sales, as if a user cannot find what they are looking for, then they cannot spend their money.

For workplace intranets, poor usability yields lower productivity meaning that employees will be being paid for doing less work. (**Nielson, 2012**).

2.6.4 The System Usability Scale

The System Usability Scale (SUS) is a set of 10 questions that gives an overview of subjective assessment of Usability (**Usability Evaluation In Industry**– 1996). It is technology independent meaning it can be used in a wide variety of situations, from Software, websites, mobile phones and even paper based systems such as the Yellow Pages.

The system behind scoring the SUS is relatively simple. For each odd item, subtract one from user response. For even numbered items, subtract the user's response from 5.

This will give values between 0 and 4 (with 4 being the best positive response), adding up all the values will give a value between 0 and 40 which has to be multiplied by 2.5 to give a range between 0 and 100. (**Jeff Sauro, 2011**) (**Brooke, J. 1996)**

2.6.5 Initial Survey Results

A selection of existing Calendar Applications were selected from the Android play Store for testing and evaluation purposes. Users were asked to test between 3 and 5 of these applications and to give feedback on each application as well as filing in a SUS for each application.

See Electronic Appendices for Participant Survey Results.

2.7 The Purpose of Creating the User Friendly Calendar Application

The purpose for the research and production of a User Friendly Calendar Application is to strengthen the understanding of User Experience & Usability and why these are important aspects for the planning and design of a product or system and applying the results of the research to produce a product that has desired functionality and Usability that would give a user a good experience.

2.8 Mobile Development Programming Languages

Programming languages are languages that are used to write instructions that allow a computer to perform tasks. They can be low-level languages such as binary or machine code that can talk directly to the machines hardware. These low level languages are fast as they can be executed without the need for translators or interpreters. The downside to low level languages is that it is very complex.

Or they can be High level languages, such as Java, Basic and C that look more like English in the way that they are written but they need to be compiled or interpreted in order to be executed on a machine, as it needs to be converted into a low level language that the computer can understand. **(BusinessDictionary.com, 2014), (Apple, 2014)**

As this project is to be a mobile application, there are 3 possible platforms, Android, iPhone or Windows Phone. The platform that the application will be developed on will be Android although the other programming languages will be researched and explained.

2.8.1 Android

Android is a mobile Operating system that is based on the Linux Kernel, and is currently being developed by Google. Its user interface is based on direct manipulation, the Operating System is primarily used on touchscreen mobile devices such as tablets and smartphones but more recently has been expanded more for use with smart watches, the Android TV set-top box, and cars (Android Auto). A secondary use for Android OS is Games Consoles such as the Ouya that runs a special version of the OS and Digital Cameras.

Android is currently the most widely used mobile Operating System, and as of 2013 it is the highest selling OS, with phone sales reaching One Billion in 2014. **(Gartner, 2014)**

On smart phones, navigation is done using touch, swiping, pinching, reverse pinching and tapping. (**Burnette, E. 2009).**

2.8.1.1 Android Development Good Practices

Android Guidelines are known as Good Practices, they dictate how features and functions should be implemented to give the best experience to the user, to get the best performance from a device and to keep the look and feel from app to app similar as to reduce confusion to a user. Below is a small sample of these practices.

2.8.1.1.1 Multiple Entry Points

Applications must be built with distinct components that can be invoked separately, for example an activity will consist of interface and a service that performs actions in the background.

2.8.1.1.2 Device Compatibility

The Android development kit allows for unique resources to be made available for different device configurations, for example, creating different layouts for different sized screens, resolutions and orientations. The Dev Kit also allows for a device to be queried to determine if specific hardware, such as a camera, is available, thus allowing you to determine whether a device is compatible or not. (**Developer.android.com, 2014).**

2.8.1.2 Java

Java is a free to use, multi-platform programming language (**Sun Microsystems, 1995**). Java is used in many places including Computers, Games Consoles, and Mobile Phones, on the Internet and in Set-top Boxes.

2.8.1.3 Advantages of Java

One of the main advantages of Java is that it is free and readily available to developers. There are free and easy to follow tutorials available that will allow beginners to learn the language. It is Class based and Object orientated, making the code easily re-usable throughout a project. It is also platform independent, meaning that typically, code can be moved from one system to another and be executed without issue. **(Java, 2014).**

2.8.1.4 Disadvantages of Java

One of the main disadvantages of Java is that it is slower than other languages such as C and C++. Another disadvantage is that it is very strictly typed; this can lead to issues when passing data around as incorrect data types can lead to inaccurate data being kept (losing decimal points from numbers) or crashes with the system/application.

2.8.2 IPhone and iOS

IPhone Operating System is a Unix-Like mobile Operating system based on Darwin Operating System. It is developed by Apple Inc., it is exclusively used on the specialised hardware of the iPhone and the iPad. Much like Android, the User Interface is based on direct manipulation that uses Multi-Touch gestures, buttons, sliders and switches. The gestures that the OS incorporates are swiping, pinching, and reverse pinching and tapping. The OS also allows screen rotation between landscape and portrait, which is detected by an internal accelerometer.

2.8.2.1 Apple Development Guidelines

Before an app can be made available on the Apple App Store, it must first be submitted for review. It is then rigorously tested to ensure that it does not break any of the rules set by apple. Some tests for functionality include testing if the app crashes, if it works as advertised or that it has no undocumented functions or features. This is only a very small selection of guidelines as Apple have well over 100 guidelines. (**developer.Apple.com, 2014**)

2.8.2.2 Swift

Swift is a new programming language developed by Apple for creating iOS 8 and OS X applications. It capitalises on the best parts of C and Objective-C without any of its constraints (**Apple 2014**.).

The Cocoa Touch is the framework that is used by iOS, it shares many of the patterns found within the Mac OS Cocoa Frameworks but with the added functionality for touch-based interfaces. Many of the basic tools are provided via UIKit for graphical and event-driven iOS applications. The iOS and Mac OS versions of UIKit share a similar structure, which includes file handling, networking and more. (**Apple 2014**.).

2.8.2.3 Advantages of Swift

Swift is designed from the ground up by Apple to make application development easier, it is still fairly young; only reaching version 1.0 on September 9th 2014 so the advantages of using Swift are not yet widely known. One advantage is its ease of use; this could lower the entry-level requirements for apps as less experienced developers will have the opportunity to develop on a widely used platform. **(Apple 2014), (Krill, P. 2014).**

2.8.2.4 Disadvantages of Swift

As the language is still fairly new, like with the advantages, the disadvantages are still not widely known or available. One disadvantage of Swift is that it is only relevant within Apple Environments and is not compatible with other platforms. This forces a company or developer to commit to Apple platforms and neglect others. It also makes it harder and more costly for developers to port their applications over to other platforms. **(Apple, 2014)**

Although it is not really a disadvantage, current developers with applications on Apples app store must update or re-write their apps in order to be compatible with the new language. This requires that developers must spend extra time learning this new language before they can do this. **(Wired, 2014),**

2.9 Data Access Objects

2.9.1 What is a Data Access Object?

A Data Access Object (DAO) is an object or an interface that provides access to an underlying database or any other type of persistence storage and they are an essential part of good application architecture. (**codefutures, n.d**)

2.9.2 Why are DAOs important?

Various applications need the ability to access data from databases and the DAO is what helps them achieve this. The DAO contains the SQL queries that the application will need to use to perform the tasks that it is designed to do, for example retrieving a list of all the Calendar Entries from the database, inserting a new entry into the database or updating an already existing entry. It also helps to map the Variables used within the application to the columns of the database to ensure that the values retrieved from the database are put into the correct places for use by application. (**codefutures, n.d**)

2.9.3 Advantages of DAOs

As the DAO is a separate object in a system, it allows for easy upgradability. The DAO and other components of the system can be modified and upgraded with little to no disruption to each other. Separating system components also helps to keep code tidy and manageable when performing maintenance and upgrades.

There are many solutions available for such as Spring DAO and Hibernate DAO and there are also tools available such as Firestorm/DAO that allows a user to quickly generate a functional DAO for their system. (**Geek Talks, 2007**), (**Jenkov , n.d**)

2.9.4 Disadvantages of DAOs

A main advantage of DAOs is that they typically only handle full objects due to the more complicated queries used when retrieving data, this means more overheads when memory usage is an issue, like in a mobile device. Using a DAO for tasks such as populating a dropdown menu, which only uses a single field from a database, a lot of the retrieved data is not used after the results have been mapped to their corresponding variables.

The format of the code is what is known in the Computer Programming world as **Boilerplate Code**. Boilerplate Code is the sections of a piece of code that must be included in multiple places that have little to no alterations. This increases the work load for a programmer as it would mean a simple piece of code that does a small job ends up being more lines than necessary. This means that writing your own DAO is a tedious and repetitive task. There are newer frameworks available that allow for greater flexibility. (**Geek Talks, 2007**; **Stackoverflow, 2009**)

2.10 Application Programming Interfaces

An Application Programming Interface (API) allows a developer to make one application communicate with another. APIs are created purposely to allow specific functions or data to pass from one application to another.

For example when developing an application for Mobile devices that requires the use of Social Media data, a developer can use the Facebook API to import data in the right format as well as use the API to post information back to Facebook from the mobile app. **(Beal, 2014)**

2.11 Mobile Application Security

Application security is the use of hardware, software or programming methods to protect the data within an application from intrusion (**Rouse, M 2014).** It has become more of a concern for mobile phones since the introduction of Smart Phones as typically, personal or business information is kept on them in some way. (**Park, 2014).**

2.11.1 Threats and Consequences

Threats to mobile security can be anything from disrupting the normal operation of the phone or transmitting, modifying or destroying the data kept on the device.

There are many types of threats to the security of an Application, below is a small sample of examples. The data kept on the device is a primary target as Smart devices are used to manage data; people can use them to purchase products online, to store other peoples contact details or manage their schedule through the use of a calendar. If this information is stored on the device than a hacker can potentially steal business information or even your credit card details as a payment is being made.

Attacks can be malicious instead of for theft if an application uses a database to store data then the application can be susceptible to **SQL injection** unless it is protected properly. SQL injection can be used to add, modify or delete specific data or even destroy the whole database by dropping the tables along with all of its data.

Buffer Overflows are another threat; this is when an application tries to store more data into memory than it was allocated, which could potentially overwrite data being used by either another application or the application in question. This overwrite can be used by hackers to gain access to sensitive data that they should not.

Bad security for an application, or even within the Operating System itself, can lead to lost, stolen or unrecoverable personal data either through malicious or accidental reasons, full system crashes or application crashes which effectively denies the user the use of the service or product, which links back to bad User Experience.

If users have personal data stolen, this could lead to legal action being taken against the company, which could cost a company massive amounts in legal fees and settlements. If a User's experience is bad, then they will seek alternative products or services and they will not recommend the product or service meaning less money for a company. **(Yoder, J. and Baracalow, J. 1997), (Enck, W. 2011).**

Chapter Three – Analysis

3.1 Introduction

The purpose of this section is to give a detailed description of the "User Friendly Calendar App". It will explore What an App is, the purpose as to why the system is being developed as well as looking at the system requirements (both functional and non-functional), the constraints of the system, the interface of the system and how the user will interact with the various functions.

This section will also give a detailed description of the Applications that were tested and the analysis of the results received from the surveys and SUS Analysis.

3.1.1 What is a Mobile Application
A Mobile Application, or app, is software that has is designed to run on mobile devices such as Tablets, Smart Phones or Smart Watches. Typically they are designed to provide the user with services or functions similar to those that are available to them on a standard PC or laptop such as Internet Browsing, Games, Social Media, Word Processing, Image processing or text and video based communication. They can also be used to provide useful utilities such as Calorie tracking, step counting and distance travelled that make life easier for a user. These apps are generally small pieces of software that are designed with a handful of functions and features in mind and they do not operate outside of those functions.

3.1.2 Who uses Mobile Applications
Mobile apps can be used by anyone, from audiences looking to chat or play games with their friends and family online, business people, students and tutors that might require the need to track their day-to-day meetings and schedules or even the elderly that might need a reminder to take medication at a specific time of day.

3.1.3 User Scenarios
It is important to define user scenarios for a project as it helps when deciding the types of features and functions to include in an application or system as well as who the product is primary being aimed at.

3.1.3.1 Reminder Tool
Students often have busy schedules and multiple deadlines to meet; they need to be able to manage their time effectively without forgetting about meetings, exam times or assignment deadlines. The product of this project will help them to achieve their goals by taking some of the edge off of their stressful schedules by allowing them to set themselves reminders and be alerted prior to an event happening.

3.1.3.2 Learning Tool
When starting a new unit or exploring a unit into more detail, the information given can be too much too quick and can feel quite daunting, especially if you do not understand the new topic straight away. As well as being a functional tool that a student can use to manage their day-to-day activities, the source code of the application will be open, allowing a student that is new to Mobile Development to look at how data is managed within a database on a mobile device in a relatively simple form. Allowing them to see a simple, working example of code in action.

3.1.4 Why am I creating the Calendar Application
The reason for creating the User Friendly Calendar application is to give students an easy way to manage their busy schedules that is separate from their daily schedules, at the same time being more user friendly than some of the applications that are currently available. Separating their University schedule from their work or life schedule would potentially reduce the chance that an event will be missed, as calendars can become full very quickly, especially for an out-going individual.

3.2 Current Existing Applications
Participants were asked to use and evaluate between three and five of the given Android calendar applications. The questions they asked consisted of the features that they did or did not like about the application, why they did or did not like the features and what they thought could be added to improve the application. They were also asked to fill in a SUS form for each application that they

tested. The most popular to be chosen were Google Calendar and Jorte Calendar as these were both tested and reviewed by all five participants.

The average score for a SUS analysis is 68, any SUS score above 68 can be considered to be above average and anything below 68 can be seen as being below average. Although a SUS score ranges from 0 to 100, it is not a percentage. If a system scores 70, it is close to the average, which can be seen as being 50%.

3.2.1 Business Calendar

The Average SUS Score based on three participants: **50**. This application was not so liked by those that tested it, receiving an average score well below the average.

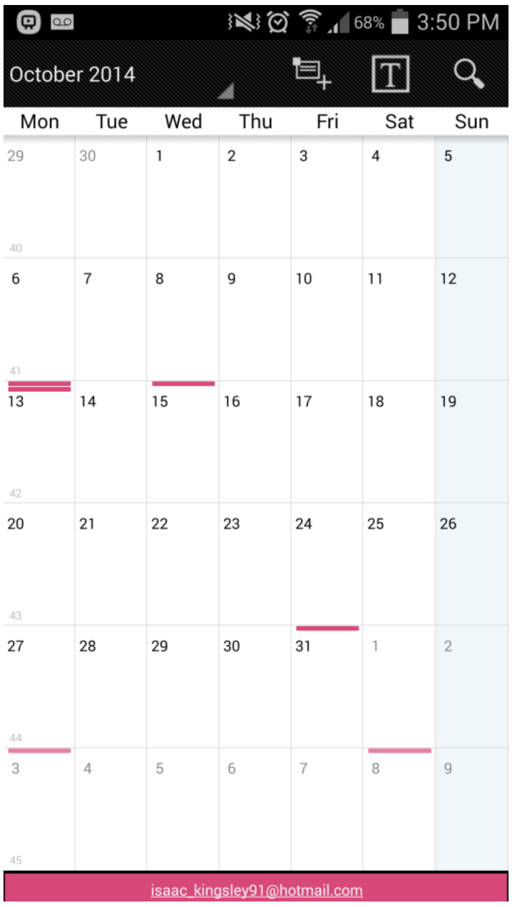


Figure 3 - Business Calendar Interface

3.2.1.1 Application Description

The Business Calendar is designed more for Business use than personal use; it gives the user the ability to label events into categories, the categories are then represented using different colours. The user can hide and unhide these different categories with a tap to make reading a schedule easier. Like the others, it provides the user with the ability to synchronise their Google calendar events.

3.2.1.2 Survey Feedback Analysis

Users liked that the calendar worked well with displaying data from multiple calendars without the need for much messing abound. They also liked the selection bar at the bottom that allowed them to quickly hide specific calendars so that they could read events quicker.

What they did not like was the fact that the buttons used to hide and show different calendar groups were so small, making them hard and awkward to press.

3.2.2 Fertility Calendar

The Average SUS Score based on two participants: **90**.
The application was highly rated by the participants, although this may have been different if there were more reviews of this app.

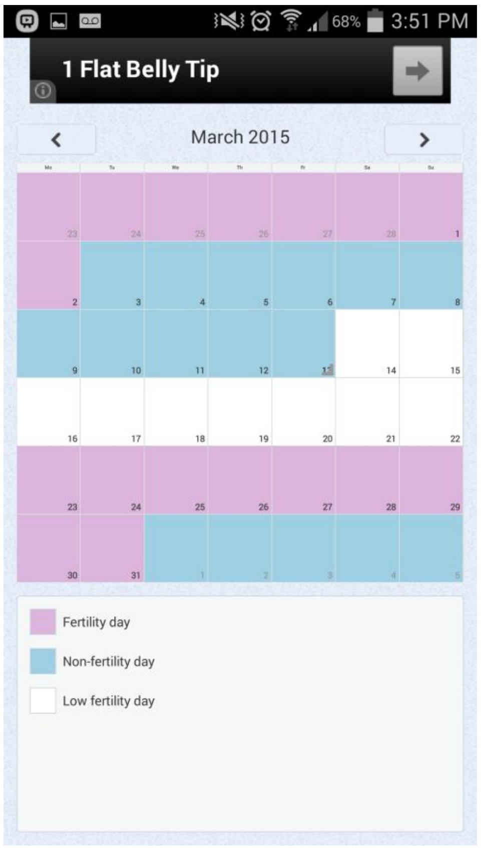


Figure 4 - Fertility Calendar Interface

3.2.2.1 Application Description

The Fertility Calendar is free app that allows the user to predict when they non-fertile, have low fertility or high fertility throughout a given month based on the information that is input by the user.

It has a simple and minimalistic UI; it is quick and easy to learn to use as well as being fast to navigate. The information given is easy to read due to the uncluttered display of the calendars.

3.2.2.2 Survey Feedback Analysis

The testers seemed to like the colour separation between fertility states as it made it quick and easy to distinguish between fertility levels at a glance.

The main feature that was not well received was the feature about temperature, as there is no explanation within the application as to why this information is relevant. Another un-liked feature was the ads, but as the app is free, the ads are the main source of income for the developer.

3.2.3 Google Calendar

The Average SUS Score based on five participants: **86.5**. The application was well received by the testing participants, scoring quite highly above the average SUS.

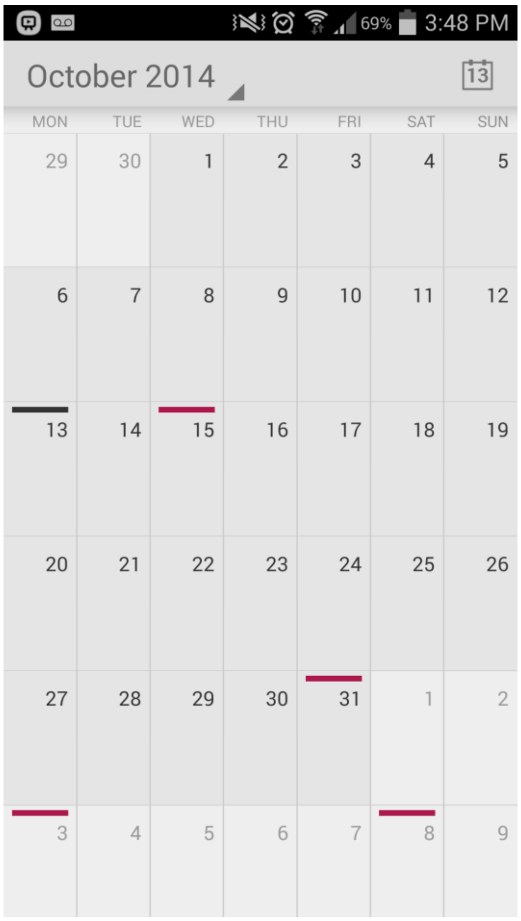


Figure 5 - Google Calendar Interface

3.2.3.1 Application Description

Google Calendar is a powerful multi-platform calendar and it is free to use, users can manage their day-to-day schedules and meetings from many devices and have all of those devices synchronise automatically with each other. For example, using the web interface to add a meeting to the calendar will result in the users' phone giving them an alert if the option to notify before the event occurs was set and also if a location was selected for the event, then directions to the location via Google maps will be made available to the user. Users can also sync dates and events to the app by logging into the account through their mobile device.

It has a simple main UI that allows the user to switch between views, including a scrollable schedule of day to day events and notifications such as Birthdays, a day view that shows events hour by hour, a 5 day view similar to the single day view just with more days, Month selection and a search function. Users can also show events from other calendars and birthdays of their Facebook friends. This can sometimes cause schedules to become quite cluttered so users can also choose to hide and show certain imported calendars with the tap of a finger.

3.2.3.2 Survey Feedback Analysis

Participants particularly liked the account-syncing feature that allowed them to edit and manage entries from their phone or computer and have all their devices synchronise with each other. Another feature that stood out was the location services available; if a location for an entry is entered then Google maps will give directions when the alert for the event goes off. Another feature that was liked was the "repeat" option that allowed the user to set an event as a repeat event for a finite or infinite amount of time each week. This was liked as it saved the user time, as they would only have to add the event once.
If the user has added their Facebook account to their android device then the app will allow the user to show their friends birthdays on their calendar.

What was not well liked was how the Month View looked as the user scrolled, as the view scrolls like a webpage rather than in sequential blocks. This means that that distinguishing between months at a glance is difficult.
The monthly view looked very messy when there multiple different events such as birthdays and meetings all squashed together.
When selecting which of the calendars to display from the menu, if there are multiple accounts such as a Samsung account, Google account and email accounts, the view can become cluttered.

3.2.4 Jorte Calendar

The Average SUS Score based on five participants: **68**.
The application scored dead on average.

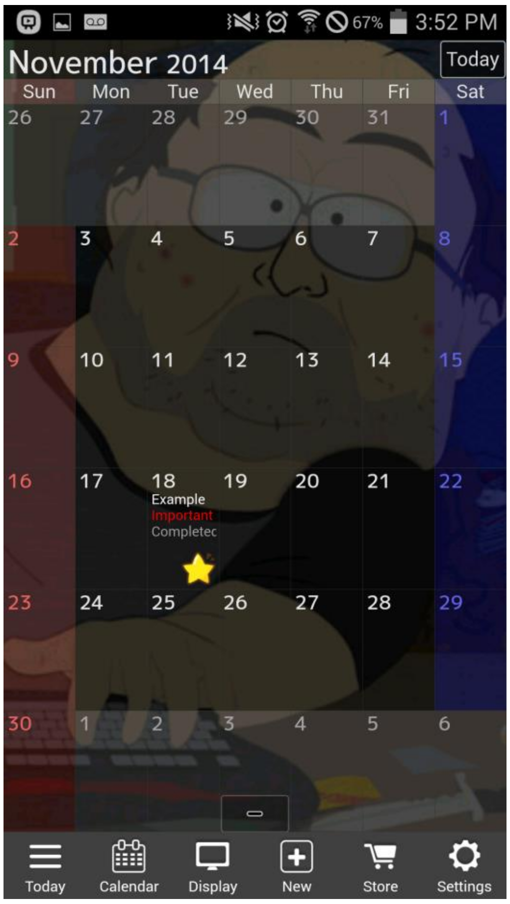


Figure 6 - Jorte Calendar Interface

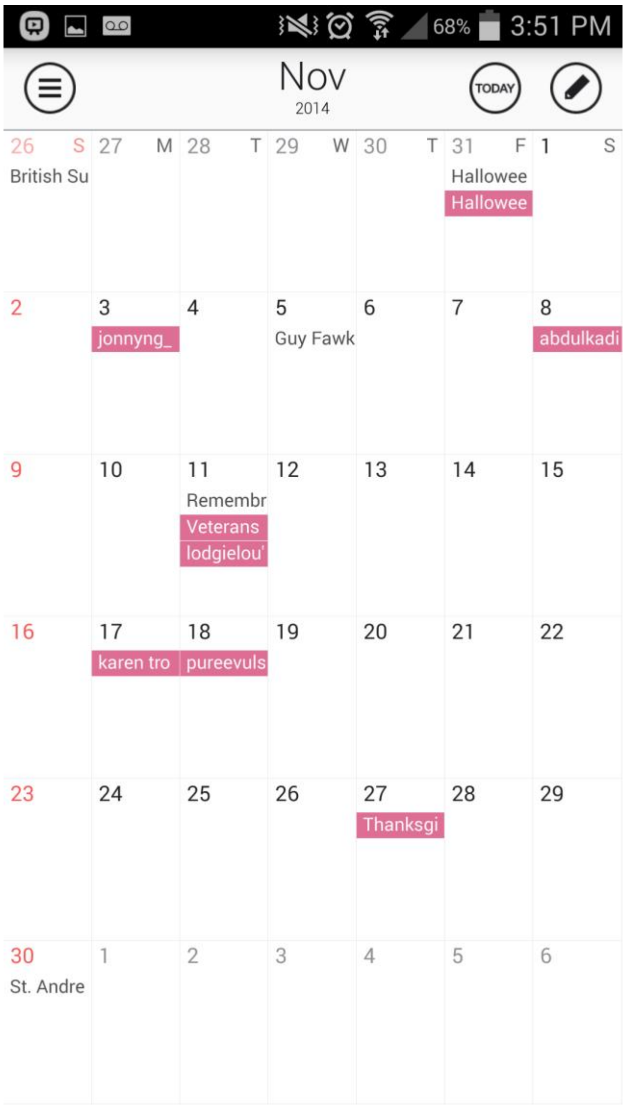
3.2.4.1 Application Description
Like Sol Calendar, Jorte Calendar also has a similar To-Do list. These tasks are displayed in a container at the bottom of the calendar rather than being a separated view like in Sol Calendar. It allows for more customisability with the look and feel as the Calendar can be given themes and backgrounds.

Unlike the others, Jorte does not give the option of different view types and sticks to the monthly style view. Tapping on an event will list that day's events in a container in a list style.

3.2.4.2 Survey Feedback Analysis
Testers liked that the navigation bar was static at the bottom of the screen allowing for quick and easy navigation. They also liked that they could change the colour scheme of the calendar, it is easier to read white text on a black background.
The diary feature was liked as it added multi-functionality to the application.

What was not liked were the backgrounds, as some backgrounds conflict with the colour of text making it hard to read.

3.2.5 SOL Calendar
The Average SUS Score based on two: **77.5**. The application scored above average.



S	M	T	W	T	F	S
26 British Su	27	28	29	30	31 Hallowee Hallowee	1
2	3 jonnyng_	4	5 Guy Fawk	6	7	8 abdulkadi
9	10	11 Remembr Veterans lodgielou'	12	13	14	15
16	17 karen tro	18 pureevuls	19	20	21	22
23	24	25	26	27 Thanksgi	28	29
30 St. Andre	1	2	3	4	5	6

Figure 7 - SOL Calendar Interface

3.2.5.1 Application Description

Sol Calendar has a sleek and modern looking UI that is smooth to navigate around. Unlike the others, it has a To-Do list to help a user manage their day to day tasks and track which of those tasks have been completed. Tasks added to the to-do list can be assigned to different days.

Like the others, the app allows the user to synchronise their other calendars through email address for example, a user's Google Account calendar entries can be imported.

3.2.5.2 Survey Feedback Analysis

Participants liked the ability to sync their Google Tasks to the calendar and To-do list. They also liked the daily view as it allowed for fluid navigation and displays uncluttered information.

What testers did not like was the sticker on new events as they take up too much space on the monthly overview.

3.3 Product Perspective

The product will consist of two main parts, a mobile android application and an internal database. The mobile application will be used to view, add, edit and delete information while the database will be used to store this information.

The mobile front end will allow the user to interact with the system, the user will provide the application with the data and inputs necessary to create, view, delete or edit entries to the system.

The application is data-centric, so the internal database will be used to store this data. As the database is essential to the functionality of the application, the application will check for the existence of a database when it runs and will create one if it does not exist. If a database already exists then the data will be called and displayed to the user. The database will be kept internal on the device to negate the need of an Internet connection, making the application portable.

The mobile application will need to have some restrictions in regards to allocation of resources. The application will have to operate using a minimal amount of Memory while it is running to avoid starving the other Android Systems or other applications of the resources that they need.

3.3.1 Assumptions about the Project

The main assumption of the Application is that the user will be using a Smart Phone that has at least the minimum recommended hardware specification (Processing power, Storage space and Memory).

If the system resources are not currently available due to another application using them or the minimum spec of the device are not met, then there may be cases where the application is used on such a device which may result in poor performance issues or the complete non-functionality of the application.

3.3.2 The User Interface

When the application is opened for the first time the user will be presented with the main calendar overview. As it will be the first time running the application there will be no data for the calendar for the user to see.

The user will be able to cycle through months in order to create future events and reminders. Tapping on a date will allow the user to manage the events associated with that particular date.

3.2.3 Communication within the Application

The application depends on the database so they need to be able to communicate. For software level communication, Data Access Objects and Database Helpers may need to be implemented that allow the two components to communicate successfully. The Android operating system handles how the application communicates with the stored database on a hardware level via imported libraries into the application.

3.4 Functional Requirements

In software engineering terms, a functional requirement defines the functions of the system; these functions include user inputs, the behaviour of the system and outputs of the system. The functional requirements for the Calendar Application have been split into two categories, Essential and Desired.

3.4 1 Essential Requirements

These functional requirements make up the core of the system. Without them the system will not be able to perform even the basic operations.

Functional Requirement Number	Title	Description	Dependencies
FR1.1	Database	The application will need a database to store user data.	No dependencies.

FR1.2	Add new Event	A user should be to create a new event on the Calendar.	1.1
FR1.3	Edit existing Event	A user should be to go back and edit the details of an existing event.	1.2
FR1.4	Delete existing Event	A user should be to delete an existing event from the calendar.	1.2

Table 1 - Essential Requirements

3.4.2 Desired Requirements

The Desired requirements are those that are not necessary for the system to operate but add extra functionality that would be useful to the user.

Functional Requirement Number	Title	Description	Dependencies
FR1.5	Download mobile Application	A user should be able to download the application either through an application store or via a website, free of charge.	No dependencies.
FR1.6	Search Function	A user should be able to search using keywords for an event, e.g. searching for 'Kingsley' would return all instances where the event contains the word Kingsley, such as a birthday.	1.2
FR1.7	Search Function – No results found	A user should be informed when a search returns no results.	1.6
FR1.8	Search Function – Sorting	The results of a successful search should be displayed in order of soonest first.	1.2, 1.6
FR1.9	Search Function – Filtering	A user should be able to filter out certain results, for example, Birthdays.	1.2, 1.6
FR1.10	Import Birthdays from Facebook	A user should be able to add their Facebook friend's birthdays to the Calendar by synchronising through Facebook.	1.1
FR1.11	Change event colour	A user should be able to change the colour of an event on the calendar to	1.2

		make quick recognition of different types of events easier.	
FR1.12	Localisation	The application should detect the default language of the system and change the language of the application to match.	No dependencies.

Table 2 - Desired Requirements

3.5 Non-Functional Requirements

In software engineering terms, the non-functional requirements describe how a system performs its functions and the constraints of those functions.

Non-Functional Requirement Number	Title	Description	Dependency
1.1	Usage of the search function	The search function should be easy for the user to find and use.	FR1.6
1.2	Usage of the search results	The search results should be displayed in a user friendly way and should allow the user to interact with the results.	FR1.6
1.3	Device Storage	The device will need storage space to store both the application and the required database. The application itself should take up as minimum space as possible.	No dependencies.
1.4	Memory Usage	The application should run using as little system memory as possible.	No dependencies.
1.5	Internet Usage	The application and its main features should be able to run independently of an Internet connection.	No dependencies.
1.6	Portability	The application will be portable, as it has no Internet dependency.	No dependencies.
1.7	Maintainability	The application and its features should be written in a way that helps promote upgradability and new	No dependencies.

		implementations. I.e. Features are contained within modules that are called when they are needed rather than all features being written into one big module.	
1.8	Testability	The system should be built in a way that allows for the complete testing of the applications functions that would represent real life use.	No dependencies.
1.9	Documentation	Detailed documentation that explains to a user how each function works.	No dependencies.
1.10	Fault Tolerance	Inputting incorrect data should not damage or render the application unusable.	No dependencies.

Table 3 - Non-Functional Requirements

3.6 Development Method:

The method that will be used during the development of the project will be the Iterative Waterfall method. This method was chosen as it allows for the project to be split into smaller, manageable chunks that can be designed, implemented, tested and evaluated individually. The components that make up the core functions and features of the project are modular by nature, as in they can mostly function on their own without depending on another function, with the exception being adding a new event which depends on the database being present. This modular approach fits perfectly with the Iterative Waterfall development method. The first thing to be implemented will be the database.

Chapter Four – Design

4. 1 Navigational Diagram

The storyboard for the flow of the system is Hierarchical. This means that the structure of the system consists of multiple levels rather being linear and on a single row.

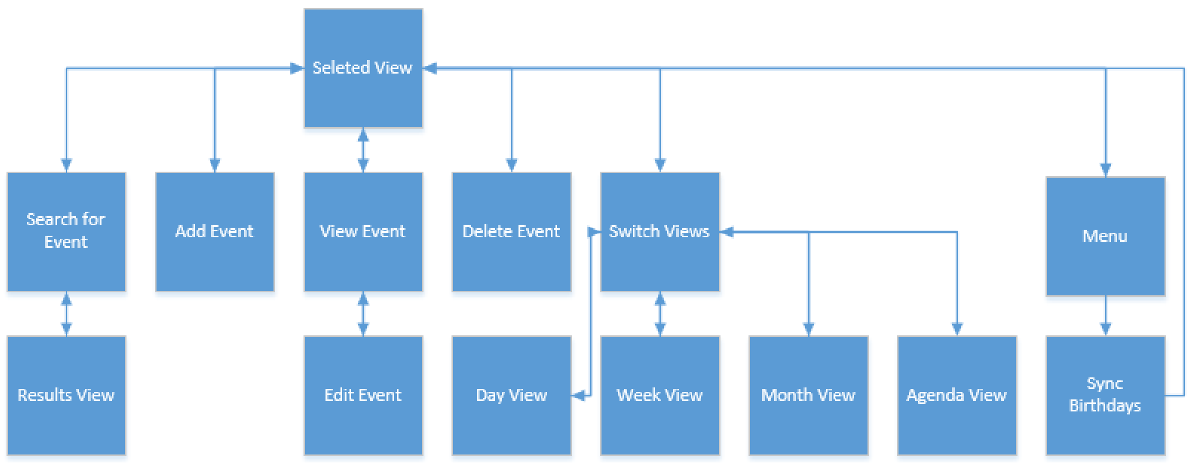


Figure 8 - Navigational Diagram

From the selected view the user will be able to search for events using keywords, the results of the search will be displayed in a linear overview. They will also have the ability to add a new event, view existing events and edit their details as well as delete events. The user has four views to choose from when selecting a view, a Daily view, a weekly view, a monthly view and an Agenda that lists that day's events. From the menu the user will be able to sync their friend's birthdays from Facebook.

4.2 Prototype One Design

4.2.1 Month View

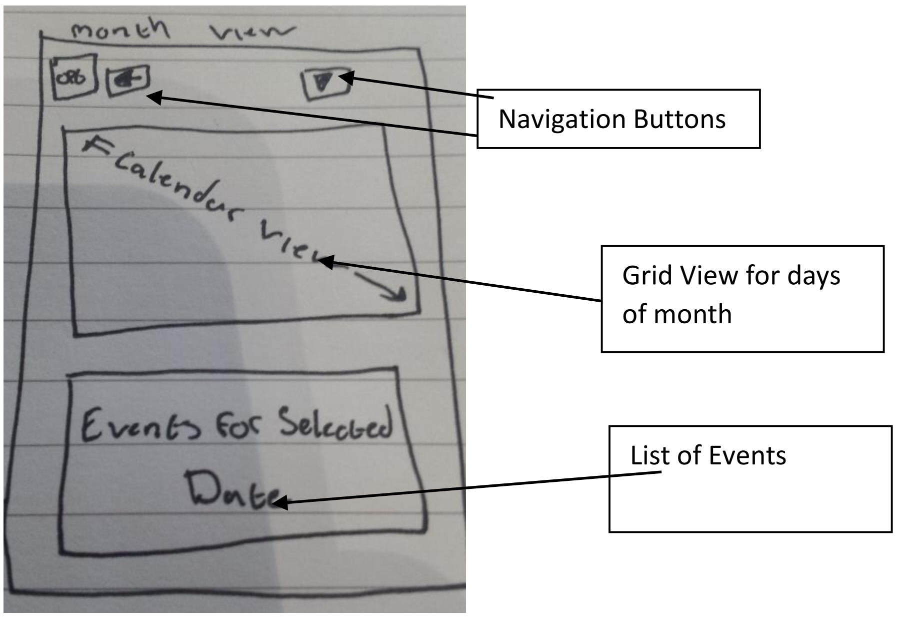


Figure 9 - Prototype One, Month View Design

The month view will have 2 sections. The first section will have the calendar, the month selection buttons and the option button. The dates for the month selected will be displayed to the user for them to select. The menu button will give the options to change the view and add a new event.

In the second section, the title of the event(s) associated with the date that the user selected will be displayed. From here they will be able to interact with an event to edit its information or delete the entry completely.

4.2.2 List (Day) View

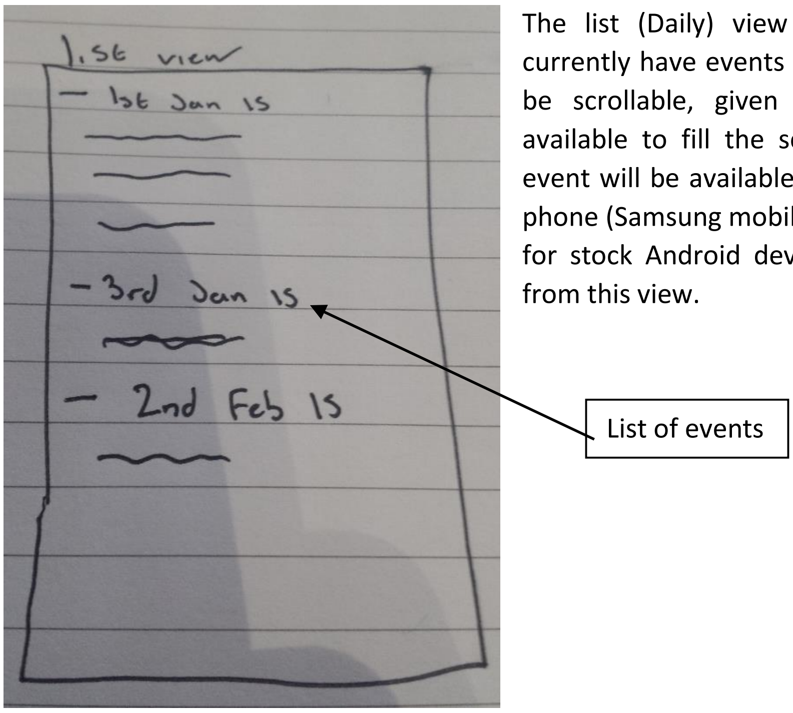


Figure 10 - Prototype One, List View (Day)

The list (Daily) view will show a list of dates that currently have events associated with them; this list will be scrollable, given that there are enough entries available to fill the screen. The option to add a new event will be available through the menu button on the phone (Samsung mobiles) and the software menu button for stock Android devices. The user can delete events from this view.

4.2.3 Add/Edit Event

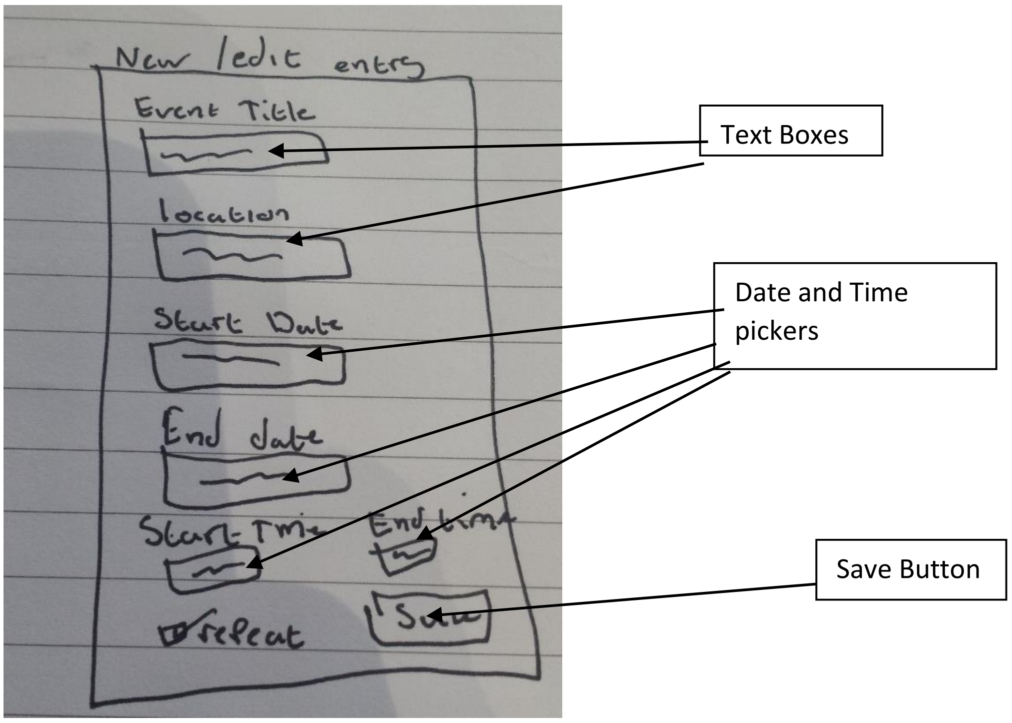


Figure 11 - Prototype One, Add / Edit Event Design

The Add entry and Edit Entry screens will be the same. If the user creates a new entry then the text boxes will be populated with default values such as the start date being 'Today', the start time being 12:00 and the end time being 12:01. Pressing the save button will save the new entry to the database.
If the user selected an existing entry then the text boxes will be populated with values retrieved from the database.

4.2.4 Menu

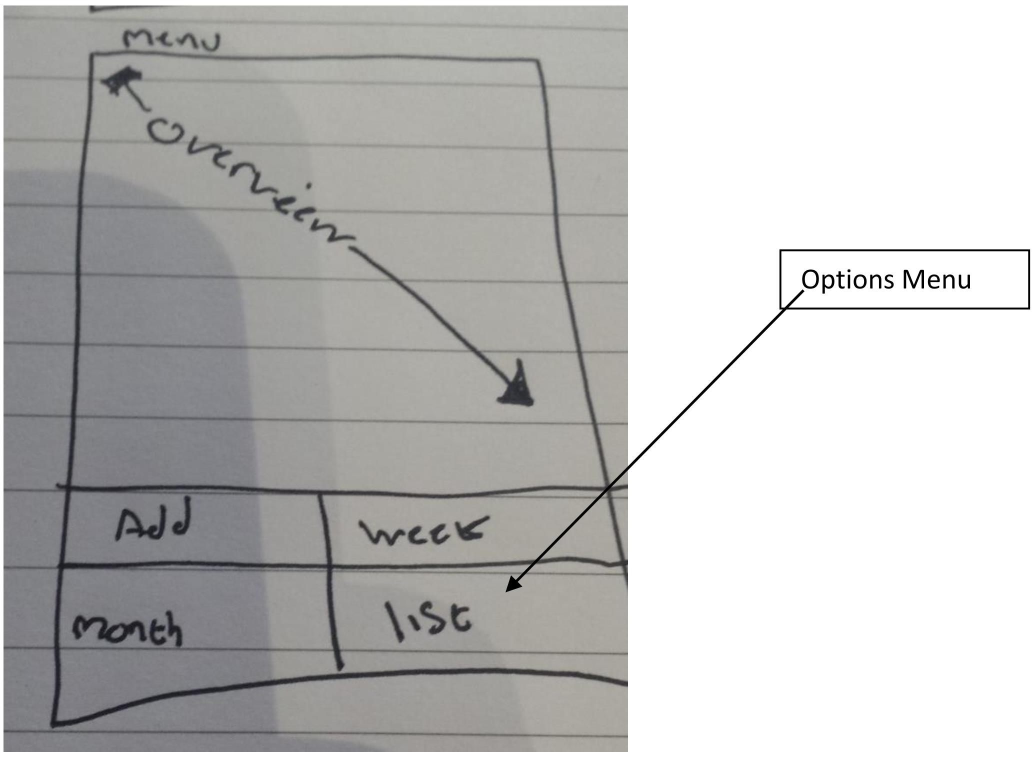
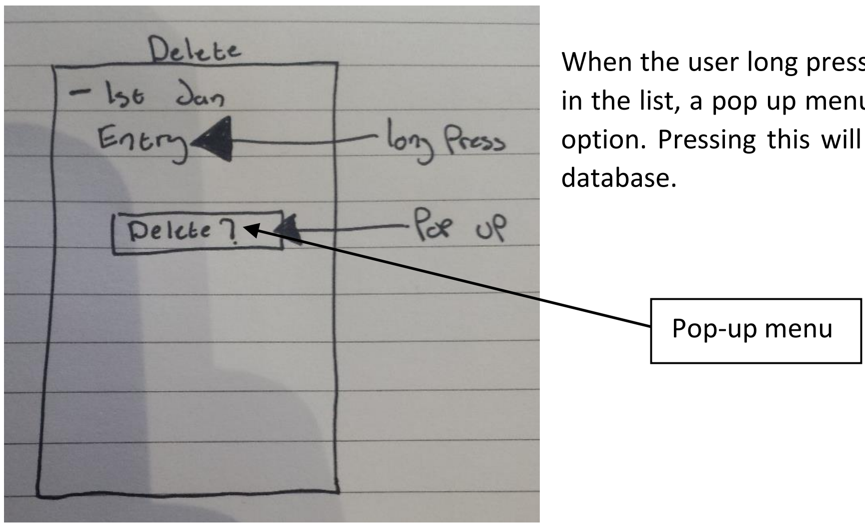


Figure 12 - Prototype One, Menu Design

When the menu button on the device is pressed the options will be shown to the user at the bottom of the screen. The options will be switching between views and creating new events for the date selected.

4.2.5 Delete



When the user long presses (taps and holds) an event in the list, a pop up menu will appear with the delete option. Pressing this will remove the entry from the database.

Figure 13 - Prototype One, Delete Option Design

4.2.6 Search

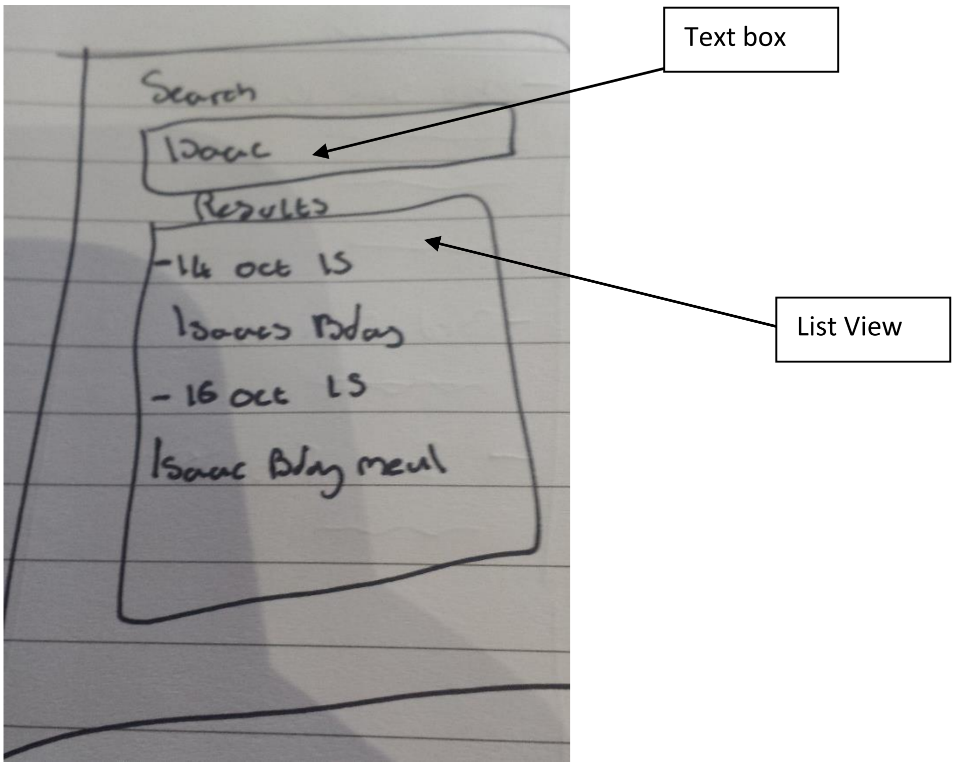


Figure 14 - Prototype One, Search Design

The search function will function like the List View but will only show the dates and events that have keywords matching the search parameters.

4.3 Use Case Diagrams

4.3.1 High Level UCD

A High Level Use Case Diagram depicts the main features of a system and how each 'user' can or cannot interact with each feature. For the Calendar App these features are, selecting a Content View, Searching for an event, managing the entries of the Calendar, importing data from other calendars and Syncing birthday data from Social Media. The calendar is only has a single actor.

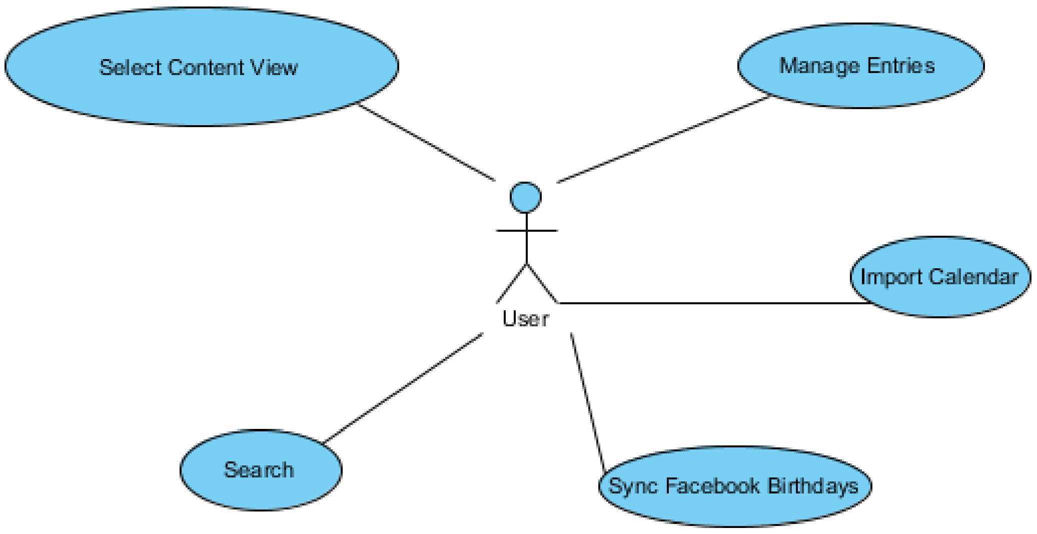


Figure 15 - High Level Use Case Diagram

4.3.2 Low Level UCD

Low Level Use Case Diagrams show a more detailed view of a how a feature works such as Inputs from the user, Outputs by the system and extension functions.

4.3.2.1 Manage

The ‘Manage Entries’ function has multiple extensions; the user can add a new entry, delete an existing entry or edit an existing entry.

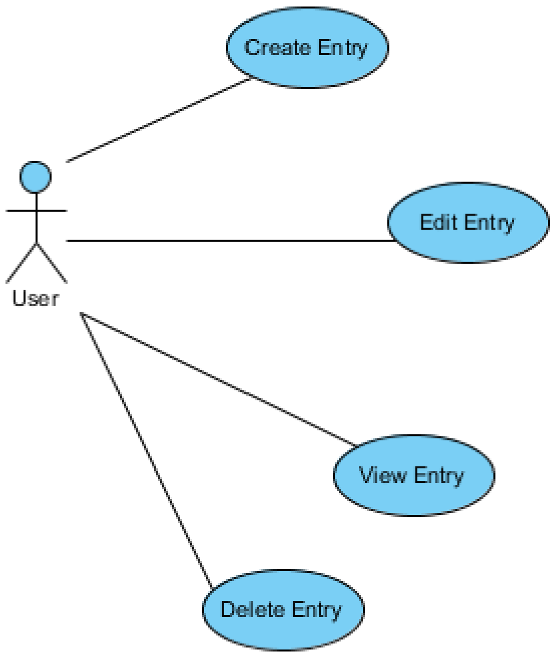


Figure 16 - Low Level Use Case Diagram, Entry Management

4.3.2.2 Content View

Similarly the ‘Select Content View’ function gives the user the choice to choose between Agenda, Daily, Weekly and Monthly views.

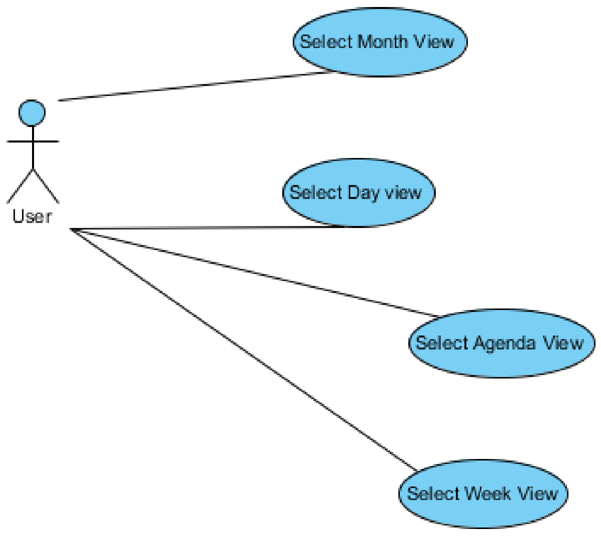


Figure 17 - Low Level Use Case Diagram, Content View

4.3.2.3 Search

When the user searches, the results are presented in an interactive result set. The user will be able to refine these results, for example, only showing results that are between month X and month Y.

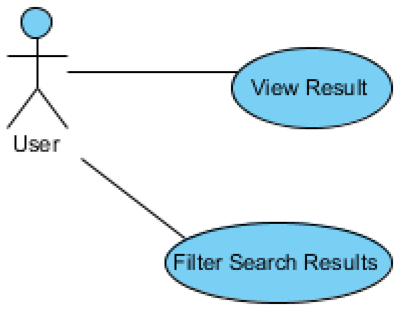


Figure 18 - Low Level Use Case Diagram, Search Function

4.3.2.4 Import

When a user wants to import from other Calendars, they must select the calendar they wish to import, for example using their Google Calendar email address or their Samsung calendar address.

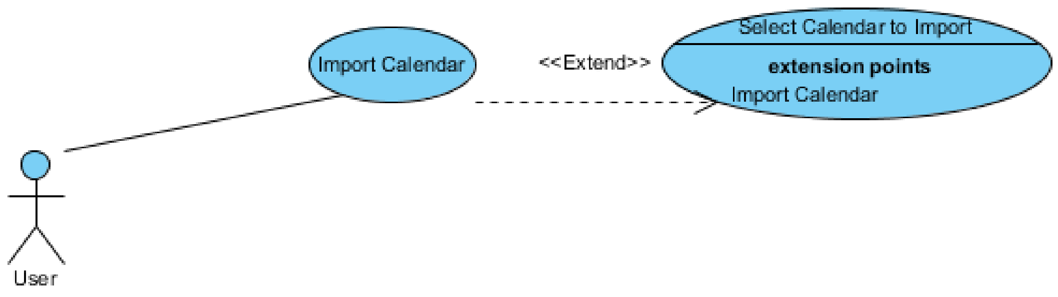


Figure 19 - Low Level Use Case Diagram, Import from Calendar

4.3.2.5 Facebook

If the user wants to sync data from Facebook then they must login to Facebook via the app to allow the data to be imported.

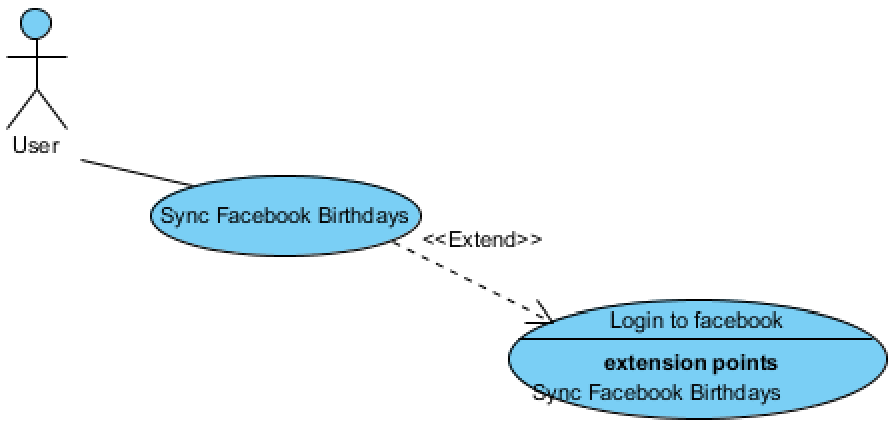


Figure 20 - Low Level Use Case Diagram, Sync with Facebook

4.4 Data Flow Diagrams

4.4.1 Context Level Diagram

A context diagram generalises the functionality of the system into a single process. The user interacts with the system by inputting data and the system interacts with the user by displaying information. The system retrieves data from the database as well as sending data to the database to be saved.

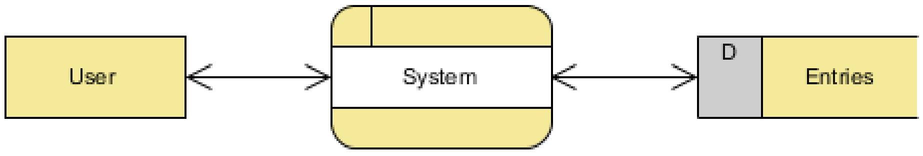


Figure 21 - Data Flow Diagram - Content Level

4.4.2 Level 1 Diagram - Edit Entry DFD

When the user needs to edit an existing entry, the data associated with the entry must be retrieved from storage and displayed to the user. Once the user has finished editing and attempts to save the changes, these changes are validated. If they are values are valid then the changes are committed back to the database, if they are not valid then the issue(s) will be highlighted for the user to see allowing them to rectify the issue(s) before attempting to save the changes again. Once the changes have been saved the screen is refreshed and displayed to the user.

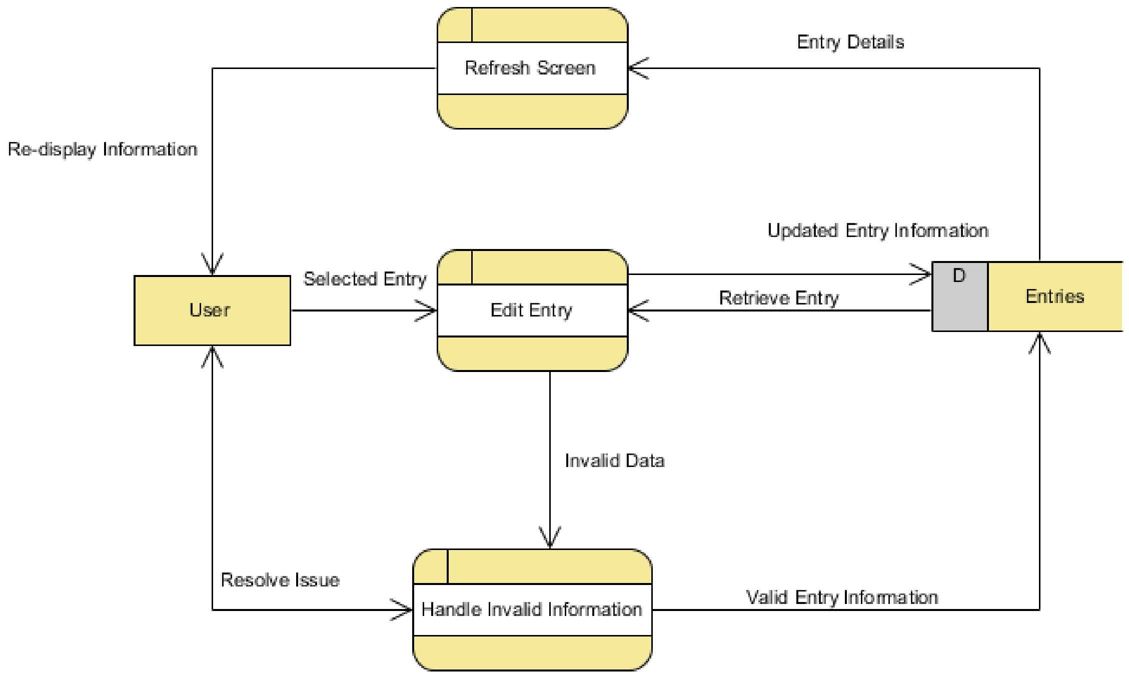


Figure 22 - Data Flow Diagram, Edit Entry

4.4.3 Level 2 Diagram – Refresh Screen

The refresh function will check if an entry has been added or modified, if this is true then data is pulled from the database and re-displayed to the user to ensure that the data and information that is being shown is accurate.

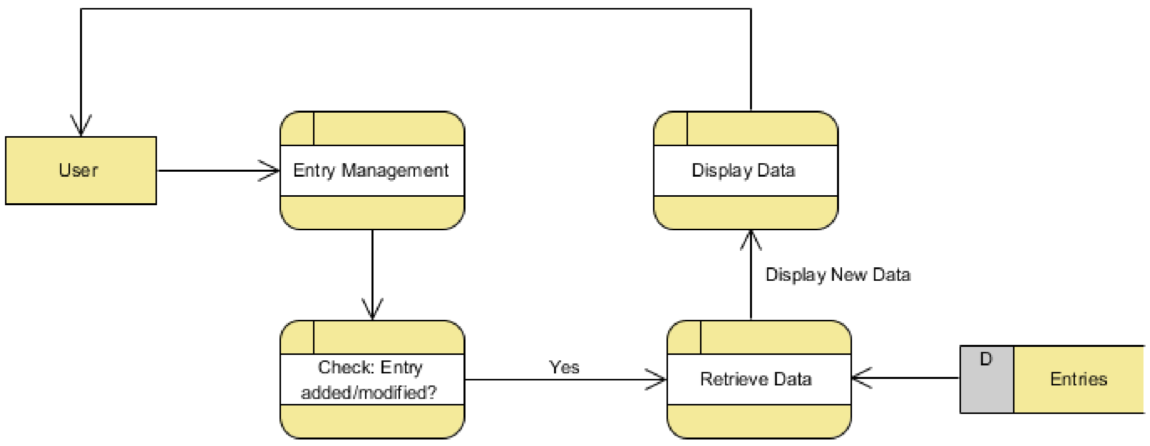


Figure 23 - Data Flow Diagram, Refresh Screen

4.4.4 Level 2 Diagram – Edit Entry

The user selects the entry they wish to edit, the data for that entry is retrieved from the database. Once changes have been made they are updated in the database. The user can cancel an edit and the changes will not be saved.

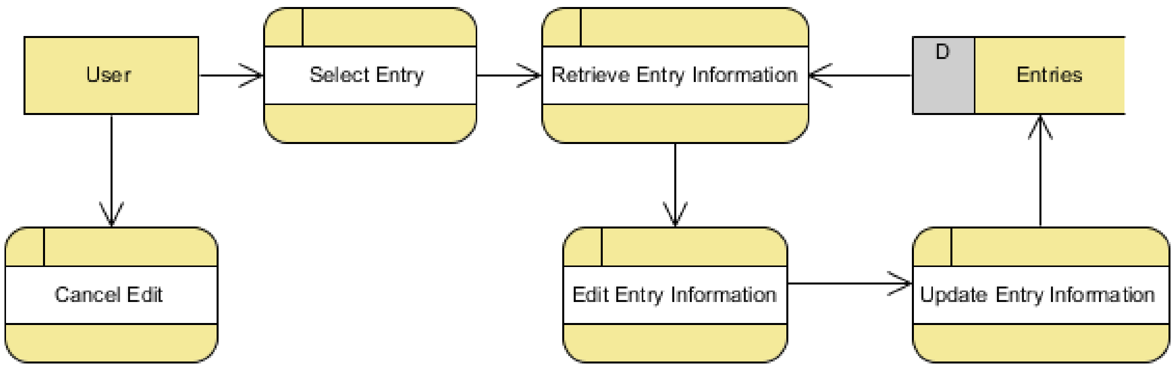


Figure 24 - Data Flow Diagram, Edit Entry

4.4.5 Level 2 Diagram – Handle Invalid Information

If edited values are detected, then the values are checked for validity, if they pass then the data in the database is updated. If the validity check fails then an error message is produced and shown to the user, this process loops until the values pass the validity check, or the user cancels the changes. In which case no update is made.

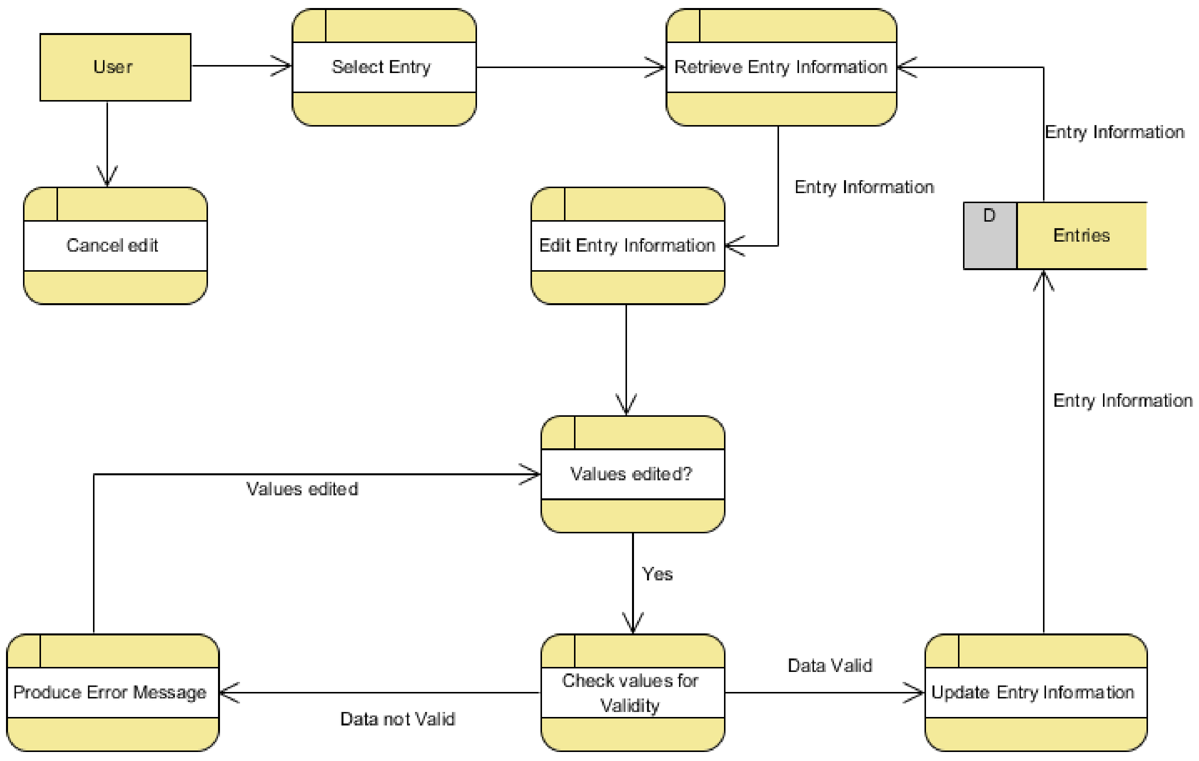


Figure 25 - Data Flow Diagram, Handle Invalid Information

4.4.6 Level 2 Diagram – Cancel Editing

If the user cancels an edit, they are returned to the Content View.

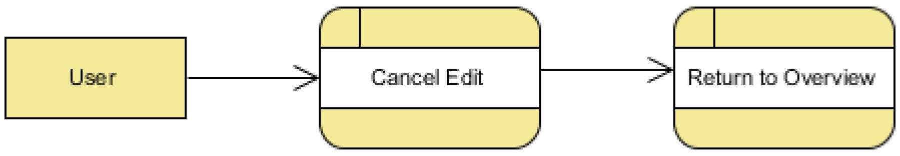


Figure 26 - Data Flow Diagram, Cancel Editing

4.5 Design: Revision One

Due to the feedback from testing the first prototype, users were not overly fond of the design of the interface and the colour scheme. What stood out the most was that interface looked quite old and needed updating to look more modern, navigation needed to be more fluid and easier to see.

To do this I decided to user an Action Bar. The action bar is a more up to date method used in mobile development that helps keep the interface tidy, navigation easier while also helping to save real estate on the display.

4.5.1 Month View

The action bar on the top has a drop down menu that allows the user to select a month, rather than the sequential method that was used in the first prototype. This allows for quicker navigation through the months. The user also has quick access to the search bar and the 'Overflow' button that expands to show more menu options.

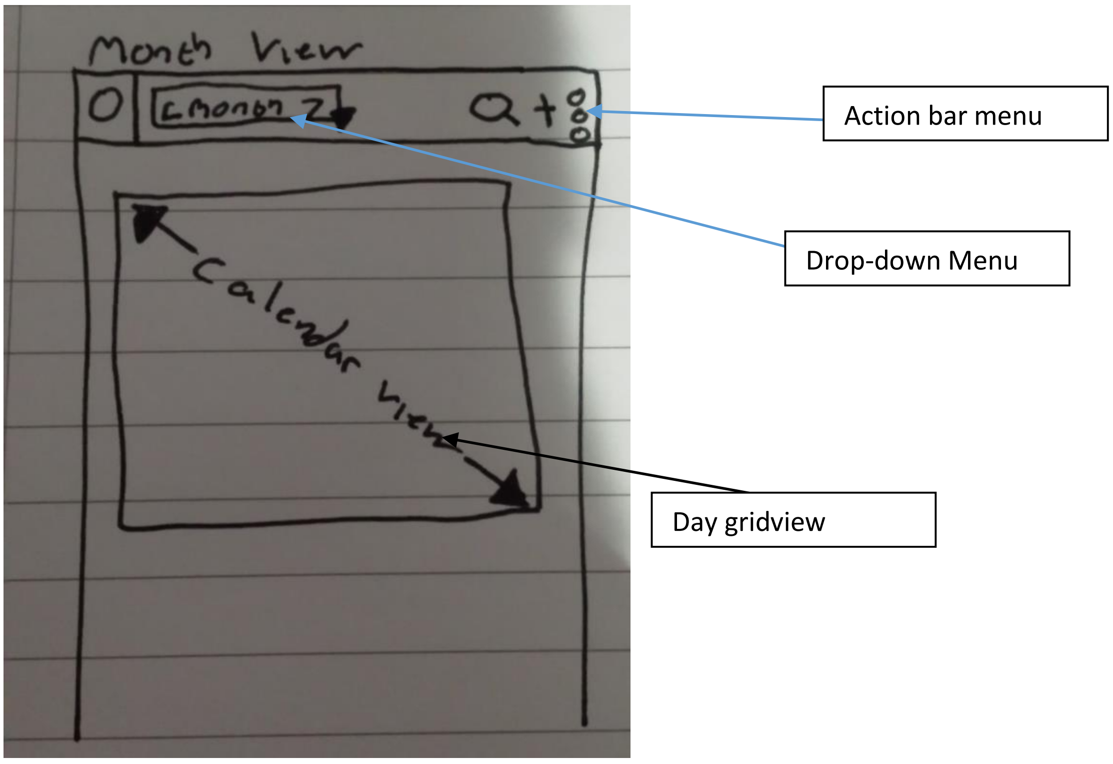


Figure 27 - Prototype Two Design Revision, Monthly View

4.5.2 List View (all)

The style of the list view has been slightly modified. It now shows the title and the date of each entry in the database, the action bar consists of a simple title and a back button that leads to the month view.

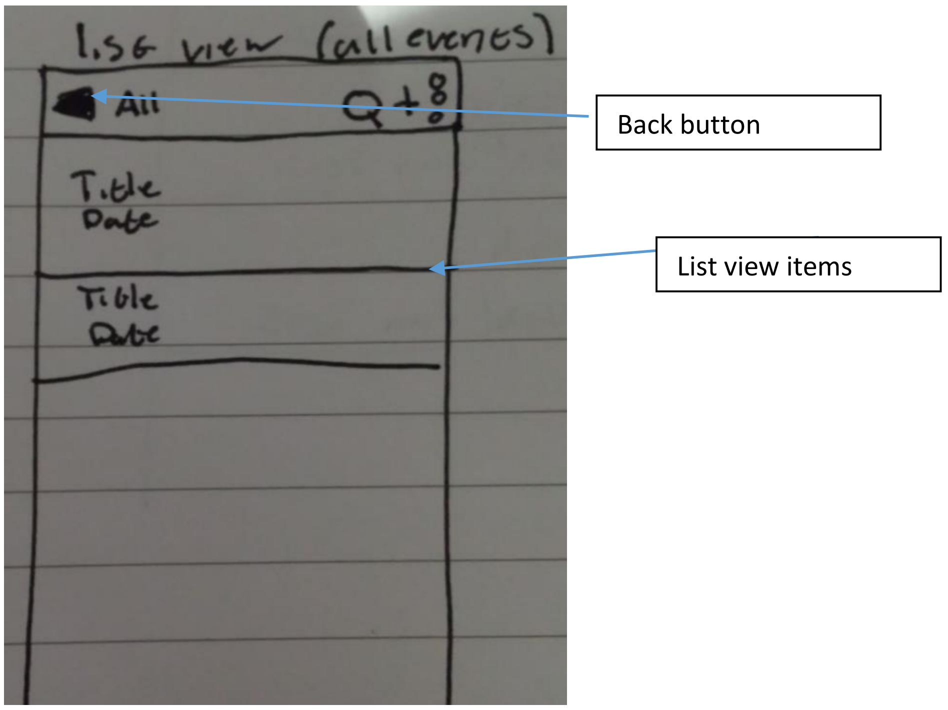


Figure 28 - Prototype Two Design, List View (All)

4.5.3 Day View

Due to limitations of my android development skills, the day view has 2 potential designs. The first is the original design that consisting of a fragmented scrollable list view on the bottom half of the month view that shows the events associated with the date selected by the user. The second is a list view like the "view all" that is shown to the user when a date is selected. The new design has an 'up' button and the date selected in the action bar. The up button returns the user to the Month view. The day view acts and looks like the list view but it only shows the events of the selected date.

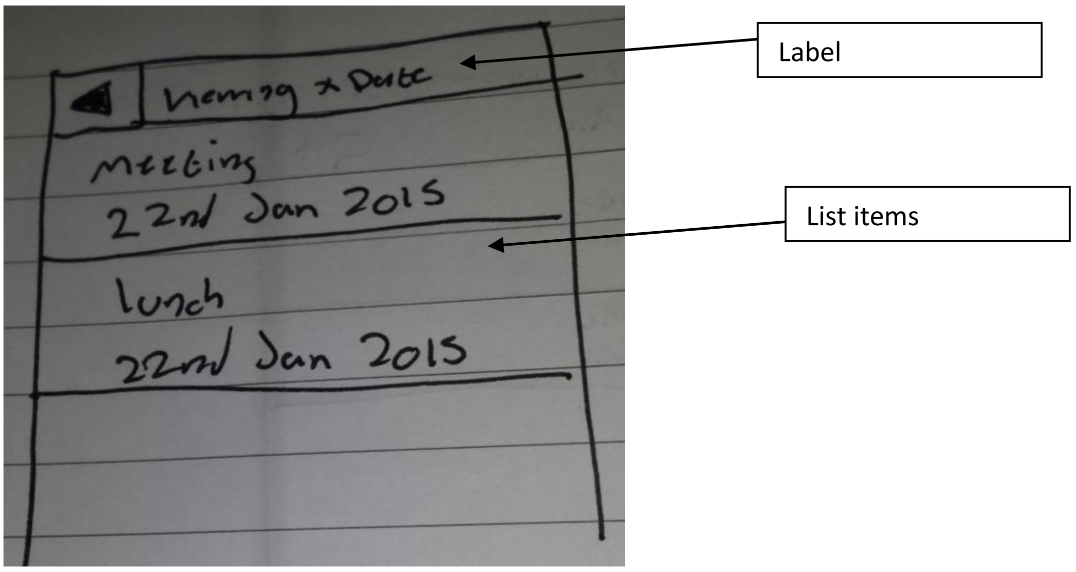


Figure 29 - Prototype Two Design, Daily View

4.5.4 Search

The search function is accessed from the action bar of the month view. When the icon is tapped it will expand into a text box that the user can type into. The results of the search will appear in a list view that the user can select and modify. Tapping the up button will return the user to the month view.

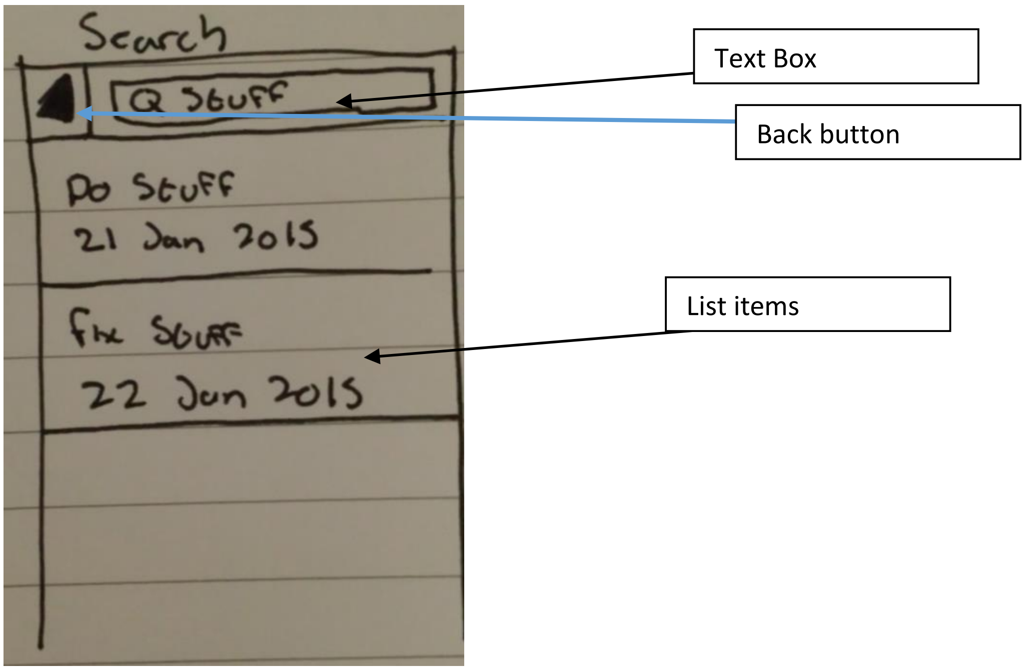


Figure 30 - Prototype Two Design, Search Function

4.5.5 Menu

The main menu is accessed from the action bar at the top of the interface; the functions that would not fit on the bar are placed into the 'overflow' sub-menu button.

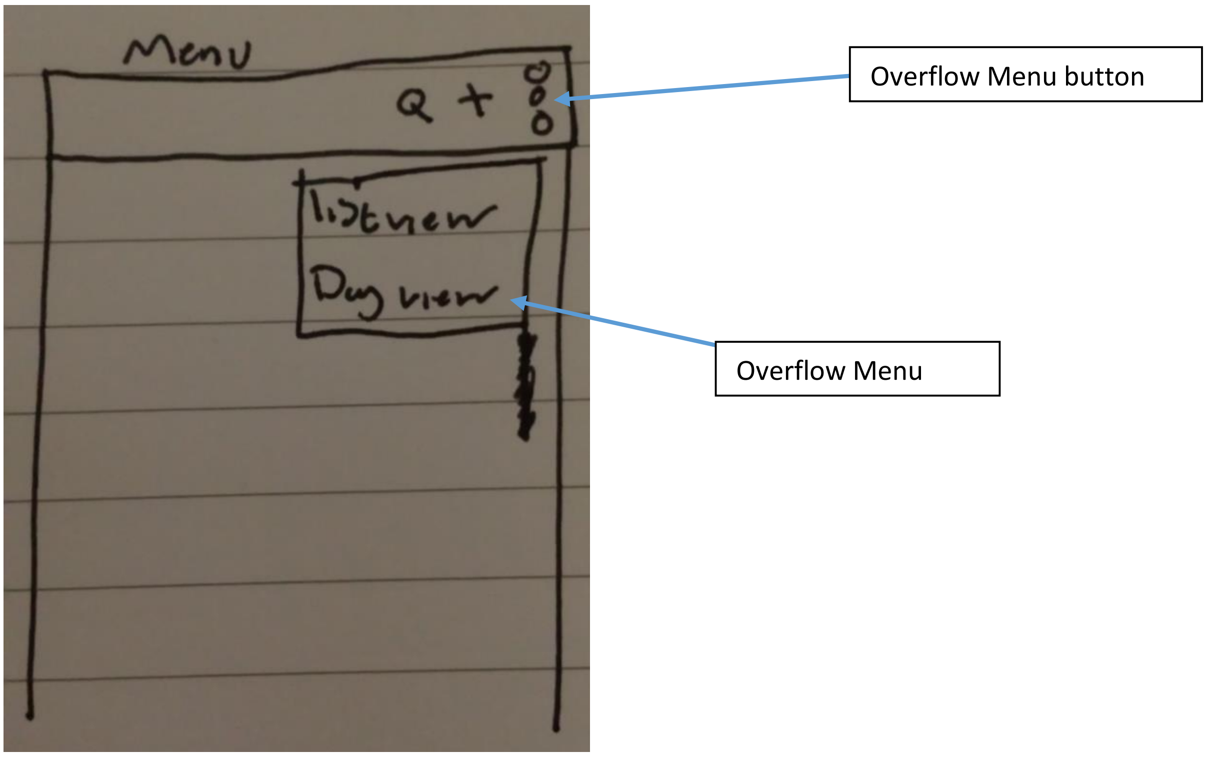


Figure 31 - Prototype Two Design, Main Menu

4.5.6 New Entry/Edit

The new entry screen remains mostly unchanged except for the addition of the action bar at the top. The back button is now located at the top of the screen next to the title. Pressing the Up button takes the user back to the 'Home' screen, which is the month view.

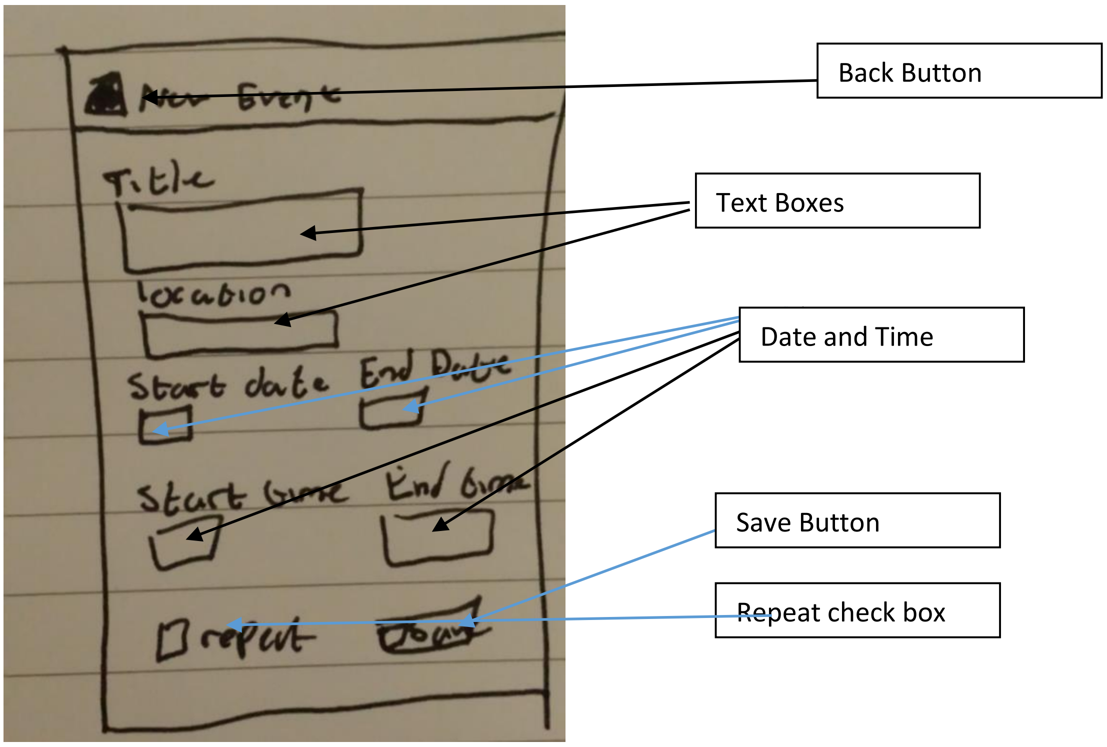


Figure 32 - Prototype Two Design, New / Edit Entry

4.6 Design: Revision Two

After implementing the action bar and modernising the UI of the prototype, users were asked to test the new prototype. The modernisation of the UI was well met but it brought to light a fatal flaw in the new method of selecting months. There was no way of selecting a year, effectively locking the calendar to 2015. This was not a hard task to fix but it meant moving around the options on the action bar of the month view.

4.6.1 Calendar View

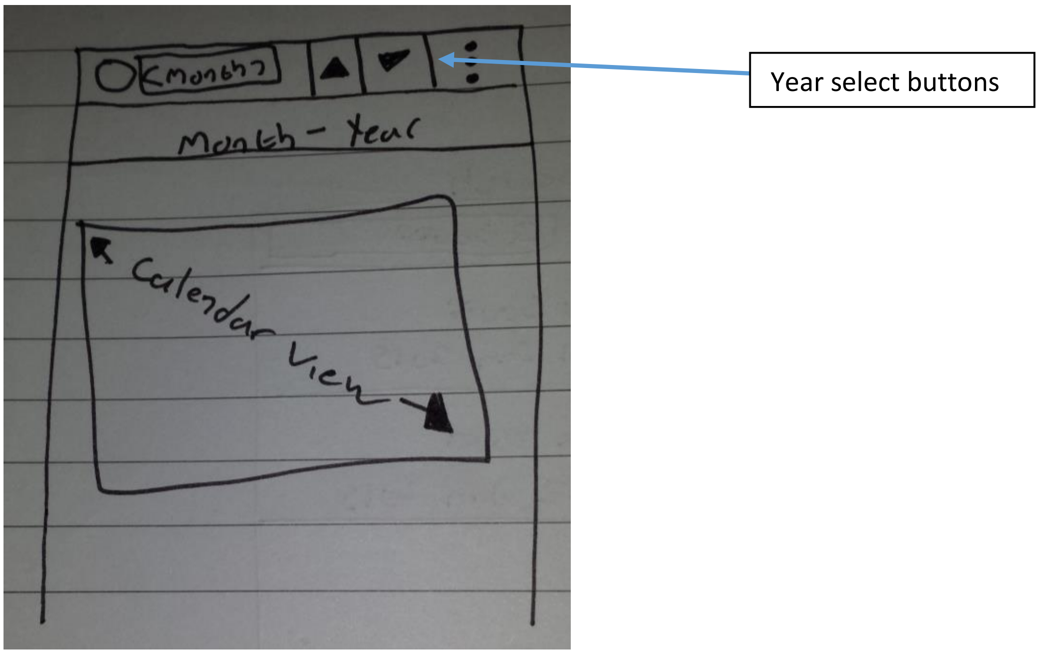


Figure 33 - Prototype Two, Revision Two Calendar View

Chapter Five – Implementation

5.1 Introduction

This chapter will provide explanations of the front-end interfaces and insight into the more complex code that has been developed for the project. In particular the chapter focuses on the back end and the database used to store information.

The front-end interfaces and the various components of each interface will be explained in detail.

Due to the number of functions and sub-routines, only snippets of code that are important, essential, or required additional research to implement will be included. The entire source code will be available in the appendices.

5.2 Front End

5.2.1 Calendar View

The month view was an open source solution; its design matched my initial design near perfectly, although this was purely coincidental. The design was used for the first prototype but was later modified to use more up-to-date and modern techniques and to modernise the look and feel of the interface (**androidhub4you, 2012**).

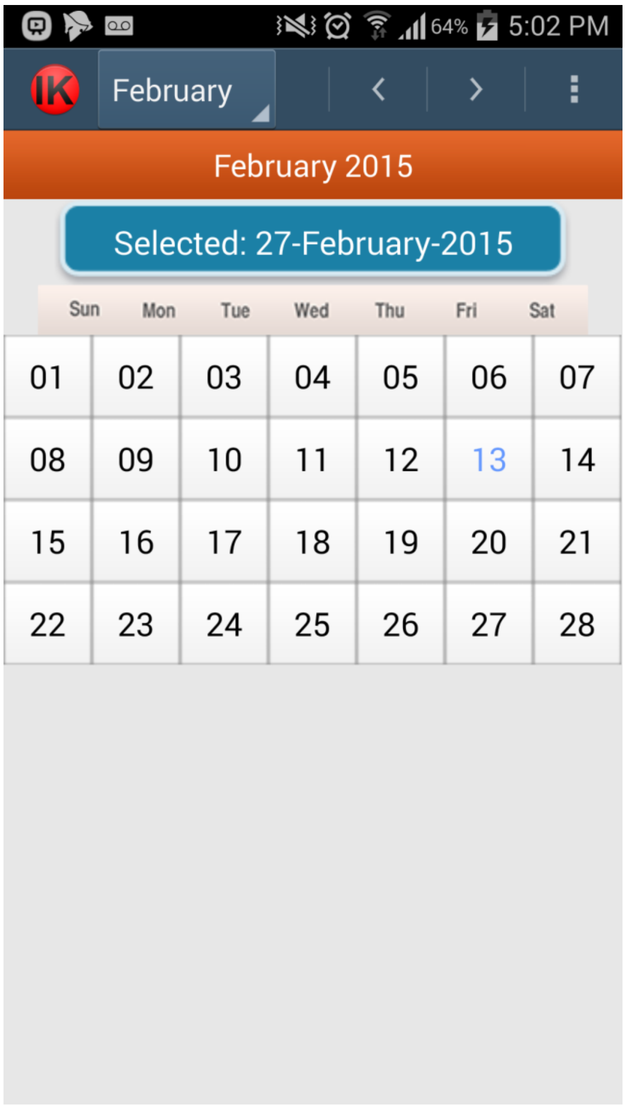


Figure 34 - Calendar View Implementation

5.2.1.1 Month Selection

Initially, months were selected using on screen arrows to scroll between months in a sequential order. This was changed to a more modern technique using a drop-down menu on an action bar. This made selecting the month faster and easier as well as looking more profession.

5.2.2.1 Year Selection

The year selection was placed onto arrow navigation buttons on the navigation bar, as year selection will not be used as often as month selection

5.2.1.3 Menu

Pressing the overflow icon causes the menu to appear with extra options. The menu is on an overflow as there was not enough space to show all of the icons as well as the month and year selection.

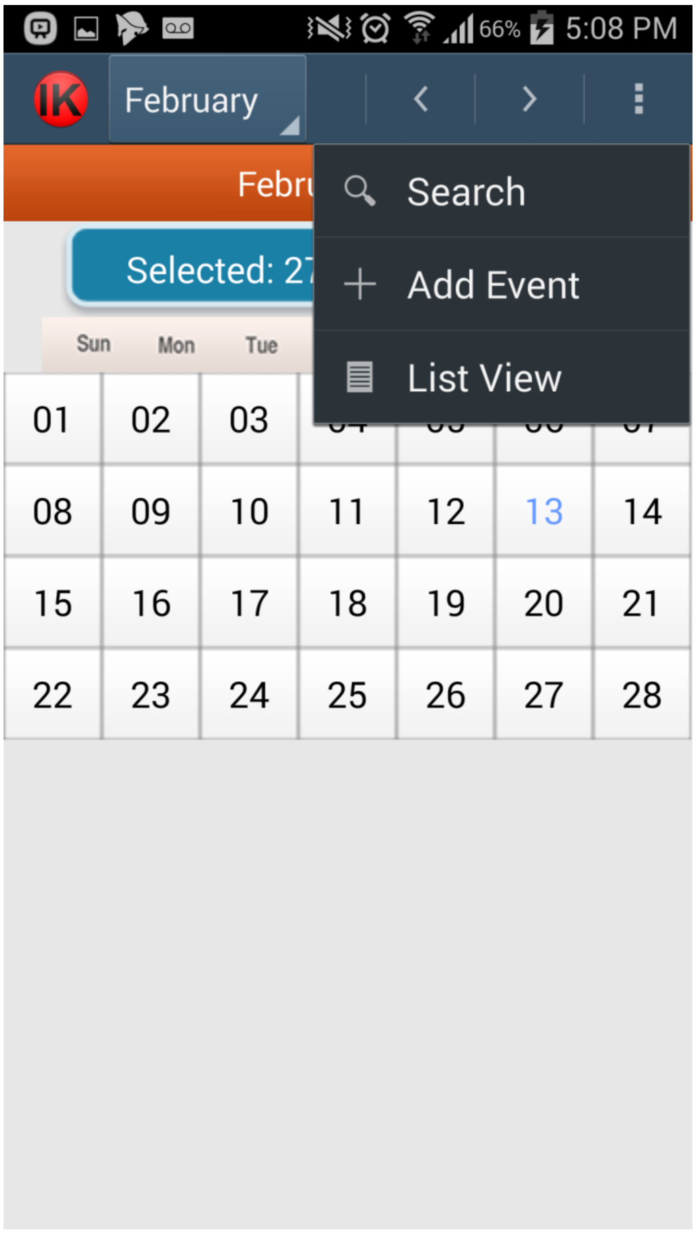


Figure 35 - Main Menu Implementation

5.2.1.4 Search Bar

The search bar is accessed from the menu, when selected the action bar at the top changes to allow the user to search using the keyboard.

Due to time constraints, the implementation of the search results being displayed to the user could not be finished.

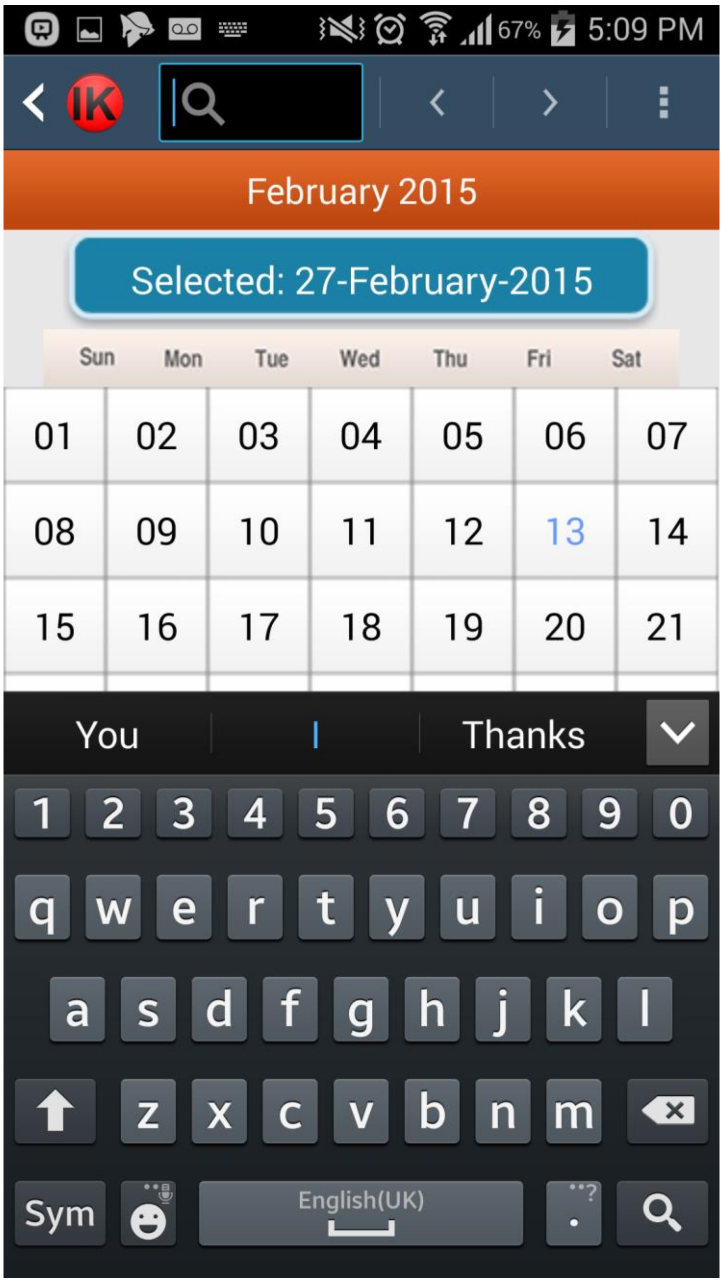


Figure 36 - Search Bar Implementation

5.2.2 New / Edit Entry

The new entry and edit entry screens share the same layout. When a new entry is created the fields are populated with default values, when an entry is selected for editing, the data for that entry are retrieved from the database and the text fields are populated with that data.

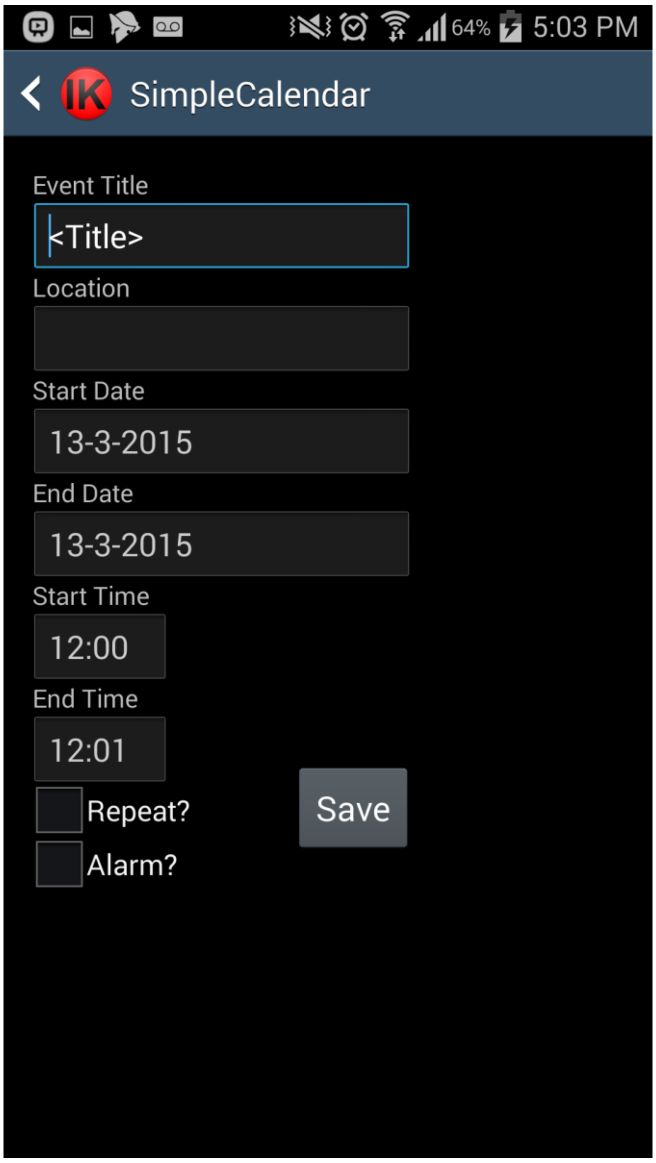
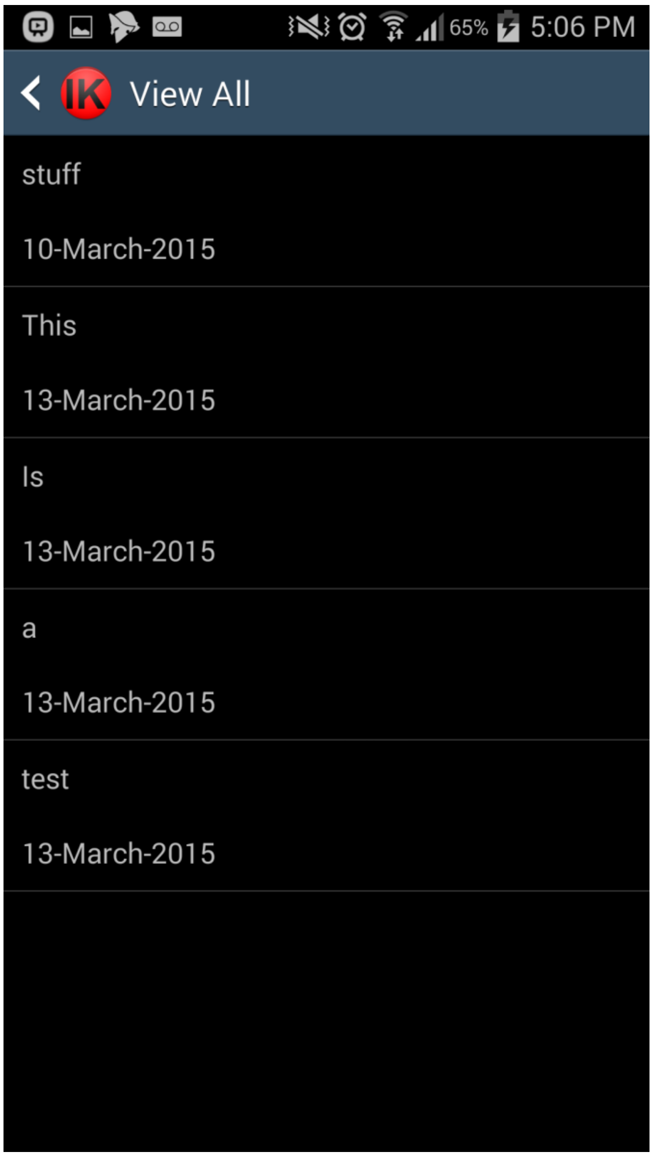


Figure 37 - New / Edit Entry Implementation

The layout consists of text boxes for the title and the location of the event, date pickers for the start and end date, and time pickers for the start and end time. A user can specify if they wish for an alarm to be set by ticking the alarm box.

5.2.3 List View

When the user selects the List View, all entries in the database are listed in a scrollable list view. The title of the event and the date are shown to the use for each entry. Events are ordered by Date order.

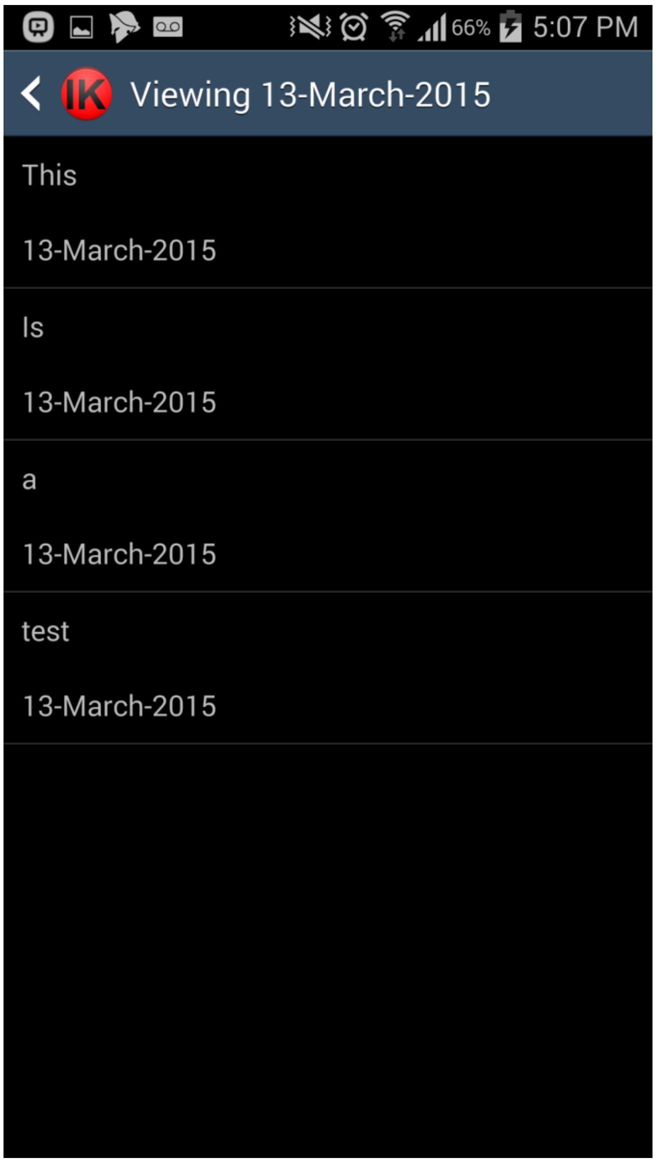


stuff
10-March-2015
This
13-March-2015
Is
13-March-2015
a
13-March-2015
test
13-March-2015

Figure 38 - List View Implementation

5.2.4 Day View

The day view is a simple list view just like the list view that only shows the events that start on the date selected from the calendar view. Like the list view, it is scrollable; given there are enough events on that day to fill the page.



This
13-March-2015
Is
13-March-2015
a
13-March-2015
test
13-March-2015

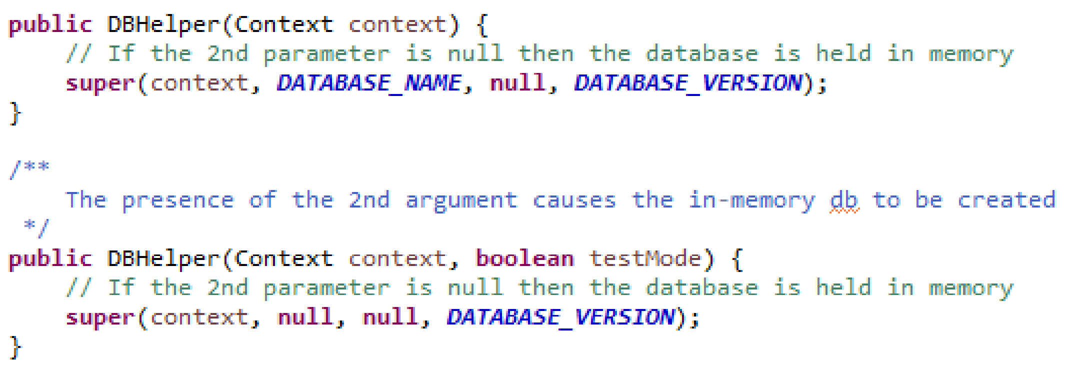
Figure 39 - Day View Implementation

5.3 Back end Code

5.3.1 Database Implementation

5.3.1.1 Checking if the Database already Exists in Memory

Before anything can happen, we need to check if the database already exists in memory. This is an essential step, as without a database the system cannot function properly so we need to create it. If it already exists then we do need to recreate it.



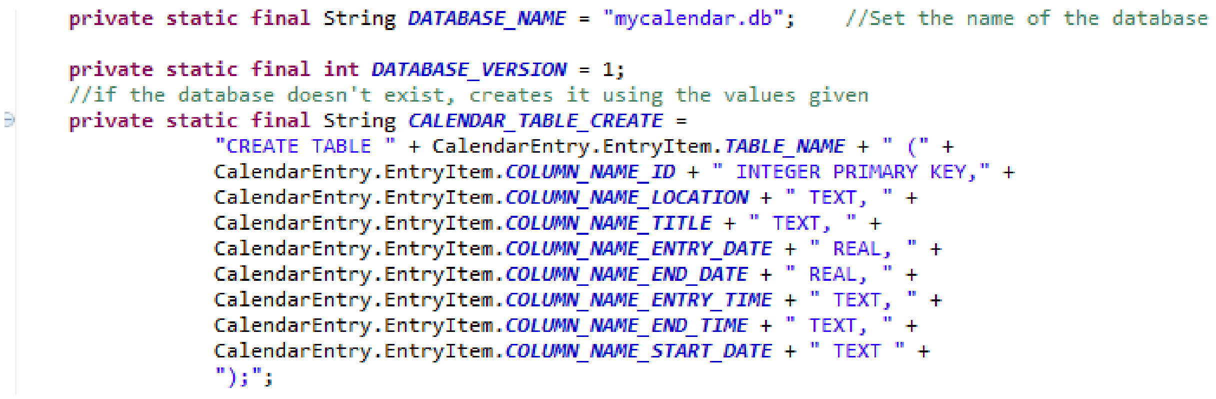
```
public DBHelper(Context context) {
    // If the 2nd parameter is null then the database is held in memory
    super(context, DATABASE_NAME, null, DATABASE_VERSION);
}

/**
    The presence of the 2nd argument causes the in-memory db to be created
 */
public DBHelper(Context context, boolean testMode) {
    // If the 2nd parameter is null then the database is held in memory
    super(context, null, null, DATABASE_VERSION);
}
```

Figure 40 - Database Helper Implementation

5.3.1.2 Database Create Statement

The create statement is created using the column names set in the Calendar Entry Class. The column names were made separate so that modifications can be done on the fly, as the column names are used in multiple places. If they were hard coded then any change would have to be made in multiple places each time something is changed.

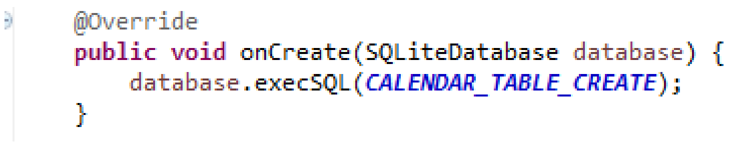
The database is a fundamental part of the system because without it; there is no way to save an entry.



```
private static final String DATABASE_NAME = "mycalendar.db";     //Set the name of the database

private static final int DATABASE_VERSION = 1;
//if the database doesn't exist, creates it using the values given
private static final String CALENDAR_TABLE_CREATE =
            "CREATE TABLE " + CalendarEntry.EntryItem.TABLE_NAME + " (" +
            CalendarEntry.EntryItem.COLUMN_NAME_ID + " INTEGER PRIMARY KEY," +
            CalendarEntry.EntryItem.COLUMN_NAME_LOCATION + " TEXT, " +
            CalendarEntry.EntryItem.COLUMN_NAME_TITLE + " TEXT, " +
            CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_DATE + " REAL, " +
            CalendarEntry.EntryItem.COLUMN_NAME_END_DATE + " REAL, " +
            CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_TIME + " TEXT, " +
            CalendarEntry.EntryItem.COLUMN_NAME_END_TIME + " TEXT, " +
            CalendarEntry.EntryItem.COLUMN_NAME_START_DATE + " TEXT " +
            ");";
```

Figure 41 - Database Create Statement

The command to execute the create statement is a simple SQL command; it uses the string we built using column names and types.

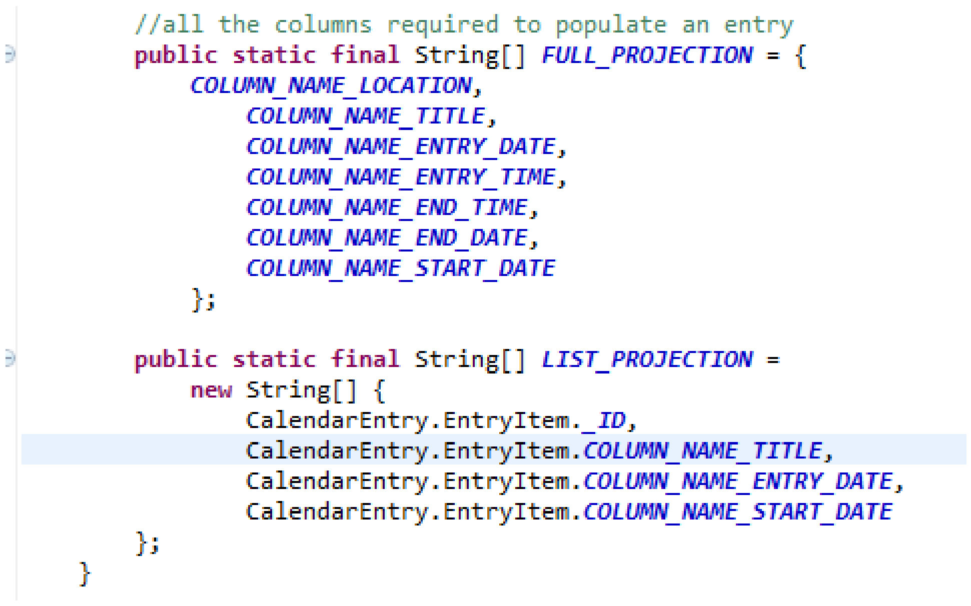


```
@Override
public void onCreate(SQLiteDatabase database) {
    database.execSQL(CALENDAR_TABLE_CREATE);
}
```

Figure 42 - Database Execute Create Statement

5.3.1.3 List Projections

When retrieving data for the list views, we do not need all of the details associated with the entries. To retrieve all data for every entry would require a greater overhead and computing time, which would decrease the overall performance of the application. To combat this, two projections were created, a Full Projection and a List projection. The full projection is used when retrieving an entry for editing. The List Projection is used when displaying the entries in the list views.



```
        //all the columns required to populate an entry
        public static final String[] FULL_PROJECTION = {
            COLUMN_NAME_LOCATION,
                COLUMN_NAME_TITLE,
                COLUMN_NAME_ENTRY_DATE,
                COLUMN_NAME_ENTRY_TIME,
                COLUMN_NAME_END_TIME,
                COLUMN_NAME_END_DATE,
                COLUMN_NAME_START_DATE
            };

        public static final String[] LIST_PROJECTION =
            new String[] {
                CalendarEntry.EntryItem._ID,
                CalendarEntry.EntryItem.COLUMN_NAME_TITLE,
                CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_DATE,
                CalendarEntry.EntryItem.COLUMN_NAME_START_DATE
        };
    }
```

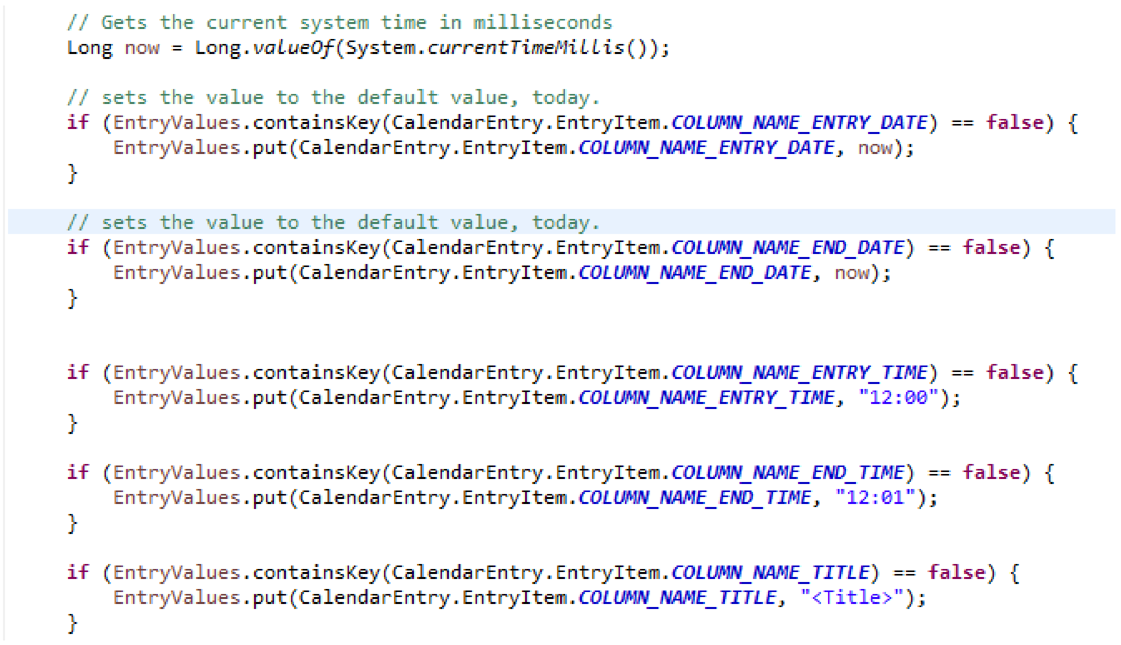
Figure 43 - Data List Projections

5.3.2 Data Access Object

The DAO deals with the retrieval, insertion and deletion of data into the applications Database. It is a class that acts like a transparent middleman between the Application and where the Data is stored. The application is unaware of where the data comes from which means the DAO and the application can be modified independently of each other.

5.3.2.1 Default Values

In order to prevent non-functionality or unwanted functionality, data fields are given default values when an event is first created.

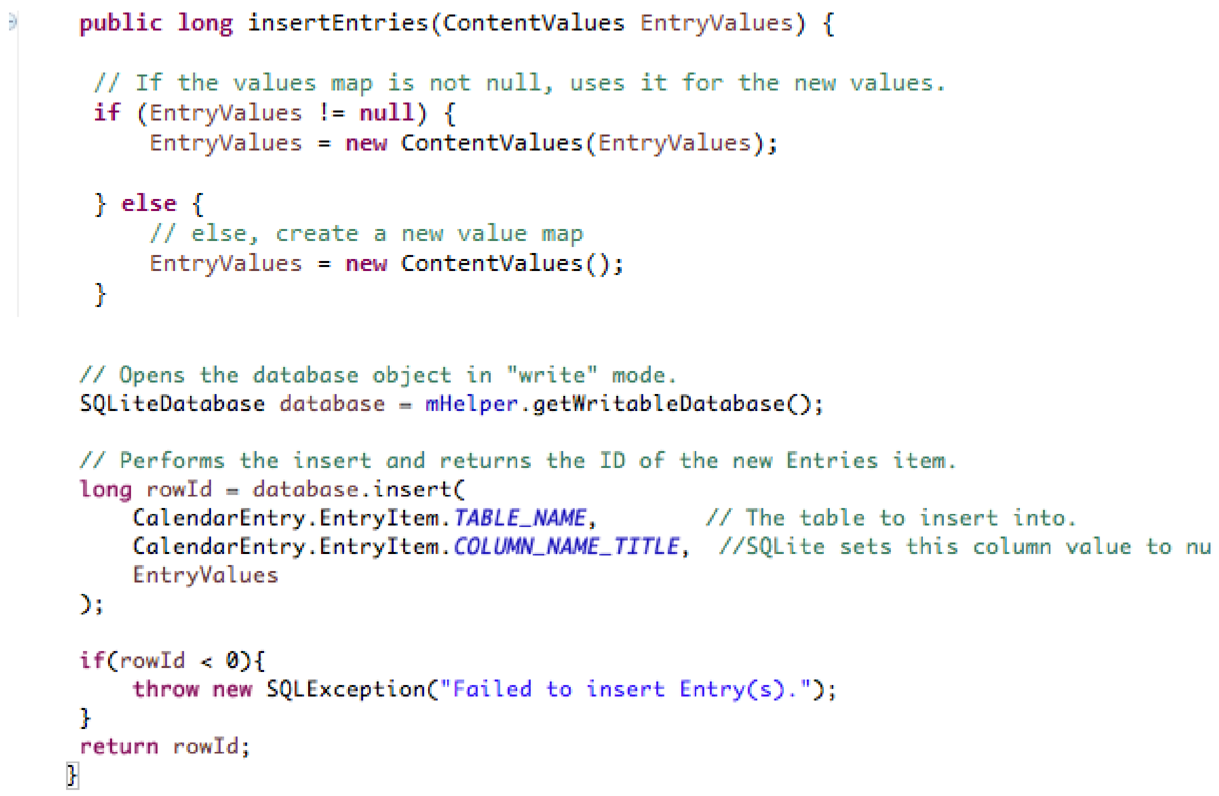


```
    // Gets the current system time in milliseconds
    Long now = Long.valueOf(System.currentTimeMillis());

    // sets the value to the default value, today.
    if (EntryValues.containsKey(CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_DATE) == false) {
        EntryValues.put(CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_DATE, now);
    }

    // sets the value to the default value, today.
    if (EntryValues.containsKey(CalendarEntry.EntryItem.COLUMN_NAME_END_DATE) == false) {
        EntryValues.put(CalendarEntry.EntryItem.COLUMN_NAME_END_DATE, now);
    }


    if (EntryValues.containsKey(CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_TIME) == false) {
        EntryValues.put(CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_TIME, "12:00");
    }

    if (EntryValues.containsKey(CalendarEntry.EntryItem.COLUMN_NAME_END_TIME) == false) {
        EntryValues.put(CalendarEntry.EntryItem.COLUMN_NAME_END_TIME, "12:01");
    }

    if (EntryValues.containsKey(CalendarEntry.EntryItem.COLUMN_NAME_TITLE) == false) {
        EntryValues.put(CalendarEntry.EntryItem.COLUMN_NAME_TITLE, "<Title>");
    }
```

Figure 44 - Default Values for new Event

5.3.2.2 Inserting an Entry

When an event is inserted, the values are being inserted are passed to the function. The connection to the database is opened and the insert command is executed. The row ID is returned from the database, if this number is less than 0 then it means the insert failed.

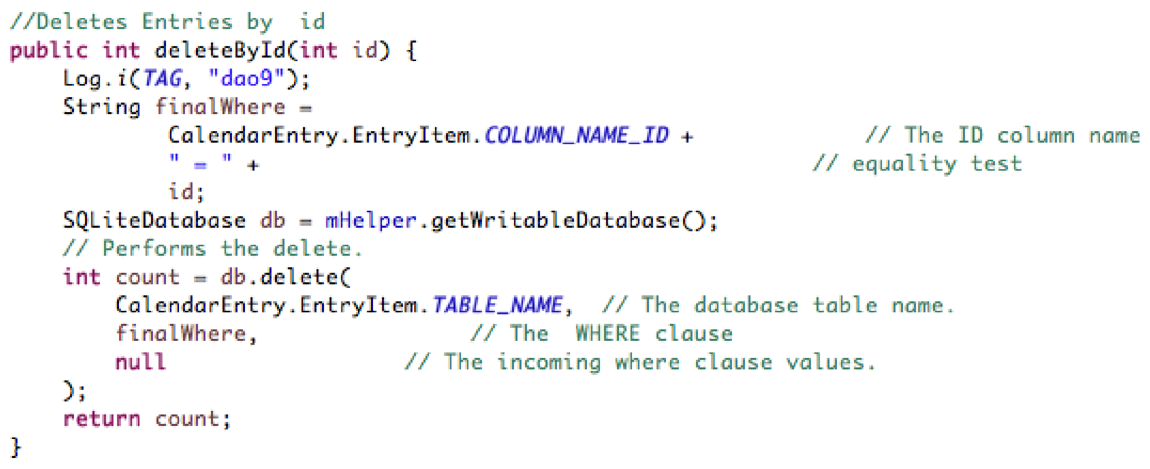


```
    public long insertEntries(ContentValues EntryValues) {

     // If the values map is not null, uses it for the new values.
     if (EntryValues != null) {
         EntryValues = new ContentValues(EntryValues);

     } else {
         // else, create a new value map
         EntryValues = new ContentValues();
     }


    // Opens the database object in "write" mode.
    SQLiteDatabase database = mHelper.getWritableDatabase();

    // Performs the insert and returns the ID of the new Entries item.
    long rowId = database.insert(
        CalendarEntry.EntryItem.TABLE_NAME,        // The table to insert into.
        CalendarEntry.EntryItem.COLUMN_NAME_TITLE,  //SQLite sets this column value to nu
        EntryValues
    );

    if(rowId < 0){
        throw new SQLException("Failed to insert Entry(s).");
    }
    return rowId;
   }
```

Figure 45 - Insert Entries

5.3.2.3 Deleting an Entry

The ID number of the entry to be deleted is passed into the function, this used to compose the where clause of the delete statement.

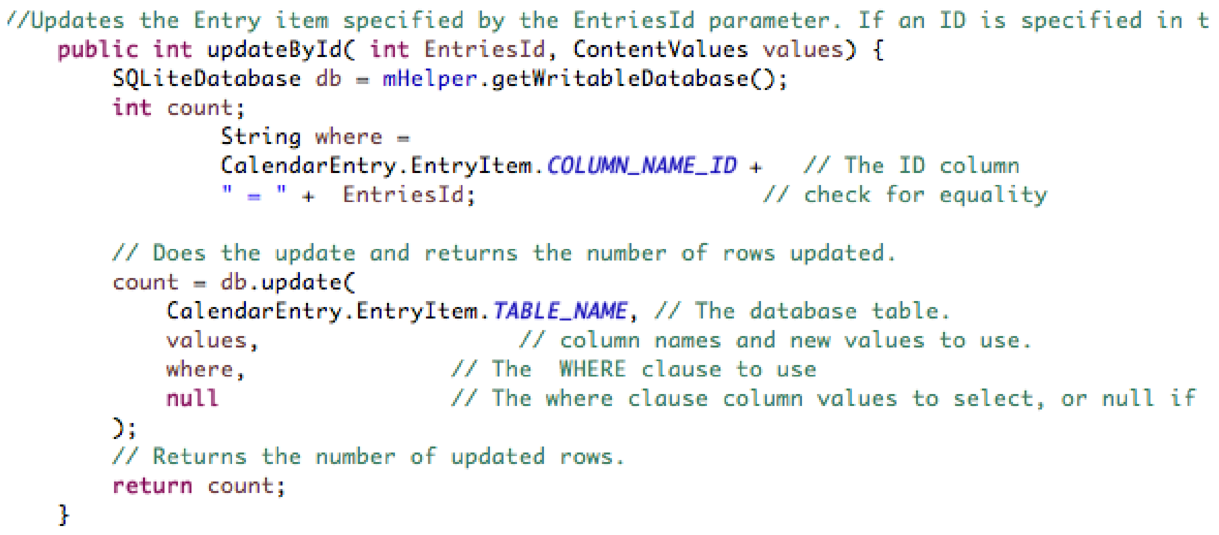


```
   //Deletes Entries by  id
   public int deleteById(int id) {
       Log.i(TAG, "dao9");
       String finalWhere =
               CalendarEntry.EntryItem.COLUMN_NAME_ID +             // The ID column name
               " = " +                                          // equality test
               id;
       SQLiteDatabase db = mHelper.getWritableDatabase();
       // Performs the delete.
       int count = db.delete(
           CalendarEntry.EntryItem.TABLE_NAME,  // The database table name.
           finalWhere,                 // The  WHERE clause
           null                    // The incoming where clause values.
       );
       return count;
   }
```

Figure 46 - Delete Entry

5.3.2.4 Updating an Entry

Similar to the delete function, the ID of the function that is being modified is passed to the function, additionally; the values to be updated are passed.



```
//Updates the Entry item specified by the EntriesId parameter. If an ID is specified in t
    public int updateById( int EntriesId, ContentValues values) {
        SQLiteDatabase db = mHelper.getWritableDatabase();
        int count;
                String where =
                CalendarEntry.EntryItem.COLUMN_NAME_ID +   // The ID column
                " = " +  EntriesId;                       // check for equality

        // Does the update and returns the number of rows updated.
        count = db.update(
            CalendarEntry.EntryItem.TABLE_NAME, // The database table.
            values,                      // column names and new values to use.
            where,                  // The  WHERE clause to use
            null                    // The where clause column values to select, or null if
        );
        // Returns the number of updated rows.
        return count;
    }
```

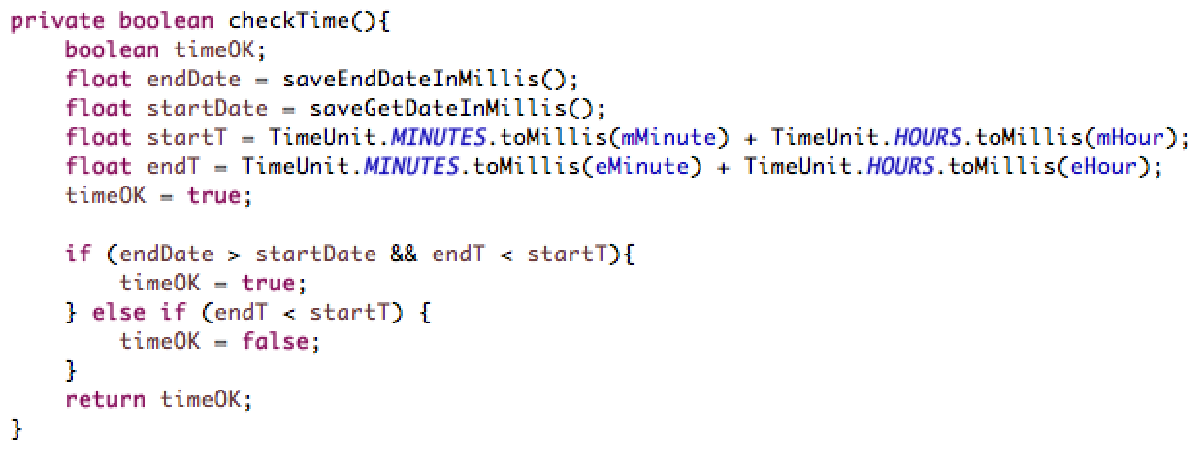
Figure 47 - Updating Entry Code

5.3.3 Calendar Entry

The code behind the Calendar Entry/Edit screen is mostly Validation Checks to ensure that the date and time are formatted correctly. It also checks whether or not the user has selected the option to create an alarm for the event in question.

5.3.3.1 Check Time

When the user attempts to save a new entry or edited entry. The time must be checked to ensure that they are not attempting to create an event that has the end time set before the start time if the start date and end date are the same, it also checks if the end date is after to start date to allow for event times to pass over into the next day.

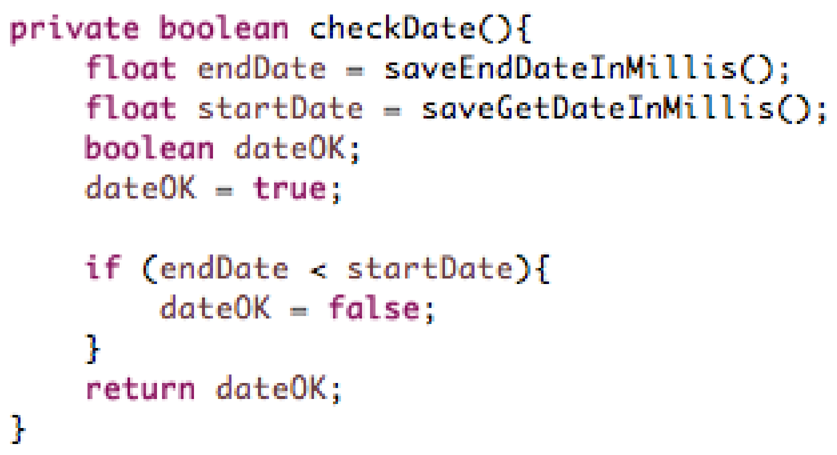


```
private boolean checkTime(){
    boolean timeOK;
    float endDate = saveEndDateInMillis();
    float startDate = saveGetDateInMillis();
    float startT = TimeUnit.MINUTES.toMillis(mMinute) + TimeUnit.HOURS.toMillis(mHour);
    float endT = TimeUnit.MINUTES.toMillis(eMinute) + TimeUnit.HOURS.toMillis(eHour);
    timeOK = true;

    if (endDate > startDate && endT < startT){
        timeOK = true;
    } else if (endT < startT) {
        timeOK = false;
    }
    return timeOK;
}
```

Figure 48 - Check Time code

5.3.3.2 Check Date

The start date and end date of an entry must be verified when a user attempts to create or edit an entry. The end date cannot be set before the start date as this would not be possible in the real world. The function returns false if the end date is before the start date.

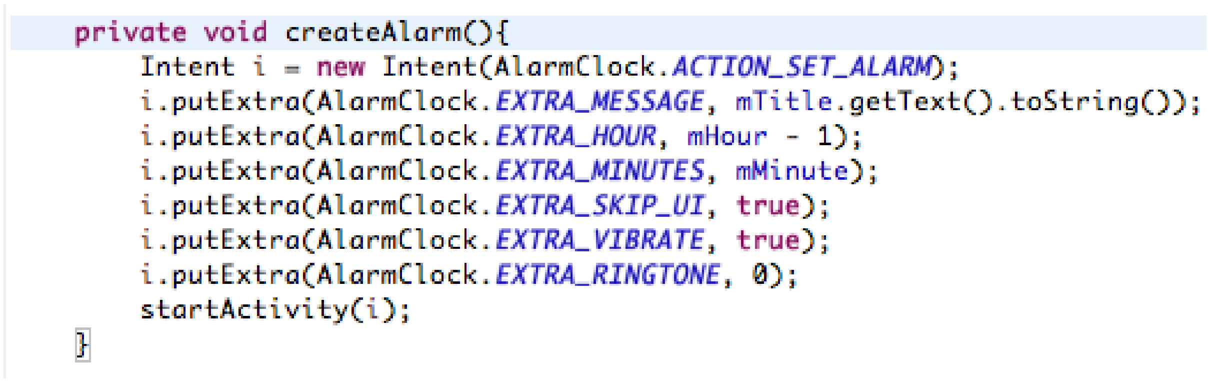


```java
private boolean checkDate(){
    float endDate = saveEndDateInMillis();
    float startDate = saveGetDateInMillis();
    boolean dateOK;
    dateOK = true;

    if (endDate < startDate){
        dateOK = false;
    }
    return dateOK;
}
```

Figure 49 - Check Date Code

5.3.3.3 Create Alarm

When a user chooses to create an alarm for an event, a new alarm is created on the android device that uses the title of the event and the time of the event minus 1 hour. There is an offset of 1 hour to give a user an early notification for the start of an event. The alert tone is set to the default ringtone of the device and the vibrate function is enabled.



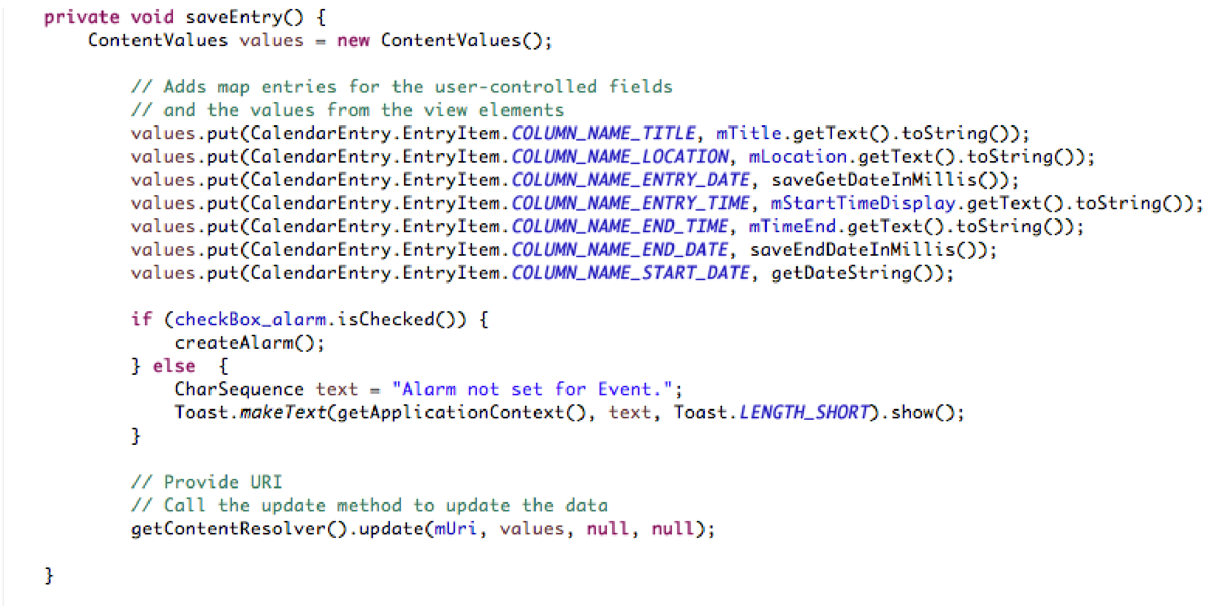
```java
private void createAlarm(){
    Intent i = new Intent(AlarmClock.ACTION_SET_ALARM);
    i.putExtra(AlarmClock.EXTRA_MESSAGE, mTitle.getText().toString());
    i.putExtra(AlarmClock.EXTRA_HOUR, mHour - 1);
    i.putExtra(AlarmClock.EXTRA_MINUTES, mMinute);
    i.putExtra(AlarmClock.EXTRA_SKIP_UI, true);
    i.putExtra(AlarmClock.EXTRA_VIBRATE, true);
    i.putExtra(AlarmClock.EXTRA_RINGTONE, 0);
    startActivity(i);
}
```

Figure 50 - Create Alarm Code

5.3.3.4 Save Entry

When a user clicks the save button, the contents of the text boxes are transferred into the variables that are mapped to the database headings, they are converted into the right variable types using built in functions and custom functions.

At the same time, the alarm check box is evaluated to determine if an alarm should be created. A notification is shown to the user if an alarm is not created in case they meant to set one and forgot.



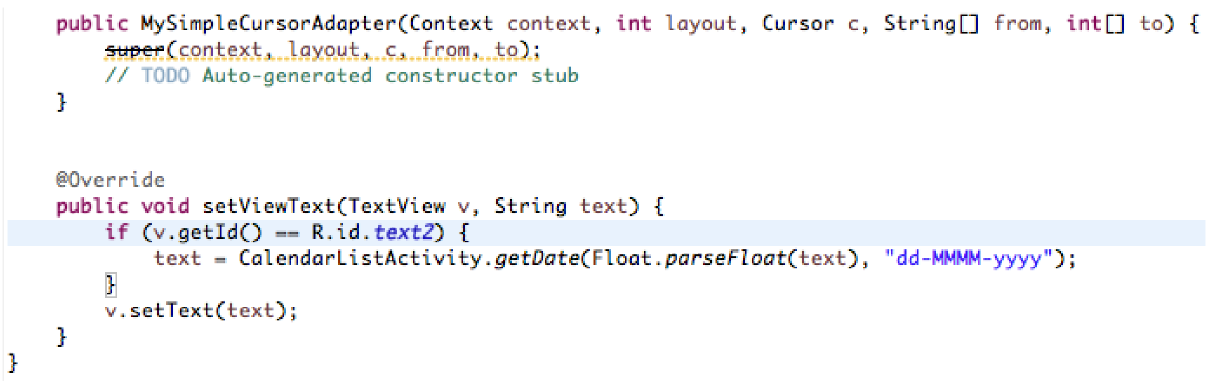
```
private void saveEntry() {
    ContentValues values = new ContentValues();

        // Adds map entries for the user-controlled fields
        // and the values from the view elements
        values.put(CalendarEntry.EntryItem.COLUMN_NAME_TITLE, mTitle.getText().toString());
        values.put(CalendarEntry.EntryItem.COLUMN_NAME_LOCATION, mLocation.getText().toString());
        values.put(CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_DATE, saveGetDateInMillis());
        values.put(CalendarEntry.EntryItem.COLUMN_NAME_ENTRY_TIME, mStartTimeDisplay.getText().toString());
        values.put(CalendarEntry.EntryItem.COLUMN_NAME_END_TIME, mTimeEnd.getText().toString());
        values.put(CalendarEntry.EntryItem.COLUMN_NAME_END_DATE, saveEndDateInMillis());
        values.put(CalendarEntry.EntryItem.COLUMN_NAME_START_DATE, getDateString());

        if (checkBox_alarm.isChecked()) {
            createAlarm();
        } else  {
            CharSequence text = "Alarm not set for Event.";
            Toast.makeText(getApplicationContext(), text, Toast.LENGTH_SHORT).show();
        }

        // Provide URI
        // Call the update method to update the data
        getContentResolver().update(mUri, values, null, null);

}
```

Figure 51 - Save Entry Code

5.3.4 Custom Classes

5.3.4.1 MySimpleCursorAdaptor

When the date is saved to the database, it is converted into milliseconds. This method works fine when retrieving the date for the entry form as it can be converted back into a readable format before displaying it back into the textbox but when attempting to display the date in a list view item, this was not the case as it would show the milliseconds instead of formatting the date. To rectify this problem I had to create my own implementation of the SimpleCursorAdaptor class that included the extra step to convert the decimal number back into a date format.



```
    public MySimpleCursorAdapter(Context context, int layout, Cursor c, String[] from, int[] to) {
        ~~super~~(context, layout, c, from, to);
        // TODO Auto-generated constructor stub
    }


    @Override
    public void setViewText(TextView v, String text) {
        if (v.getId() == R.id.text2) {
            text = CalendarListActivity.getDate(Float.parseFloat(text), "dd-MMMM-yyyy");
        }
        v.setText(text);
    }
}
```

Figure 52 - Convert Date and Time code

5.4 Challenges Faced

5.4.1 Calendar View Challenge

The main challenge faced during the development of the application was trying to implement the scrollable list view onto the Calendar View screen. Due to being unable to find a working example application already on the market or tutorial online, this idea had to be changed into a separate list view that the user can access by selecting the day that they wish to see (*See sections 4.2.1 & 4.5.2*).

5.4.2 List View Challenge

Another challenge faced was retrieving the date from the database for the List and Day views, as the date is stored in the database as Milliseconds. This required the modification and implementation of a customised Simple Cursor Adaptor (*See section 4.2.2, 4.5.2 & 5.3.1.1*).

Chapter Six – Testing

6.1 Introduction

The aim of this chapter is to discuss the testing strategies that are available, and use them to identify and implement a testing strategy that is suitable the usability of the application and the user experience that comes with it, this will involve designing a plan. To do this, further research must be done on the different types of testing strategies that are available. The two test strategies that will be focused on will be Black Box and White Box testing, these testing strategies will be discussed in the section below.

6.2 Testing Strategies

Testing has two purposes; it is the process of using a piece of software or a system with the intention to find bugs and errors as well as being used to ensure that the features and functions of a system function as intended.

Myers, G. J. 2004 highlights two key testing strategies, White-Box and Black Box testing.

6.2.1 White Box Testing

White Box testing is logic testing. The tester is normally the developer, as the tester needs to know how the code works. The strategy behind White Box testing is to pick a number of tests that will use as many functions and features as possible.

White Box testing has four common methods (**Myers G, J 2004**), Condition Coverage, Decision Coverage, Multiple Condition Coverage and Statement Coverage.

6.2.1.1 Condition Coverage

Condition coverage is used to make sure that each condition in a decision is met at least once, for example classifying the validity of an input as being true or false. One test must be completed that is deemed false and another must be completed that returns true.

6.2.1.2 Decision Coverage

Decision Coverage ensures that there is an adequate number of tests performed, so that each outcome of a test is seen at least once.

6.2.1.3 Multiple Condition Coverage

When conditions are combined for testing, for example when testing multiple inputs of an entry form, tests must be created so that each possible combination of values are used to produce all possible conditions. (**Marcotty, M 1991**).

6.2.1.4 Statement Coverage
Statement coverage says that each function of a system must have a test case, meaning that all of the code is executed once at a minimum. Although **Myers G, J 2004** states that this criteria is weak and deems it useless for testing purposes.

6.2.3 Black Box Testing
Black box testing is data driven; it can also be called input-output testing. The method behind black box testing is to test the system or applications behaviour rather than thinking about the code behind it. Tests are performed in order to find unexpected or unwanted behaviour from the system. This can be done by entering combinations of correct and incorrect data and checking that the system handles the incorrect data properly.

There are a number of approaches to Black Box testing but Myers G, J 2004 focuses on four of these. These are Boundary Value analysis, Cause Effect graphing, Error Guessing and Equivalence Partitioning.

6.2.3.1 Boundary Value Analysis
Boundary Value analysis tests the Boundaries of inputs, also known as extreme values. Inputs and values that are close to being invalid or valid are used, for example if an input box accepts the values 0 to 100, then the values 99 and 101 will be used to test the boundary. It is usually the case that these types of inputs and values have a major role when looking for errors in a system.

6.2.3.2 Cause Effect Graphing
Cause Effect graphing is used to systematically select a set of tests that will show the highest results. It helps reveal ambiguities and incompleteness (**Myers G, J 2004**). As the cause effect graphing technique does not handle large specifications very well, the specification is split into smaller, more manageable parts. An example of a manageable test for this project would be selecting the alarm feature being the cause/input for the test and effect/output being the alarm being created and a verification message being displayed to the user.

6.2.3.3 Error Guessing
With Error Guessing, the tester attempts to predict where errors will occur and what will cause them to occur. Error testing is used to simulate human error by performing tests such as leaving fields blank like the title of a calendar entry or by trying to input duplicate records into a database. Quite often, this method reveals errors that the other methods may have missed.

6.2.3.4 Equivalence Partitioning
Equivalence partitioning is to test as many input conditions as possible. Each test performed acts like a representative for the part of the system that is being tested. For example, if testing the start date for an entry has validation errors like allowing the user to enter the end date before the start date, then more tests are constructed for the entry page, if no errors are found within the initial tests then no extra tests are carried out.

6.3 Grey Box Testing
Combining the methods of White Box and Black Box testing is known as Grey Box testing. **Myers G, J 2004** considers this to be the best practice for testing rather than focusing on one method. For the testing of this application, Grey Box testing will be used to a degree, although most of the tests will be White Box. Code for the application was tested during the development stage of the project, but the system as a whole was not. The main aims of testing the application are to test the validation and usability of the application.

6.3 Testing Guidelines

(Myers H, J, 2003) Identifies 10 vital testing guidelines that can be used to create a test plan, these guidelines are as follows:

Guideline	Description
1	The expected outcome of the test must be clearly defined
2	A programmer should avoid testing their own software.
3	The organisation that developed the software/system should not test their own software. The testing should be outsourced.
4	The results of the testing should be thoroughly inspected for validity.
5	Tests must be written for inputs that are unexpected and invalid, not just for what is valid and expected.
6	The system should be tested to ensure that it does what it is supposed to do, as well as testing for what it is not supposed to do.
7	Do not use tests that are disposable.
8	Do not create a testing plan with the assumption that there will be no errors discovered.
9	The probability of finding more errors in a given section is proportionate to the number of errors that have already been found in that section.
10	Creating a test plan is a creative task and benefits more from an individual that is an innovative thinker.

Table 4 - Testing Guidelines

It is also to identify what to test as well as following the guidelines above, as it can be nearly impossible to test everything for errors. For this reason, the components that have been identified as high priority, as in, if the components that are crucial for functionality that would cause the application to fail, will be tested.

6.4 Test Plan

The test plan below depicts the necessary testing of the features and functions of the application that are vital. A complete failure of a component could lead to non-functionality of the application; miss-functionality could lead to poor usability of the application. In addition to showing the Procedure and expected results for a test, the outcome off the test will be indicated in the correct column. Failures and their solutions will be analysed and discussed in more detail.

6.4.1 Calendar View testing

Number	Test	Procedure or Input	Expected Result	Result
1	Launch the Application	Tap the application Icon to Launch	Application Launches, current month is displayed	Pass / Fail (See 6.5.1)
2	Use the calendar view to view events <today>	Tap the date in the calendar View (Blank list will be shown if no entry for that date)	Entries associated with that date will be shown.	Pass
3	View all events using the menu	Use the action bar to select "all Events"	All Events in the database are shown in date order	Pass
4	Return to the Calendar View	Use the 'Up' navigation arrow to return to the Calendar	Calendar view is shown in the state it was left in	Fail (See 6.5.2)

		View		
5	Navigate to December 2015 and view the 25th of December	Use the month selection and year selection	The events are shown for December 25th 2015	Pass

Table 5 - Calendar View Test Plan

6.4.2 Day View testing

Number	Test	Procedure or Input	Expected Result	Result
1	Edit the start date of an existing event	View an existing event, tap the start date and change it to a 3 days prior	New date is displayed in the box, event updates fine.	Pass
2	Delete an event from the day view	Long press an event in the list and press the Delete pop up that appears	Event is removed from the database	Pass

Table 6 - Day View Test Plan

6.4.3 List View testing

Number	Test	Procedure or Input	Expected Result	Result
1	Edit the start date of an existing event	View an existing event, tap the start date and change it to a 3 days prior	New date is displayed in the box, event updates fine.	Pass
2	Delete an event from the List view	Long press an event in the list and press the Delete pop up that appears	Event is removed from the database	Pass

Table 7 - List View Test Plan

6.4.4 New Event

Number	Test	Procedure or Input	Expected Result	Result
1	Add a new event <today>	Select new event from menu. View event in list view.	New event added to database.	Pass
2	Create a new event with the end date set before the start date	Set the End Date before the start date.	Error message pops up when trying to save the event.	Pass
3	Create an event with an alarm set	Tick the “alarm” box on the new event from	Pop up saying Alarm set.	Pass
4	Create a new event for 25th December 2015	Create a new event and change the date to 25/12/15.	Event adds successfully.	Pass
5	Create an event with the end date after the start date, and the end time before the start time	Create an event with the start date of 12-02-15, End date of 13-02-15 and the Start time of 12PM and the end Time of 9AM	Event will save as the Start Time is part of the following day	Pass

6	Create an event with the start and end date the same, and the end time before the start time	Create an event with the start date of 12-02-15, End date of 12-02-15, Start time of 12PM and the end Time of 9AM	Error message will show as the Start time is before the End time, On the same day	Pass
7	Edit an existing entry to have an alarm.	View an event and click the alarm button, save the edit.	Alarm set for time in time start box	Fail (See 6.5.3)

Table 8 - Create Event Test Plan

6.5 Problems

The problems found whilst carrying out the test plan and the action, or the plan of action that could be taken to rectify these problems will be discussed below.

6.5.1 Month View

When the application is launched, the month view is initialised and the current month and day is set. At the same time, the array that holds the values for months for the drop-down list is initialised. This causes the system to see the month selected as the first value within the Array and the month changes to January.

6.5.2 Returning to the Calendar View

When using the Navigation Bar to return to the Calendar View, the Calendar View resets to January by default, this was not the case when using the Android 'Back' button on the device. This is because the Action Bar 'Up' button causes the Calendar View activity to be restarted rather than returning to the state that the user left it in when they launched a new activity. This is not so much an error as it is more of design flaw

6.5.3 Editing Alarm

The problem found was that when editing an entry to add an alarm, the time had to be re-entered otherwise the alarm would be set to 12pm. This is because of the date is retrieved from the database as a string, for example "12:15", rather than individual hour and minute values. The action needed to rectify this problem would require a redesign of multiple functions of the application as well as the database. Firstly, the database would need to be changed to hold the Hour and Minute value in separate fields. Next, the save and load functions as well as the structure of the data access object would need to be changed to reflect the changes made to the database.

6.6 Conclusion

To conclude, the test was a success in terms of finding errors and issues with the application. Three problems were found within the application. Unfortunately due to time restrictions, the solutions for the problems found could only be theorised as two out of three of the problems were due to the architecture of the system and fixing them would require a re-design of a large portion of the system. The problems found were not major in terms of affecting the overall functionality of the application, but they do affect the usability and the ease of use of the application.

Chapter Seven – Evaluation

7.1 Introduction

In this chapter, the methods used to evaluate the application that was produced during the project will be discussed, concentrating on usability and user-friendliness. The aim is to gain knowledge on how easy the application is to use. There are numerous ways to evaluate a system or application, including Questionnaires, Heuristics Evaluation, the System Usability Scale and Usability Testing. These are as follows:

7.2 Questionnaires

Questionnaires can be a vital tool for providing data to a developer from users, data that can be used to produce figures and facts. They also have high reliability, meaning that if the process is repeated then similar results will be returned. For this project, questionnaires were used early on to gain useful information from users about the current existing calendar applications available in order to help design the features and functions of the new application as well as some basic information about each user including how they rate themselves with the use of Computers and smart Devices as well the types of applications and websites they use the most on these devices.

7.3 Development Method: Iterative Waterfall

The project fell victim to one of the disadvantages of the Development Method that was used during the implementation of the Application, which was the underlying architecture and design of the application was incompatible with integrating Social Media functionality.

7.4 Heuristic Evaluation

Molich, R and Nielson, J (1990) state that Heuristic Evaluation is method used for usability engineering to help find usability problems with a user interface design so that they can be rectified as part of the iterative design process. It is a method for systematically inspecting a user interface, carried out by a usability specialist in an attempt to identify problems based on their experience.

7.5 Usability Testing

Usability testing is different from Heuristic Evaluation, **Bebit (2001)** *defines it as a method used to find problem areas by having a subject similar to the target users that will actually be using the system*. Usability testing is very effective tool as it gives the evaluator the ability to watch the participant and learn how they behave whilst using the system, allowing for problem areas to be spotted while the user is using the system while it is live.

7.6 The Sample Group

The size of the sample group is important, so is selecting the right participants for the sample group. **(Sudman, 1976)** says that this is because defining a sample population that is too narrow can make it difficult to obtain of individual elements. Basically, this means that if you are too specific when choosing the sample group, the sample group will be specialised which makes it non-existent.

Nielson, J (2000) says that 'Elaborate usability tests are a waste of resources' and that the best results come from tests with a sample group of no more than 5 users running many smaller tests.

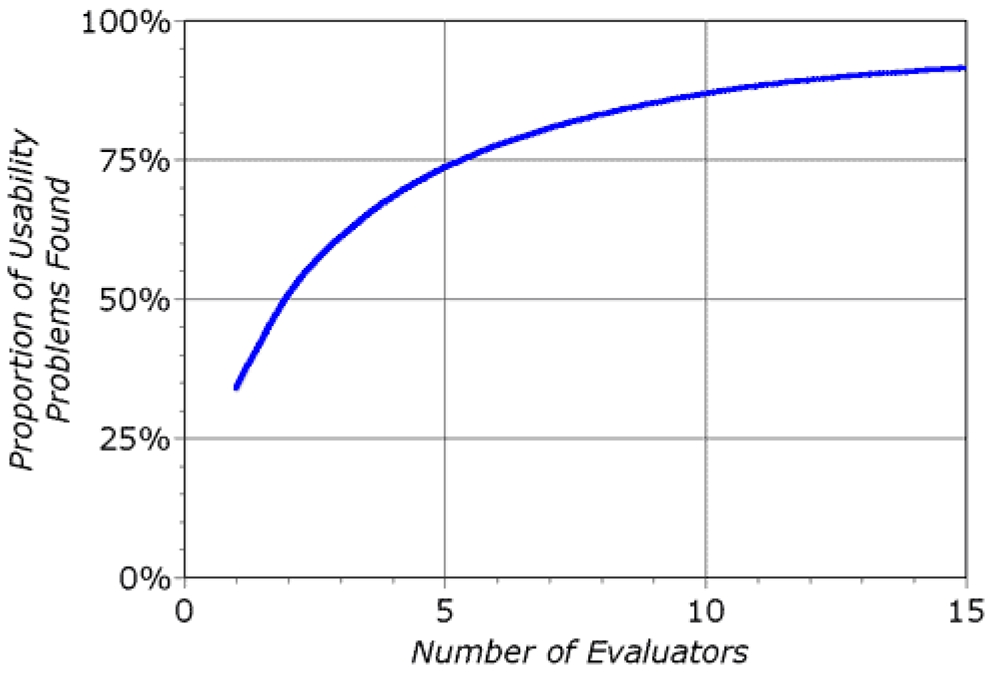


Table 9 - *Curve showing the proportion of problems found compared to the number of evaluators. (Nielson, 1995)*

For this project, the sample group is the same group of 5 volunteers that helped evaluate, test and give feedback on the current existing solutions. They are a group with mixed skill levels of computer literacy. Volunteers were asked their gender, age, application types they use the most and how they rate themselves as computer / smart device users.

7.6.1 Gender

The first question asked was the Gender of the participant. Although this is not necessarily relevant to the usability of the project, it was useful during the analysis stage for the current existing solutions, as the features and functions of the Fertility App could not be reviewed properly by only Male participants.

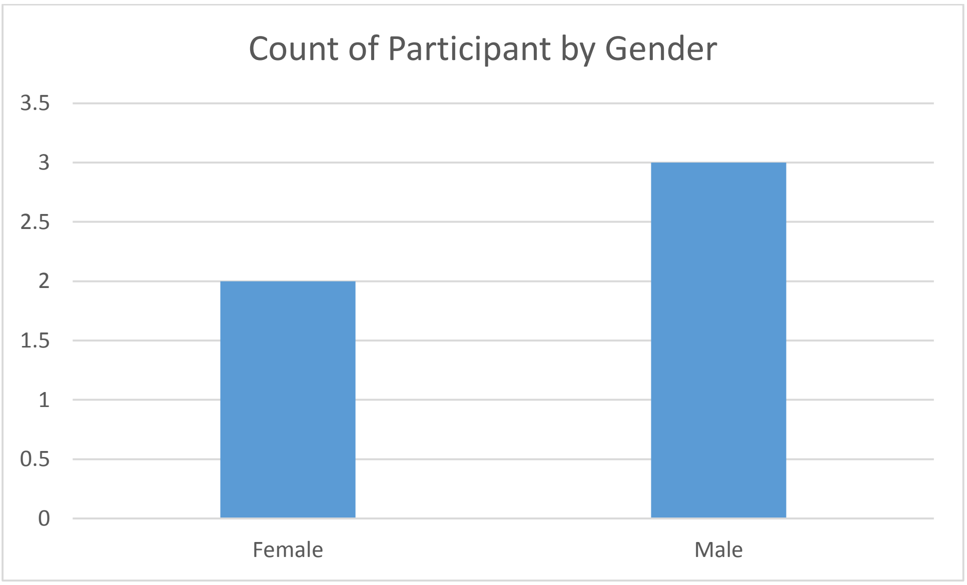


Figure 53 - Sample Group, Gender

7.6.2 Age

The age of a participant helped to verify if the sample group was in the target population, for this project the primary target population was students. Although students can be any age, usually new University student are aged 18 to 21, but this is my own speculation.

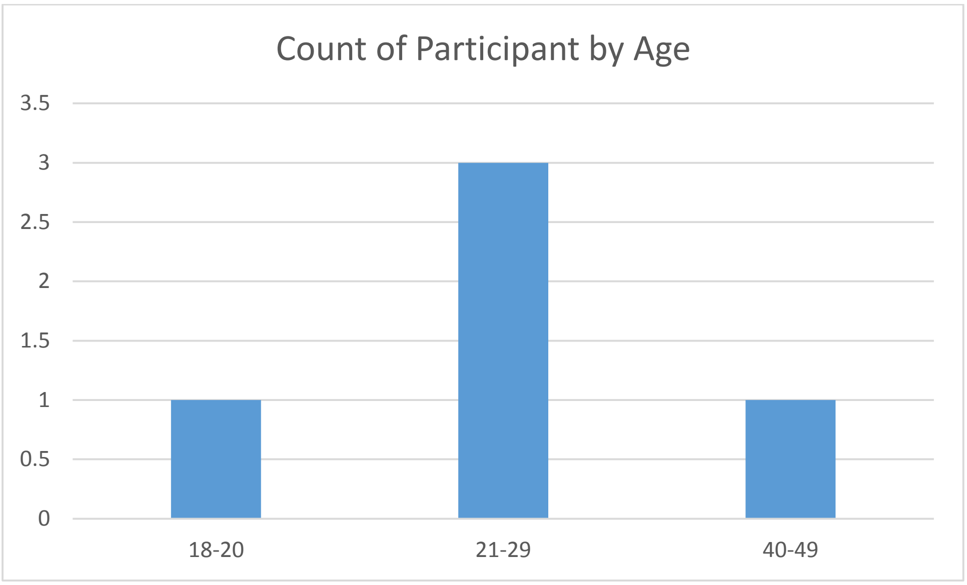


Figure 54 - Sample Group, Age

7.6.3 Types of Applications Used

The forth question the sample group was asked was about the types of applications that they used. The entire sample group chose Social Apps as their main use for Smart Devices. Although this question was not directly linked to Usability Testing, it was useful during the analysis and design stages for picking which features to include in the project. Even though 100% of the sample group chose Social Media over other types of applications such as utility, the group was small and therefore cannot be said to represent the entire target population.

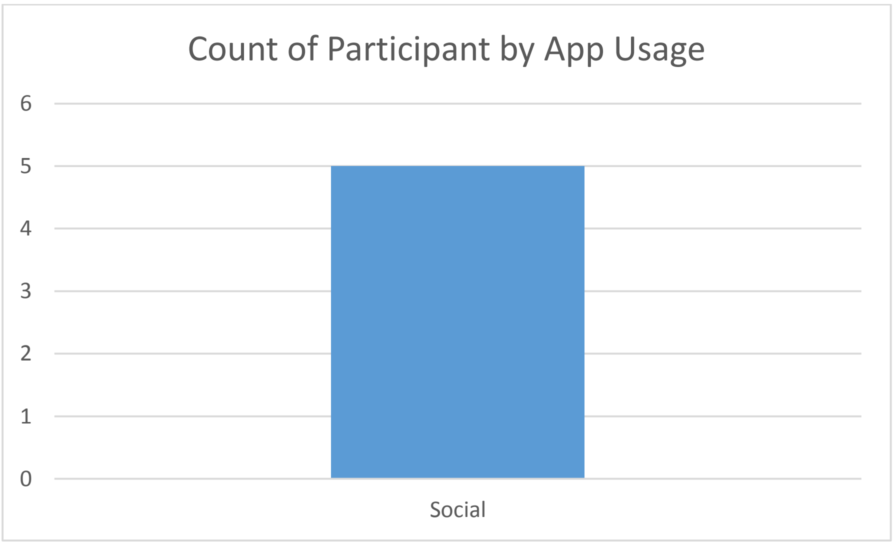


Figure 55 - Sample Group, Types of Applications used

7.6.4 User Rating

The last question asked was important as the results could affect the usability test. The question was about the computer literacy of the users, rating themselves from Novice to Expert. This is important as a user who is a novice or beginner is more likely to say that the application was complicated and hard to use and vice-versa with Intermediate and Expert users. As you can see from Figure 56, 60% (3 users) identified themselves as intermediate which means that they are competent enough with computers and mobile phones to perform every day, computer based tasks without great difficulty, whereas a novice user may struggle to complete a test that is simple to the average user.

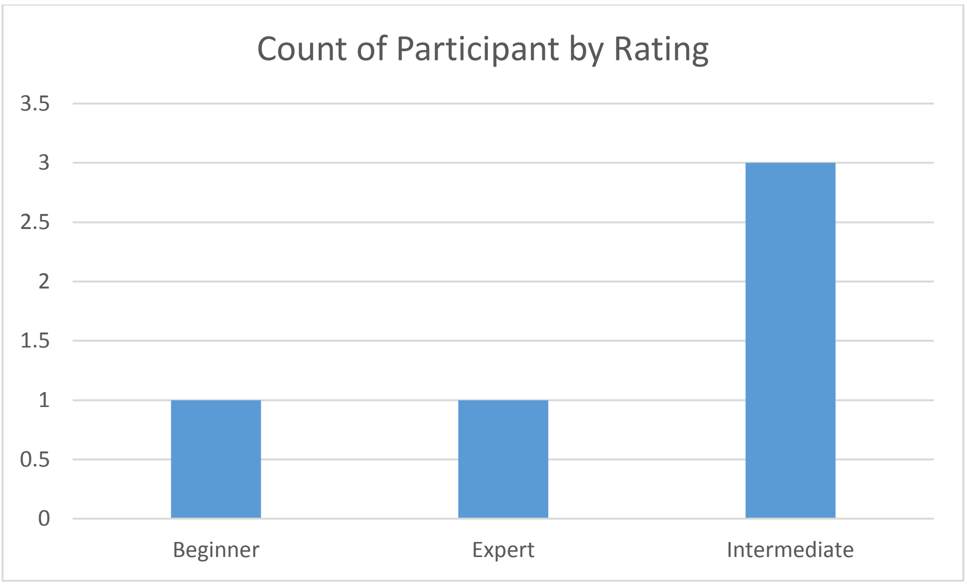


Figure 56 - Sample Group, Computer Literacy ratings

7.7 SUS

The same five participants that helped to test the current existing solutions tested both versions of the prototype. The participants were asked to give feedback via questionnaires on the look & feel, interface, usability and the features of the application, as well as this, they were also asked to fill out a SUS form for each version, the SUS scores for each prototype are as follows.

Participant	Prototype One	Prototype Two	Score Difference
Adam	75	82.5	7.5
Jake	67.5	75	7.5
Katie	60	65	6
Luke	60	65	5
Yvonne	60	75	15
Average Score	64.5	72.5	8

Table 10 - Prototype SUS Analysis comparison Results

In terms of Usability, the prototype application scored quite highly, being 4.5 points above the average of 68. Although this score does not reflect on the different features that have been implemented into the Prototype. There is still much room for improvement of the User Interface and the Usability of the features. From the results of the SUS analysis on both of the Prototypes, you can see a significant increase in usability from the first prototype to the second, although 2 of the scores were still below the 68 score average. This shows that the interfaces and functions of the application are user friendly but could still benefit from some refining or a slight de-design.

7.7.1 SUS Comparison

Application	Business Calendar	Fertility Calendar	Google Calendar	Jorte Calendar	SOL Calendar	Prototype Two
Average SUS	50	90	86.5	68	77.5	72.5

Table 11 - Average SUS Comparison, Current Existing Applications & Prototype Two

A SUS analysis is a crucial piece of feedback that allows for a system or application to continue to evolve and improve its usability. The average SUS score of the averages is 74.4.

When you compare the average of the Prototype with the current existing solutions, the prototype scores quite highly and competitively but the average SUS Score of the prototype is still slightly lower than the overall average of the five. This shows us that there is still room for improvement with the Usability of the application as it is not yet on the same level of Usability as the current existing solutions.

7.8 Usability Checklist

This section will contain the results of A Usability Checklist (**keepitusable, 2011**) carried out and discussed on the Calendar Applications analysed and reviewed in *section 3.2,* a Usability Checklist will also be carried out for the prototype and these results will then be compared. The comments section of the checklist is used to provide feedback as to why points were awarded or not, although not all users chose to use this. At the end, the average points awarded for each application will be displayed in a table. The Usability Checklist used can be found in *Appendix 1.4,* the results of the Usability Checklist filled out by participants can be found in the *Electronic Appendices*.

7.8.1 Usability Checklist Score Comparison

Application	Business Calendar	Fertility Calendar	Google Calendar	Jorte Calendar	SOL Calendar	Prototype Two
Average Usability Score	18.3	18	23.6	18.8	24	18.2

Table 12 - Average Usability Score, Current Existing Applications & Prototype Two

The results shown above are the final score for the Usability Checklist from the five existing solutions and the prototype the average score of the five existing solutions was **20.5** and the score for the average score for the prototype was **18.2**, the difference being -**2.3**. This shows us that although the prototype is usable, it is still not as usable as the existing applications. Even though the prototype did not score as highly as anticipated, it is still a prototype and therefore shows us that it still requires improvement and more feedback from users for future work.

7.8.2 Business Calendar Usability Checklist

The business calendar was fairly easy to use; it scored lower than the average of the five applications. It lost points mainly due to ambiguous help text and lack of instructions and from a small font size on higher resolution screens, which is poor optimisation.

7.8.3 Fertility Calendar Usability Checklist

The Fertility Calendar did not score higher than the average five even though it is very simple to use. Its major flaws were its colour scheme of grey on pink for certain squares and lack of error messages.

7.8.4 Google Calendar Usability Checklist

Google calendar scored higher than the average of the five applications, almost a perfect score. Its only flaw was that when a user made a mistake such as setting the end date before the start date, there were no error messages shown to the user and the date was auto-corrected.

7.8.5 Jorte Calendar Usability Checklist

Jorte Calendar scored slightly higher than the average. Similar to some of the other applications, there was not much in terms of error feedback, which lost the application some points. As well as this, there were many options within options meaning that the application did not have a minimalistic design.

7.8.6 SOL Calendar Usability Checklist

Sol Calendar scored near perfect with the highest average score; what failed this application was its lack of user help and instructions. Much can be learnt from this application about User Experience Design and Usability.

7.8.7 Prototype Usability Checklist

The prototype application lost points due to the application not containing any audio or audio settings, so no points were awarded for check 19 and 20. If the alarm manager was implemented differently, then the option for push notifications and custom alerts could be introduced (*See section 8.4.2*). The main concern with the usability checklist results is the lack of UI consistency and UI feedback; this will be addressed in the Conclusion section *(see section 8.4.3).* The prototype also lost points due to the style of the error messages during data entry as poorly formatted data & time does not define which of the date or time is incorrect, just that one of them is incorrect.

The Usability Checklists carried out on the prototype will be very beneficial to the future Improvements of the application, as the points that were not awarded and the comments left by the testers will allow me to focus on those specific areas.

7.9 Application Design

The testers were asked to rate four aspects on the design of the application; these were the Layout, the Colour Scheme, the consistency of the application, the font size and its ease of use. The graph below shows how users rated each aspect using a 1 – 5 rating, with 1 being Needs Improvement and 5 being Excellent. The majority voted good on aspects of the design but the Layout and the Colour Scheme received a small number of Poor and Needs Improvement ratings. Due to the results of the design evaluation, the colour scheme and layout will be the one of the focus points for future improvements.

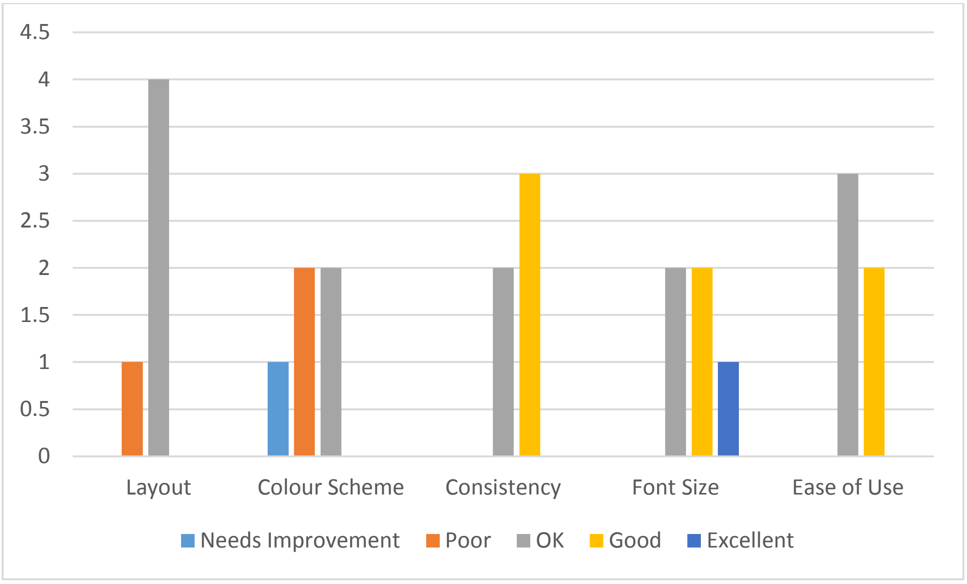


Figure 57 - Design Analysis results for Prototype Two

Chapter Eight – Conclusion

8. 1 Chapter Summaries

This section includes a brief conclusion for each of the previous chapters: Literature Review, Analysis, Design, Implementation, Testing and the Evaluation.

8.1.1 Literature Review Conclusion

The literature review provides information on the importance of User Experience, User Experience Design, Heuristics the available development Languages and Software Security, as well as technical background of the various design patterns used when developing for mobile platforms to maximise the use of the limited screen real estate and ease of use for the user.

8.1.2 Analysis Conclusion

In the Analysis chapter, the Target Population is defined and current existing calendar applications are examined for functionality and usability. The section then goes on to discuss the perspective and the assumptions about the product, such as the application being data-centric and that a database is required for functionality. The requirements of the application are identified, each for a brief description.

8.1.3 Design Conclusion

The design chapter uses the functions identified in the analysis chapter as well as the Usability information from the Literature Review to create prototype Interfaces, Structure and Navigational Diagrams, Data Flow Diagrams and Use Case Diagrams that represent the entire application.

8.1.4 Implementation Conclusion

The Implementation Chapter contains details on how the Application was created, including illustrated drawings for interface design and examples of code along with descriptions of what the code does.

8.1.5 Testing Conclusion

The testing involved an in-depth test plan that lead to the discovery of errors within the application. The errors found were discussed in detail that explained the error and how it could be resolved. The test plan devised covered all the components of the system that a user would have direct interaction with. Due to the lack of resources available, the tests that were conducted could not be conducted to the industry standard.

8.1.6 Evaluation Conclusion

The evaluation chapter is the critical analysis of the application in terms of Usability. The prototype was tested by five participants that have been assisting the project since the beginning. The results of the Evaluation showed that the application was usable and user-friendly but still needed much improvement as the performance of the application was still below that of the averages found from the current existing calendar applications.

8.2 Project Aims and Objectives Evaluation

The aims of the project were to produce a Calendar application that was open-source, and user friendly that would allow a new student to see how data management between an Android device and an internal database works with the addition of social media integration.

In order to successfully achieve the aims of the project, all the objectives listed in chapter one must be achieved. Four out of five of these objectives were achieved; it was the implementation objective that was not fully achieved. This meant that the aim was only partially successful, as time constraints and the architecture design of the application produced meant that the integration of social media would not be possible without a re-design of the architecture. The initial assumption was that an API would be used to integrate the Facebook Calendar and Events, but this was not the case after conducting completing the research chapter (*See Section 8.4.1 for more information*).

Another challenge came during the Testing and Evaluation stage of the project as testing and evaluation to the high standard required was a new experience. This meant investing more time than was initially set to complete these stages to ensure that each task was completed to the high standard that was necessary.

After conducting the Evaluation of the application, it was found that design and usability of the product produced was not on the level of the all of the current existing applications even though all the functions were implemented that allowed for the addition, modification and deletion of calendar entries.

8.3 Limitations

During the project there were a number of things that slowed down or hindered progress. These limitations have been identified and will be discussed below:

8.3.1 Possible Feedback Bias

The testing candidates consisted of Family & Friends; this could have caused feedback to be potentially biased in my favour when the prototypes were being evaluated. Given the opportunity, I would repeat the research process using different methods of collection such as student forums and social media as this would produce a wider range for a sample group and would rule out the possible Bias from only using friends.

8.3.2 Time Constraints

Due to Time Constraints, there were some features that did not get implemented into the product as there implementation would have require a re-write of the applications architecture. This was down to the fact that the project fell victim to one of the downfalls of using the Iterative Waterfall process for development. *See Section 2.5.2.1.*

8.3.3 Personal Knowledge

Due to my existing knowledge of both Android Development techniques and Heuristics evaluation, progress was slow to begin with as with the Android Development; new techniques were being learnt and implemented at the same time during development. This made troubleshooting harder and made some sections of the implementation take longer.

My lack of Heuristics knowledge was more of a learning experience than a limitation as it forced me to research into a subject area that I would not have done otherwise.

8.4 Future Improvements

Although the calendar is functional, there is still much room for improvement on the look and feel of the applications interfaces and usability, and growth in terms of adding in new features or re-working the existing features to function more fluidly. These improvements will be discussed below.

8.4.1 Social Media Integration

Although the application is functional, social media integration was not possible. If I was to continue to develop the application and there was enough time available, I would re-design the architecture of application to allow this integration to be possible. Initial research was done into the Facebook API in order to sync event and birthday data to the calendar of the Prototype but the testing of the current calendar applications on the market revealed that these applications automatically pull data from the accounts that are logged into the account manager on the Android device. If any details of event are changed then these changes can be seen mirrored across all calendar applications that are set to show the events of the email account. To implement the social media syncing in the manor that the others did would have required the database structure to be changed as well as re-writing the Data Access Object.

8.4.2 Alarm Manager Integration

During development, a feature that was requested was repeat alarms, by Day, by Week and by Year. An unsuccessful attempt was made as Alarms can be set but they cannot be made to repeat. Further research lead to a built in Android Alarm Manager Function for Android Version 4.4.2 and the new, more Advanced Job Scheduler for Android Version 5.0. The Alarm Manager and Job Scheduler allows for repeat alarms and timers as well as other things. Integrating the Alarm Manager into the current prototype would have require a complete overhaul of how the time and date values are handled within the application, but given extra time these changes could be made and a fully functional alarm manager could be developed and integrated. Implementing this feature would allow for custom notification sounds and push notifications to be used for the application.

8.4.3 Calendar Interface

If there was more time for re-designing and re-implementing the calendar view, I would make the days of the months more interactive by increasing the size of each Day cell, allowing for marks or symbols to be shown on the dates that have events easier. As currently, the user must know what day the events are on or they must view all the events if they do not know the date it is on. Another addition to the Calendar Interface would be to highlight the days as a user moves their finger over the cells and introduce haptic feedback (vibration of the phone) as another form of confirmation that the users input has been registered by the application.

The colour scheme was not highly rated by users, a future improvement would be to add the ability for users to select their own colour schemes or at least include a selection of pre-set schemes and themes that a user would be able to select.

8.4.3 Use of Different Test groups

As the feedback on both the existing applications and the prototypes was received via Questionnaires, if I were to repeat the process I would collect data from different groups. For example, collecting data from a range of age groups and Computer Literacy Levels so that the results could be compared. This would show if the application was as friendly to those with lower levels of Computer Literacy as it is to someone who is an intermediate or an expert.

8.5 Personal Gains

Over the course of the project, a lot of experience was gained in my Android Development as well as my understanding of the important of Usability testing and the effort that developers and designers put into Usability to ensure a product is suitable and useable by its target audience.

My biggest gain was in Android Development. During my 2nd year at University I did half a term of Android Development, but the content of the course was slightly out-dated. The Android Version we used was 2.3.3 when at the time the current Android version was 4.1; this meant that there were a number of newer techniques and functions built in that I had no experience of. Developing the application for this project forced me to do more research then I had done previously that revealed techniques and functions that would suit my needs. Unfortunately, as the device I own (Samsung Galaxy S4) that was used for testing is only updateable up to Android Version 4.4, with the newest version being 5.0. There were a small selection of new, built in functions that I was not able to try and test.

8.6 Project Summary

The project can be considered to be successfully, although only partially as one function was not successfully implemented.

In terms of a user-friendly application, the project managed to design and successfully implement the following functions and features:

- Adding a new event
- Modifying an existing event
- Deleting an existing event
- A database to store calendar entries

- A sub-system that checks for the existence of database to ensure full functionality of the application.
- A Data Access Object to manage the retrieval, updating and deletion of data from the database.
- A calendar overview, although the overview that was used was open-source.

Due to the number of goals achieved compared to the one goal that was not achieved, as well as the Usability and Heuristics feedback received from the prototype, the project can be seen as a success.

Appendices

1 Questionnaires

1.1 Basic Information Survey

1. Gender.

◯ Male

◯ Female

2. Age.

◯ 17 or younger

◯ 18-20

◯ 21-29

◯ 30-39

◯ 40-49

◯ 50-59

◯ 60+

3. In a typical Day, which types of apps do you use on your Computer/Smart Device most often.

◯ Utility apps (calculate, convert, translate, etc.)

◯ Entertainment apps (movie trailers, celebrity gossip, radio station guides, etc.)

◯ Game apps (puzzles, charades, etc.)

◯ News apps (local news, national headlines, technology announcements, etc.)

◯ Productivity apps (calendar, to do list, price checker, etc.)

◯ Search tool apps (directions,) phone numbers, recipes, etc.)

◯ Social networking apps (location check-ins, friend status updates, etc.)

◯ Sports apps (sports schedules, scores, headlines, etc.)

◯ Travel apps (airplane tickets, tourist guides, public transportation info, etc.)

◯ Weather apps (local forecasts, natural disaster updates, etc.)

4. How would you rate yourself when it comes to using Computers/ Smart Devices.

○ Novice

○ Beginner

○ Intermediate

○ Expert

1.2 Application Feedback

1.Which app did you test?

○ Google Calendar

○ Fertility Calendar

○ Jorte Calendar & Organiser

○ Business Calendar

○ SOLCalendar

2. Which feature(s) did you like?

3. Why did you like these feature(s)?

4. Which feature(s) didn't you like?

5. Why didn't you like these feature(s)?

6. What do you think needs improving and/or adding and why?

1.3 SUS Analysis

SUS Analysis - [App Name Here]

1. I think that I would like to use this system frequently

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

2. I found the system unnecessarily complex

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

3. I thought the system was easy to use

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

4. I think that I would need the support of a technical person to be able to use this system

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

5. I found the various functions in this system were well integrated

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

6. I thought there was too much inconsistency in this system

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

7. I would imagine that most people would learn to use this system very quickly

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

8. I found the system very cumbersome to use

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

9. I felt very confident using the system

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

10. I needed to learn a lot of things before I could get going with this system

○ Strongly Disagree ○ Disagree ○ Neither Disagree Not Agree ○ Agree

○ Strongly Agree

1.4 Usability Checklist

		✓	Comments
1	Easy to Navigate		
2	Clear and Consistent Way to go back on every screen		
3	Labels and button text are clear and concise		
4	Retains overall consistency and behaviour with mobile platform		
5	Minimalist Design – excess features removed		
6	Content is concise and clear		
7	Provides feedback to the user of system status		
8	Number of buttons / links is reasonable		
9	UI elements provide visual feedback when pressed		
10	Ensure any visual feedback is not obscured by the user's finger		
11	Colours used provide good contrast		
12	Colours used provide good readability		
13	Icons are clear to understand – No ambiguity		
14	Font size and spacing ensures good readability		
15	If changes can be made, ensure there is a'Save' button.		
16	Present users with a confirmation when deleting,		
17	Allow users to tailor frequent actions to make them easier and quicker to do.		
18	Speak the users' language		
19	Auditory feedback is timely and appropriate		
20	Settings to turn off auditory feedback & sound.		
21	Help users to recognise, diagnose and recover from errors.		
22	Error messages are free of technical language.		
23	Error messages clearly explain how to correct the problem.		
24	Any help text should be clear and unambiguous		
25	Instructions easily visible or easily retrievable whenever appropriate		
	Total		Out of **125**

1.5 Prototype Feedback

1. What was your first reaction this product?

2. Did you find it easy to navigate the app?

○ Very Hard ○ Hard ○ Neurtal ○ Easy ○ Very Easy

3. What do you like about the app?

4. What do you dislike about the app?

5. What Features & Functions would you like to see added to the app?

1.6 Design Heuristics Questions

Layout

☐Needs Improvement ☐Poor ☐OK ☐Good ☐Excellent

Colour Scheme

☐Needs Improvement ☐Poor ☐OK ☐Good ☐Excellent

Consistency

☐Needs Improvement ☐Poor ☐OK ☐Good ☐Excellent

Font Size

☐Needs Improvement ☐Poor ☐OK ☐Good ☐Excellent

Ease of Use

☐Needs Improvement ☐Poor ☐OK ☐Good ☐Excellent

References

Beal, V. (2014). *What is Application Program Interface (API)? Webopedia*. [online] Webopedia.com. Available at: http://www.webopedia.com/TERM/A/API.html [Accessed 15 Nov. 2014].

Brooke, J. (1996). *Usability Evaluation In Industry*. [online] Google Books. Available at: http://books.google.co.uk/books?hl=en&lr=&id=IfUsRmzAqvEC&oi=fnd&pg=PA189&dq=Usability&ots=G9jwGcmr2j&sig=o0whmYlKq4ch6oJDx1Iurmg6aRk#v=onepage&q=Usability&f=false [Accessed 14 Nov. 2014].

Burnette, E. (2009). Hello, Android: Introducing Google's Mobile Development Platform. *Pragmatic Bookshelf*. [online] Available at: http://dl.acm.org/citation.cfm?id=1816808 [Accessed 13 Nov. 2014].

BusinessDictionary.com, (2014). *What is programming language? definition and meaning*. [online] Available at: http://www.businessdictionary.com/definition/programming-language.html [Accessed 14 Nov. 2014].

Carol, J. and Rosson, M. (2014). *Usability Engineering*. [online] Google Books. Available at: http://books.google.co.uk/books?hl=en&lr=&id=JCYTOCOugWAC&oi=fnd&pg=PP2&dq=info:2avTUB62X-gJ:scholar.google.com&ots=2yrt-stPil&sig=ucGs5Lab9rTFH41Kun0c2-wg-RE&redir_esc=y#v=onepage&q&f=false [Accessed 10 Nov. 2014].

Codefutures, (n.d). *Data Access Object - Codefutures*. [online] Available at: http://codefutures.com/data-access-object/ [Accessed 19 Dec. 2014].

Developer.android.com, (2014). *Introduction to Android | Android Developers*. [online] Available at: https://developer.android.com/guide/index.html [Accessed 9 Nov. 2014].

Developer.apple.com, (2014). *App Store Review Guidelines - Apple Developer*. [online] Available at: https://developer.apple.com/app-store/review/guidelines/ [Accessed 10 Nov. 2014].

Developer.apple.com, (2014). *Cocoa Touch - iOS Technology Overview - Apple Developer*. [online] Available at: https://developer.apple.com/technologies/ios/cocoa-touch.html [Accessed 15 Nov. 2014].

Enck, W. (2011). *A Study of Android Application Security*. [online] usenix.org. Available at: https://www.usenix.org/legacy/event/sec11/tech/slides/enck.pdf [Accessed 17 Nov. 2014].

Gartner.com, (2014). *Gartner Says Annual Smartphone Sales Surpassed Sales of Feature Phones for the First Time in 2013*. [online] Available at: http://www.gartner.com/newsroom/id/2665715

[Accessed 11 Nov. 2014].

Geeks Talk, (2007). *About DAO Design pattern*. [online] Available at: http://www.geekinterview.com/talk/5827-about-dao-design-pattern.html [Accessed 19 Dec. 2014].

Google Books, (2014). *Elements of User Experience,The*. [online] Available at: http://books.google.co.uk/books?hl=en&lr=&id=9QC6r5OzCpUC&oi=fnd&pg=PT4&dq=The+Elements+of+User+Experience&ots=mG-9LtccVy&sig=CYyLM6i6d48oMVAB2Zg-POidGl8#v=onepage&q=The%20Elements%20of%20User%20Experience&f=false [Accessed 10 Nov. 2014].

Halvorson, K. and Rach, M. (2014). *Content Strategy for the Web*. [online] Google Books. Available at: http://books.google.co.uk/books?id=KIIQ3Fq9CM8C&printsec=frontcover&dq=Kristina+Halvorson,+Content+Strategy+for+the+Web&hl=en&sa=X&ei=r7RsVIXlIurksAT47oLwBg&ved=0CDUQ6AEwAA#v=onepage&q=Kristina%20Halvorson%2C%20Content%20Strategy%20for%20the%20Web&f=false [Accessed 11 Nov. 2014].

Historylearningsite.co.uk, (2014). *The Personal Computer*. [online] Available at: http://www.historylearningsite.co.uk/personal_computer.htm [Accessed 19 Nov. 2014].

Istqbexamcertification.com, (2014). *What is Waterfall model- advantages, disadvantages and when to use it?*. [online] Available at: http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/ [Accessed 12 Nov. 2014].

Java.com, (2014). *What is Java and why do I need it?*. [online] Available at: https://www.java.com/en/download/faq/whatis_java.xml [Accessed 9 Nov. 2014].

Jenkov, J. (n.d.). *The DAO Design Pattern | tutorials.jenkov.com*. [online] Tutorials.jenkov.com. Available at: http://tutorials.jenkov.com/java-persistence/dao-design-pattern.html [Accessed 19 Dec. 2014].

Johnnyholland.org, (2010). *Content Lifecycle: Closing the loop in content strategy | Johnny Holland*. [online] Available at: http://johnnyholland.org/2010/10/content-lifecycle-closing-the-loop-in-content-strategy/ [Accessed 11 Nov. 2014].

Joy, B. and Gosling, J. (2000). *The Java Language Specification*. [online] Google Books. Available at: http://books.google.co.uk/books?hl=en&lr=&id=Ww1B9O_yVGsC&oi=fnd&pg=PA1&dq=Java&ots=Sf-GfkUdoA&sig=2O_LfeIsP0Q8lUUX5Yw1HUbK0NA#v=onepage&q=Java&f=false [Accessed 10 Nov. 2014].

Krill, P. (2014). *Apple's Swift is instant hit among top programming languages*. [online] InfoWorld. Available at: http://www.infoworld.com/article/2608054/application-development/apple-s-swift-is-instant-hit-among-top-programming-languages.html [Accessed 10 Nov. 2014].

Kuniavsky, M. (2014). *Smart Things: Ubiquitous Computing User Experience Design*. [online] Google Books. Available at: http://books.google.co.uk/books?hl=en&lr=&id=-WLyUCBBUVAC&oi=fnd&pg=PP2&dq=user+experience+design&ots=HB-Zz5yvlx&sig=ZfNWeCYM0Wqdc4eEnQUA_ZZWous#v=onepage&q=user%20experience%20design&f=false [Accessed 8 Nov. 2014].

Mall, R. (2014). *FUNDAMENTALS OF SOFTWARE ENGINEERING*. [online] Google Books. Available at: http://books.google.co.uk/books?id=ppIjvQdEDhIC&pg=PA42&dq=The+Waterfall+Model&hl=en&sa=X&ei=ibVsVPbnHOLHsQTQ34KIDw&ved=0CCAQ6AEwAA#v=onepage&q=The%20Waterfall%20Model&f=false [Accessed 12 Nov. 2014].

Measuringu.com, (2014). *10 Essential Usability Metrics: MeasuringU*. [online] Available at: https://www.measuringu.com/blog/essential-metrics.php [Accessed 12 Nov. 2014].

Metz, C. (2014). *Why Coders Are Going Nuts Over Apple's New Programming Language | WIRED*. [online] WIRED. Available at: http://www.wired.com/2014/06/apple-swift-language/ [Accessed 9 Nov. 2014].

Nielsen, J. (1994). Usability inspection methods. *Conference companion on Human factors in computing systems - CHI '94*.

Nielson, J. and Norman, D. (2014). *The Definition of User Experience (UX)*. [online] Nngroup.com. Available at: http://www.nngroup.com/articles/definition-user-experience/ [Accessed 7 Nov. 2014].

Nielson, J. (2014). *Usability Engineering*. [online] Google Books. Available at: http://books.google.co.uk/books?hl=en&lr=&id=DBOowF7LqIQC&oi=fnd&pg=PP1&dq=Usability&ots=Bk16XSHYxQ&sig=fi1bDLYmurcKtgAVfxhaCcgkNlc#v=onepage&q=Usability&f=false [Accessed 12 Nov. 2014].

Nielson, J. (2014). *Mobile Usability*. [online] Available at: http://ptgmedia.pearsoncmg.com/images/9780321884480/samplepages/0321884485.pdf [Accessed 13 Nov. 2014].

Nielson, J. (2014). *Usability 101: Introduction to Usability*. [online] Nngroup.com. Available at: http://www.nngroup.com/articles/usability-101-introduction-to-usability/ [Accessed 13 Nov.

2014].

Park, M. (2014). *Mobile Application Security: Who, How and Why*. [online] owasp. Available at: https://www.owasp.org/images/c/cf/ASDC12-Mobile_Application_Security_Who_how_and_why.pdf [Accessed 14 Nov. 2014].

Quora.com, (2014). *What are the benefits of User Experience design? - Quora*. [online] Available at: http://www.quora.com/What-are-the-benefits-of-User-Experience-design [Accessed 11 Nov. 2014].

Rockly, A. and Cooper, C. (2014). *Managing Enterprise Content*. [online] Google Books. Available at: http://books.google.co.uk/books?id=82X6jGY_dHMC&pg=PT117&dq=Content+Lifecycle&hl=en&sa=X&ei=FrVsVJ3eNrDHsQTO0oCQCg&ved=0CCsQ6AEwAA#v=onepage&q=Content%20Lifecycle&f=false [Accessed 12 Nov. 2014].

Rouse, M. (2014). *What is waterfall model? - Definition from WhatIs.com*. [online] Searchsoftwarequality.techtarget.com. Available at: http://searchsoftwarequality.techtarget.com/definition/waterfall-model [Accessed 12 Nov. 2014].

Rouse, M. (2014). *What is application security? - Definition from WhatIs.com*. [online] Searchsoftwarequality.techtarget.com. Available at: http://searchsoftwarequality.techtarget.com/definition/application-security [Accessed 15 Nov. 2014].

Sauro, J. (2011). *Measuring Usability with the System Usability Scale (SUS): MeasuringU*. [online] Measuringu.com. Available at: http://www.measuringu.com/sus.php [Accessed 14 Nov. 2014].

Searchsoftwarequality.techtarget.com, (2014). *Five application security threats and how to counter them*. [online] Available at: http://searchsoftwarequality.techtarget.com/tutorial/Five-application-security-threats-and-how-to-counter-them [Accessed 16 Nov. 2014].

Stackoverflow.com, (2009). *Pros and Cons of the use of DAO pattern*. [online] Available at: http://stackoverflow.com/questions/1748238/pros-and-cons-of-the-use-of-dao-pattern [Accessed 19 Dec. 2014].

The Huffington Post, (2014). *Why Most Developers Should Avoid Apple's New Swift Language*. [online] Available at: http://www.huffingtonpost.com/gabe-sumner/why-most-developers-shoul_b_5454013.html [Accessed 9 Nov. 2014].

Travis, D. (2014). *Usability testing, discount usability and usability metrics*. [online] Userfocus.co.uk.

Available at: http://www.userfocus.co.uk/articles/discount.html [Accessed 12 Nov. 2014].

Tullis, T. and Albert, W. (2014). *Measuring the User Experience*. [online] Google Books. Available at: http://books.google.co.uk/books?hl=en&lr=&id=bPhLeMBLEkAC&oi=fnd&pg=PA1&dq=user+experience&ots=R8KghqUUpM&sig=p3Z2MgqcoIH5ZW979ejVLVpZvyo#v=onepage&q=user%20experience&f=false [Accessed 8 Nov. 2014].

Usability.gov, (2014). *User Interface Design Basics*. [online] Available at: http://www.usability.gov/what-and-why/user-interface-design.html [Accessed 10 Nov. 2014].

Usabilitynet.org, (2014). *UsabilityNet: Definition of usability*. [online] Available at: http://www.usabilitynet.org/management/b_what.htm [Accessed 12 Nov. 2014].

User Experience. (n.d.). [image] Available at: http://www.texavi.com/blog/wp-content/uploads/2012/12/User-Experience_5E_Focus-on-Users-1024x724.png [Accessed 8 Nov. 2014].

UXPin, (2014). *Navigating the Mobile Application: 5 UX Design Patterns - UXPin*. [online] Available at: http://blog.uxpin.com/5308/navigating-mobile-application-5-ux-design-patterns/ [Accessed 10 Nov. 2014].

Webopedia.com, (2014). *What is SDLC? Webopedia*. [online] Available at: http://www.webopedia.com/TERM/S/SDLC.html [Accessed 13 Nov. 2014].

Wikipedia, (2014). *Swift (programming language)*. [online] Available at: http://en.wikipedia.org/wiki/Swift_(programming_language) [Accessed 12 Nov. 2014].

Williams, O. (2014). *What Developers Think About Apple's New Swift Programming Language*. [online] The Next Web. Available at: http://thenextweb.com/apple/2014/06/03/developers-apples-swift-huge-potential/ [Accessed 15 Nov. 2014].

Yoder, J. and Baracalow, J. (1997). *Architectural Patterns for Enabling Application Security*. [online] Available at: http://www.idi.ntnu.no/emner/tdt4237/2007/yoder.pdf [Accessed 16 Nov. 2014].

Anroidhub4you. (2012) Custom Calendar In Android. [online] Available at: http://www.androidhub4you.com/2012/10/custom-calendar-in-android.html [Access 27 Nov. 2014].

Myers, G. (2004). *The Art of Software Testing*. [online] Google Books. Available at: http://www.carlosfau.com.ar/nqi/nqifiles/The%20Art%20of%20Software%20Testing%20-%20Second%20Edition.pdf [Accessed 13 Mar. 2015].

Marcotty, M. (2008). *Formal Methods and Testing*. [online] Google Books. Available at: https://books.google.co.uk/books?id=WnQM6rulf74C&pg=PA155&lpg=PA155&dq=Marcotty,+M.+1

991+testing&source=bl&ots=c9Vbjq-PXN&sig=23dtpW4Dj3xv62VX2Lk5IctHw4w&hl=en&sa=X&ei=VxjqVLuCHZHgaIvpgMAP&ved=0CCEQ6AEwAA#v=onepage&q&f=false [Accessed 13 Mar. 2015].

Nielson, J. (1995). *Heuristic Evaluation: How-To: Article by Jakob Nielsen*. [online] Nngroup.com. Available at: http://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/ [Accessed 27 Feb. 2015].

keepitusable, (2011). *mobile app usability checklist*. [online] Available at: http://www.keepitusable.com/keepitusable-mobile-app-usability-checklist.pdf [Accessed 4 Mar. 2015].

BBC. (No Date). Life Of Crime. Retrieved November 12, 2010, from BBC: http://news.bbc.co.uk/hi/english/static/in_depth/uk/2001/life_of_crime/cybercrime.stm

BeBit. (2001, June 14). Usability Columns. Retrieved February 15, 2011, from BeBit: http://www.bebit.co.jp/english/column/column008.html

Nielsen, J., and Molich, R. (1990). Heuristic evaluation of user interfaces, Proc. ACM CHI'90 Conf. (Seattle, WA, 1-5 April), 249-256. Sociology.org.uk. (No Date). Semi-structured Interviews. Retrieved November 12, 2010, from Sociology: http://www.sociology.org.uk/methfi.pdf

Sudman, Seymour. 1976. Applied Sampling. New York: Academic Press Tedeschi, B. (2001, February 20). E-commerce Report: New Alternatives to Banner Ads. New York Times.

Nielsen, J. (2000). Designing Web Usability. Indiana: New Riders.

Nielsen, J. (2000, March 19). Why You Only Need to Test with 5 Users. Retrieved February 15, 2011, from UseIt: http://www.useit.com/alertbox/20000319.html