**Smart Shopping List application for Android OS - Report 2**

Multimedia Year 4

# 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.*

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

Contents

[Declaration of Authenticity 2](#_Toc279422727)

[1. Abstract / Description of the title 5](#_Toc279422728)

[2. Goals 6](#_Toc279422729)

[2.1 What is the purpose of this title? 6](#_Toc279422730)

[Core Goals 6](#_Toc279422731)

[Secondary Goals 6](#_Toc279422732)

[Personal Goals 6](#_Toc279422733)

[Technical Goals 6](#_Toc279422734)

[Performance Goals 6](#_Toc279422735)

[2.2 What is the scope of the title? 8](#_Toc279422736)

[3. User Experience 9](#_Toc279422737)

[3.2 Scenarios – typical user 16](#_Toc279422738)

[3.2.1 Scenario 1 16](#_Toc279422739)

[3.2.2 Scenario 2 17](#_Toc279422740)

[4. Methodology 18](#_Toc279422741)

[4.1 The Problem from a process Viewpoint 18](#_Toc279422742)

[4.2 Assessment of Candidate Methodologies 18](#_Toc279422743)

[4.3 Process Outline 26](#_Toc279422744)

[5. Feasibility Study 27](#_Toc279422745)

[5.1 Risk Identification and Management 27](#_Toc279422746)

[6. Tools and Technologies 32](#_Toc279422747)

[6.1 Hardware 32](#_Toc279422748)

[6.2 Software 33](#_Toc279422749)

[7. Title Content 34](#_Toc279422750)

[7.1 Content Inventory 34](#_Toc279422751)

[7.2 Content Grouping, Labelling and Structure Listing 35](#_Toc279422752)

[7.3 Initial Architectural Blueprints 37](#_Toc279422753)

[7.4 Storyboard 38](#_Toc279422754)

[7.5 Database 42](#_Toc279422755)

[7.5.1 Introduction to System / User Specification 42](#_Toc279422756)

[7.5.2 Pre-normalisation tables 42](#_Toc279422757)

[7.5.3 Normalisation 43](#_Toc279422758)

[7.5.4 Table Structure 45](#_Toc279422759)

[7.5.5 Sample Implementation (using Oracle) 46](#_Toc279422760)

[*7.5.5.1* *DDL (Data Definition Language) Statements* 46](#_Toc279422761)

[*7.5.5.2* *DML (Data Manipulation Language) Insert Statements* 48](#_Toc279422762)

[*7.5.5.3* *Database Testing* 48](#_Toc279422763)

[8. Visual Design 49](#_Toc279422764)

[8.1 Layout Grids 49](#_Toc279422765)

[8.2 Design Sketches 52](#_Toc279422766)

[8.3 Mock-ups 56](#_Toc279422767)

[9. Schedule 60](#_Toc279422768)

[10. Security 65](#_Toc279422769)

[References 67](#_Toc279422770)

# 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.

Once the user’s shopping list is finished, it is hoped to offer the functionality where the user can choose to finalise their list. This will result in their Android mobile device connecting to another site (either an existing price comparison site or a new site specific to the Smart Shopping List App). This connection will be made either via Wi-Fi or else a physical connection to a PC or Laptop. A comparison will be made (based on the items in their shopping list that week) as to which of the four main supermarkets is the cheapest for their list of goods. There may also be an option to connect automatically to a Supermarket’s online shopping and order the goods on their online shopping portal. It is also hoped to implement a feature that will allow people who both have the application installed to share shopping lists. For example if there was a situation where one member of a household who normally does the grocery shopping is unable to for some reason, they can send the list to another person so that they can complete the shopping using the 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. In addition, the functionality that allows users to send their finished shopping list to be compared with the main supermarkets to see which is the cheapest for their chosen list this week is unique among existing applications. The same is true of the functionality that allows users to share their list with another person who also has the application installed.

2. Goals

2.1 What is the purpose of this title?  
The overall goal of this project is to create an application that will assist households with their weekly grocery shopping in a number of useful ways and not to be just another novelty application. There are however a number of different types of goals for this project. Core goals are goals that will have to be met in order to successfully implement this project. Secondary goals are goals that are not crucial to the successful completion of this project, but which are desirable and should (and will, subject to time constraints) be implemented in order to add to the functionality of this project. Technical goals are related to designing and creating a secure application, as well as extending the availability of the application by getting it to operate successfully on different devices or platforms. Personal goals are set by the developer in order to ascertain what they have learned over the course of this project. Finally, performance goals relate to how the finished application will perform in comparison to similar applications.

Core Goals  
The core goals of this project are to create a Smart Shopping List Application in which users can create, edit, delete and manage a number of shopping lists.

Secondary Goals  
The secondary goals of this project relate to added functionality around core goal. They are desirable, but not crucial to the successful completion of this project. The secondary goals of this project are:

* Functionality that allows users to share a list with another user who has the application installed.
* Functionality that allows users to check the price of a given list in the four major supermarkets.

Personal Goals   
There are also a number of personal goals related to this project. To successfully undertake the design, implementation and testing of a full project from beginning to end, complete with full documentation is a personal goal.

## Technical Goals

Making the application work on a number of different android operated devices would also be technical goal. Also designing and implementing a secure application is a technical goal.

## Performance Goals

These are goals that relate to how well the application will perform, and depend on a number of factors. In order meet these goals, the application must out-perform, “on balance” the traditional method of keeping a shopping list (a pen and paper). The phrase “on balance” refers to a number of factors:

* Is it easier to use than what’s currently available?
* Does it give a better result?
* Is it more reliable?

While this application will not be easier to use than what’s currently available; as the time taken to take out the mobile device, unlock it, launch the application and record the selection will be longer and more inconvenient than just writing down your selection on a piece of paper. Therefore, this application must out-perform the traditional method of making a shopping list, which is through reliability and giving better results.

To achieve greater reliability, there already exists a characteristic of Mobile Device usage that will help in this goal; that is that users tend to carry their mobile device on or near to their person at all times. This increases the reliability of this application because the user will normally have their mobile device on them at all times which may not be true of a pen and paper (even if the user had a pen and paper, it may not be the same paper that has their shopping list on it). Another aspect this application that will help to increase the general reliability is that it allows users to take and store a photograph of a particular item, thus avoiding confusion about certain products that may be part of a range of similar products or come in a variety of sizes.

In attempting to achieve the other goal of giving a better end result to users, this can be achieved in a number of ways. Firstly, as already mentioned there will be the option of using the inbuilt camera on the mobile device to take a picture of a particular product. Secondly, the planned functionality that will allow users to share lists should be a very convenient feature under certain scenarios. Thirdly, the planned functionality that allows users to compare the price of their particular list could be very useful and has the potential to save the users money each week.

## 2.2 What is the scope of the title?

The Smart Shopping List application will be developed for the HTC Desire Smart Phone running the Android operating system. It is an application that contains a number of screens, and it will provisionally allow users to create, delete, browse and edit up to 99 shopping lists. Users can name these shopping lists, and add, delete, browse or edit up to 999 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 using the keyboard, create a new list using the speech recognition system or view the lists that are currently stored on this device.

The List screen contains details of all the lists currently saved in the application. Users can view the existing lists, edit them or delete them. Users can also choose to go to the main menu, create a new list or share an existing list with another person who has the application installed.

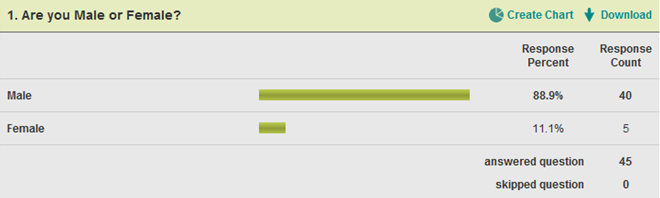
The Add Item screen is where users can items to their lists. Users are brought here when they create a new list or when they click to edit an existing list from the List Item Screen. In the Add Item screen users can add an item to a list by using the keyboard, barcode scanner or the speech recognition 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.

The List Item screen is where users can see the items on a specific list. Users can get to this screen by choosing to browse or edit a list on the List screen. Here users can edit any details of an item, delete an item or move or copy an item from one list to another. 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 simple by touching a particular item on a list.  
The Checkout screen is where users can check the price of a list in the four major supermarkets. Users can get to this screen by pressing the Check Price button on the List Item screen. Users can also go back and edit their list or click on any of the four major supermarket buttons to edit their lists.

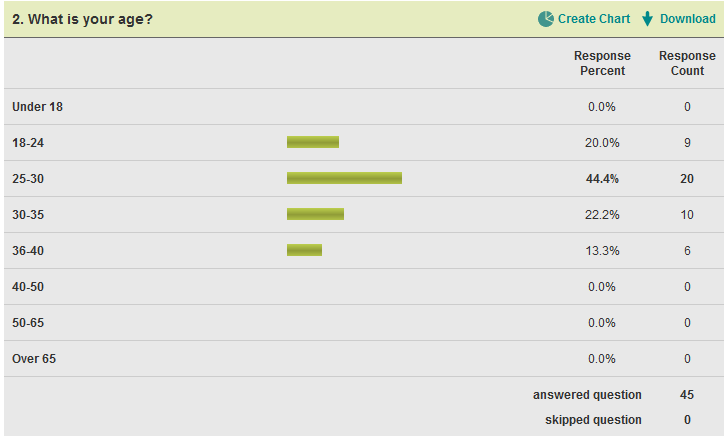
3. User Experience  
3.1 Audience Definition

In order to define the target audience rather than rely on assumptions of what the average user might be, the decision was taken to conduct some market research in the form of an online survey. The survey was created using SurveyMonkey (SurveyMonkey, 2010), and a link to the survey was posted onto various Smart Phone related online forums, such as Boards.ie (Boards, 2010) and Vodafone.ie (Vodafone, 2010). The only requirement for this survey was that the user owned a Smart Phone device. Over the space of five weeks between late October and early December the survey was completed by 45 people and the results are shown below.

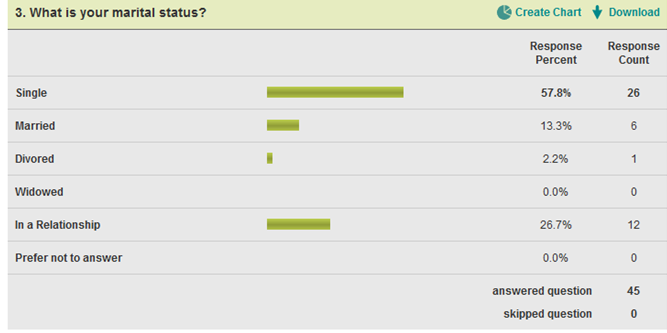
**Question 1.**

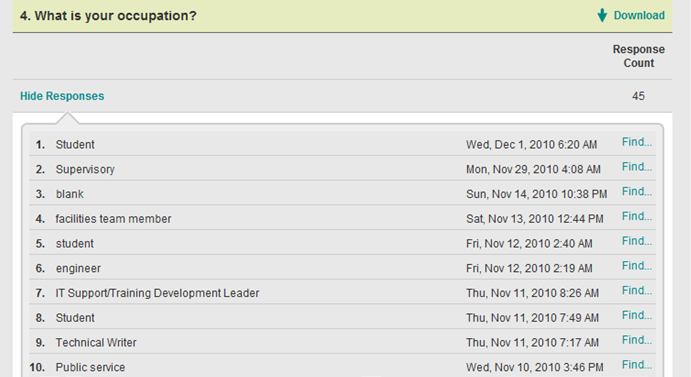


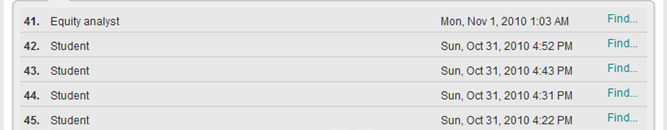
**Question 2.**

****

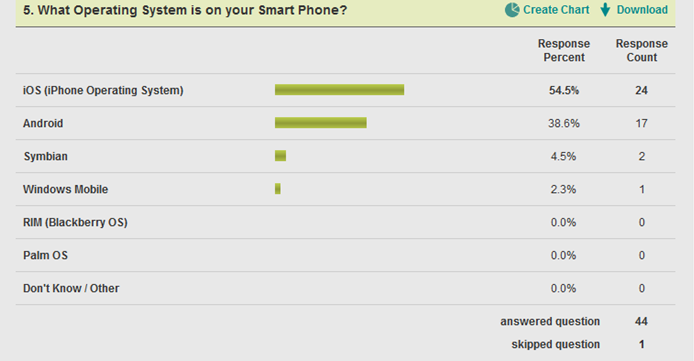
**Question 3.**

****

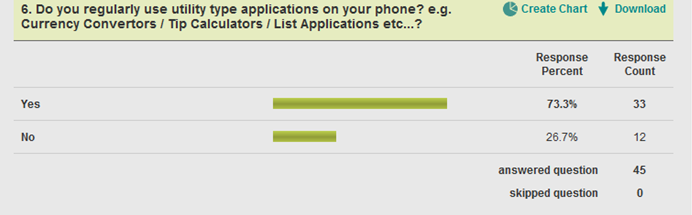
**Question 4. **

** **

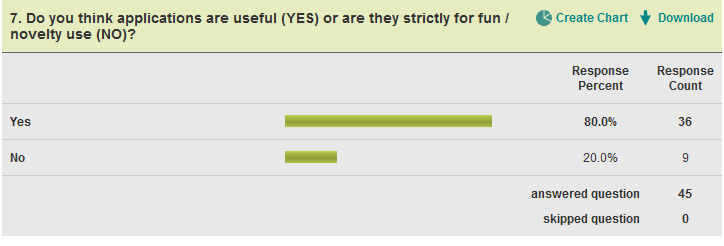
**Question 5.**

****

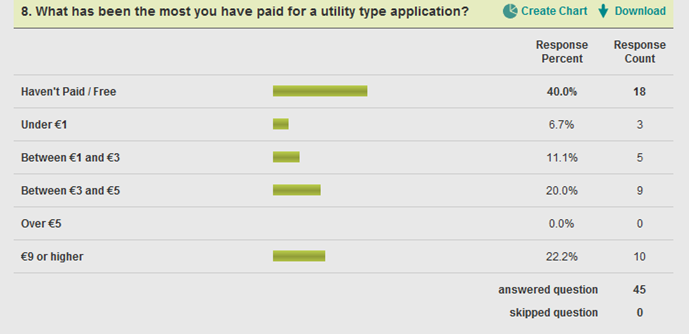
**Question 6.**

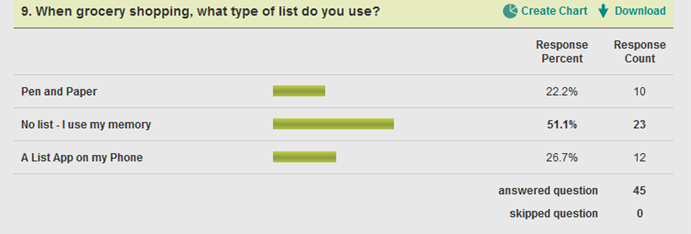
****

**Question 7.**

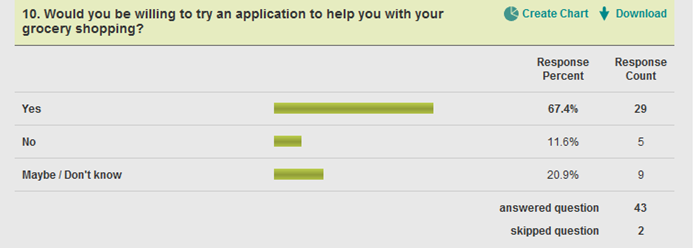
****

**Question 8.**

****

**Question 9.  
**

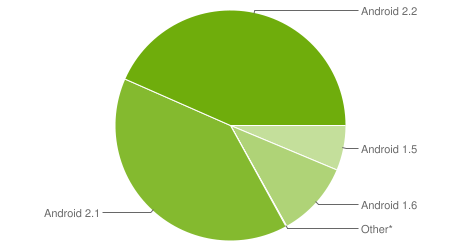
**Question 10.**

****

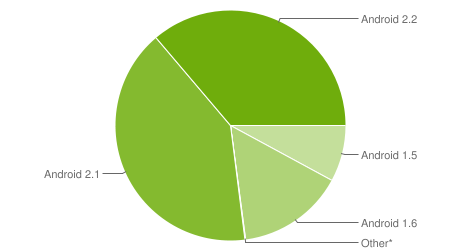
Looking at the results of this survey is it possible to come to the conclusion that the average user “is male, between the ages of 18-40, single, a student or working in I.T. owns an iPhone, regularly uses free Apps, finds them useful, doesn’t use a list while shopping but would be prepared to try a shopping list app for their phone.”  
It is important however not to rely totally on this survey, as there may be anomalies in the data gathered, for example, perhaps online forums are used predominantly by men between the ages of 18 and 40, who are single and have iPhones. Therefore the results of this survey should only be used as a guide to an “average user”.

In relation to the software currently in use, Google’s own Android Developers website (Google, 2010) gives a picture of which version of the Android platform, users accessing the Android Marketplace over the last 2 weeks have been using.

|  |  |  |
| --- | --- | --- |
| **Platform** | **API Level** | **Distribution** |
| Android 1.5 | 3 | 6.3% |
| Android 1.6 | 4 | 10.6% |
| Android 2.1 | 7 | 39.6% |
| Android 2.2 | 8 | 43.4% |

*Data collected during two weeks ending on December 1, 2010*\* *Other: 0.1% of devices running obsolete versions*

|  |  |  |
| --- | --- | --- |
| **Platform** | **API Level** | **Distribution** |
| Android 1.5 | 3 | 7.9% |
| Android 1.6 | 4 | 15% |
| Android 2.1 | 7 | 40.8% |
| Android 2.2 | 8 | 36.2% |



*Data collected during two weeks ending on November 1, 2010  
\* Other: 0.1% of devices running obsolete versions*

If a comparison is made between the 2 weeks ending the 1st of November and the 2 weeks ending the 1st of December, it shows that in that time, the Android 2.2 platform has overtaken the Android 2.1 platform as the most popular among users accessing the Android Marketplace. This is unsurprising as the Android 2.2 platform was realised during the summer and was rolled out in stages depending on when the mobile network operators finished testing the platform on their system. This coupled with the fact that all new handsets over the last few months would have come with the Android 2.2 platform pre-installed explains the reason it is only in the last month that Android 2.2 has overtaken Android 2.1. Also, as the Android operating system only gained popularity in the last year, (now almost on a par with the market leader: “iOS” - iPhone operating system) it means that the percentage of users using the Android 2.2 platform will continue to increase as time goes on.

With this in mind it may be tempting to develop the Smart Shopping List application for the Android 2.2 platform. However bearing in mind that Android platforms are guaranteed to be forward compatible but not guaranteed to be backward compatible, by designing this application for the Android 2.1 platform it is possible to target approximately 80% (on current figures) of the available market. Therefore unless these is some specific functionality included in the Android 2.2 platform that isn’t available in the Android 2.1 platform,(and based on research so far this seems not to be the case), then it would be wise to develop this application for the Android 2.1 platform.

## 3.2 Scenarios – typical user

Scenarios are used to specify typical interactions between the average user and the system. The template used below is taken from an online article by Karl Kapp (n.d.).

### 3.2.1 Scenario 1

**Scenario Name:** Make List, add a number of items, then delete some items later.  
**Scenario Level:** Typical **Instructional Objective:** To show the user how to create a list using the keyboard, add items to the list using the keyboard and delete items from the list. **Storyboards & Instructions:** The user wants to make a shopping list. They take out their Smart Phone, unlock the device then launch the Smart Shopping List App. When they get to the splash screen they touch the button to create a new list using the keyboard. When the keyboard pops-up they enter the name of the list as “Weekly Shopping” and press enter. They are then brought to a screen where they can add items to their list. The users touches the input item with keyboard button, they type in “Kelloggs Cornflakes” and press enter. They only want one box so they skip the quantity input box (which has a default of one). They move onto the unit input and they change the quantity type from “piece” to “kg” as they want a 1kg box of Kelloggs Cornflakes. They then touch the “Save Item” button, this adds the item to their list. The user presses the “Add More” button which clears the “Add Item” screen and resets the quantity and unit input boxes to their original defaults. The user wishes to add another item, so they touch the input item with keyboard button again. The keyboard pops-up and they type “Fairy washing up liquid”, then press enter. They move to the quantity input box and press on the “1” default value. This brings up a text input box. They enter “500”, press enter, and then change the value in the “unit” input box to “ml”. Then the user presses the “Save Item” button which saves this item to the list. The user presses the “Add More” button which then resets the input screen to its original state and allows them to enter another item. The user repeats this process for a number of items. Once the list is complete the user exits the application by pressing the back button on their Smart Phone device. Sometime later the user discovers that there was a new bottle of washing up liquid in their cupboard. The user remembers that they had added washing up liquid to their shopping list earlier so they decide to remove it from their list. The user takes out their Smart Phone, unlocks the device and launches the Smart Shopping List App. At the splash screen they press the “View Lists” button and are brought to the “Lists” screen. In this screen the user scrolls down to their list, then presses the button that says “Weekly Shopping”. The user is brought to the “List Item Screen”. The user scrolls down to button that says “Fairy washing up liquid” and they press this button. A list of options appear on the screen: “done”, “edit”, “move”, “copy”, “delete”. The user presses “delete”, and prompt appears on screen asking the user if they are sure they want to delete this item. The user chooses “Yes” and the item is deleted from the list. The user then exits the application by pressing the back button on their Smart Phone device.

### 3.2.2 Scenario 2

**Scenario Name:** Make List using Barcode Scanner   
**Scenario Level:** Simple **Instructional Objective:** To show the user how to create a shopping list and items to this list by using the barcode scanner. **Storyboards & Instructions:** The user wants to make a shopping list using the barcode scanner. The user has kept the empty packaging from a number of daily household items in order to create this shopping list. They take out their Smart Phone, unlock the device then launch the Smart Shopping List App. When they get to the splash screen they touch the button to create a new list using the keyboard. When the keyboard pops-up they enter the name of the list as “Daily Shopping” and press enter. They are then brought to a screen where they can add items to their list. The user touches the input item using barcode button. A prompt comes up asking the user to pick a barcode scanning application from the list installed. The user chooses one, and their camera becomes active and they can see the view from the camera on their screen. The user holds the barcode from an empty box of “Kelloggs Rice Krispies” in the viewfinder of their camera until the software recognises the barcode. When the barcode is recognised, the item is added to the list including the correct quantity and unit type e.g. “Kelloggs Rice Krispies 500g”. The user touches the “Save Item” button to save this item to the list, then the user touches the “Add More” button to begin the process of adding another item to the list. The user repeats this process for all the items that they have barcodes for. The user then exits the application by pressing the back button on their Smart Phone device.

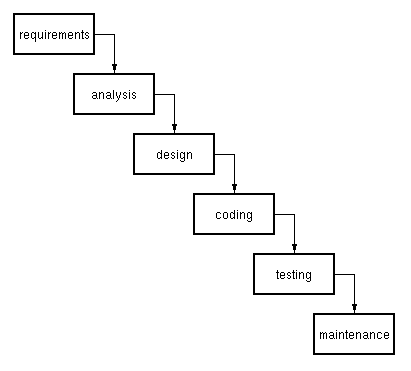
# 4. Methodology

## 4.1 The Problem from a process Viewpoint

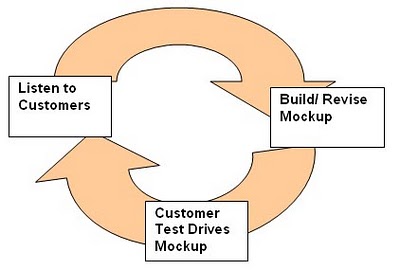
There are potentially a number of problems that may arise with this project. Firstly, this project has relatively new technology at its core. Although, this technology is established, it is fast moving and it will be a challenge to develop a project using this technology, having little or no experience of it. Secondly, there are also a number of different types of goals with regard to this project. The Core goals can be implemented without too much trouble, the secondary goals for this project will require a lot more research however and may not even be feasible for this project. The secondary goals will still be investigated however and if possible implemented.

## 4.2 Assessment of Candidate Methodologies

In assessing the candidate methodologies, it is obviously a good idea to look at all the software methodologies that are out there. A number of the main Software Development Methodologies are identified and discussed below. Their suitability with regard to this project is also assessed.  
  
**Waterfall:** This is a sequential development approach. It usually consists of 5/6 phases: Requirements, Analysis, Design, Implementation, Testing, (possibly) integration, and maintenance. After each phases there should be some quality of the assessment done in the previous phase, (Van Vliet, 1996) in the form of validation and verification, and if the previous phase is deemed to have not been successful it is essential to go back and implement that step again. While the Waterfall methodology was highly popular in the past, it has gained a certain amount of notoriety in the intervening years and is now thought of as a somewhat outdated methodology. This is because of its rigidness, it doesn’t allow for an easy change of requirements. However requirements change is generally accepted as a factor in modern software development.

  
*The Waterfall Model*

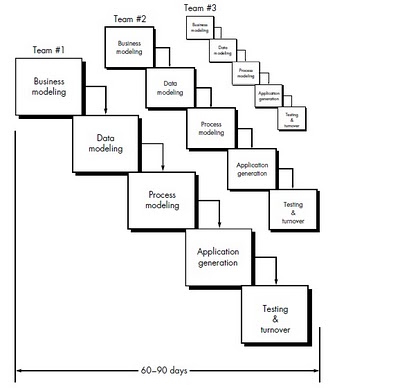
As there are a number of secondary goals related to this project that may be implemented given time, this type of requirements change makes the Waterfall approach an unsuitable method for this project.  
  
  
**Prototyping:** Prototyping is an approach best used when the requirements elicited from a customer are not detailed, or when the developer is unsure about certain core aspects of their project such as a crucial algorithm or the suitability of a CASE tool. The prototyping paradigm (shown below) begins with “listening to customers”, otherwise known as requirements elicitation: where the customer and developer discuss the overall objectives of the project, what the requirements that are currently known are, and the areas where further development is needed (Pressman, 2000). A “quick-design” is then produced which focuses on the aspects of the application that are visible to the user such as GUI’s (graphical user interfaces). From this “quick-design” a working prototype is then developed. This prototype is then evaluated by the customer and their feedback is used to refine the requirements for the software to be developed. This process is then iterated so that the customer can have their requirements met more closely and this enables the developer to get a better understanding of the development that has to be undertaken. One important aspect of the prototyping methodology is that often once the prototype has served its purpose it is discarded.

  
*The prototyping paradigm*

Prototyping would be unsuitable for this project as it requires developing prototypes in an iterative process in order to refine the customer’s requirements. This process of creating prototypes, garnering feedback from the customer, discarding the prototype then creating a new prototype based on feedback from the customer would be unsuitable given the limited timeframe of this project. Also the possibly introduction of secondary goals if there is time once the core goals have been met would not be suited to this methodology.

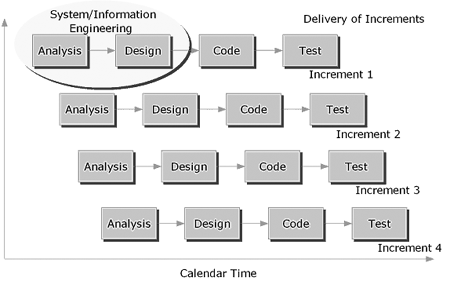
**Rapid Application Development:** Pressman (2000) describes RAD as an incremental software development process which emphasises an extremely short development cycle. It is a “high-speed” version of the Linear Sequential Model in which rapid development is achieved using component-based construction. RAD consists of the following phases:

* Business Modelling
* Data Modelling
* Process Modelling
* Application Generation
* Testing and Turnover



*The RAD Model*

The amount of modelling required in the early stages of RAD make it an unsuitable methodology for this project. There will not be sufficient time to conduct this modelling and develop a working application. Also once the core goals are met, if the secondary goals are to be implemented it will mean some requirements change, and RAD is not the best suited methodology to this type of requirements change.  
  
  
**Incremental:** This methodology combines the linear sequential model with the iterative approach of prototyping (Pressman, 2000). As shown in the diagram below linear sequences are applied in a staggered fashion as calendar time progresses. Each of these linear sequences produces a deliverable increment of the software. For example the first increment can be used to deliver the core functionality. This first increment is either used or reviewed and from this a plan for the next increment is developed. In the plan the core functionality is modified (if needed) and additional functionality is added. This process is repeated until the complete product is produced. The Incremental approach is similar to Prototyping in that it is iterative, it differs however in that each iteration delivers an operational product.

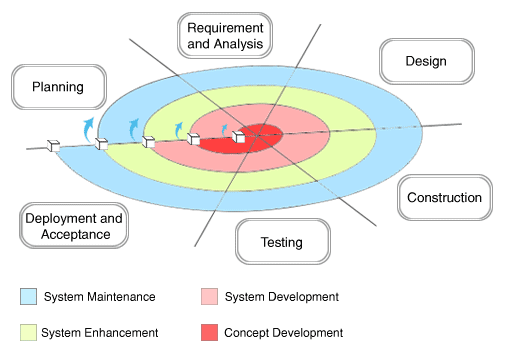
*The Incremental Model*

The incremental model would be a suitable model for this project. It would allow a number of increments which could incorporate additional functionality. The incremental approach is very similar to the SCRUM Agile methodology, however there are subtle differences and the iterative and incremental nature of the SCRUM Agile methodology would be best suited to his project.

**Spiral:** This methodology originally proposed by Boehm (1988) combines the iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model (Pressman, 2000). It provides the potential for rapid development of incremental versions of software. Early iterations are extremely basic prototypes, maybe even paper prototypes. Later iterations add increasing complexity. As shown in the diagram below the methodology is divided into six task regions:

***Customer Communication:*** These are tasks that are required to establish effective communication between the developer and the customer.  
***Planning:*** These are tasks required to define resources, timelines and other project related information.  
***Risk Analysis***: Tasks required to assess both technical and Management risks.  
***Engineering:*** Tasks required to build one or more representations of the application.  
***Construction & Release:*** Tasks required to construct, test, install and provide support (e.g. documentation and training).  
***Consumer Evaluation:*** Tasks required to obtain customer feedback based on evaluation of the software representations that were created during the engineering stage and implemented during the installation stage.

Each of these six regions is populated by a set of work tasks called a task set. As the evolutionary process begins, development begins at the centre and move in a clockwise direction. The first circuit of the spiral may, for example be used to develop a product specification. The next circuit may be used to develop a prototype, and so on. After each circuit, adjustments are made to the project plan. Cost and schedule are also adjusted based on feedback derived from the customer. The Spiral methodology remains operative until the software is retired, although it may Spiral may become dormant, but whenever a change is initiated the process starts again from the appropriate point.

  
*The Spiral Model (Boehm, 1988)*  
  
The Spiral methodology would be a suitable methodology for this project. However, much like the Incremental methodology, the additional requirements to be added in order to implement secondary goals (once the core goals have been met,) means that this methodology is not the most suitable for this project.

**Agile:** Agile Development is an umbrella term for several iterative and incremental software development methodologies (VersionOne, Inc, 2010). Agile development fundamentally incorporates iteration and the continuous feedback that it provides to successively refine and deliver a software system. They all involve continuous planning, continuous testing, continuous integration, and other forms of continuous evolution of both the project and the software. Agile development is lightweight and inherently adaptable

The 12 principles of the Agile manifesto are as follows (Wikipedia, 2010a):

* Customer satisfaction by rapid delivery of useful software
* Welcome changing requirements, even late in development.
* Working software is delivered frequently (weeks rather than months)
* Working software is the principal measure of progress
* Sustainable development, able to maintain a constant pace
* Close, daily cooperation between businesspeople and developers
* Face-to-face conversation is the best form of communication (co-location)
* Projects are built around motivated individuals, who should be trusted
* Continuous attention to technical excellence and good design
* Simplicity
* Self-organizing teams
* Regular adaptation to changing circumstances
* work is confined to a regular, repeatable work cycle, known as a sprint or iteration

**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 seems to be the methodology that is best suited to this project. As there is a number of core and secondary goals to this project, the core goals can be identified and then implemented over three, 4 week sprint iterations. If some of these goals are not met in the first two they will be added to the backlog to be included in the next iteration. This rapid iteration process will allow the functionality to be developed over a number of iterations. This iterative and incremental approach will allow the secondary goals to be added to the backlog (if there is sufficient time) and they will be implemented in the next available iteration. As the addition of these secondary goals may affect the implementation already carried out, SCRUM’s iterative approach will allow for changes to this completed work in order to implement the secondary goals, whereas a methodology such as Incremental would not. Because each iteration delivers a potential shippable product increment, once an iteration has been completed, and all the required functionality is included then the product can be released.

## 4.3 Process Outline

**Stage 1: Analysis and Design:**This project began with the some research into a number of potential topics, before one topic was decided on and a project proposal was due. This proposal gave a brief outline as to what the goal of the project was as well as the disciplines and potential hardware and software to be used. Following on from the project proposal was Project Report 1. This report contained detailed information about the goals of the project, the user experience, feasibility study, risk management, candidate methodologies for the project, initial design work and architectural blueprints as well as schedule. Shortly after report 1, an abstract was required. This abstract was similar to the original project proposal except while the project proposal was subject to change, the abstract had to contain finalised details about the project topic, disciplines and hardware and software used. Report 2 was due halfway through the project schedule. This report carried on from report 1: with some topics built upon and other topics that were absent from report 1 now included.

**Stage 2: Implementation and Testing:**  
As the SCRUM methodology is based on the Agile methodology, the preliminary stages including: analysis, design and documentation are not that important. This is because Agile methodology focuses on rapid delivery and expected requirements change, so the analysis and design stages are not as important as in other methodologies. The SCRUM methodology starts with a Product Backlog. Here the all the work items to be included in the project are listed. From the Product Backlog a number of work items are moved to the Sprint Backlog. As the SCRUM methodology is an iterative and incremental methodology, it requires a number of incremental iterations to create a complete project. Therefore, only the work items that are expected to be implemented in the first Sprint are included in the Sprint Backlog. In the case of this project the Sprint Backlog for the first sprint will include basic functionality relating to the core goals. Once the sprint has been completed a review is held and a fully functional complete product is required, no matter how basic it may be. If certain functionality was not implemented in this Sprint it must go back into the Sprint Backlog in order to be included in the next Sprint. The Sprint iterations continue incrementally adding to the product until after a certain number of iterations (four in the case of this project) results in a finished product.

# 5. Feasibility Study

Although limited progress with regard to implementation has been made so far with this project, all indications point to the intended software working together. There are a number of reasons for this. Firstly, any experience in using the Android SDK and the Eclipse IDE such as completing the suggested tutorials, have not resulted in any compatibility issues. Secondly there are a huge number of applications that have been created using these tools, and documented cases of compatibility issues are minimal at best.

There are also many support outlets, from the official Google Android Developers site and the official Google Groups Forum, to many other third party sites and unauthorised forums giving tips and advice. Also because the Android SDK is based on a subset of the Java language, and by already possessing a moderate amount of familiarity with Java, this will hopefully mean that there will be less compatibility issues and general beginner errors.

## 5.1 Risk Identification and Management

There are also issues with regard to Risk Management that need to be addressed. Risk Management is a four phased procedure:

* Identify the risks
* Calculate the probability of the risks occurring based on all available evidence
* Specify what impact the realisation of a threat would have on the project
* Outline a strategy on how to minimise the risk happening or the effect of the risk if it were realised.

|  |  |
| --- | --- |
| ***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 plugins 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). |

# 6. Tools and Technologies

## 6.1 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 – 1Ghz  
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
* Geotagging

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

## 6.2 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) Plugin, 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, 2010b). 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, 2010c).

**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, 2010d). SQLite is the database of choice for mobile applications and as such it will be the database used in this project.

**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.

# 7. Title Content

## 7.1 Content Inventory

Application Icon  
Splash Screen Image  
View List Button  
Create New List - Keyboard Button  
Create New List –Speech Button  
List Screen Image  
List Button1  
List Button2  
List Button3  
List Button4 etc...  
Main Menu Button  
Share List Button  
New List – Speech Button  
New List – Keyboard Button  
List Item Screen Image  
Item1 Button  
Item2 Button  
Item3 Button  
Item4 Button etc...  
Add More Button  
Check Price Button  
Add Item Screen Image  
Add Item – Keyboard Button  
Add Item – Speech Button  
Add Item – Barcode Button  
Attach Photo? Button  
Quantity Number Button  
Quantity Plus Button  
Quantity Minus Button  
Quantity Type Button  
Save Item Button  
Add More Button  
Checkout Screen Image  
Dunnes Stores Button  
SuperQuinn Button  
SuperValu Button  
Tesco Ireland Button  
Edit List Button

## 7.2 Content Grouping, Labelling and Structure Listing

**Content Grouping:**

|  |
| --- |
| **Application Icon** |

|  |
| --- |
| **Splash Screen** |
| Splash Screen Image |
| View List Button |
| Create New List - Keyboard Button |
| Create New List –Speech Button |

|  |
| --- |
| **List Screen** |
| List Screen Image |
| List Button1 |
| List Button2 |
| List Button3 |
| List Button4 |
| Main Menu Button |
| Share List Button |
| New List – Speech Button |
| New List – Keyboard Button |

|  |
| --- |
| **List Screen** |
| List Item Screen Image |
| Item1 Button |
| Item2 Button |
| Item3 Button |
| Item4 Button |
| Main Menu Button |
| Add More Button |
| Check Price Button |

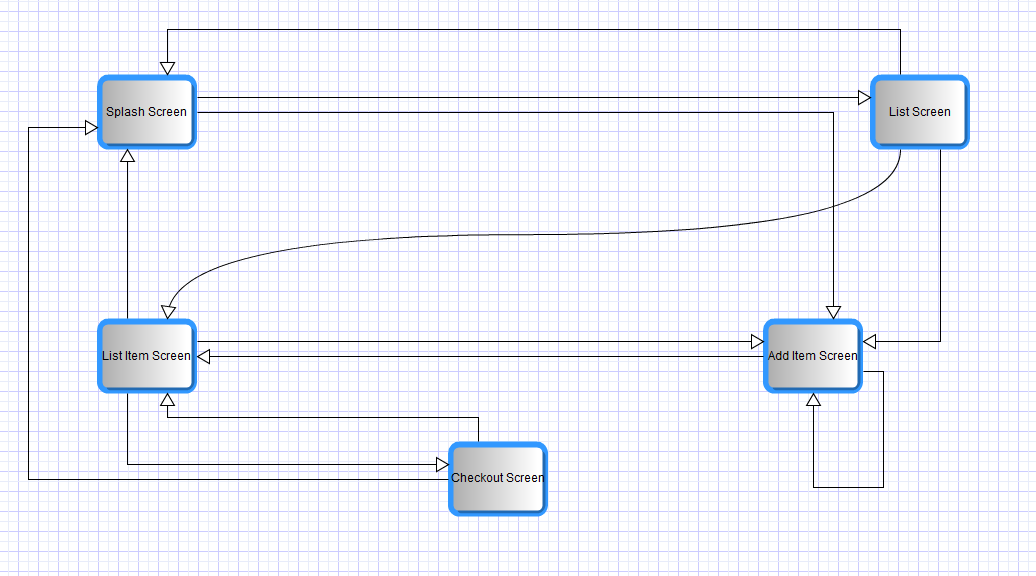
|  |
| --- |
| **Add Item Screen** |
| Add Item Screen Image |
| Add Item – Keyboard Button |
| Add Item – Speech Button |
| Add Item – Barcode Button |
| Attach Photo? Button |
| Quantity Number Button |
| Quantity Plus Button |
| Quantity Minus Button |
| Quantity Type Button |
| Save Item Button |
| Add More Button |
| View List Button |

|  |
| --- |
| **Checkout Screen** |
| Checkout Screen Image |
| Dunnes Stores Button |
| SuperQuinn Button |
| SuperValu Button |
| Tesco Ireland Button |
| Edit List Button |
| Main Menu Button |

**Naming Convention:**

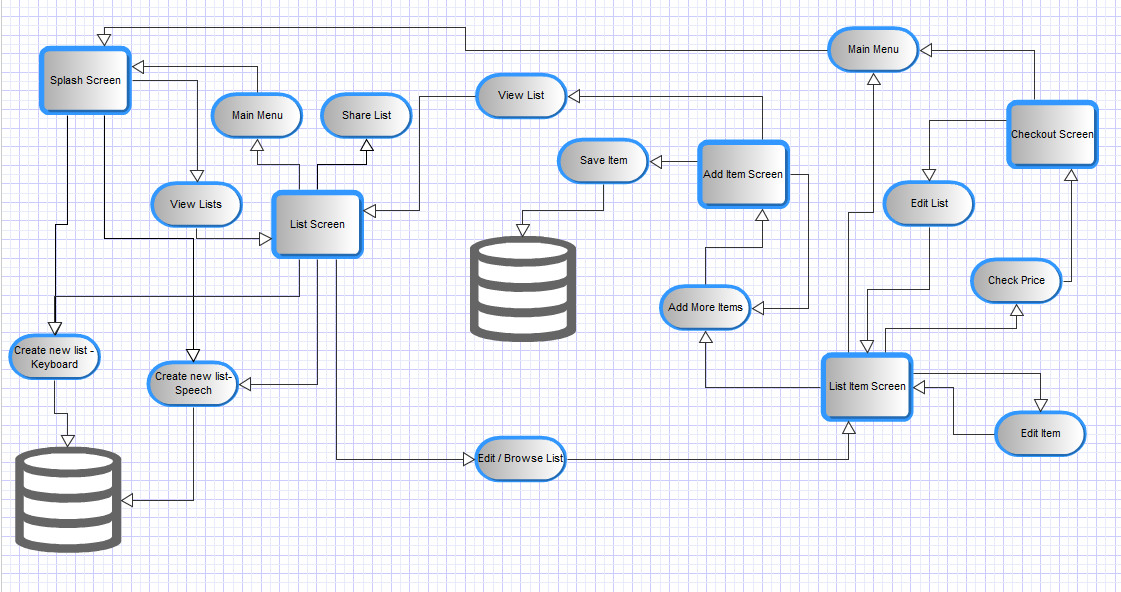
|  |  |
| --- | --- |
| **Image Name** | **File Name- Naming Convention** |
| Application Icon | icon.jpg |
| Splash Screen Image | splash.jpg |
| View List Button | view\_list.jpg |
| Create New List - Keyboard Button | createnew\_keyboard.jpg |
| Create New List –Speech Button | createnew\_speech.jpg |
| List Screen Image | list.jpg |
| List Button1 | list\_button1.jpg |
| List Button2 | list\_button2.jpg |
| List Button3 | list\_button3.jpg |
| List Button4 | list\_button4.jpg |
| Main Menu Button | main.jpg |
| Share List Button | share.jpg |
| New List – Speech Button | newlist\_speech.jpg |
| New List – Keyboard Button | newlist\_keyboard.jpg |
| List Item Screen Image | list\_item.jpg |
| Item1 Button | item1\_button.jpg |
| Item2 Button | Item2\_button.jpg |
| Item3 Button | Item3\_button.jpg |
| Item4 Button | Item4\_button.jpg |
| Add More Button | add.jpg |
| Check Price Button | check.jpg |
| Add Item Screen Image | additem.jpg |
| Add Item – Keyboard Button | additem\_keyboard.jpg |
| Add Item – Speech Button | additem\_speech.jpg |
| Add Item – Barcode Button | additem\_barcode.jpg |
| Attach Photo? Button | photo.jpg |
| Quantity Number Button | qty\_number.jpg |
| Quantity Plus Button | qty\_plus.jpg |
| Quantity Minus Button | qty\_minus.jpg |
| Quantity Type Button | qty\_type.jpg |
| Save Item Button | save.jpg |
| Add More Button | addmore.jpg |
| Checkout Screen Image | checkout.jpg |
| Dunnes Stores Button | dunnes.jog |
| SuperQuinn Button | superquinn.jpg |
| SuperValu Button | supervalu.jpg |
| Tesco Ireland Button | tesco.jpg |
| Edit List Button | edit.jpg |

## 7.3 Initial Architectural Blueprints



*Simplified Navigation Diagram*

\* Note: Users may exit the application at any time by pressing the “home” or “back” key on the device.



*UML Diagram of the system*

## 7.4 Storyboard

**Screen name:** *Splash Screen*  
This is the screen that users will be greeted with when they launch the application on their mobile device. Users will be presented with three options which are explained below.  
**Screen Option 1:** *Create a new list with Keyboard*Here users are given the option to create a new list using the Mobile Device’s inbuilt keyboard. Users simply have to touch the button that says “Create New List” and the Mobile Device’s inbuilt Keyboard will pop up along with a text input box. Once a user has created their list they will be brought to the add item screen.  
**Screen Option 2:** *Create a new list using Speech Recognition System*Here users are given the option to create a new list using the Android Speech Recognition function. Users simply have to touch the “microphone” button which will bring up a prompt which tells users to “Speak now”. When they do, the Android Speech Recognition System’s interpretation of what they said will be entered in the text input box. If it is wrong or incomplete (e.g. if the user said “Shopping List” and the Android Speech Recognition System interpreted it as “Chopping List”), it will be simple enough to edit or delete and enter the correct selection using the keyboard. Once a user has created their list they will be brought to the add item screen.  
**Screen Option 3:** *View current lists*Here users will be brought to the “List” screen in order to see which lists are currently stored on their device simply by touching the button that says “View Lists”.

**Screen Name:** *List Screen*Users will be brought to this screen by touching the “View Lists” button on the “Splash” screen. It contains a list of all the lists entered by the user as well as an overview of what was contained on these lists. If there are a lot of lists, users can scroll through them using the touchscreen controls.  
**Screen Option 1 :** *View List*Here users can view all existing lists. The list names are displayed on a button in large font, while the items on their list are displayed in a smaller font on the same button, so that users can get an overview of what items are on their list. Touching a list sends users to the “List View” screen where they can view and edit individual items or add more items. Holding a list from 1 second brings up a menu where users can choose to edit, view, copy, share or delete a list.  
**Screen Option 2 :** *Edit List*Users can edit a list by holding one of the list buttons for 1 second, which will bring up a menu where users can edit a list, view it, share it, copy it or delete it.**Screen Option 3:** *Delete List*Users can delete a list by holding one of the list buttons for 1 second, which will bring up a menu that will allow users to edit the name of the list, let the user view, edit or copy the list by sending them to the “List Item” screen, share it with someone or else delete it.  
**Screen Option 4 :** *Copy List*Users can delete a list by holding one of the list buttons for 1 second, which will bring up a menu where users can copy a list (amongst other things). The new list will have the same name as the old list except “(copy)” will be appended to it.  
**Screen Option 5:** *Go to Main Menu*Touching the button labelled “Main Menu” will bring users back to the “Splash” screen.  
**Screen Option 6:** *Create New List*Users have the option to create a new list here. Users can do this by either touching the button saying “New List” which will bring up the inbuilt keyboard and a text input box, or alternatively they can touch the “Microphone” icon which will bring up the Android Speech Recognition prompt and a related text box.  
**Screen Option 7 :** *Share List*   
When users touch this button they are given the option to share one of their lists with someone else who also has the application installed. If users are in close proximity they will have the option of sharing the list by Bluetooth, if not they can share the list over Wi-Fi or mobile internet. Alternatively user can share a list by holding the list name for 1 second, which will bring up a menu where users can choose to edit, view, delete or share a list.

**Screen Name:** *Add Item*This screen will allow users to add an item to their list. Users will be brought to this screen by creating a list using the keyboard, creating a list using the speech recognition system or by touching the “Add More” button on the “List Item Screen”.  
**Screen Option 1:** *Input item with keyboard*Touching the “Input Item” button will bring up the inbuilt keyboard and an input text box where users can type in the name of their item.  
**Screen Option 2:** *Input item by scanning barcode*Touching the Barcode icon will allow users to add an item to their list by scanning the barcode from a particular product if they happen to have it handy, (the barcode scanning function may be implemented using the Barcode API or may use an existing third party barcode application).**Screen Option 3:** *Input a picture*Touching the Camera icon will allow users to add a picture to their item description or else instead of a description.  
**Screen Option 4:** *Input item using voice recognition software*Touching the Microphone icon will bring up the Android Speech Recognition prompt and a related text box. When the user speaks, the speech recognition’s interpretation of what the user said will appear in the text box. This makes it easy to delete or edit, if the speech recognition system misinterprets what the user said.**Screen Option 5:** *Change item quantity using keyboard*Touching the numerical Quantity box will bring up the inbuilt keyboard and a related text input box where users can enter a new number for the quantity using the keyboard. The default for this value will be “1”.  
**Screen Option 6:** *Change item quantity using “****+****” or “-“ keys*Alternatively users can increment or decrement the number in the numerical Quantity box, by touching either the “**+**” or “**-**“ buttons. The minimum quantity cannot go below 1 and the maximum quantity cannot go above 999.  
**Screen Option 7:** *Change item quantity measurement e.g. pound / loaf/ piece etc...*Touching this button will expand a drop down list of popular measurements that will allow users to better define the quantity of their selected item. It will include popular measurements such as pieces, grams, bottles, boxes, cans, kilograms etc... The default for this value will be piece.  
**Screen Option 8:** *Save Item to a list*Once users have entered whatever information they deem relevant they can add their chosen item to a list by touching the “Save Item” button. Users will then be presented with a list of all available lists, so they can choose which list to add this item to.  
**Screen Option 9:** *Add another item*Once users have added an item to a list, they will remain on the “Add Item” screen and the fields they entered for the last item will still be present. Touching the “Add New Item” button will clear all fields (expect the quantity and quantity type fields which will return to their original defaults i.e.”1” and “Piece” respectively). Users can then enter the details of a new item.  
**Screen Option 10:** *View List*Touching this button will bring users to the “List Item” Screen.

**Screen Name :** *List Item Screen*This screen contains the contents of an individual list. This is the screen users will have open while doing their shopping. This screen allows users to mark off items from their list, delete items from their list, edit the details of their lists such as quantity, or move an item in their list to another list. Users can also choose to add additional items or else to navigate to the “Splash” screen or the “Checkout” screen.  
**Screen Option 1:** *View an individual list*By default, going to the “List Item” screen will present users with all the items on a particular list. If the list is long, users can scroll using the touchscreen controls.  
**Screen Option 2:** *Edit the name, quantity or quantity type of an existing item*Users can edit an item or its details by either holding the particular item button for 1 second which will bring users to a menu where they can choose to mark the item as “done”, “edit” it, “delete” it , “move” it or “copy” it. Alternatively users can touch the notepad icon to the left of the item, which will bring up the same “done”, “edit”, “delete”, “move” or “copy” options.  
**Screen Option 3:** *Delete an item from a list*An item can be deleted from the list by using the same two methods outlined above (holding the particular item for 1 second or touching the notepad icon), which will bring up the menu containing the “done”, “edit”, “delete” or “move” options. A prompt will ask the user if they are sure they want to delete the item.   
**Screen Option 4:** *Mark an item as already picked up*Again users can mark an item as done or picked up by using the two methods outlined above. Users can also mark an item as done by simply touching it, rather than holding their finger on it for 1 second. When an item is marked as “done” or picked up it will still appear on the list however it will have a line through it. To unmark an item, users can simply press the “done” button again in the menu or else simply tapping the item button rather than holding it for 1 second, which will remove the line through the particular item.  
**Screen Option 7:** *Move an item from one list to another*Using the two methods previous outlined in the last few options users can move an item to another list. When they select the “move” option from the menu a list of the other shopping list will be displayed, users can simply scroll to the list they want to move the item to and touch it. A prompt will tell the user the move has been made.  
**Screen Option 6:** *Copy an item from one list to another*Using the two methods outlined in the last few options (touching or holding an item for 1 second) users can copy an item from one list to another. The process is exactly the same as moving an item except the item is kept in the original list and a copy is moved to the new list. A prompt will tell the user the item has been copied. **Screen Option 7:** *Go to Main Menu*Touching the “Main Menu” button will bring users to the “Splash” screen.**Screen Option 8:** *Add another Item*Touching the “Add More” button will bring users to the “Add Item” screen**Screen Option 9:** *Check Price*Touching the “Check Price” button will bring users to the “Checkout” screen.  
  
  
**Screen Name :** *Checkout* **Screen Option 1:** *Dunnes Stores*Touching this button will bring users to a screen which lists their items and the price of each one as charged by Dunnes Stores. **Screen Option 2:** *Superquinn*Touching this button will bring users to a screen which lists their items and the price of each one as charged by Superquinn. **Screen Option 3:** *Supervalu*Touching this button will bring users to a screen which lists their items and the price of each one as charged by Supervalu **Screen Option 4:** Tesco  
Touching this button will bring users to a screen which lists their items and the price of each one as charged by Tesco.  
**Screen Option 5:** *Go to Main Menu*Touching the “Main Menu” button will bring users to the “Splash” screen. **Screen Option 6:** *Edit List*Touching the “Edit List” button will bring users to the “List Item” screen, from where they can easily edit their list.

## 7.5 Database

As previously specified, the database used in this project will be SQLite (SQLite, 2010).

### 7.5.1 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*** and ***Description***. The Application will limit users to 99 lists at any one time due to the fact that with this being a mobile application, storage and processing are at a premium. 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). Users will also have the option to attach an optional photograph to a particular item. 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.

This optional storage of an image raises an interesting point in relation to databases. Specifically how images should be stored in a Multimedia database. The image could be stored as BLOB (Binary Large Object) however binary data has the potential to become corrupted and is not very scalable (the amount of data to be stored can increase very rapidly) which makes it particularly unsuited to mobile applications where processing and storage are at a premium. A better solution would be to store the photograph in a file and assign it a URI (unique resource identifier, in this case the path to where the image is stored on the device may suffice) which could then be used to point to the image when it needs to be retrieved.

### 7.5.2 Pre-normalisation tables

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

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

### 7.5.3 Normalisation

#### 1 N.F.

Test Table to check for repeating data

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

#### 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** | **List\_Description** |
| 5 | Birthday | John’s Bday |
| 5 | Birthday | John’s Bday |
| 5 | Birthday | John’s Bday |
| 1 | Weekly | Weekly Shopping List |
| 20 | Daily | Daily Shopping List |

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

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

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

#### 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** | **List\_Description** |
| 5 | Birthday | John’s Bday |
| 1 | Weekly | Weekly Shopping List |
| 20 | Daily | Daily Shopping List |

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

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

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

### 7.5.4 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 |
| List\_Description | Varchar | 50 | Items to get on way home from work |

### 7.5.5 Sample Implementation (using Oracle)

### 7.5.5.1 DDL (Data Definition Language) Statements

Create Table ITEM  
(  
 Item\_ID Number(3) constraint Item\_ID\_PK Primary Key,  
 Barcode Varchar(13),  
 Item\_Description Varchar(45),  
 Quantity Number(3) ,  
 Quantity\_Type Varchar(10),  
 Photo Varchar(80)  
)

CREATE SEQUENCE item\_id\_seq  
MINVALUE 1  
MAXVALUE 999  
START WITH 1  
INCREMENT BY 1  
CYCLE  
CACHE 20;

Alter Table ITEM  
MODIFY Item\_Description constraint Item\_Desc\_not\_null NOT NULL;

Alter Table ITEM  
MODIFY Quantity constraint Quantity\_not\_null NOT NULL;

Alter Table ITEM  
Modify Quantity DEFAULT 1;

Alter Table ITEM  
Modify Quantity\_Type DEFAULT 'Piece';

**Note:** During actual implementation it is hoped that the attribute Quantity\_Type will consist of a predetermined list of the most popular quantity types e.g. carton, box, bottle, g(grams) ml (millilitres), etc... Users will simply choose whichever quantity type is most appropriate from a drop-down list. However as this will take some research and testing before implementation, it will not be implemented at this point.

Create Table LIST  
(  
 List\_ID Number(2) constraint List\_ID\_PK Primary Key,  
 List\_Name Varchar(40),  
 List\_Description Varchar(50)  
)

Alter Table List  
MODIFY List\_Name constraint List\_Name\_not\_null NOT NULL;

Alter Table List  
MODIFY List\_Name DEFAULT to\_char(sysdate, 'Dy DD-Mon-YYYY HH24:MI:SS');

CREATE SEQUENCE list\_id\_seq  
MINVALUE 1  
MAXVALUE 99  
START WITH 1  
INCREMENT BY 1  
CYCLE  
CACHE 20;

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)  
)

### 7.5.5.2 DML (Data Manipulation Language) Insert Statements

These statements will be used to test the constraints and defaults used in the create / alter tables section.

INSERT into Item  
Values (item\_id\_seq.nextval, null, 'Head and Shoulders Shampoo', 500, 'ml', ‘sdcard/CAMERA/ 1287513967339’);

INSERT into Item  
Values (2, null, 'Kelloggs Corn Flakes', default, 'kg');

INSERT into Item  
Values (item\_id\_seq.nextval, '5010934002860', 'Odor Eaters', default, 'packet');

INSERT into Item  
Values (item\_id\_seq.nextval, null, 'Birthday Cake', 2, default);

Insert into List  
Values (list\_id\_seq.nextval, default, null);  
  
Insert into List  
Values (list\_id\_seq.nextval, 'Marys Birthday', 'Stuff for Marys Birthday');

Insert all   
into List\_Item  
Values(1,1)  
into List\_Item  
Values(1,2)   
into List\_Item  
Values(1,3)  
into List\_Item  
Values(2,4)  
select \* from dual;

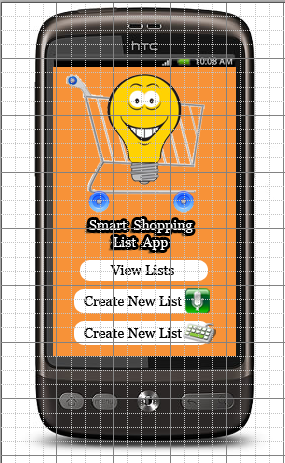
### 7.5.5.3 Database Testing

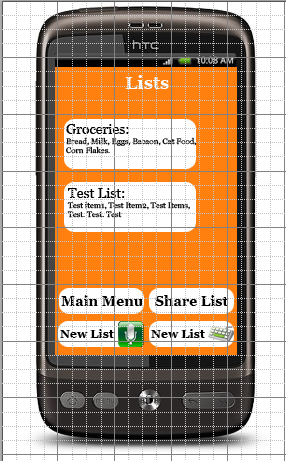
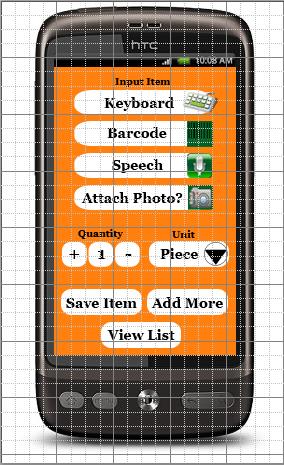
Referential integrity can be tested by trying to drop either the List or Item tables without first dropping the List\_Item table.

Testing null-able attributes can be tested using insert statements that leave fields that are defined as not null, as null.

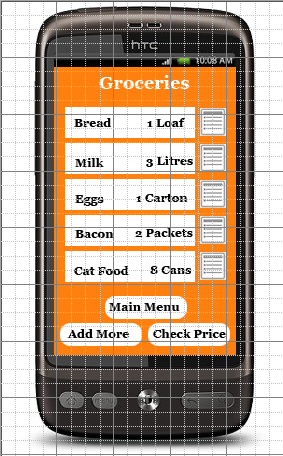
# 8. Visual Design

## 8.1 Layout Grids

  
*Application Icon Splash Screen*

**

*Lists Screen Add Item Screen*

*  
List Item Screen Checkout Screen*

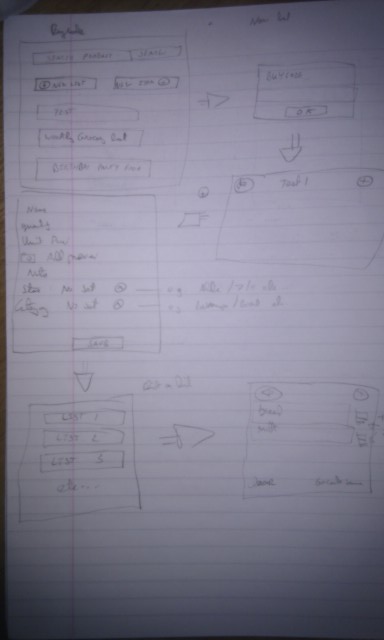
## 8.2 Design Sketches

  
*Initial Sketch of Splash Screen and List Screen layouts*

## IMAG0484.jpg

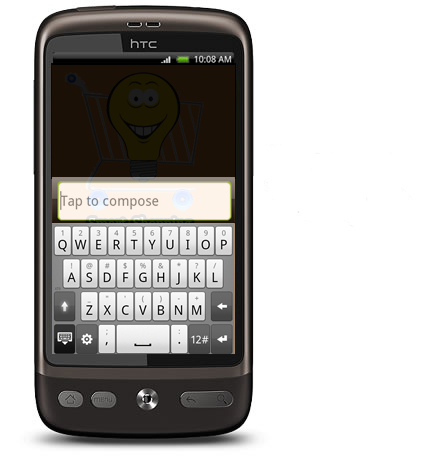
*Initial Sketch of Add Item and List Item Screens*

  
*Initial Sketch of Alternative Design Ideas that were not implemented*

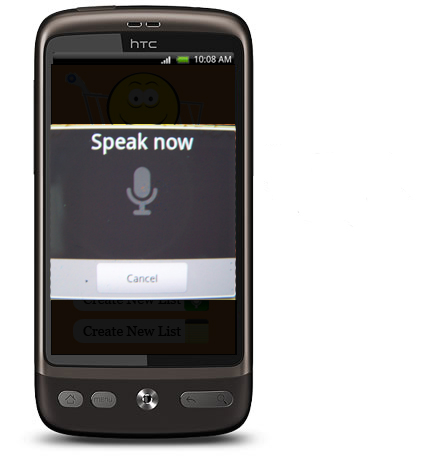
  
*Initial Sketch of Alternative Design Ideas that were not implemented*

## 8.3 Mock-ups

*Application Icon Splash Screen*



*Splash Screen – Create New List with Keyboard*

*Splash Screen – Create New List using Speech Recognition*

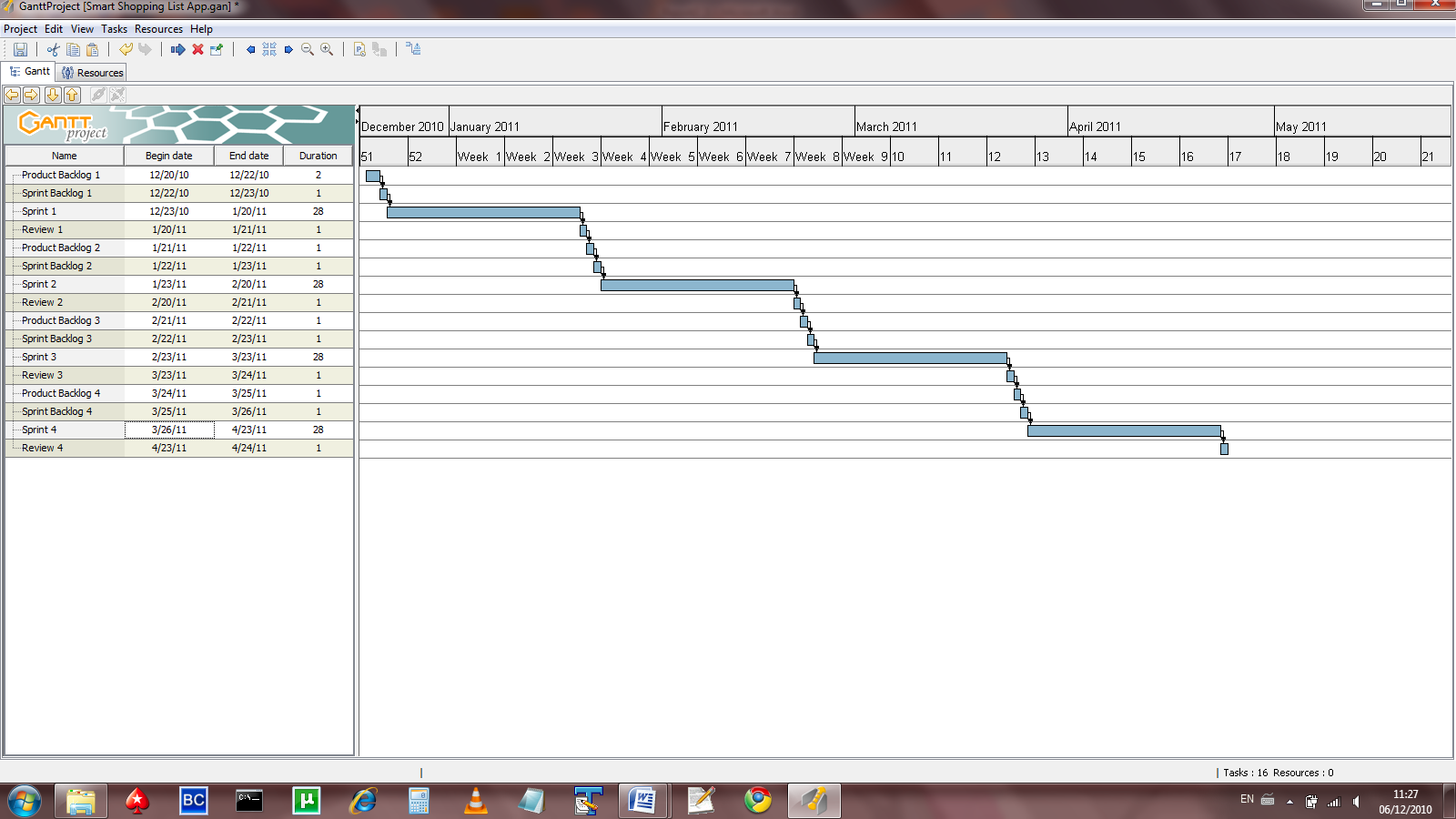
  
*Lists Screen*

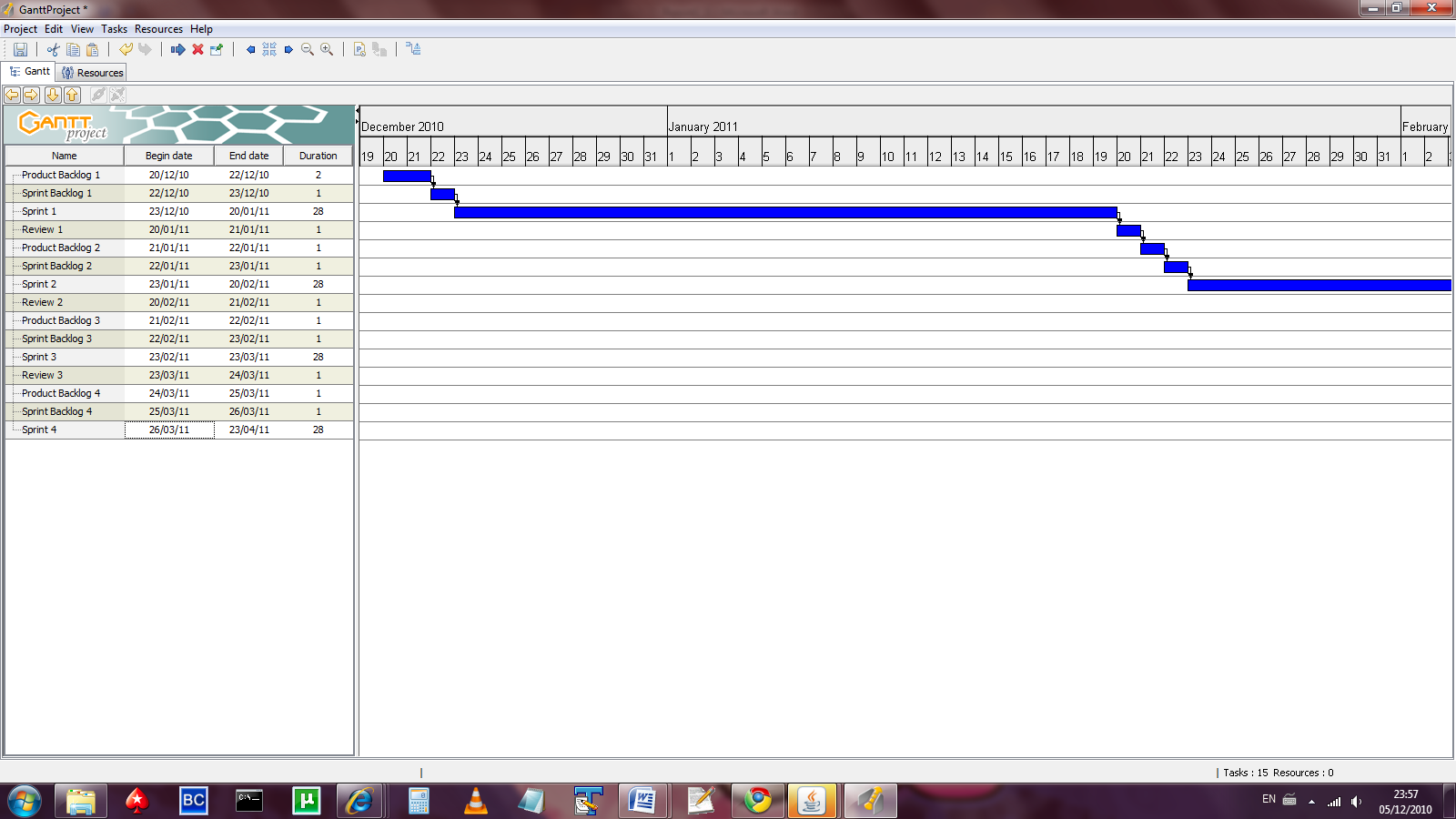
*List Item Screen*

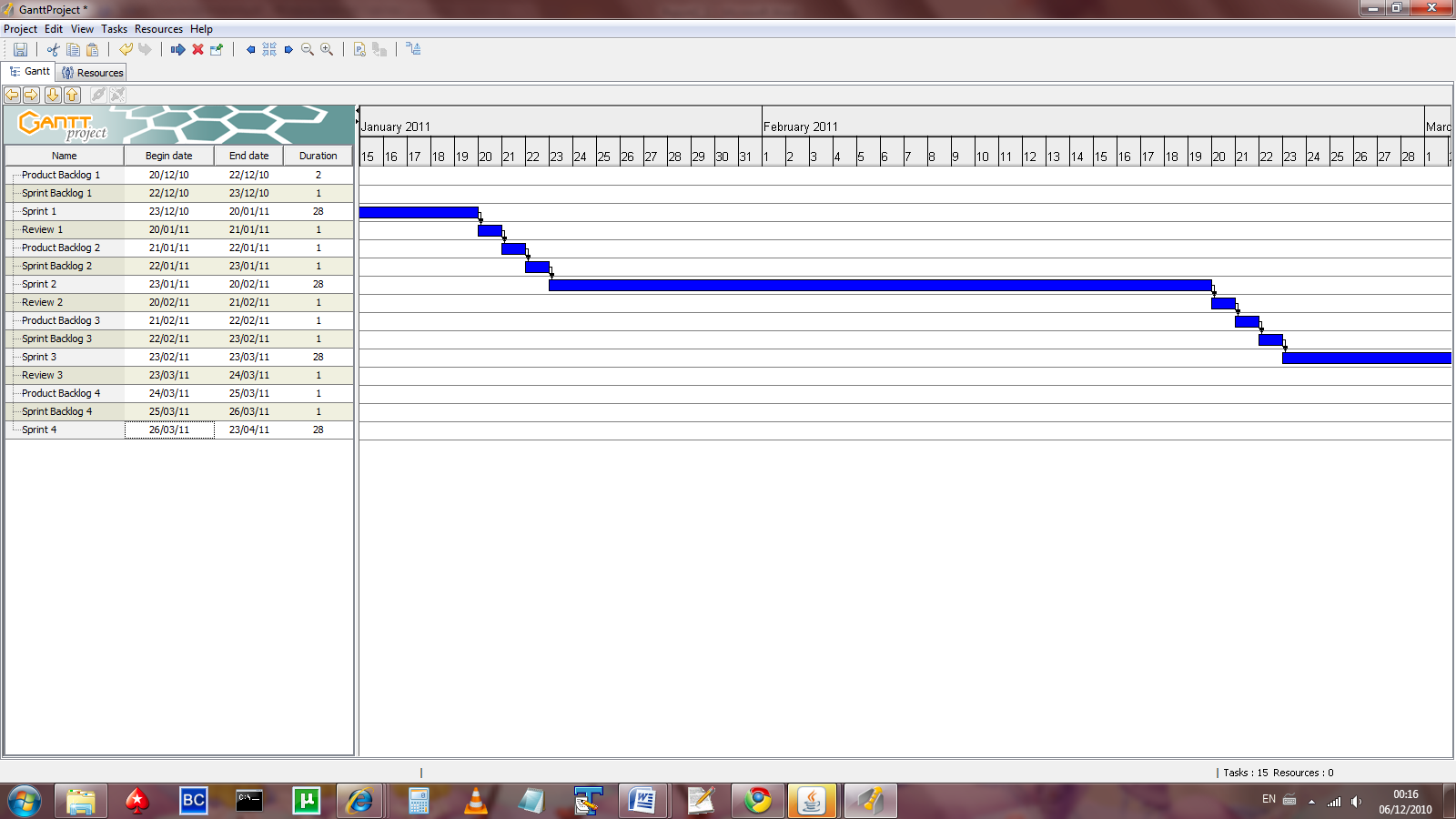
*Add Item Screen*

Checkout Screen

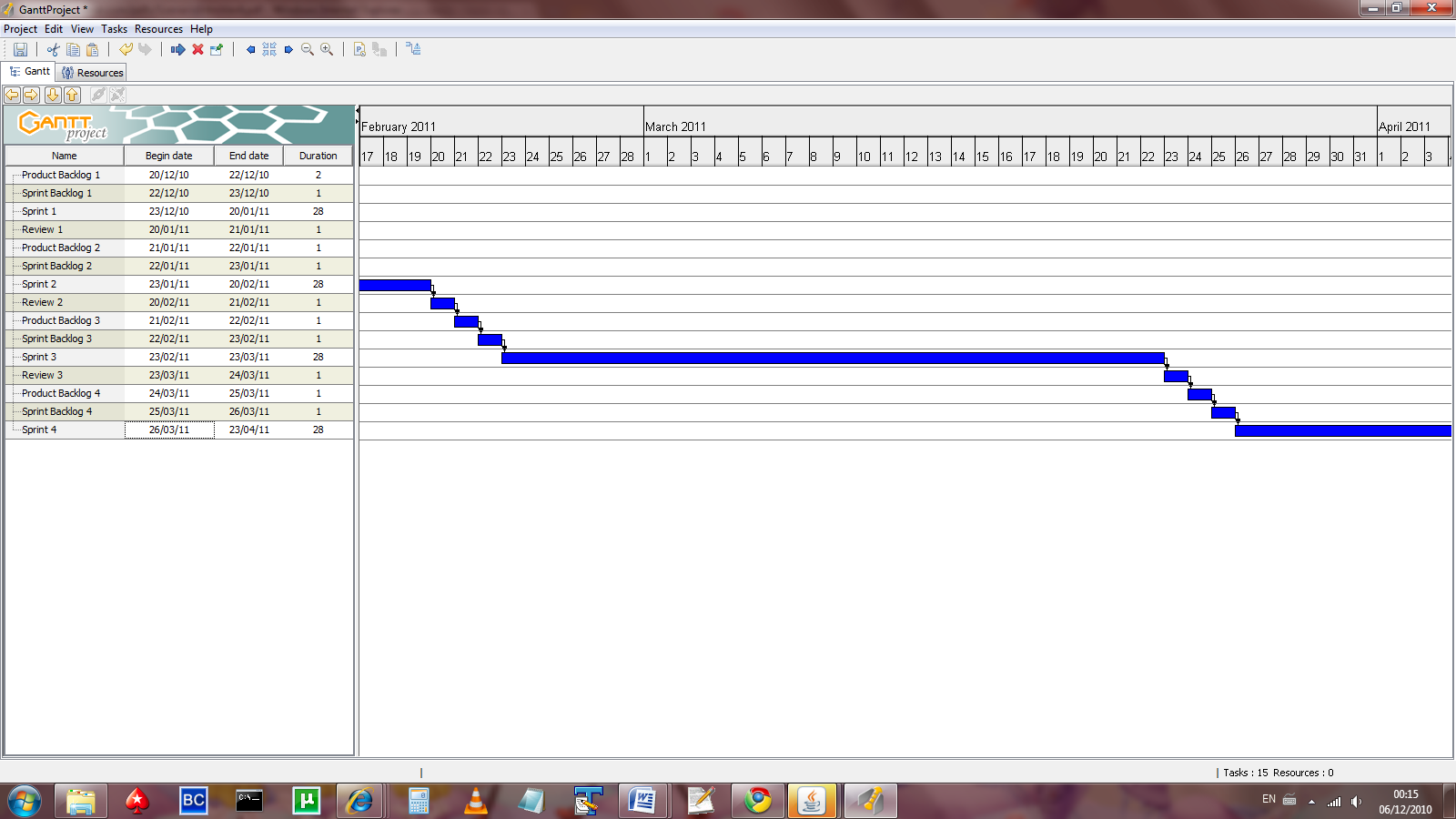
# 9. Schedule

*****Overall Project Schedule*

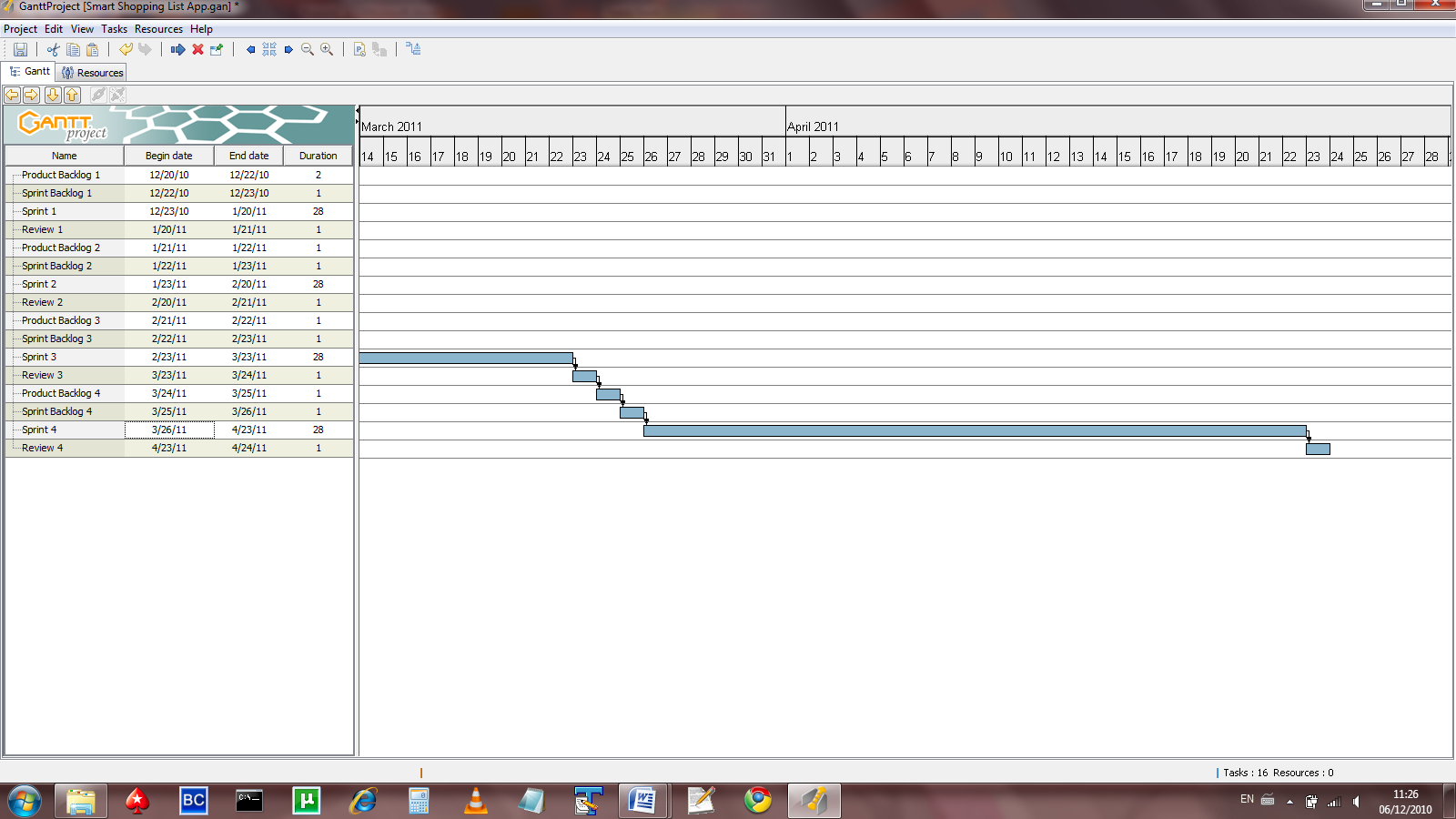
*****Schedule showing initial SCRUM sprint cycle*

****

*Schedule showing second SCRUM sprint cycle*

****

*Schedule showing third SCRUM sprint cycle*

****

*Schedule showing fourth SCRUM sprint cycle*

**Schedule Activities and Deliverables**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Start Date** | **Finish Date** | **Project Phase** | **Activity** | **Deliverable** | **Estimated Hours** |
| 20/12/10 | 22/12/10 | 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 |
| 22/12/10 | 23/12/10 | 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 |
| 23/12/10 | 20/1/11 | Sprint 1 | First Sprint iteration of the project implementing the most basic core goals | Functioning shippable product containing the most basic of the core goals | 32 |
| 20/1/11 | 21/1/11 | 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 of the first functional shippable product | 4 |
| 21/1/11 | 22/1/11 | Product Backlog 2 | Review of work items to be included in the project | Updated list of items to be included in the project | 3 |
| 22/1/11 | 23/1/11 | 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 | 3 |
| 23/1/11 | 20/2/11 | Sprint 2 | Second Sprint iteration of the project implementing the more advanced core goals | Functioning shippable product containing the most basic and more advanced core goals | 32 |
| 20/2/11 | 21/2/11 | 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 of the second functional shippable product | 4 |
| 21/2/11 | 22/2/11 | Product Backlog 3 | Review of work items to be included in the project | Updated list of items to be included in the project | 3 |
| 22/2/11 | 23/2/11 | 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 |
| 23/2/11 | 23/3/11 | Sprint 3 | Third Sprint iteration of the project implementing either the final core goals or else the most basic secondary goals | Functioning shippable product containing either the final core goals or else the most basic secondary goals | 32 |
| 23/3/11 | 24/3/11 | 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 of the third functional shippable product | 4 |
| 24/3/11 | 25/3/11 | Product Backlog 4 | Review of work items to be included in the project | Updated list of items to be included in the project | 4 |
| 25/3/11 | 26/3/11 | 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 | 4 |
| 26/3/11 | 23/4/11 | Sprint 4 | Fourth Sprint iteration of the project implementing either the final the most basic secondary goals or the more advanced secondary goals | Functioning shippable product containing either the most basic secondary goals or the more advanced secondary goals | 32 |
| 24/3/11 | 26/3/11 | 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 of the fourth functional shippable product | 10 |

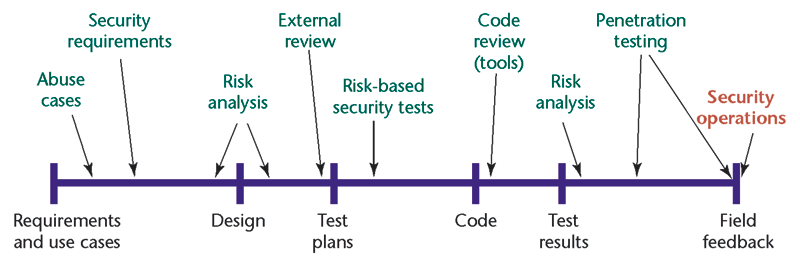
**Total Hours: 180**

# 10. Security

Traditionally, security was never a major concern. Before I.T. became the cornerstone of society that it is today, it was a relatively niche field. Computers were generally so expensive that only a handful of organisations such as universities, governments and the military had use for computers and more importantly the finance to provide them. A lot of the systems and protocols that are in use today are built on these early technologies. Therefore the legacy from a lot of these systems and protocols that have been inherited mean that they have built-in security vulnerabilities. Even today, the complexity of modern programs and operating systems often mean that it is impossible to test every use-case, and the economics of the software industry favour a “release then patch” model.

However, there is a growing acceptance of the need to factor security into the software development process. With regard to the core goals of this project there is perhaps little need for security. If however the core goals of this project are reached, the secondary goals of this project could benefit from having security in mind during the software development process. This is especially true as traditionally when new technology is released, (such as the smart phone technology this application will run on) the vulnerabilities aren’t known until they are exploited. By thinking about security now, it allows a proactive approach, rather than a reactive approach to be taken.

One of the most widely cited papers on the subject is called “Bridging the Gap between Software Development and Information Security” (van Wyk & McGraw, 2005). Although not all the aspects of the paper discussed below are applicable for this project, those that are will be discussed with regard to this project.



***(van Wyk & McGraw,2005)***

**Abuse Cases:** are considered during the “requirements and use case” stage. Abuse cases are similar to use cases in which the behaviour is a system is modelled in response to a request, however in abuse cases the developer must consider how the software may be deliberately misused. The developer must identify the risk and also how the system should respond. Examples of abuse cases may be buffer overflow or SQL injection attacks, where certain strings or code may be entered into user input fields in order to exploit the system. Therefore for this project all user input will be assumed to be malicious and handled accordingly in order to secure against the system being exploited.

**Business Risk Analysis:** is part of the Design stage and is particularly important for certain secondary goals of this project (if the implementation proceeds that far). Business risk analysis assesses the impact on a business if a security threat is realised. Some of the secondary goals of this project may rely on integrating with some of the major supermarket chains systems. If a security breach were to occur, the damage to these companies could be very high. There may be damage in relation to direct costs such as loss of revenue and cost incurred in resolving a problem as well as indirect costs such as damage to reputation.

**Architectural Risk Analysis:** is also part of the design stage and is similar to business risk analysis. Architectural Risk Analysis “assesses the technical security exposures in an applications design” (van Wyk & McGraw, 2005). This means examining the whole system and every element e.g. every interface and interaction against known security weaknesses. As the field of Smart Phone applications is relatively new, there are not many known security weaknesses (there has yet to be a spate of viruses or known vulnerabilities published for mobile devices). However that is not to say that they do not exist and for this project every interface and interaction will be tested against know security weaknesses.

**Security Functionality Testing:** is part of the test planning stage. It involves tests that verify such as encryption, user authentication, logging, confidentiality and so on.

**Risk Driven Test:** is again part of the test planning stage. Risk based test scenarios are the natural result of the process of assessing and prioritizing software’s architectural risks. Each architectural risk and abuse case considered should be described and documented down to a level that clearly explains how an attacker might go about exploiting a weakness and compromising the software. Such descriptions can help generate a priority-based list of test scenarios for later “adversarial” testing.

**Code Review:** is part of the implementation stage and is used to protect against implementation bugs that may be introduced through the coding of this project. The SCRUM methodology used in this project for a review after each sprint implementation, so this may be an opportunity to conduct a code review also.

**Penetration testing:** is part of the system testing stage. Penetration testing focuses on human and procedural failures made during the software’s conﬁguration and deployment. The best kinds of penetration testing are driven by previously identiﬁed risks and are engineered to probe risks directly to ascertain their exploitability

**Deployment and operations:** Careful conﬁguration and customization of any software application’s deployment environment can greatly enhance its security posture. Designing a smartly tailored deployment environment for a program requires following a process that starts at the network-component level, proceeds through the operating system, and ends with the application’s own security conﬁguration and setup.

# References

Appelo, J. (2008) *The Definitive List of Software Development Methodologies.* Available at: http://www.noop.nl/2008/07/the-definitive-list-of-software-development-methodologies.html  
(Accessed: 28 October 2010).

Boards.ie Limited. (2010). *Boards.ie - Now Ye’re Talking.* Available at: www.boards.ie.   
(Accessed: 31 October 2010).

Google. (2010) *Platform Versions | Android Developers.* Available at: http://developer.android.com/resources/dashboard/platform-versions.html.  
 (Accessed: 3 November 2010).

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

Kapp, K. (n.d.). *Software Functionality Scenario Template and Example For Teaching a Software Procedure.* Available at: http://www.karlkapp.com/materials/Scenario%20Example%20and%20Template.pdf. (Accessed: 1 December 2010).

Pressman, R. (2000). *Software Engineering – A Practitioner’s Approach.* 5th Ed. London: McGraw-Hill Publishing Company.

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

Van Vliet, H. (1997) *Software Engineering Principles and Practice.*4th Ed. Chichester: John Wiley & Sons.

Van Wyk, K & McGraw, G. (2010). *Bridging the gap between software development and information security*. Available at: http://www.cigital.com/papers/download/bsi10-ops.pdf. (Accessed: 3 December 2010).

Vodafone Ireland Limited. (2010). *Get Mobile Phone Services, Home Phones & Broadband - Vodafone Ireland.* Available at: www.vodafone.ie. (Accessed: 8 November 2010).

Wikipedia. (2010a) *Agile software development.* Available at: http://en.wikipedia.org/wiki/Agile\_Software\_Development  
(Accessed: 20 November 2010).

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

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

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