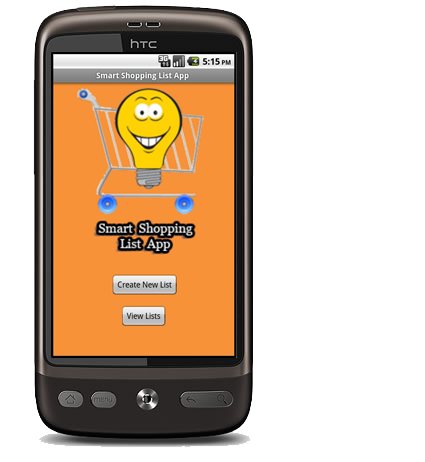
Smart Shopping List App for Android OS



Final Report

Table of Contents

[1. Abstract / Description of the title 3](#_Toc292459509)

[2. Declaration of Authenticity 4](#_Toc292459510)

[3. Introduction 5](#_Toc292459511)

[3.1 Overview 5](#_Toc292459512)

[3.2 Risk Management 6](#_Toc292459513)

[3.2.1 Risk Identification and Management 6](#_Toc292459514)

[3.2.2 Risk Realisation and Response 11](#_Toc292459515)

[3.3 Approach / Considerations 14](#_Toc292459516)

[3.3.1 Hardware 15](#_Toc292459517)

[3.3.2 Software 16](#_Toc292459518)

[3.3.3 Similar Projects 17](#_Toc292459519)

[3.3.4 Constraints 17](#_Toc292459520)

[4. Project Analysis and Specification 18](#_Toc292459521)

[4.1 Project Size 18](#_Toc292459522)

[4.2 Economic Feasibility 18](#_Toc292459523)

[4.3 Organisation Feasibility 19](#_Toc292459524)

[4.4 Technical Feasibility 19](#_Toc292459525)

[5. Requirements Analysis 20](#_Toc292459526)

[5.1 Software Functions 20](#_Toc292459527)

[5.2 Constraints 21](#_Toc292459528)

[6. Methodology 22](#_Toc292459529)

[7. Specification of Data Structures 24](#_Toc292459530)

[7.1 Introduction to System / User Specification 24](#_Toc292459531)

[7.2 Pre-normalisation tables 24](#_Toc292459532)

[7.3 Normalisation 24](#_Toc292459533)

[7.4 Table Structure 27](#_Toc292459534)

[7.5 DDL Statements 28](#_Toc292459535)

[8. User Interface 29](#_Toc292459536)

[9. Management of the Process 43](#_Toc292459537)

[10. Project Evaluation 46](#_Toc292459538)

[10.1 Problems 46](#_Toc292459539)

[10.2 Immediate project outcome: 46](#_Toc292459540)

[10.3 Long-Term project outcome 46](#_Toc292459541)

[10.4 Purpose of the Project 47](#_Toc292459542)

[11. Testing 48](#_Toc292459543)

[12. User Manual 49](#_Toc292459544)

[13. Conclusion 51](#_Toc292459545)

[14. References 52](#_Toc292459546)

[15. Appendices 53](#_Toc292459547)

[Appendix A 53](#_Toc292459548)

[Appendix B 0](#_Toc292459549)

# 1. Abstract / Description of the title

People these days are increasingly under pressure from time constraints and financial worries, and the weekly grocery shopping is one activity that every household must endure. It takes a great deal of time out of a household’s leisure time and is one of the biggest weekly financial expenditures in a household’s budget. With this in mind, this project will attempt to develop a mobile Application for the Google Android operating system. This should not only makes the activity less demanding with regard to time and money, it also makes shopping more effective by ensuring you don’t forget any items or choose the wrong ones.

When this Smart Shopping List App is launched, it will allow the user to build up a weekly shopping list, simply and easily, through a variety of input methods. Users can scan the barcode on a product (e.g. just before you put the box of Cornflakes in the bin, scan the bar code and it will add it to your personalised shopping list), they can also type the item in manually using the keyboard and attach a photograph using the mobile devices camera if they so desire or, finally, they can speak into the mobile device and use the voice recognition system to add items to their list.

While there are numerous Shopping List Applications already on the market, most only offer a subset of the input methods that are planned in this implementation. Some of these applications are only available commercially (i.e. the user must purchase them before use) while this application will be available for free.

# 2. Declaration of Authenticity

*Except where explicitly stated, this report represents work that I have done myself. I have not submitted the work represented in this report in any other course of study leading to an academic award.*

*---------------------------------------------------------------------------------------------------------------------------*

# 3. Introduction

## 3.1 Overview

This goal of this project is to develop a commercially viable Smart Phone application for the Android OS.

The application itself is designed to help users with their grocery shopping. The idea came from my own experience of using a Smart Phone; previously I made a lot of lists using pens and paper in order to remember things, however upon getting a Smart Phone I was impressed by the power and convenience of it to create notes and reminders. Another advantage of using a Smart Phone is that users generally have their device on or near their person at all times. It was the combination of these two observations that gave me the initial idea for the application.

The application is a specialist application that allows users to create a number of lists of groceries and then add items to these lists. Once a list is created users can add an item name (using a keyboard or speech recognition system), a quantity and a quantity type. Users can also attach an image to their description in order to better remember which product they want. Alternatively users can scan a barcode on an item and have this item added to their list. Once the list is created, users can then add, edit or delete items from this list.

Smart Phone sales have increased rapidly over the last number of months and the Android OS has overtaken Apple’s iOS and is now the world’s most popular Smart Phone platform. Google purchased Android Inc. in a 2005 acquisition and in 2007 with collaboration from the Open Handset Alliance (a consortium of technology companies) it released the first Android platform. The Android Market now has almost 300,000 apps available to download and is poised to overtake Apple app store for the first time in August 2011 (Mack, 2011).

During the Analysis and Design stage in order to better understand the potential audience some market research was conducted in the form of an online survey. The only requirement to take part was that participants must own a Smart Phone. Over the course of 5 weeks, 45 people participated and a summary of the findings are described below.   
The average Smart Phone user is:

* Male
* Aged between 25 and 30
* Single
* Either a student or works in an I.T. related field
* Regularly uses utility applications on their device, such as calculators / currency   
  convertors etc...
* Believes that these type of application are genuinely useful and not a novelty
* Expect these applications to be free
* Rely solely on their memory when shopping – don’t use a shopping list
* But would be willing to use a shopping list application on their device

However it should be stated not to rely too heavily on the results of this survey as there may be anomalies in the participants: perhaps males between 25 and 30 are more likely than any other group to participate in online surveys.

## 3.2 Risk Management

A risk identification and management study was undertaken during the analysis and design stage of this project and the findings are listed below.

### 3.2.1 Risk Identification and Management

|  |  |
| --- | --- |
| ***Risk:*** | Primary and / or Secondary storage failure |
| ***Probability:*** | ***Moderate*:**  The probability of something like this happening (at least once) over the duration of a project like this cannot be ruled out. |
| ***Effect:*** | ***Serious*:** Depending on how long it has been since a back-up was last taken, it could put the project behind schedule by anything from a matter of hours to a matter of weeks. |
| ***Minimisation Strategy:*** | Any files or documentation should be backed up to a number of locations e.g. Memory Stick, Laptop and/or PC, free online storage facility at the end of every session. Thus if the worst were to happen all that would be lost would be one day of work (at most). |

|  |  |
| --- | --- |
| ***Risk:*** | Illness |
| ***Probability:*** | ***Moderate*:** Again over the duration of this project the probability of illness affecting the project schedule cannot be ruled out. |
| ***Effect:*** | ***Insignificant – Serious:*** The effects on the project depend on how long the illness lasts for and how serious it is. |
| ***Minimisation Strategy:*** | Unfortunately this type of risk cannot be ruled out, and as this project is being undertaken by one person, if the worst were to occur there would not be much that could be done about it. The risk could be slightly minimised by keeping up to date with the planned development schedule, so that if an illness were to occur for a number of days, that the project would only be that number of days behind schedule. |

|  |  |
| --- | --- |
| ***Risk:*** | Hardware Unavailability – Mobile Device / Laptop failure |
| ***Probability:*** | ***Low - Moderate:*** The probability of either of the primary hardware devices used in this project failing are possible but unlikely. |
| ***Effect:*** | ***Tolerable – Serious:*** Depending on when the failure took place, the effects could be serious or tolerable. If the Laptop failed it would be very inconvenient as it would mean having to use either inferior equipment in a less convenient location, while the Laptop was awaiting repair. Also if the Mobile Device being developed for were to fail or became lost coming up to a demonstration it could have serious consequences. |
| ***Minimisation Strategy:*** | By running anti-virus checks on the Laptop and general good practice: making sure updates are installed, hard disk isn’t too full, etc... the risk of a failure could be minimised. With regard to the Mobile Device, it should be insured against breakage. There has also been a contingency list developed containing the names and contact details of colleagues with a similar Mobile Device, in case the planned Mobile Device is unavailable. |

|  |  |
| --- | --- |
| ***Risk:*** | Requirements Change |
| ***Probability:*** | ***Low – Moderate:*** Because implementation of the application has not yet begun, it is impossible to say which aspects of the planned functionality will be possible to implemented, and whether or not the planned functionality is too ambitious or not ambitious enough. |
| ***Effect:*** | ***Tolerable:*** Once implementation begins, it should become apparent very quickly if the complexity and time budgeted for the planned functionality have been underestimated or overestimated. |
| ***Minimisation Strategy:*** | In order to minimise the risk of requirements change affecting the delivery of the project a multi-pronged approach can be adopted. Firstly, thorough research at the outset as to what is required and how it is implemented can help minimise the risk of the requirements having to be changed later on in the project. Secondly, strict adherence to the planned schedule means that implementation will start on time and will indicate whether the planned functionality has been underestimated or overestimated. Thirdly, choosing a suitable development methodology which will accommodate requirements change will greatly help to minimise the risk. The chosen methodology for this project is SCRUM which is very suited to requirements change, as it allows for a series of rapid iterations after which a potentially shippable product increment is produced. |

|  |  |
| --- | --- |
| ***Risk:*** | Size Underestimation |
| ***Probability:*** | ***Moderate:*** Again as implementation has not begun except for a few exploratory tutorials, it is very possible that the size of this project has been underestimated. |
| ***Effect:*** | ***Tolerable:*** Although it is possible that the size of this project has been underestimated, it is unlikely that the size has been so grossly underestimated that it is not possible to meet the core goals of the project and deliver enough basic functionality for the application to function. Although a situation such as this is very undesirable, it should still be possible to deliver a fully functional prototype as a worst case scenario. |
| ***Minimisation Strategy:*** | Thorough research into what is required at the outset of the project can give a better indication as to whether the size of the project is being underestimated. Also the methodology used to develop the software can have a big impact as to how to react should a situation arise where the size of a project has been underestimated. A methodology that is flexible with regard to requirements change would be useful in minimising the risk of underestimating the size of the project. As the chosen methodology for this project is SCRUM, this allows for requirements change through a number of iterations that result in a potentially shippable product increment. This is the most suitable development methodology for minimising the risk of size underestimation. |

|  |  |
| --- | --- |
| ***Risk:*** | CASE Tools Underperform |
| ***Probability:*** | ***Low:*** Initial exposure to the Android SDK and the Eclipse IDE, demonstrate that both tools seem to be well-designed and are powerful and easy to use. In addition, the number of Android applications that have already been developed (approximately 80,000: the majority using these technologies) show that the CASE tools perform as expected, if not better. Therefore the risk of them underperforming is low. |
| ***Effect:*** | ***Serious:*** If however it was the case that the CASE Tools did underperform it would have serious implications on this project. Although it is not necessary to use the Eclipse IDE to develop Android applications, it is the recommended method for development and it also includes various plug-ins such as emulators which can make the development process much more efficient and effective. |
| ***Minimisation Strategy:*** | The only way to mitigate against the CASE Tools underperforming is to research and plan. Research beforehand can be used to gauge the performance of the CASE tools and highlight whether there will be any problems down the line. The more research that is done at this stage then the more the risk of the CASE tools underperforming is minimised. Planning how to react should the CASE tools underperform is another method of minimising the risk. |
| ***Risk:*** | Technology Change |
| ***Probability:*** | ***Very low:*** The Android SDK is based on a subset of the Java programming language, there are approximately 80,000 applications developed using this SDK and the Android operating system is installed on millions of devices (and these figures are growing rapidly). So it would be very unlikely that the underlying technology would change, except for Android updates which occur a couple of times per year. However as these updates are backward compatible this should not affect the development of this application. |
| ***Effect:*** | ***Catastrophic:*** If the underlying technology were to change it would have a catastrophic effect on this project. There is no easy way to convert one programming language to another and even if it were possible it would be highly unlikely that the internal architecture would be the same (if it were the same, what would be the point of the change?). In this case, it would mean starting the project again and possibly having to learn a new programming language and new development tools. |
| ***Minimisation Strategy:*** | There is not a lot that can be done to minimise the risk of the underlying technology changing. Keeping up to date with news and updates from official and unofficial sources could help to give developers a warning about an impending change. The risk could be minimised by honing development skills in relation to other technologies in case the underlying technology changes. Investigation about how to implement the planned functionality in another programming language is also another possible means to mitigate against a change of the underlying technology. How practical an idea this would be in practice however is debatable at best. |

|  |  |
| --- | --- |
| ***Risk:*** | Product Competition |
| ***Probability:*** | ***Low – Moderate:*** The results of research conducted found a series of similar shopping list applications. The majority of these Android based applications were free, and while most of them offered option methods such as using inbuilt keyboards, barcode scanners, cameras or speech recognition, none of the applications allowed users all of these input methods. Also none of the applications found during research allowed users to send their finished list to be price-checked with the four main supermarkets. The same is true of the functionality that allows users to share their list with others who have the application installed. With so many shopping list application (free and paid) out there already, the prospect of a competitor releasing an application in such an already crowded market place is low to moderate. If a competitor were to release that sort of application however, it is likely that it would include a lot of the functionality planned for this project which would prove problematic. |
| ***Effect:*** | ***Serious:*** As this is a student project while the effect of a competitor releasing a similar product would be relatively serious and discouraging, the application would still be developed. However, If this were a real life project the consequences of a competitor releasing a similar project would be much more severe and may even lead to termination of the project as well as monetary loss of the time and resources used in developing the project thus far. |
| ***Minimisation Strategy:*** | There is not a lot that can be done to minimise the risk of a competitor releasing a similar product. Ensuring that this project is running on schedule in order to avoid delays in which time competitors who may have started their development later, have the time to catch up can help minimise the effect of a competitor developing a similar application. It may be possible to investigate trying to copyright and patent designs so that a competitor cannot copy the ideas outlined in this project. Or perhaps it may be possible to publicise this project which would discourage competitors from developing a similar application (although this might have the reverse effect and give our competitors ideas about this application and an insight into how this project has been developed). |

### Risk Realisation and Response

Over the course of the development of this project the following risks became a reality.

|  |  |
| --- | --- |
| ***Risk*** | Illness |
| ***Description*** | Just after Analysis and Design stage was complete, I contracted flu which meant the project development could not start on schedule and the project was approximately two weeks behind schedule when the development did start. |
| ***Handling*** | As outlined in the Risk Analysis section, this type of risk cannot be avoided, and unfortunately if it does occur there is little that can be done about it. To counteract the effect of this delay an updated schedule was created and more resources (time) were put into the project in the initial stages in order to catch up. |
| ***Effect*** | Slight: Although the realisation of this risk was less than ideal, particularly at the beginning of the project development, it was not a major setback. It simply meant having to devote more time to the project when the development did actually start in order to catch up. |

|  |  |
| --- | --- |
| ***Risk*** | Size Underestimation |
| ***Description*** | The Analysis and Design stage may have been slightly too optimistic with the specification for this project. While the core functionality of the project was sound, some of the more ambitious aspects of the project e.g. allowing users to share shopping lists or calculate the price of their given list in the four main supermarkets were not feasible to implement in the time available. |
| ***Handling*** | In order to guarantee the successful delivery of the core goals and functionality of the project, a decision was made to the secondary goals from the project. |
| ***Effect*** | Slight – Moderate: Although the decision to remove some of the secondary goals for the project was not taken lightly, it needed to be done to ensure the overall success of the project. Thankfully the use of a flexible methodology such as SCRUM enabled these features to be stripped from the project with minimum disruption. |

|  |  |
| --- | --- |
| ***Risk*** | CASE Tools underperform |
| ***Description*** | The Eclipse IDE and the Android SDK like most other software development tools contain bugs. Although not a regular occurrence, sometimes perplexing issues would arise with the Eclipse IDE and Android SDK which needed to be overcome before work on the project could recommence. |
| ***Handling*** | In order to handle these issues that would occasionally arise, the best form of help came from internet forums devoted to Android, Java and Eclipse. Usually another developer had experienced the same issue, and often various different actions were offered as solutions, so there was an element of trial and error in order to find a satisfactory solution to a problem. A log (Appendix A) was kept containing a description of the problem and action taken in overcoming the issue once it was satisfactorily resolved, so that if the problem was encountered again it could be quickly and easily looked-up in this knowledge base and then resolved. |
| ***Effect*** | Slight: These errors and bugs were occasionally frustrating and took time to be overcome but they are to be expected with any project, and in overcoming them my knowledge and understanding of Eclipse and the Android SDK increased, so it ended up being a valuable experience. |

|  |  |
| --- | --- |
| ***Risk*** | Technology Change |
| ***Description*** | Although the underlying technology did not change there was an issue with the compatibility of different Android OS versions. The Android OS has many different platforms: three new OS platforms have been launched in the last 6 months, since the Analysis and Design stage was concluded. The targeted platform for this project did not change; it would still be developed for the Android 2.2 platform (as the Smart Phone device used in development ran this platform). The difficulty arose however at the start of the development process, in completing background tutorials for technologies similar to those that would be implemented in this project; Some of them technologies detailed in tutorials were only available for the newer Android platforms, while others contained code that had been deprecated in previous versions. |
| ***Handling*** | Just like the scenario of the CASE Tools underperforming that is described above, research was needed to overcome these problems caused by the occasional incompatibility of Android platforms. Internet forums for Android developers usually contained workarounds to deprecated code or gave explanations as to why a certain technology would not work with the targeted Android 2.2 platform. |
| ***Effect*** | Slight – Moderate: The effect of the occasional incompatibility between Android platforms ranged from a minor inconvenience to a major stumbling block. More often than not, some research would reveal solutions to deprecated code, however on certain occasions areas of functionality would not be possible to implement due to incompatible Android OS versions. |
| ***Risk*** | Complexity Underestimation |
| ***Description*** | The subject of this project was originally going to be an AR (Augmented Reality) application but was changed to a Shopping List application late on, due to a lack of support for the AR technology. This had a knock-on effect which meant that during the Analysis and Design stage there was not enough time to get familiar with the Eclipse and Android development tools. This then led to a situation where the initial stages of the development phase were spent completing beginner’s tutorials and getting accustomed to Eclipse and the Android SDK instead of actually developing the application. This coupled with the greater than expected complexity with regard to the Android SDK led to a situation where development of the project struggled to get started. |
| ***Handling*** | In order to overcome this issue outside help was needed. Support was found in the form of online forums and also from hands on assistance from a third-party who had some experience in developing Android applications. Over a couple of weeks a number of Android concepts were explained to me and I was shown examples of code which were somewhat similar this project and which formed a starting point to for the development of this project. |
| ***Effect*** | Moderate – Serious: This issue was perhaps the most serious of all the risks encountered in this project. Without sufficient support it was difficult to know where and how to start. The whole project may have been jeopardised without this support but thankfully it was available and it appropriate to the needs of this project. |

## 3.3 Approach / Considerations

In assessing this project the underlying technology and language was taken into consideration, in fact they were one of the main motivations in deciding to implement this project. There are a few reasons behind this decision: firstly the Android platform was chosen as it has close ties to Google, and in the past I have enjoyed working with Google API’s (Application Programming Interface) and have been impressed by the support available for them from Google and also from third-parties. Also the underlying development language of the Android platform was a modified version of Java, and as I already possessed a modest amount of Java experience I decided to proceed with this project. Although I did not have previous experience with the Eclipse IDE, I was aware that it was the industry standard for Java development so I was not daunted by using it.

### 3.3.1 Hardware

**HTC Desire Android device:** This is the mobile device that the application being developed will be targeted for. It has the following specifications (HTC Corporation, 2010):

CPU Processing Speed – 1 GHz  
Platform – Android 2.2 with HTC Sense  
ROM Storage – 512mb  
RAM Storage – 576mb  
Camera:

* 5 Megapixel Colour Camera
* Face Detection Capability
* Auto Flash and Focus
* Widescreen Photo Capture
* Geo-tagging

Internet:

* 3G – Up to 7.2 Mbps download speed
* 3G – Up to 2 Mbps upload speed
* GPRS – Up to 114 kbps downloading
* EDGE – Up to 560 kbps downloading
* Wi-Fi – IEEE 802.11 b/g

Tethering – Internet sharing through USB

Network - Europe:

* HSPA/WCDMA: 900/2100 MHz
* GSM: 850/900/1800/1900 Mhz.

Sensors:

* G-Sensor
* Digital Compass
* Proximity Sensor
* Ambient Light Sensor

### 3.3.2 Software

**Google Android SDK:** The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator (based on QEMU), documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.4.9 or later, Windows XP or later. The officially supported integrated development environment (IDE) is Eclipse (currently 3.4 or 3.5) using the Android Development Tools (ADT) Plug-in, though developers may use any text editor to edit Java and XML files then use command line tools (Java Development Kit and Apache Ant are required) to create, build and debug Android applications as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely). (Wikipedia, 2010a). The Android SDK will be development tools used in this project.

**Eclipse IDE:** Eclipse is a multi-language [software development environment](http://en.wikipedia.org/wiki/Software_development_environment) comprising an [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) and an extensible [plug-in](http://en.wikipedia.org/wiki/Plug-in_(computing)) system. It is written mostly in [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) and can be used to develop applications in Java and, by means of various plug-ins, other [programming languages](http://en.wikipedia.org/wiki/Programming_language) including [Ada](http://en.wikipedia.org/wiki/Ada_(programming_language)), [C](http://en.wikipedia.org/wiki/C_(programming_language)), [C++](http://en.wikipedia.org/wiki/C%2B%2B), [COBOL](http://en.wikipedia.org/wiki/COBOL), [Perl](http://en.wikipedia.org/wiki/Perl), [PHP](http://en.wikipedia.org/wiki/PHP), [Python](http://en.wikipedia.org/wiki/Python_(programming_language)), [Ruby](http://en.wikipedia.org/wiki/Ruby_(programming_language)) (including [Ruby on Rails](http://en.wikipedia.org/wiki/Ruby_on_Rails) framework), [Scala](http://en.wikipedia.org/wiki/Scala_(programming_language)), and [Scheme](http://en.wikipedia.org/wiki/Scheme_(programming_language)). (Wikipedia, 2010b).

**SQLlite:** SQLite is an [ACID](http://en.wikipedia.org/wiki/Atomicity,_consistency,_isolation,_durability)-compliant [embedded](http://en.wikipedia.org/wiki/Embedded_database) [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) contained in a relatively small (~275 [kiB](http://en.wikipedia.org/wiki/Kibibyte)) [C](http://en.wikipedia.org/wiki/C_(programming_language)) programming [library](http://en.wikipedia.org/wiki/Library_(computer_science)). The [source code](http://en.wikipedia.org/wiki/Source_code) for SQLite is in the [public domain](http://en.wikipedia.org/wiki/Public_domain) and implements most of the SQL standard. In contrast to other databases, SQLite is not a separate process that is accessed from the client application, but an integral part of it. (Wikipedia, 2010c). SQLite is the database of choice for mobile applications and as such it will be the database used in this project.

**SQLite Manager:** SQLite Manager is a developer plug-in for the Mozilla Firefox browser that allows developers to create SQLite Databases through a simple GUI. The database can then be simply exported to the “assets” directory of the Eclipse project.

**DroidDraw**: DroidDraw is a standalone third party UI (User Interface) designer and editor for the Android OS. Initial layouts and designs were completed using DroidDraw as it offered a more intuitive and user-friendly environment than the one provided in Eclipse. Once these design templates were generated in DroidDraw, they were brought into the Eclipse XML GUI editor for refinement.

**Adobe Illustrator (various versions):** Adobe Illustrator is a vector graphics editor and will be used to create some of the artwork for this project.

**Adobe Photoshop (various versions):** Adobe Photoshop is a graphics editing program. It will also be used to create some of the artwork for this project.

### 3.3.3 Similar Projects

There are already a number of similar applications readily available on the Android Market. These applications are not dissimilar to this application; however this application offers more convenience in certain areas, particularly with the amount of options available to users to add items to their list. There also exists a number of Supermarket branded applications which offer users standard shopping list functionality however they do this by providing the users with a list of all products that the supermarket stock, so users build up a list from choosing from products already listed in the application instead of adding the details of the items themselves. These branded applications also have the benefit of letting users see the price of items. This kind of application was not feasible for this project firstly due to the complexity and time which would be involved in creating it, but more importantly as access to a particular supermarket chains product and price list is a valuable asset that is not available to the public or third-party developers like myself.

### 3.3.4 Constraints

Time was one of the major constraints of this project. Although there was approximately an eight month period to analyse, design and implement the project, another idea was being considered initially before this shopping list application was chosen. This meant that the Analysis and Design stage started late and did not allow time to build a prototype and get familiarised with the development environment. This then had a knock-on effect on the development phase as part of this stage had to be give over to becoming familiar with the development environment and completing beginners Android tutorials. This meant that the secondary goals of this project had to be dropped in order to ensure the completion of the core goals.

Complexity was the other main constraint of this project. Although the underlying language of the Android SDK was Java, of which I already possessed a modest of amount of knowledge of, the Android SDK uses a specialised form of Java. This meant there were certain programming concepts specific to the Android SDK that I was unaware of and needed to thoroughly familiarise myself with in order to successfully complete the project.

# 4. Project Analysis and Specification

## 4.1 Project Size

This project is an individual project for a commercially viable Smart Phone application running the Android 2.2 OS (Operating System). The application is a standalone product, however it does require an Android powered handset to be displayed (or alternatively an Eclipse IDE with the Android SDK installed). To ensure full functionality a free barcode scanner application is also required but this can be downloaded to the device free at any time.

## 4.2 Economic Feasibility

There is potential for this project to be marketed and sold. There should also be scope to re-develop this application for different handsets running different Android OS versions without the need for too much time (or other resources) to be invested. However, although possible it would require substantial resources in order to develop the application for another Smart Phone OS, such as Apple’s iOS, as the underlying technologies are not compatible. In this case the benefits would outweigh the investment in resources that would be needed.

There are a few options for generating revenue from the application:

1. Sell the application to a company and allow them to brand it, i.e. the application could be sold to a supermarket chain e.g. Superquinn, who would brand it with their logo and colours and market it to their customers. This company could use this application with its’ database of products and price list to create a powerful application which would allow customers to select items from a pre-existing list and see the price of the item, rather than having to enter the detail manually themselves.
2. Revenue from selling the app on Android Market. This is the simplest and most obvious way to generate revenue from the application. The tested application is complied into an .APK file which is then uploaded to the Android Market, where developers can set the fee for the public to download and use the application. All that is required is an Android Developers licence which costs $25. In contrast to Apple’s iPhone however there is not the same prevalence of paid applications, so charging a fee for this application while other free alternatives are available may not make good economic sense.
3. The third option to generate revenue from the application is to release the application for free on the Android Market but include advertisements on it, which would generate revenue for us as developers. A strategy applied by many companies who develop Android applications seem to be to develop a full feature application that can be downloaded for a fee, while offering a free or “lite” version which contains more basic functionality and also includes advertisements.

## 4.3 Organisation Feasibility

Listing this application on the Android Market (which is the usual method of deploying applications onto user’s devices) allows developers to issue updates to users. Users are simply notified when connected to Wi-Fi or Mobile internet that an update for a particular application(s) are available and users can then decide if they would like to install any updates. This allows developers to issue updates for bug fixes or new features to users quickly and easily.

## 4.4 Technical Feasibility

Before undertaking this project I had a number of years of experience of Database Theory, SQL and Java programming. Java is the underlying language of the Android SDK (with XML used for screen layout) and as I already possessed a modest knowledge of each, this was one of the main motivations in decided to implement this project. From comparing this finished application to similar applications on the Android Market, this application ranks adequately with what is commercially available. This application could be refined further to improve the UI experience for the user and it may also be possible to add in some of the secondary technical goals which had to be dropped due to time and complexity constraints. If these secondary goals were realised, then this application from my research into the competition would be in the highly rated in comparison to competing applications available.

# 5. Requirements Analysis

## 5.1 Software Functions

During the Analysis and Design stage a phase of requirements analysis was conducted. This took the form of market research among Smart Phone users and also a study into existing similar applications to see what technologies they offered. From the requirements analysis phase it was decided to implement the following functional features into the project.

Create New List: Users need to be able to create new lists for different occasions in order to add items to.  
View List: Users need to be able to view the lists that they have created in the past so that they may use the list while shopping, edit it for their needs or else delete it.  
Choose List:Users need to be able to choose a list from the lists available in order to use it while shopping or else edit it for their needs.  
Delete List:Users need to be able to delete lists that they no longer need.  
Edit List: Users may want to edit existing lists instead of deleting them.  
Add Item – Description / Quantity / Quantity Type: Users need to be able to add items to their list specifying the item name, the quantity and the quantity type.  
Add Item - Using Barcode Scanner: Users may want to have the option of scanning a barcode on an item and then have it added to their list. This is possible with the Barcode Scanner.  
Add Item - Using Speech Recognition: Users may decide to enter an item’s description using speech recognition instead of the keyboard. This is possible with the Speech Recognition system available on all Android platforms from 2.1 onwards.  
Add Photo to item using Camera: When adding an item to a list, users may decide to use the Smart Phones camera to take a photograph of an item to help them better remember that item on their list.  
View Items on List: Users need to be able to view the items on their list, so that they can see what items they have already added and also so that they can mark items off their list as they go shopping.  
Delete item from list: Users may need to delete items from their list, if they are editing an existing list or else if they have added an item by mistake.  
Edit item on List: Users may need to edit an item if they are changing part of the item such as the quantity, or else if they have wrongly entered all or part of the item.  
Mark item on List: Users need to be able to mark an item as “got” while shopping so they know they have already added this item to their trolley.

## 5.2 Constraints

During the Analysis and Design stage, size and processing requirements were identified as possible constraints to the development of the application. As the application would run on a mobile device the overall size and processing required by the application would have to be as small as possible in order to allow it to run on more limited devices rather than just top of the range Smart Phones. In order to ensure that the application size is as small as possible, all images and sound must be optimised as these media elements contribute greatly to the overall size of the application. Application size is becoming less and less of an issue however as most Smart Phones now have several GB of internal memory and Micro SD cards are now available in sizes up to 32GB. What may be a bigger issue however is the processing power required, because while storage can be upgraded relatively cheaply, the Smart Phone processer cannot. The code would need to be optimised and the best OOP (Object Oriented Programming) practices put into place to ensure that code was reused and there were no needless reads or writes to the database. The overall feel of the application must be responsive to users and the performance must be fast in order for users to continue using the application and not uninstall it.

The Android platform itself may also be a constraining factor; the targeted device and platform for this application is a HTC Desire handset running Android Version 2.2, while some other handsets and Android versions may also be able to able to run the application, this may not be guaranteed without re-development.

In designing the application the decision was taken to maximise the chances of the application working on different platforms by ensuring that all interactions between the user and the application took place using the touch-screen. This would ensure that the application was be as compatible as possible as the alternative would have been to use some of the physical buttons on the HTC Desire device which may have caused compatibility issues on other devices that would not necessarily have the same buttons.

# 6. Methodology

The chosen methodology for this project is described below. The initial problems in starting the development of the project meant the sprint iterations had to be shortened significantly, however the structure and constitution of the methodology did not change.

**SCRUM**: This is a form of an Agile Project Management methodology. Work is confined to a regular, repeatable work cycle, known as a sprint or iteration. Each iteration can last anywhere between 2 and 4 weeks, but each iteration must be of a consistent duration. During each sprint, a shippable product is created, no matter how basic that product is. Due to the accelerate timeframe only the most essential functionality is built. This means that testing is conducted during the sprint iteration rather than afterwards to enable a functional shippable product to be produced at the end of the sprint. By placing an emphasis on working code, the developer prioritises a release’s most essential features and the developer must focus on the short-term goals. As each release requires many iterations to develop a satisfactory product, each iteration of work builds on the previous one. This is how SCRUM differs from a methodology like Incremental: SCRUM is “Iterative” and “Incremental”. Each sprint begins with deciding what functionality will be moved from the product backlog in to the sprint backlog, in order to be developed in the next sprint iteration. Once the sprint begins the work items in the sprint backlog cannot be changed, however if a work item cannot be implemented it is removed from the current sprint iteration and goes back into the sprint backlog for the next iteration.

 *SCRUM Software Model*

SCRUM was chosen initially as the methodology that was best suited to this project: this was due to there being a number of core and secondary goals for the project defined during the Analysis and design stage.

As implementation of the project progressed there were a number of problems that arose which are discussed elsewhere in this report. These problems meant that the SCRUM methodology could not be followed as was first anticipated as the schedule needed to be modified a number of times and that the secondary goals of the project needed to be discarded. A significant amount of the time set aside for development was instead exhausted getting familiar with the Eclipse environment, the Android SDK as well as a number of Android tutorials. Luckily the SCRUM methodology as it is based on an Agile methodology is highly flexible and can accommodate such changes relatively easily. At the culmination of this project I believe that this methodology was the correct one as it allowed great flexibility all while adding functionality to the overall project in an iterative and incremental manner.

# 7. Specification of Data Structures

The database used in this application was developed using an SQL GUI (Graphical User Interface) called SQLite Manager. This application is available for free as a Mozilla Firefox browser plug-in and allows users to create a database using a GUI and then export it to the “assets” directory in an Eclipse project.

### 7.1 Introduction to System / User Specification

The Shopping List Application will allow users to create, edit (update), copy, browse or delete a **List**. Each list will have the following details stored on them: ***List ID***, ***List Name* /** ***Description***. Once a List is created Users will then add various items to these lists, the items will have the following details stored on them: ***Item ID***, ***Barcode, Item Name / Description***, ***Quantity*** and ***Quantity Type*** (e.g. box, carton), ***Photograph***. User have the option to add items to a list, delete items from a list, move or copy items from one list to another, edit items on a list and browse items on a list.

The photograph will be stored in a file and assigned a URI (unique resource identifier, in this case the path to where the image is stored on the device may suffice) which will then be used to point to the image when it needs to be retrieved.

### 7.2 Pre-normalisation tables

Extrapolating the information from the User Specification results in the following pre-normalised table.

**Lists**  
(**List ID**, List\_Name, **Item ID**, Barcode, Item\_Description, Quantity, Quantity Type, Photo)

### 7.3 Normalisation

#### 1 N.F.

Test Table to check for repeating data

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***List ID*** | ***List Name*** | ***Item ID*** | ***Barcode*** | ***Item Description*** | ***Quantity*** | ***Quantity Type*** | ***Photo*** |
| 5 | John’s Bday | 1 | 1234567891011 | Candles | 1 | Box(es) | sdcard/CAMERA/ 1287513967339 |
| 5 | John’s Bday | 2 | 1110987654321 | Balloons | 3 | Bag(s) |  |
| 5 | John’s Bday | 3 | 1111111111111 | Cake | 1 | Piece(s) |  |
| 1 | Weekly Shopping List | 50 | 5555555555555 | Washing up Liquid | 1 | Bottle(s) |  |
| 20 | Daily Shopping List | 10 | 7878787878787 | Bread | 2 | Loaf(loaves) |  |

**List\_Item**  
(**List\_ID,** **Item\_ID,** Barcode, Item Description, Quantity, Quantity Type, Photo)

**List  
(List\_ID,** List\_Name)

#### 2 N.F.

Test Tables to check for repeating data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **List\_ID** | **Item\_ID** | **Barcode** | **Item\_Description** | **Quantity** | **Quantity\_Type** | ***Photo*** |
| 5 | 1 | 1234567891011 | Candles | 1 | Box(es) | sdcard/CAMERA/ 1287513967339 |
| 5 | 2 | 1110987654321 | Balloons | 3 | Bag(s) |  |
| 5 | 3 | 1111111111111 | Cake | 1 | Piece(s) |  |
| 1 | 50 | 5555555555555 | Washing up Liquid | 1 | Bottle(s) |  |
| 20 | 10 | 7878787878787 | Bread | 2 | Loaf(loaves) |  |

|  |  |
| --- | --- |
| **List\_ID** | **List\_Name** |
| 5 | John’s Bday |
| 5 | John’s Bday |
| 5 | John’s Bday |
| 1 | Weekly Shopping List |
| 20 | Daily Shopping List |

**Item  
(Item\_ID,** Barcode, Item\_Description, Quantity, Quantity\_Type, Photo)

**List\_Item  
(List ID, Item\_ID)**

**List**  
(**List\_ID,** List\_Name)

#### 3. N.F.

Test Tables to check for repeating data

**Item**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Item ID*** | **Barcode** | **Item\_Description** | **Quantity** | **Quantity\_Type** | ***Photo*** |
| 1 | 1234567891011 | Candles | 1 | Box(es) | sdcard/CAMERA/ 1287513967339 |
| 2 | 1110987654321 | Balloons | 3 | Bag(s) |  |
| 3 | 1111111111111 | Cake | 1 | Piece(s) |  |
| 50 | 5555555555555 | Washing up Liquid | 1 | Bottle(s) |  |
| 10 | 7878787878787 | Bread | 2 | Loaf(loaves) |  |

**List\_Item**

|  |  |
| --- | --- |
| **List\_ID** | **Item\_ID** |
| 5 | 1 |
| 5 | 2 |
| 5 | 3 |
| 1 | 50 |
| 20 | 10 |

**List**

|  |  |
| --- | --- |
| **List\_ID** | **List\_Name** |
| 5 | John’s Bday |
| 1 | Weekly Shopping List |
| 20 | Daily Shopping List |

**Item  
(Item\_ID,** Barcode, Item\_Description, Quantity, Quantity\_Type, Photo)

**List\_Item  
(List ID, Item\_ID)**

**List**  
(**List\_ID,** List\_Name)

### 7.4 Table Structure

**Item**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Field Size Limit** | **Sample Data** |
| Item\_ID | Number (PK) (Not Null) | 3 | 500 |
| Barcode | Varchar | 13 | 7702018982356 |
| Item\_Description | Varchar (Not Null) | 45 | Gillette Body Spray |
| Quantity | Number (Default = 1) (Not Null) | 3 | 051 327300 |
| Quantity\_Type | Varchar (Default = ‘piece’) | 10 | Can |
| Photo | Varchar | 80 | sdcard/CAMERA/ 1287513967339 |

**List\_Item**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Field Size Limit** | **Sample Data** |
| List\_ID | Number (PK) (Not Null) (FK) | 2 | 10 |
| Item\_ID | Number (PK) (Not Null) (FK) | 3 | 500 |

**List**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Field Size Limit** | **Sample Data** |
| List\_ID | Number (PK) (Not Null) | 2 | 10 |
| List\_Name | Varchar (Not Null) (default Sysdate – date & time) | 40 | 04-Dec-2010 12:00:00 |

Once the structure of the database and tables has been exported to Eclipse, the Java code can then interact with this database using an Adapter.

### 7.5 DDL Statements

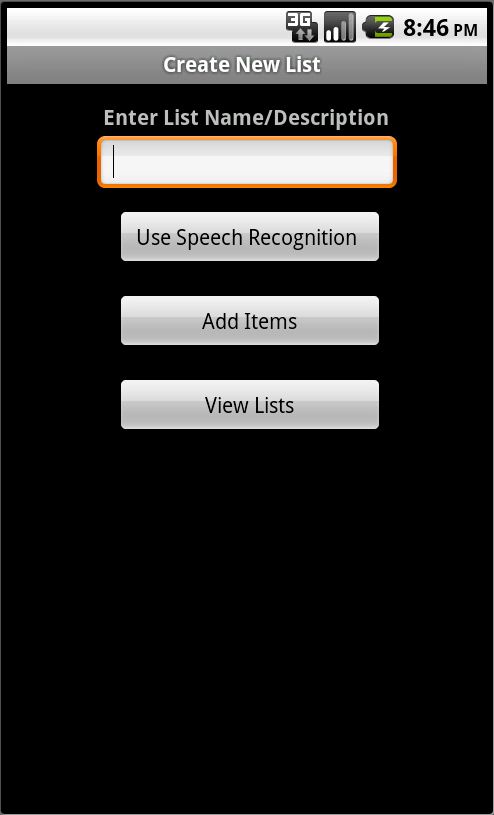
CREATE TABLE "item" ("item\_id" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL , "item\_barcode" VARCHAR DEFAULT no\_barcode, "item\_description" VARCHAR NOT NULL , "item\_quantity" INTEGER NOT NULL DEFAULT 1, "item\_quantity\_type" VARCHAR NOT NULL DEFAULT Piece, "item\_photo" VARCHAR DEFAULT no\_photo)

CREATE TABLE "list" ("list\_id" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL , "list\_name" VARCHAR NOT NULL DEFAULT CURRENT\_TIMESTAMP, "list\_description" VARCHAR)

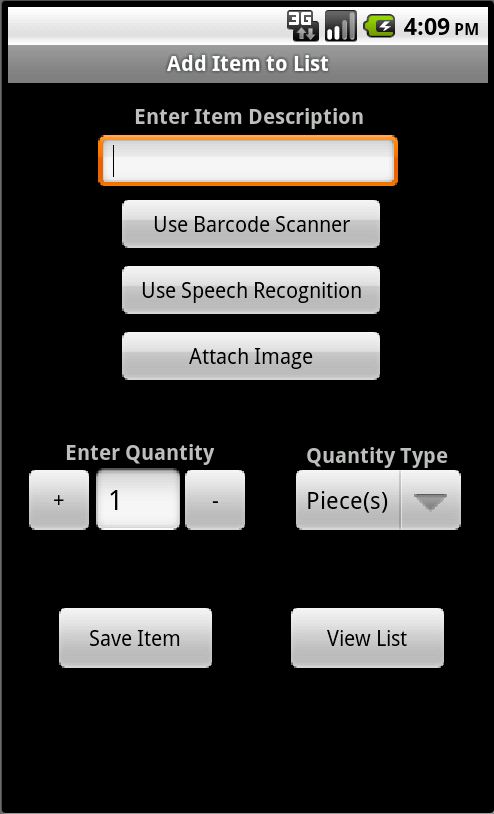
Create Table List\_Item (List\_ID Number(3) constraint List\_ID\_FK references List(List\_ID), Item\_ID Number(4) constraint Item\_ID\_FK references Item(Item\_ID), Primary KEY (List\_ID, Item\_ID))

# 8. User Interface

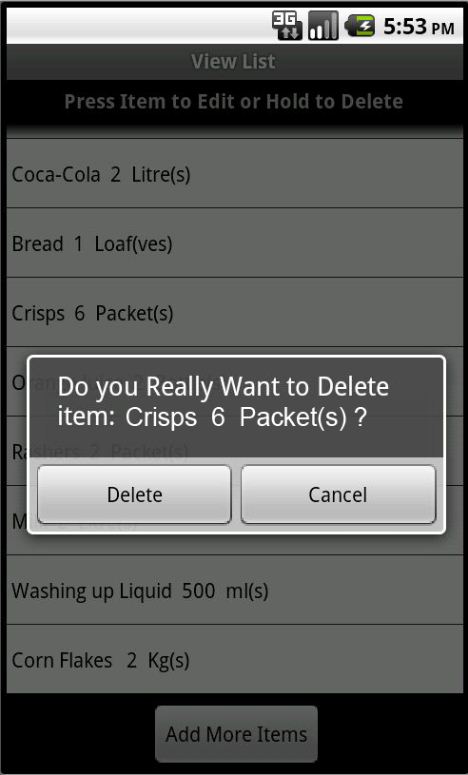
  
***Introduction Screen***

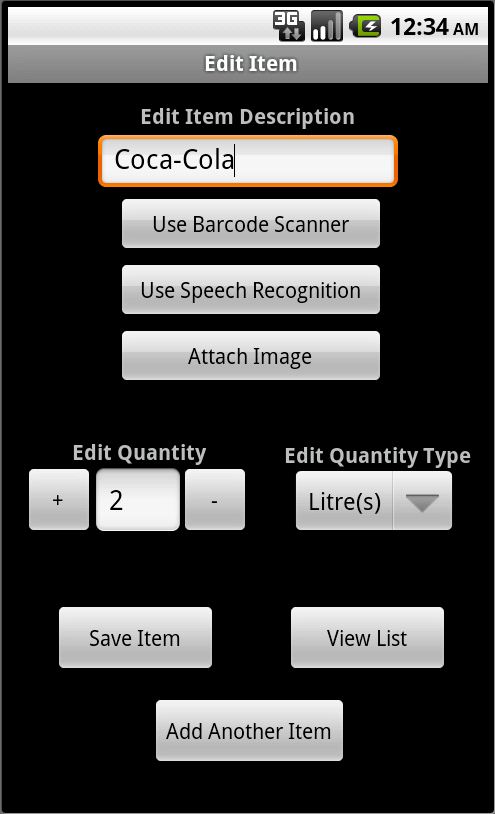
  
***Add List Screen***

***  
View Lists Screen***

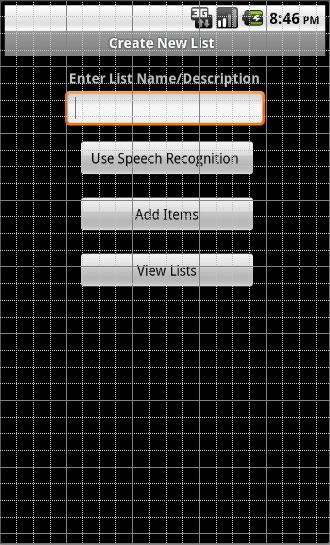
  
***Add Item Screen***

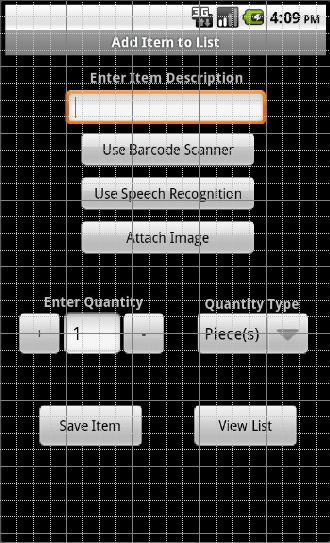
  
***List Items Screen***

***  
Delete Item Screen***

***  
Edit Item Screen***

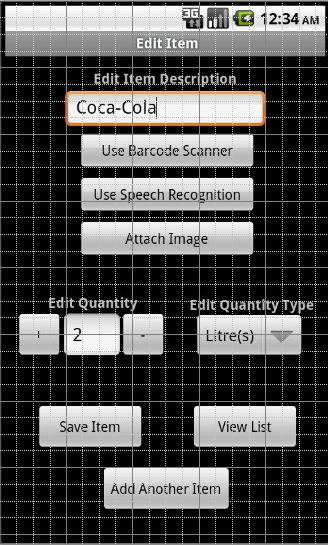
***  
Introduction Screen***

***  
Add List Screen***

  
***View Lists Screen***  
***Add Item Screen***

  
***List Item Screen***

  
 ***Delete Item Screen***

  
***Edit Item Screen***

# 9. Management of the Process

In order to manage the project more effectively a detailed schedule was drawn up showing details of the activities and deliverables expected. The start and finish date, and the duration of each of these activities were also detailed. Because of the problems already outlined in this report the schedule needed to be revised a number of times. The final revised schedule can be seen here. A Gantt chart detailing the schedule can be seen in Appendix B.

**Schedule Activities and Deliverables**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Start Date** | **Finish Date** | **Project Phase** | **Activity** | **Deliverable** | **Estimated Hours** |
| 9/1/11 | 1/4/11 | Research | Implementing various tutorials to better understanding the process of creating applications, the Android SDK, the Eclipse IDE and the underlying Android architecture | Working (Implemented & Tested) finished sample applications | 200 |
| 2/4/11 | 2/4/11 | Product Backlog 1 | Making a list of work items to be included in the Product Backlog | List of work items to be included in Project | 4 |
| 2/4/11 | 2/4/11 | Sprint Backlog 1 | Making a list of work items to be included in the first Sprint: e.g. the most basic core goals | List of work items to be included in the first Sprint | 2 |
| 3/4/11 | 11/4/11 | Sprint 1 | First Sprint iteration of the project implementing the most basic core goals | The applications navigation was completed. A series of screens were created with buttons linking each screen. The Database was also created and added to the project. | 60 |
| 12/4/11 | 12/4/11 | Review 1 | Review of the first Sprint iteration, examining the code, what has been delivered, what has to be added to the product backlog, etc... | Review and testing of the first functional shippable product | 8 |
| 13/4/11 | 13/4/11 | Product Backlog 2 | Review of work items to be included in the project | Updated list of items to be included in the project | 4 |
| 13/4/11 | 13/4/11 | Sprint Backlog 2 | Making a list of work items to be included in the second Sprint e.g. more advanced core goals | List of work items to be included in the second Sprint | 2 |
| 14/4/11 | 22/4/11 | Sprint 2 | Second Sprint iteration of the project implementing the more advanced core goals | A Database adapter was created in order for the Java code to be able to talk to the database. Setter and Getter methods for all fields in the tables were also created. The application design was refined. | 60 |
| 23/4/11 | 23/4/11 | Review 2 | Review of the second Sprint iteration, examining the code, what has been delivered, what has to be added to the product backlog, etc..  . | Review and testing of the second functional shippable product | 8 |
| 24/4/11 | 24/4/11 | Product Backlog 3 | Review of work items to be included in the project | Updated list of items to be included in the project | 3 |
| 24/4/11 | 24/4/11 | Sprint Backlog 3 | Making a list of work items to be included in the third Sprint e.g. the final core goals or else the most basic secondary goals | List of work items to be included in the third Sprint | 3 |
| 25/4/11 | 3/5/11 | Sprint 3 | Third Sprint iteration of the project implementing either the final core goals or else the most basic secondary goals | The basic Create, Edit, Update and Delete functions of the application were implemented. | 60 |
| 4/5/11 | 4/5/11 | Review 3 | Review of the third Sprint iteration, examining the code, what has been delivered, what has to be added to the product backlog, etc... | Review and testing of the third functional shippable product | 8 |
| 5/5/11 | 5/5/11 | Product Backlog 4 | Review of work items to be included in the project | Updated list of items to be included in the project | 2 |
| 5/5/11 | 5/5/11 | Sprint Backlog 4 | Making a list of work items to be included in the fourth Sprint e.g. the most basic secondary goals or the more advanced secondary goals | List of work items to be included in the fourth Sprint | 2 |
| 6/5/11 | 14/5/11 | Sprint 4 | Fourth Sprint iteration of the project implementing either the final the most basic secondary goals or the more advanced secondary goals | The functionality regarding the Barcode Scanner, Speech Recognition and Camera were added to the application. | 90 |
| 15/5/11 | 16/5/11 | Review 4 | Review of the fourth Sprint iteration, examining the code, what has been delivered, what has to be added to the product backlog, etc... | Review and testing of the fourth functional shippable product | 12 |

**Total Hours: 528**

# 10. Project Evaluation

10.1 Problems: While this project has come to a successful conclusion, there have been many problems and stumbling blocks along the way. The main reoccurring problem was underestimating the complexity of the project. As already outlined in this report it was difficult to start the development of this project when the research and tutorial stage had finished. Although there were a wealth of tutorials available on the internet, most only covered a small area of functionality and offered no guidance in creating a full project. As the Android SDK used a modified version of the Java language this also contributed to the problem as there were certain ideas and concepts specific to Android that I was unaware of and struggled to understand.

10.2 Immediate project outcome: The project’s immediate outcome is that a functional, shippable, potentially viable Android application has been produced. The application could potentially be released onto the Android Market as it is, however it may also benefit from some refinement before release. If the application is released as it is, it is possible to update the application by including refinements and new features. This can easily be done by uploading the updated application to the Android Market, and the next time users who have the application use Wi-Fi or mobile internet they will receive and update from Android Market informing them that an update to the application is available.

10.3 Long-Term project outcome: As previously stated this application may potentially be ready for release as it is, but may also benefit from some refinement. In this case there are a number of options available in order to do so. Firstly the UI may benefit from being cleaned up a small bit; there a quite a lot of screen and although navigation is generally intuitive, the application could benefit from a reduction in screens. This would mean however that more functionality may have to be included in existing screens in order to ensure the same functionality which may lead to a less rewarding user experience.

Secondly, this application could potentially be made available for all Android handsets running all Android platforms. This goal may be actually be achievable without too many resources needing to be invested. This is because a lot of the features and code used in this application are backwards and frontwards compatible with other Android platforms.

Thirdly, there is the potential to add more technical aspects to the project. One of the original secondary goals of the project was to enable users to share a list with others who had the application installed using Wi-Fi or mobile internet, however this feature was discarded as it was neither the time nor the technical ability were present to enable this goal to be feasible.

Lastly, there exists the potential to collaborate with a company in the grocery sector in order to use this application as the basis for a supermarket application. This company may be a supermarket chain or some other group involved in the grocery sector. There already exists a number of applications (mostly in the U.S. but Tesco U.K also have an application) similar to this application which interact with a particular supermarket chain’s database of products (Gallagher, 2010), allowing users to see all the items a particular supermarket stocks and the price of each item. This was also originally a secondary goal of this project but was discarded due to time and technical constraints as well as the unattainability of details of a supermarket’s database of products without their co-operation.

## 10.4 Purpose of the Project

The aim of this project was to develop a Shopping List Application for the HTC Desire Smart Phone running the Android 2.2 operating system. It is an application that contains a number of screens, and it will allow users to create, delete, browse and edit a number of shopping lists. Users can name these shopping lists, and add, delete, browse or edit up items to each list.

The Splash screen will greet the users when they launch the application. On this screen users can create a new list or view the lists that are currently stored on this device.

The Create New List Screen allows users to create a new list by typing a name into the text box or using the Speech Recognition system. Users can then either add items to their list by going to the Add Item Screen, or see all the lists currently created by going to the View List screen.

The View List screen contains details of all the lists currently saved in the application. Users can either delete a list (and all the items contained in this list) by holding their finger on the list. Users can press the list name to move to the Add Item screen and add items to that particular list.

The Add Item screen is where users can items to their lists. In the Add Item screen users can add an item to a list by using the keyboard, barcode scanner or the speech recognition system. Users also have the option to add a picture of a specific item to their list. Users can also specify the quantity and the quantity type (e.g. box, carton, etc...) of an item. Users can then save an item to their list and also view all the items on that list by going to the View Item page.

The View Item screen is where users can see the items on a specific list. Here users can edit any details of an item by pressing the item and are then brought to the Edit Item screen. Users can delete an item by holding down the Item Name. This screen is the one the user will have open when they are completing their shopping as it allows users to check items off their list.

The Edit Item screen is where users can edit a particular item on their list. It allows users to change any detail of the item; description, quantity, quantity type, photograph etc. Once users have edited their item to their satisfaction they can Save the item, add more items by going to the Add Item screen or view their list by going to the View Items screen.

# 11. Testing

As the development of the application is still on-going at the time of writing this, final testing has still to be completed. The SCRUM software development methodology allows for testing at the end of every sprint iteration, in which black-box regression testing was conducted.

**Black-box testing :** This is testing which is conducted with no knowledge of the internal workings of the code. It can be thought of as testing similar to that conducted by the end user of the product. Black-box testing as such will focus on making sure that the application works as it is expected to. Black-box Testing could include testing of basic functionality as well as trying to enter unexpected inputs into an input field in order to see how the application reacts.

**White Box testing:** White box (or the more appropriately titled Clear Box) testing is the antithesis of Black Box testing; it is testing that is performed with a knowledge of the inner workings of the code. It is testing as would be performed by a Software Developer. White Box testing can be performed by using J Unit testing which is available in Eclipse.

**Regression Testing:** This is testing that is conducted on every new build (sprint iteration) to ensure that the new changes to the software have not had any unintended side-effects on the existing code. Black-box regression testing was conducted at the end of every sprint iteration.

**Smoke Testing:** Is a wide and shallow approach to testing when all the basic functionality is tested without delving too deep into any one area. Smoke testing was conducted as various stages during the development of this process, particularly after adding an area of significant functionality in order to ensure the application was functioning correctly before continuing.

**Sanity Testing :** This is a narrow and deep approach to testing where an area of functionality is . The testing of the “Add Item” screen followed this testing methodology as this screen contained a lot of interrelated functionality that needed to be tested in order to verify that all elements were working as expected.

**Testing on different devices / Emulators:** Eclipse allows for the emulation of different handsets running differing Android OS platform versions. Once the application has been finished a number of the most popular handsets will be emulated running different Android OS platforms in order to see if the application behaves as expected. It may also be possible to test the application on different physical handsets owned by colleagues in order to see how the application behaves in a real world environment.

# 12. User Manual

Most Smart Phone applications (for Android devices anyway) do not come with a user manuals. As such applications should be designed to the highest UI (User Interface) and HCI (Human Computer Interaction) standards. This however, is not always the case. Many applications have a series of controls (buttons, checkboxes, etc...) and it is simply up to the user to figure out what these controls do, usually by trial and error. As most Smart Phone applications are limited in scope, covering only one particular area of functionality or using a specific technology, this discovery by trial and error is an accepted, if far from ideal situation for users. Games are perhaps the only areas in which user manuals are regularly include with applications. Scenarios for new and existing users are described below:

**New User:**  
Michael has just downloaded the Smart Shopping List Application for the first time. He opens up his applications and launches the Smart Shopping List app where he is greeted by the Introduction screen. Michael clicks the “Create New List” button and is brought the list creation screen. Here Michael clicks the “Use Speech Recognition” button, when the dialog appears he says “Weekly Shopping” and the words “Weekly Shopping” appear in the textbox. Michael then clicks the “Add Items” button and is brought to the Add Item screen. On the Add item screen Michael types “Bread” into the textbox, chooses 2 as the quantity and selects “Loaf(ves)” from the quantity type, he then clicks the “Save Item” button. Michael sees an empty bottle of Coca-Cola near him which reminds him he needs to get more, so he picks up the bottle and presses the “Use Barcode Scanner” button. Michael does not have a Barcode Scanner installed on his Smart Phone so upon being prompted he chooses to download one. Once downloaded the Barcode Scanner launches and Michael scans the barcode on the Coca-Cola bottle. Once the barcode has been read, the Barcode Scanner exits and the Add Item screen is brought back with the details of the Coca-Cola bottle populating the fields. Michael then clicks the “Save Item” button. Michael continues adding items until his list has been completed. Once it has, Michael clicks the “View List” button and is brought to the View Items screen where he can see all the items on his list.

**Existing User:**Kate has been using the Smart Shopping List Application for a number of weeks. Two weeks ago she organised a party for her best friend Tom and created a list to help her. Next week is her sister’s birthday and again Kate has to organise the party. Kate launches the Smart Shopping List app and is greeted with the Introduction screen. She wants to modify the list she used for Tom’s birthday so she clicks the “View Lists” button. She is then brought to the View Lists screen. She sees the birthday list amongst the other lists and presses it in order to select it. She is then brought to the Add Items screen. Kate cannot remember exactly what items were on the list since last time so she clicks the “View List” button and is brought to the View Items screen where she can see which items are on that list. She remembers that she got Tom a football birthday cake last time, and is sure her sister would not like that kind of cake for her birthday so she holds her finger on that item for a couple of seconds and a dialog box asks her if she would like to delete that it. She chooses “Yes”. Looking further down the list of items Kate notices that for the last party she bought 24 packets of crisps. She knows her sister prefers chocolate to crisps, so she clicks the item and this brings her to the Edit Item screen. When this screen appears, the item’s description is already set to “Crisps”, the quantity is set to “24” and the quantity type is set to “packets”. Kate knows her sister doesn’t like crisps, but she knows some others at the party might so she changes the quantity to “12”. Kate then clicks the “Save Item” to save the modified item to the list. Kate then clicks the “View List” item and is brought back to see all the items on her updated list.

# 13. Conclusion

In conclusion, this project has been a very valuable and worthwhile experience. There is a great sense of satisfaction in seeing through a full project from the initial concept, through the Analysis and Design stage right up to the implementation and testing of the project. It was also very interesting to be involved in such a cutting-edge project using modern Smart Phones and technologies such as Speech Recognition and Barcode Scanning.

During this project I have increased my knowledge in the following ways:

* Thoroughly familiarising myself with the Eclipse IDE
* Increased my understanding of XML (Screen Layouts were created using XML)
* Familiarised myself with SQLite database engine
* Increased my Java OOP (Object Oriented Programming) skills
* Familiarised myself with the Android SDK

In conclusion, although this hasn’t been an easy project by any means, I am extremely happy with the end result. The finished application is not dissimilar to existing applications that have thousands of downloads from the Android Market and I intend to launch it either as it is or with a small amount of further refinement in order to gauge reaction from users.

# 14. References

Gallagher, J n.d., 'Mobile Apps Do Better Than Expected', *Supermarket News*, 58, 37, pp. 49-50, WilsonWeb: Wilson OmniFile Full Text Mega Edition, EBSCO*host*, viewed 14 March 2011.

HTC Corporation. (2010). *HTC – Products – HTC Desire – Overview.* Available at: http://www.htc.com/www/product/desire/specification.html. (Accessed: 4 December 2010).

Mack, E. (2011). *Android Market Poised to Overtake App Store in August*. Available at: http://www.pcworld.com/article/227253/android\_market\_poised\_to\_overtake\_app\_store\_in\_august.html. (Accessed: 5th May 2011).

SQLite. (2010). *SQLite Home Page.* Available at: http://www.sqlite.org/. (Accessed: 4 December 2010).

Wikipedia. (2010a). *Android Operating System.* Available at: http://en.wikipedia.org/wiki/Android\_(operating\_system)#Software\_development\_kit. (Accessed: 4 December 2010).

Wikipedia. (2010b). *Eclipse (Software).* Available at: http://en.wikipedia.org/wiki/Eclipse\_(software). (Accessed: 4 December 2010).

Wikipedia. (2010c). *SQLite.* Available at: http://en.wikipedia.org/wiki/SQLite. (Accessed: 4 December 2010).

# 15. Appendices

## Appendix A

**Error Handling**

When dealing with R cannot be resolved or is not a field change the import statement to the following:  
**import de.vogella.android.intent.explicit.R;**

Instead of   
**import android.R;**

Android.R should never be imported

match\_parent has been deprecated by fill\_Parent in API Versions 8 and above.

new DialogInterface.OnClickListener() (This should be .OnclickListener and not .onClickListener)

If a “Layout .XML” file is run instead of a Java file the following error will occur:

[2011-04-06 13:15:03 - SmartShoppingListApp] Error in an XML file: aborting build.

[2011-04-06 13:15:05 - SmartShoppingListApp] res\layout\add\_item.xml:0: error: Resource entry add\_item is already defined.

[2011-04-06 13:15:05 - SmartShoppingListApp] res\layout\add\_item.out.xml:0: Originally defined here.

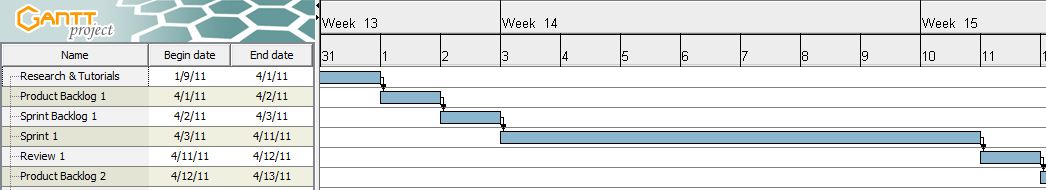
[2011-04-06 13:15:05 - SmartShoppingListApp] C:\Users\John\workspace\SmartShoppingListApp\res\layout\add\_item.out.xml:1: error: Error parsing XML: no element found

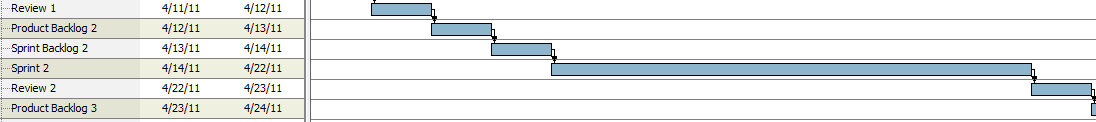
In this case you must delete the “name.out.xml” and clear the console. If errors are still present in the package explorer the go to “project” - > “clean” and select the appropriate project. If this cause the “R.java” file in the “Gen” Directory to be deleted, try changing the file by commenting out a line then un-commenting it which should

## Appendix B

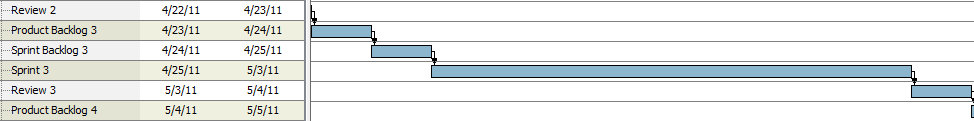
***Overall Project Schedule***

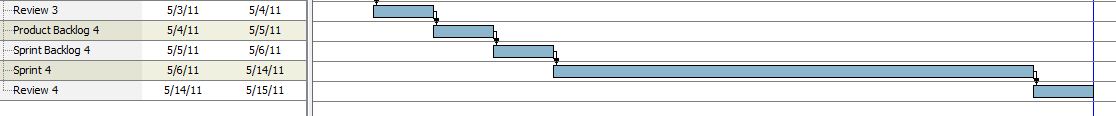
***Research & Tutorial Stage***

  
***First Sprint Iteration***



***Second Sprint Iteration***

  
***Sprint Iteration 3***



***Sprint Iteration 4***