COM814 Project Dissertation

School of Computing & Information Engineering

Supervisor: Shuai Zhang

Second Marker: Gaye Lightbody

Declaration of Originality

When submitting your work you are agreeing with the following statement:

I declare that this is all my own work and that any material I have referenced to has been accurately reference. I have read the University's policy on plagiarism and understand the definition of plagiarism. If it is shown that material has been plagiarised, or I have otherwise attempted to obtain an unfair advantage for myself or others, I understand that I may face in accordance with the policies and procedures of the University. A mark of zero may be awarded and the reason for that mark will be recorded on my file

Signed _____

(Daniel Willis, B00615338)

Date: 1/9/2016

Contents

Abstract

As society becomes increasingly mobile and globalised it is ever-common that familial and social circles become spread across the globe. Consequently, it can thus be difficult to organise gift buying at times of celebration like birthdays, weddings and other momentous occasions. Not only is it important to ensure that the recipient does not already have the item, buyers may also worry about buying an item the recipient will enjoy and, perhaps most essential, they should not have to worry about buying the same item as another person.

In addition to a more globalised international community there has been a concurrent shift in the world of technology – one that has led to the meteoric growth of the mobile technology market. This includes the evolution of the traditional mobile phone into a smartphone capable of offering more than just simple phone calls and text messaging services. The introduction of other smart devices like tablets offering similar functions has led to a dynamic new mobile marketplace, allowing users flexible, accessible, relatively affordable access to a previously unknown world of computing on-the-go.

With this in mind, a mobile application has been developed in order to remedy to problem alluded to above. The following dissertation is an attempt to document the development process of the application in question.

1 – Introduction

1.1 – Introduction

The dramatic and continued growth of the mobile technology market has opened many avenues for both businesses and users in terms of providing and accessing content and services on the move via mobile applications.

1.2 – Problem Statement

As social mobility escalates exponentially and the international community becomes increasingly globalised it is now common to find that families and groups of friends find themselves scattered across the globe. Accordingly, at times of celebration it can be difficult to coordinate the purchase of gifts for those at the centre of festivities, while avoiding the infamous "ten toaster problem" – when several individuals of good intention inadvertently purchase the same item. Additionally, buyers often worry that the item they are thinking of buying may not be appreciated or might already be owned by the recipient. To this end, the following problem statement has been created to summarise the issue at the centre of this project:

"To ensure recipients are not gifted multiple copies of the same item it is necessary to develop a gift suggestion and allocation system. This will allow recipients to generate a wish list of items they want which can be seen by specified contacts, who can use it to determine what gift to buy – safe in the knowledge they are buying a desired gift, while also not duplicating the selection of another purchaser"

1.3 – Aim

In brief, the aim of the mobile application to be discussed in this dissertation is to relieve the stress of gift buying, ensuring that purchasers have a selection of desired items to choose from without having to worry about buying the same item as someone else. This selection of items is based on a wish-list which is created by the recipient that can be added to and updated in the period running up to the corresponding event.

1.4 – Objectives

To carry out the required aim, a list of objectives has been drawn up in order to outline the purpose of this project:

1. Analyse the problem to clearly understand what current difficulties are faced by those planning for upcoming events
2. Carry out competitive market analysis in order to see what solutions currently exist, identify potential gaps there are and how the developer can attempt to fill them
3. Develop a focus group to determine what users want in a gift wish list app and use feedback to shape overall system design
4. Design and develop a prototype in response to feedback and research in order to ensure functionality and content of the app match requirements
5. Test this prototype to ensure that the system functions as required, allowing for necessary adjustments or modifications to be made in good time

6. Use the focus group to test the app in order to meet the specified requirements
7. Evaluate the system after testing with focus group in order to provide recommendations for future development

1.5 – Dissertation Outline

- Chapter Two: *Analysis* – This chapter outlines and details the initial problem solving undertaken in order to remedy the aforementioned problem. It will comprise all background research carried out into the problem, including detailing the problem itself, existing solutions, a proposed solution and how to actually solve the problem. System requirements will also be defined at this stage in response to focus group feedback.
- Chapter Three: *Design* – This section will encompass an analysis of the system design, including the graphical user interface (GUI), software architecture, data definitions and models illustrating how the system works.
- Chapter Four: *Implementation* – Building on the work done in the design chapter, this section will focus on the technical makeup of the project. It will detail the development environment and coding process as well as a detailed examination of some key functionality of the system.
- Chapter Five: *Testing and Evaluation* – Here the developer will outline the testing process of the system, including in-house and external

testing by a focus group. Feedback and results from focus group testing will be analysed in comparison with the original system requirements to determine how successful the developer has been in achieving the original aims.

- Chapter Six: *Conclusions and Recommendations* – The project as a whole will be summarised and considered in terms of effectiveness. The evolution of the project will be discussed and areas for potential future development will be indicated and acknowledged

2 – Analysis

2.1 – Introduction

This section of the dissertation is intended to act as the initial research carried out by the developer in their attempt to

- understand and define the problem at the heart of this project
- analyse existing solutions and identify gaps in the market and
- propose a new solution aimed at solving the problem.

It will also be necessary here to deconstruct the professional issues associated with developing such a project and outlining how the system in question adheres to relevant legislation and professional concerns.

2.2 – Current Problem

Bearing the problem statement outlined at the beginning of this dissertation in mind, a rich picture describing the problem central to this project follows:

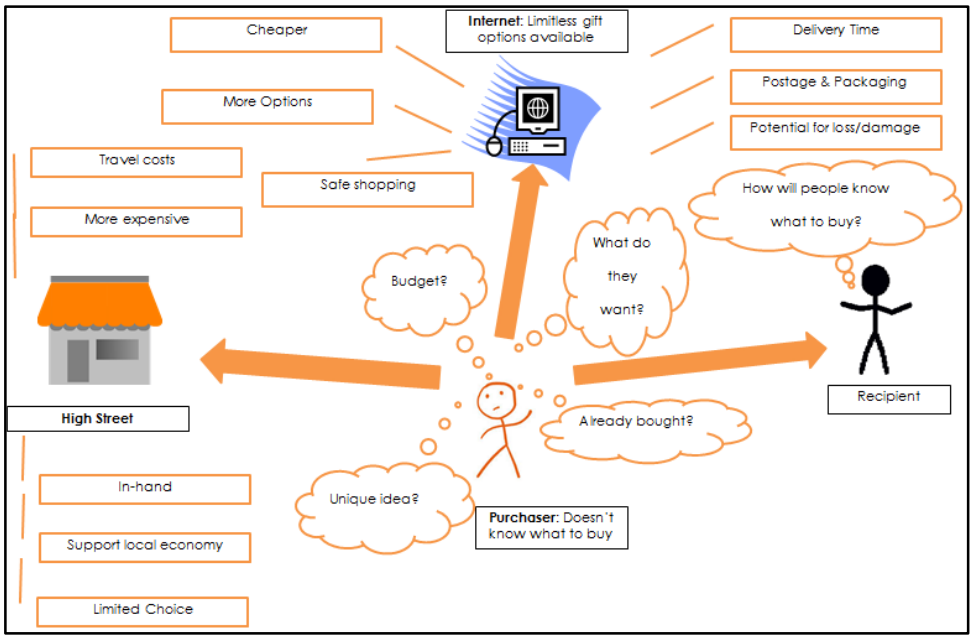


Figure 1 – Rich Picture illustrating the current problem

As illustrated above, we can see the plethora of problems facing a gift purchaser. The main concern is obviously deciding what to buy and this can be affected in a number of ways, for example

- they may be working within a strict budget
- they might not know what has already been bought
- they are potentially oblivious as to what the recipient actually wants
- they may not be sure if their gift idea is genuinely unique

Additionally, the shopper must also decide where to begin looking for gifts – either on the high street or online, and both options entail their own considerations.

To begin with online, the user is going to be overwhelmed with choice. Although this can be seen as a positive feature enabling them to find a great many gift ideas, it can often be difficult to know exactly where to begin, and some users may not have the confidence or time necessary to trawl through the mass available online shopping outlets available to them. This increased variety of online vendors is furthermore problematic as it is difficult to discern a trustworthy and reliable company to buy from. Additionally, another important concern for online shopping is that users can leave themselves open to the possibility of cyber-crime if they are not careful with their personal payment details.

While online shopping becomes increasingly popular – highlighted through research carried out by groups such as eMarketer (2015) and the Office for National Statistics (2015) – which can be done from the comfort of your own armchair it also offers the potential for cheaper items. However

there is a trade-off in that delivery charges are normally incurred. Furthermore, there is the possibility that items may be badly damaged or even lost in transit – therefore not arriving on time or in acceptable condition. More often than not it also means that the purchaser will have to be in their house at the time of delivery, particularly for larger items or those requiring a signature, and this can be problematic if delays occur or tracking information is not provided.

On the other hand if the purchaser instead decides to search for gifts on the high street there are a number of different but equally important issues to consider. High street shops may not offer the same variety of options as their online competitors, thus leaving customers disappointed and left wanting. Additionally, physically travelling to shops involves the inconvenience of leaving the house – a problem for somebody who may not live locally to them or who is simply not that mobile. Despite this it must be noted that footfall and trade within the area has undeniable benefits for the local economy by benefitting regional commerce. Browsing in a shop offers the added benefit of being able to examine an item up close before buying, allowing customers to see exactly what they will be paying for. Relatedly, while high street shops may offer items at higher prices than online alternatives the user gets what they pay for in hand when they pay for it – alleviating the stress of having to wait for an item to be delivered, worrying about its condition during delivery and having to be at home for delivery.

2.3 – Existing Solutions

Currently there are a number of existing services which aim to reduce the stress surrounding gift buying, a number of which have been described in the following table

Service	Benefits	Limitations
Gift Planner https://itunes.apple.com/gb/app/giftplanner-gift-list-organizer/id479882306?mt=8	• Suitable for various occasions • Allows a budget to be set • Tracks spending • Contacts can be input as new or via address book • Barcode scanning	• iOS only • No companion site • Requires in-app purchases to enhance functionality • No user accounts • User manually enters item details • User doesn't know if item is wanted • No ability to share lists • App aimed at users monitoring their own spending
Giftster https://www.giftster.com/	• Suitable for various occasions • User accounts • Group creation • List-sharing • No account required to reserve gifts on a list • iOS, Android and companion site • Free-to-use • Social media integration	• Reports of buggy performance • Accounts are specific to this system
iWishFor http://www.iwishfor.ca/	• Suitable for various occasions • Barcode scanning • Social media integration • Free-to-use • iOS, Android app	• Non-responsive website • Buggy application, poor performance
John Lewis – the Gift List https://www.johnlewisgiftlist.com/giftint/JSPs/GiftList/glsmain.jsp	• Store delivers gifts • Single source for shopping	• Store specific • Event specific • Single source limits variety

mGifts https://itunes.apple.com/gb/app/mgifts-gift-list-manager/id297631123?mt=8	• Suitable for various occasions • Budget can be set • Tracks spending • Group creation	• iOS only, no companion site • Manual list sharing via email • App aimed at users monitoring their own spending • User doesn't know if item is wished for • Pay-to-use • Security is optional
No More Socks: The Christmas List Genius https://itunes.apple.com/gb/app/no-more-socks-christmas-list/id399220659?mt=8	• Suitable for various occasions despite name • Budget can be set • Tracks personal spending • Contact customisation	• Pay-to-use • iOS only, no companion site • Aimed at users monitoring their own spending • User doesn't know if item is wished for • No list sharing
Santa's Bag https://itunes.apple.com/gb/app/santas-bag-christmas-gift/id397698040?mt=8	• Budget can be set • Tracks spending • Tracks gifts left to buy • Countdown to event	• iOS only, no companion site • Event-specific • Aimed at users monitoring their own spending • In-app purchases

Table 1 – Competitive analysis of existing services

A number of conclusions about the existing gift list application market can be drawn based on the findings outlined in the previous table. Excluding the Giftster service, the other options are aimed at allowing users to create their own lists in order to allow them to monitor their individual gift shopping activities. While this enables users to keep track of personal spending habits – by allocating a budget, showing what they have already bought, showing other ideas – it doesn't solve the "ten-toaster" problem outlined at the beginning of this section. It also means that, because these lists are created by the purchasers themselves, gifts being bought may not be wished for by

the recipient in the first place – in other words, there is no guarantee that the items are desired. Giftster solves this problem by allowing users to share their personal wish lists among social circles. This is based on users signing up to a group, within which they have the opportunity to create their desired list of items and share it among the group members. This means members of this group know exactly what to look for; ensuring they are buying from a pool of sought for items, and also avoids the potential for multiple copies of the same item being bought thanks to real-time list updates indicating which have already been purchased. As outlined in the table, however, a drawback is that users must have accounts specific to the service.

Additionally, a number of the services share similar problems in that they are only accessible on a particular device – the iOS platform – and thus limit the potential exposure of the service. Furthermore, the fact that some services are applications only means that those who happen to be without a compatible device have no way to user the service. Moreover, some of the services described are either pay-to-use or offer in-app purchases in order to access the full variety of functionality options, a common gripe among users of mobile applications who feel this is a shameless way for companies to make more money. In terms of accessibility, the least convenient option considered is the John Lewis Gift List service because it is only available for use within the company's own setup. Another issue with this service in particular is that it is tailored specifically for wedding gift lists, further limiting its application in the real world.

To solve the problem a solution is required which allows users to create and share a wish list of gifts within their circle of friends and family. As the list will be made up of wanted items, this guarantees that purchasers have a selection of items to choose from – removing the stress of worrying about buying a unique gift – and ensures that undesired items will not be purchased. Furthermore the developer is proposing to implement a list that updates to reflect the status of items on the list in an attempt to resolve the possibility of several of the same item being bought.

2.4 – Proposed Solution

Having deeply considered the problem central to this project and carried out relevant research into the pertinent domain the developer has decided to develop a mobile application to simplify the process of gift shopping for those involved in upcoming celebratory events. With an increasing shift towards mobile devices – as described by mobile marketing analyst Dave Chaffey (2016) – this seems like an obvious choice to make as it enables a wide range of users to access and use the proposed application. The following rich picture illustrates the proposed solution to the problem at the centre of the project.

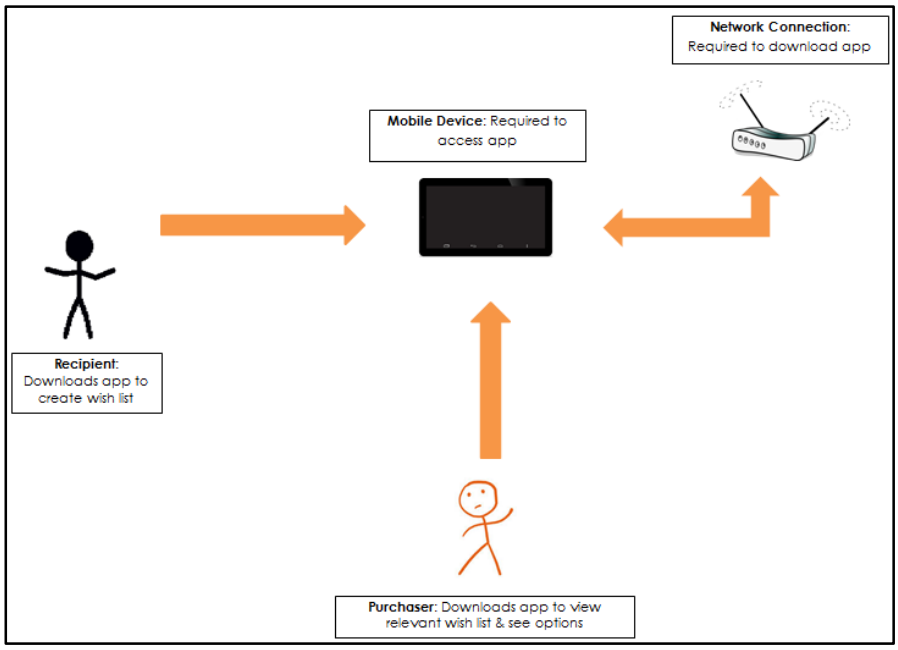


Figure 2 – Rich picture illustrating the proposed solution

As shown, the problem is solved relatively simply thanks to the application proposed by the developer. Users will simply download the app and use the system as follows –

For a Recipient

- The user logs in to the app with their Facebook account information
- The recipient creates a new event in the system – giving it an appropriate description and date – by entering this information before saving it
- The user then shares this event to their Facebook timeline, if necessary limiting who can see it by editing who can see it online
- The user saves items they want for the upcoming event by sharing items to the event from external apps or browsers – for example,

sharing an item while browsing the Amazon Shopping app or a website within their device's web-browser

- The list updates to reflect new additions to the event wish-list even after the user has shared it to their timeline

For a Contact

- Facebook friends of the user who shared the event will see this activity on their Facebook newsfeed
- A friend can click on the shared link and view the wish-list of items
- They then have the ability to view these items and choose which one to buy
- The contact selects an item they wish to buy by reserving it in the list – thus updating the item status, informing other viewers that item has been reserved
- Google cloud messaging sends a notification to the recipient that their wish-list has been updated

Alongside using the competitive research and market analysis previously described as part of developing this new solution, the developer felt it beneficial to create a focus group. This was done with the intention of gathering information from a variety of real-world users in order to gauge what exactly is required from the application in question. Their feedback has been used to inform the functional and non-functional system requirements described after the following feedback analysis.

2.5 – Questionnaire Feedback

Having already carried out market research to analyse pre-existing alternatives and identify gaps in the sector, the developer felt it important to carry out consumer analysis to pinpoint what exactly target users want from a gift wish-list system. This was done through the founding of a focus group and the use of a questionnaire – a template of which can be found in Appendix A – which provided invaluable feedback used to inform the development of the system at hand.

The first question was aimed at determining how problematic people currently find it both organising their gift shopping habits – buying on time, staying within budget – as well as providing gift ideas for upcoming events of their own. While the following graph illustrates that, of the sample quizzed, there is currently little or no difficulty in organising shopping habits, they do have difficulty in terms of providing gift ideas for impending events.

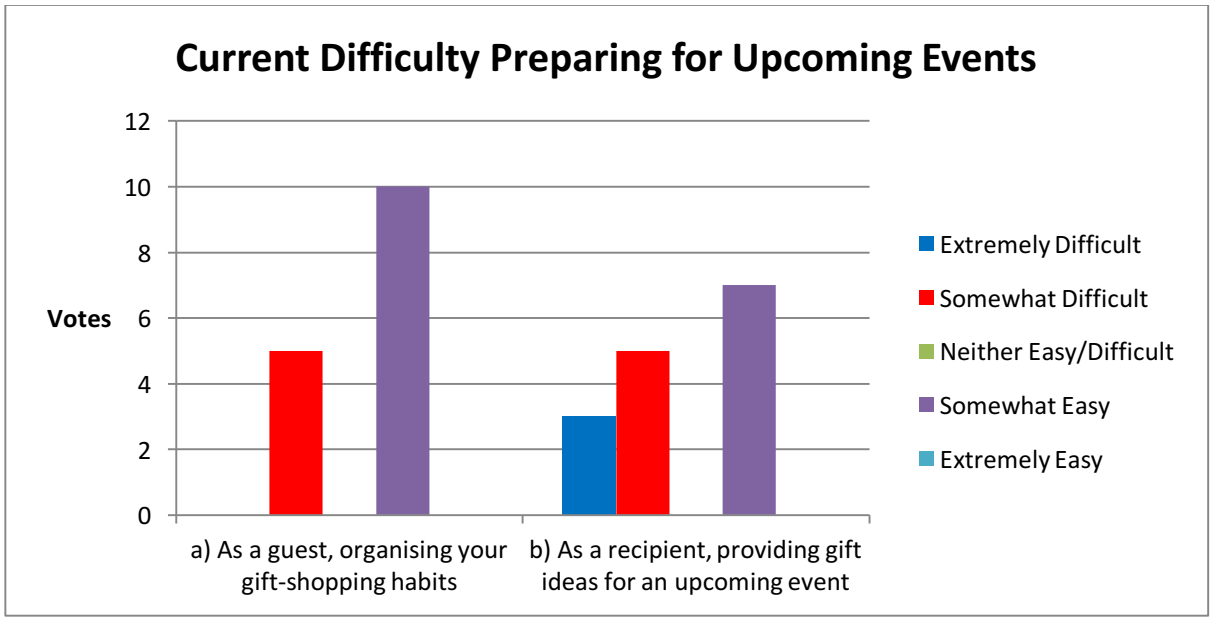


Figure 3 – Graph illustrating user difficulty a) organising shopping habits and b) providing gift ideas

Following this the members of the group were asked to indicate how beneficial a mobile gift wish-list app would be for them and, as illustrated below, the feedback reveals there is a strong demand for such a solution despite an extreme minority indicating otherwise.

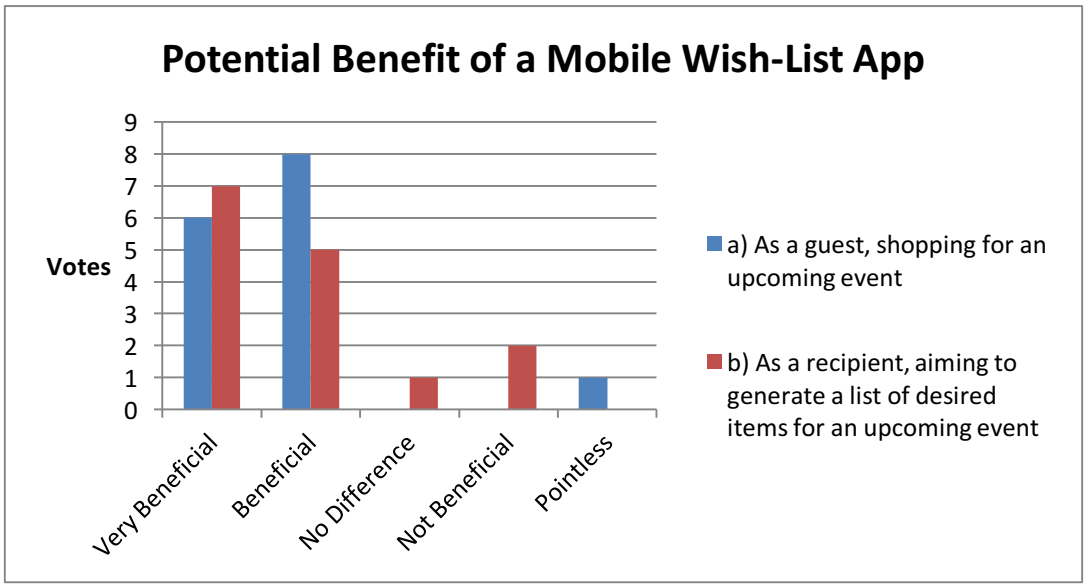


Figure 2 – Graph illustrating potential benefit of a mobile gift wish-list app

Having gauged existing demand for a wish-list application users were then given a list of shopping-related concerns and asked to rank them in order of importance, the results of which are summarised below –

1. Knowing what to buy
2. Knowing what has been asked for
3. Knowing where to buy
4. Worrying an item is already owned
5. Worrying an item has already been bought
6. Getting the best deal
7. Worrying the recipient will not like the item
8. Buying within a set budget

9. Buying in time

Based on these results it is clear to see that users are primarily concerned with buying an item that is unique and will be well received by the recipient. Additionally, as was highlighted in the earlier problem analysis, they also face difficulty knowing where to shop for items. Bearing this in mind, it will be necessary for the system to allow users to view a list of items they can buy and, because this list is generated by the recipient, they can be sure that any of the options will be well received. Furthermore, to avoid having to worry about buying the same item as another user, this list will update to reflect the current status of items on it.

In an attempt to gauge user awareness of similar services the members of the group were given a list of comparable wish-list managers – made up of those used for prior competitive analysis – and asked to indicate their knowledge of them, if they had used them and, if applicable, which they would recommend. The most popular alternative service was John Lewis' "The Gift List", while only a select few had encountered any of the other options listed. This is an important consideration because, in order to reach maximum user-awareness, it will be necessary to provide a practical system that generates good publicity based on an effective, unique functionality that caters to the needs of users.

To ascertain what these needs may be the group was given a list of features and asked to rank them in order of descending importance from 1 to 10, the following graph indicating the top three requested features.

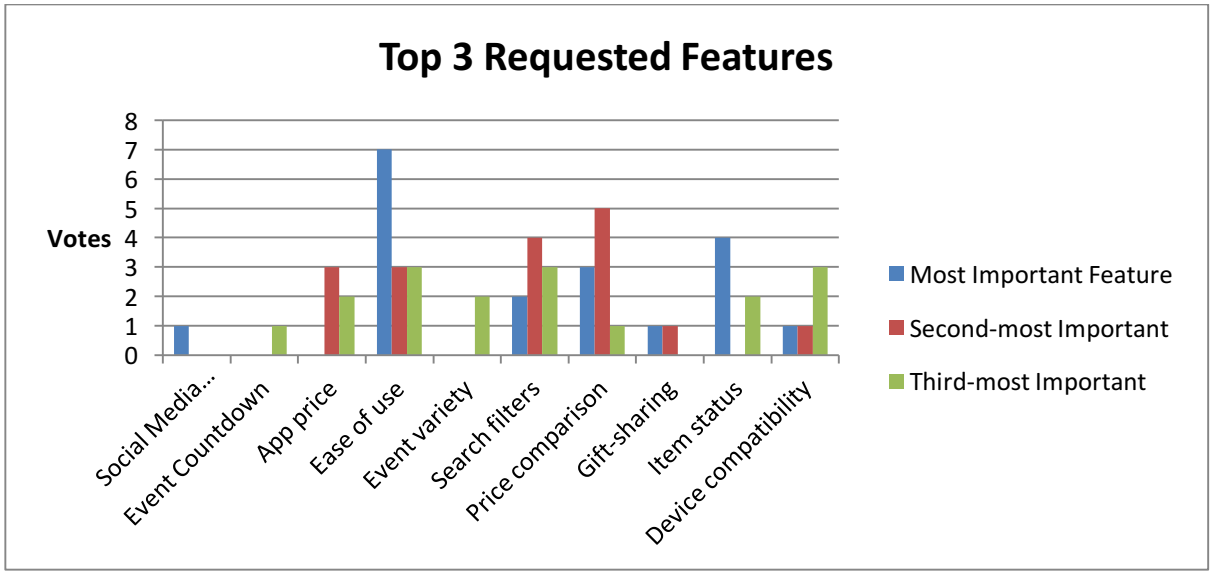


Figure 5 – Graph illustrating top 3 requested features

As can be seen, the focus group determined that making the app easy to use was of highest importance. Despite viewing an item's status receiving the second-highest amount of votes for most important feature, the feature voted second-most important was the ability to compare prices of items on the list. Results show the group then tied device compatibility with the ability to filter list searches. Gift-sharing rounded off the top five, followed by –

- app price
- the ability to generate lists for different occasions
- viewing an item's status
- a countdown to the event
- social-media integration

Given that a user will be required to access the app via their Facebook account a second list of features related to accounts was provided and again, users were asked to rank these in order of preference.

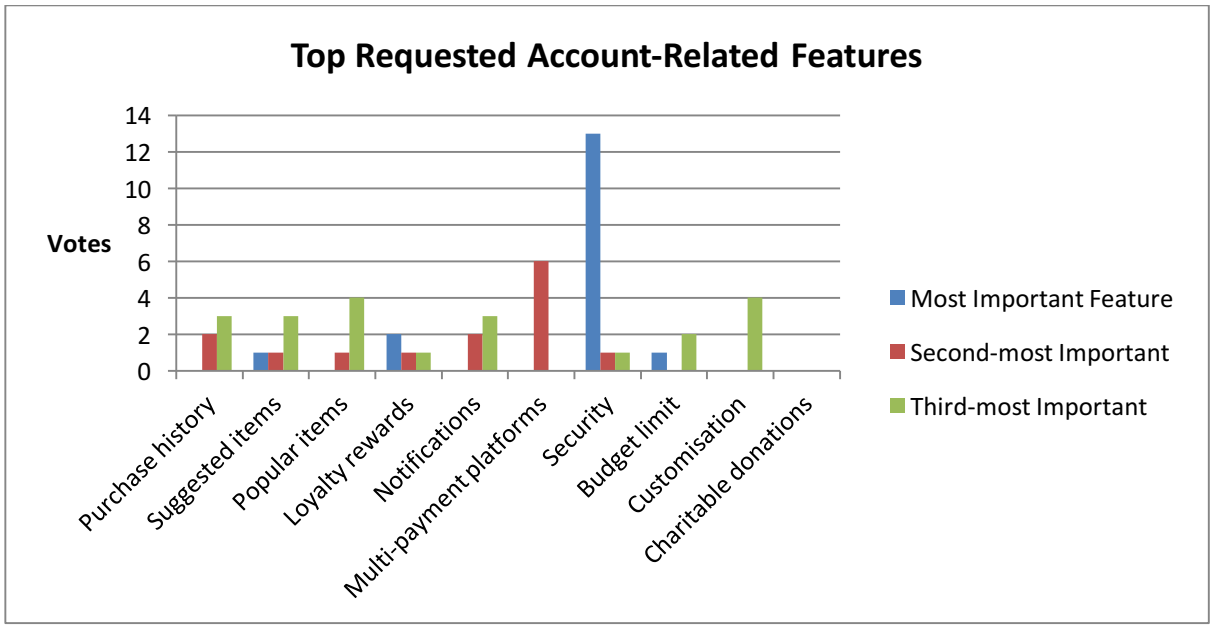


Figure 6 – Graph illustrating most requested account-related features

Perhaps unsurprisingly the group was primarily concerned with security and requested this be of most importance. Results indicate that the ability to link their account with multi-payment platforms was the second-most requested. While the app is currently envisaged as a simple gift-list manager facilitating the ability to manage wish-lists, this is undoubtedly a unique feature which would remove the hassle of having to use different accounts for a variety of online retailers and is ripe for consideration, however this may fall outside of the realm of the project. Finally customisation was jointly ranked with the ability to view popular items. These would both be based on user preferences to allow a uniquely individual experience during use of the application while at the same time allowing users to browse current shopping trends and find

ideas for their own lists. Viewing personal purchase history completed the top five while the remaining features are listed in order of descending preference below –

- view suggested items
- set a budget limit
- loyalty rewards
- personalised notifications
- charitable donations

In an attempt to decide on a suitable name for the app the group was provided with a number of options – as well as the opportunity to provide alternative suggestions – and indicate their preference, with the chart below indicating a clear preference for GiftBox.

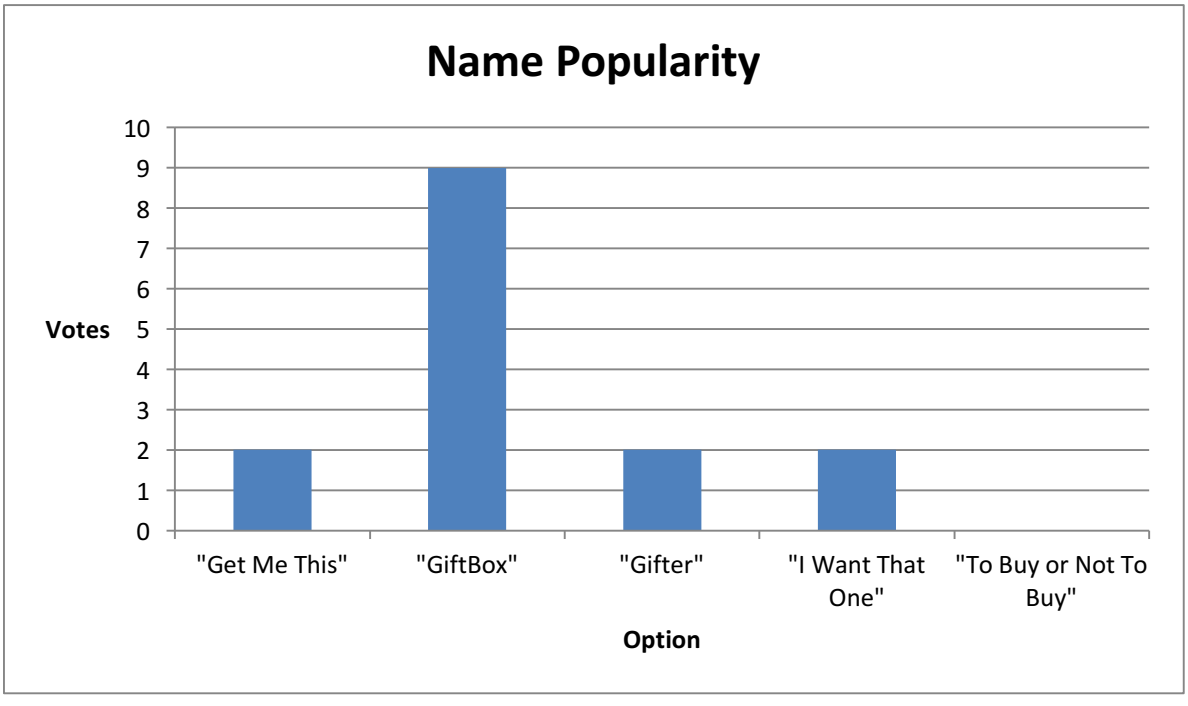


Figure 7 – chart indicating name popularity

Finally, in the interests of maximising readability and accessibility, the group was asked to indicate their preferred font style for use within the app, with the results indicated below –

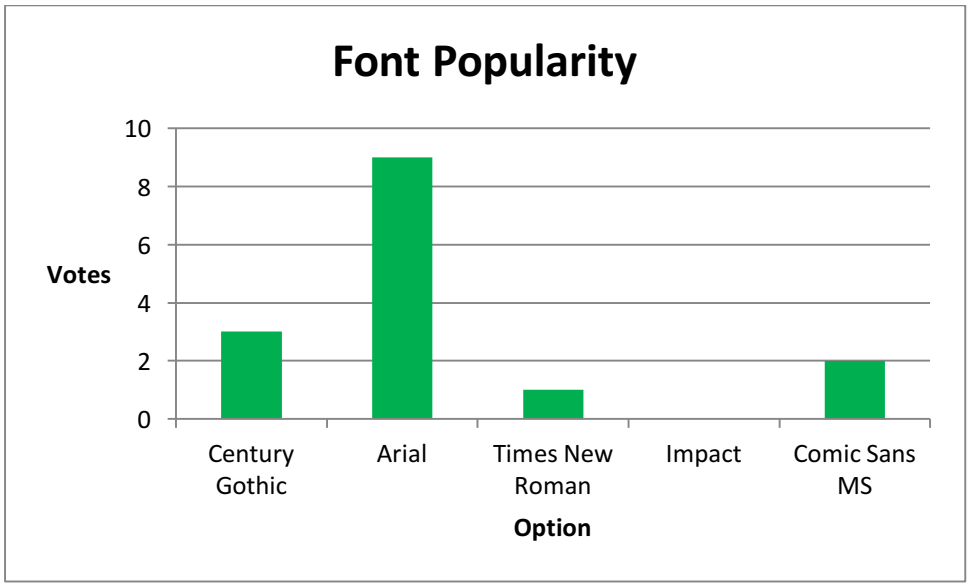


Figure 8 – Chart indicating font popularity

2.6 – System Requirements

Based on feedback received from the questionnaire completed by members of a focus group, a number of functional and non-functional system requirements have been decided upon in response.

2.6.1 – Functional Requirements

Concerning the actual operations of the system at hand the following list summarises required functionality –

- Users should be able to log in with their existing Facebook account
- Users should be able to create an event

- Users should be able to add items to a gift wish-list for a corresponding event
- Users should be able to share the wish-list(s) to their Facebook timeline
- Users should be able to view a gift wish-list online

2.6.2 – Non-functional Requirements

While not regarding functionality itself, non-functional system requirements are just as important in order to allow an effective implementation of the system and the relevant requirements have been summarised henceforth –

- Any personal details required should be dealt with securely
- The system should be intuitive and user-friendly
- The system should be accessible by a wide range of users
- The system should feature an attractive design
- The system should be run on a variety of compatible devices

2.7 – Professional Issues

The proposed application is intended to provide users with a costless, hassle-free way to create, manage and share wish lists with their inner circle of friends and family. This will reduce the stress of gift purchasing in the run up to celebratory events, because users will be able to see exactly what kind of items are sought after by those creating the list. It also negates the possibility that the same gift will be bought by multiple people because once an item has been purchased the list updates to reflect this by greying-out the item in question – thus preventing it from selection by another user. As well as

eradicating the potential for several of the same gift being bought, the app is also intended to reduce the last-minute panic buy by providing users with a countdown to the event itself – allowing them to plan well in advance when and what to buy. Finally, perhaps most importantly, it also ensures that the giftee only receives those items that they have indicated their interest in, thus avoiding awkward embarrassment on the day of the event.

This project is classified as a Category Z project. The project does not involve invasive interaction with people and does not require a full ethical review. This includes the use of anonymous questionnaires for testing software. I can confirm that this project meets the definition for research in the category above. All risks and ethical procedural implications have been considered. The project will be conducted at all times in compliance with the research description protocol and in accordance with the University's requirements on recording and reporting. This application has not been submitted to/rejected by another committee, and if the project category changes as development progresses the project must be resubmitted.

The following risk assessment table considers the main risks associated with this project and are listed along with relevant contingency plans.

Risk Source	Probability (1-5)	Severity (1-5)	Threat (P x S)	Contingency Plan
Physical Risks				
N/A	-	-	-	-
Technical Risks				
Computer Failure	3	5	15	Access to multiple computers

Network Connectivity Failure	3	1	3	Access to multiple networks
Data Loss/Corruption	2	5	10	Regular data backups
Software Issues	2	4	8	Don't rush coding, follow professional practice
Project Management Risks				
Unforeseen Circumstances	3	3	9	Aim to be well ahead of schedule
Time Management	2	5	10	Create & adhere to realistic timeline
Project Document Loss	3	5	15	Perform regular backups

Table 2 – Risk assessment and relevant contingency plans

2.8 – Summary

This opening chapter has detailed the developer's background research into the current problem, including competitive analysis and end-user feedback, in order to prove the feasibility of the proposed solution. A number of key system requirements have been determined in response to this research have informed the developer well in moving forward with developing the new solution

3 – Design

3.1 – Introduction

The aim for mobile developers is to create attractive applications that are intuitive and easy to use, thus the design stage is crucial in determining the overall success of the system in question. It is during this important phase that the both user interface and internal system architecture is clearly defined and outlined in order to provide a firm basis to move forward with development. The following chapter examines in depth the design of the user interface as well as an analysis of the system architecture.

3.2 – Design Methodology

While it may seem an obvious point to make, it is imperative that a mobile application is designed to be as accessible and easy-to-use as possible. With this in mind, alongside user feedback that the system be user-friendly and intuitive, the developer found Ben Shneiderman's rules of interface design (2010) extremely beneficial in designing the system to be as intuitive and accessible as possible. These guidelines have been summarised in the following table and, as illustrated in the screen design wireframes subsequently provided, they have been implemented effectively within the system.

Consistency	This allows a coherent flow throughout the system, enabling user familiarity and confidence
Accessibility	This ensures a wide variety of users will be able to use the system and is reinforced via a consistent design
Informative Feedback	Each user interaction should provide system feedback, informing them of the success/failure of their action e.g. the successful creation of a wish-list
Design Dialogs to Yield Closure	Closure here ensures that users are safe in the knowledge that an action has been complete and they can move on e.g. a notification that they have successfully reserved an item
Error Prevention	Users should not be able to do things they are not meant to do e.g. insert numbers into a character entry field
Undo Options	Users should be allowed to undo or fix actions e.g. edit a wish-list
Give the User Control	Users should feel that they are initiating the interaction process rather than simply responding
Reduce short-term memory load	Avoid the need to remember large quantities of data from one screen to use another

Table 3 – Summary of Shneiderman's *Eight Golden Rules* (Shneiderman, 2010)

3.3 – Application Architecture

Before designing the user interface for the system it was important for the developer to understand how the user should navigate between activities. To this end, the following diagram illustrates the flow of the application

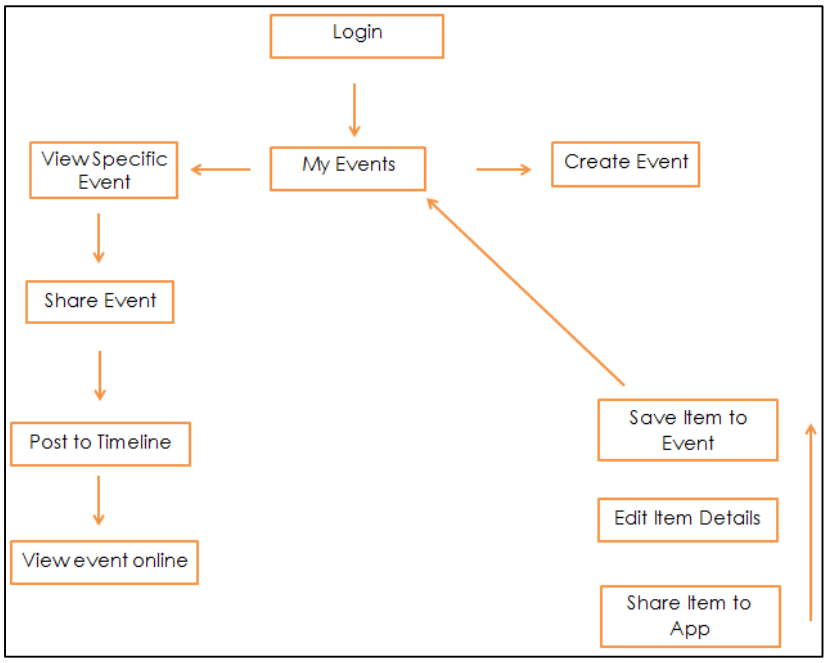


Figure 9 – Application architecture

Once this navigation map was done, the developer proceeded by designing the actual user interface of the system. This was done via the use of wireframe mock-up illustrations and provided the basis for the appearance of the activity screens within the application.

3.4 – User Interface

In the interest of maximising consistency the subsequent activity screens have been designed to follow a similar layout. By providing a consistent experience for the user this allows them to quickly become familiar with the system and rapidly increases confidence as they use it. On loading the application the user will be presented with the login screen at which juncture they are prompted to login via Facebook. Upon clicking the login button the user is redirected to an external Facebook login screen, allowing the user to enter their details

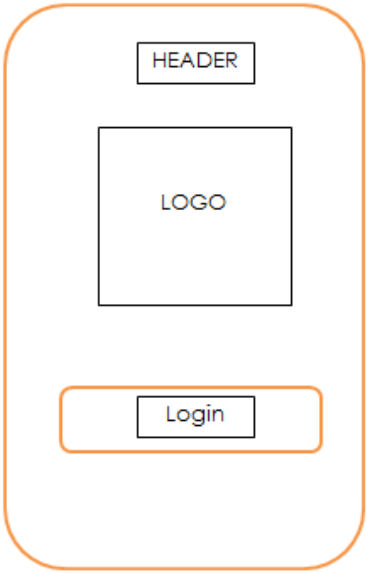


Figure 10 – Login screen

Once successfully signed in, the Facebook login screen disappears to be replaced by a screen containing the user's list of events. These events are contained within a scrollable ListView as clickable buttons, allowing the user to select a particular event and view it in another screen in more detail. Finally this screen contains a button which allows the user to create a new event. This navigates the user to another screen which prompts them to enter the details of this new event.

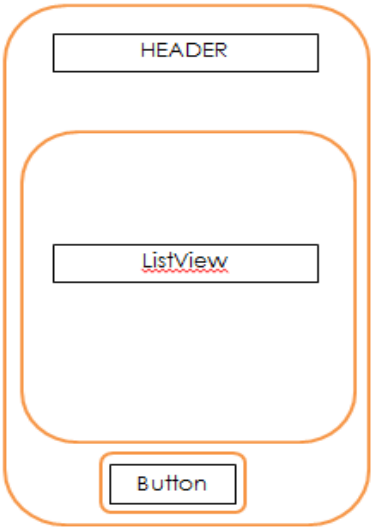
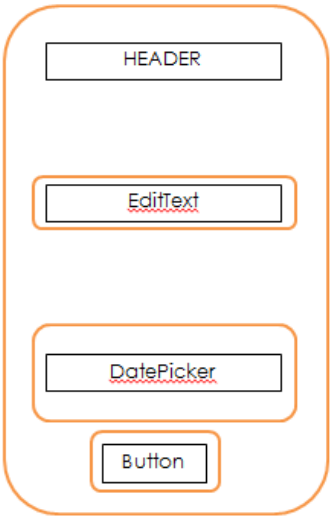


Figure 11 – My Events

To preserve consistency, the screen allowing users to view a specific event looks much the same as the one listing their list of events. A header contains the event description, as well as the date, and a ListView is implemented containing the wish-list of items for the particular event as a series of buttons. This screen also contains a button which allows the user to share the event to their Facebook timeline.

The activity allowing users to share an event to their timeline simply contains a header and a button allowing the user to perform the share operation. When clicked, this directs the user to a Facebook share dialog which allows them to add a comment description and limit who can see the post.



The screen to create an event contains a header, prompts to enter required details, a text field allowing the user to enter the item description, a date picker widget and a button to save the event. Once an event is created the user is returned to the previous screen with their list of personal events updated to reflect the new addition.

Figure 12 – Create Event

Just as viewing a particular event redirects the user to a new screen containing information on the selected event, choosing a certain item allows the user to view it in further detail. This item view screen contains a header, a TextView containing the item description and two buttons – one allowing the user to view the item source and one allowing them to delete the item if they wish. While the former opens the item URL in a web-browser, the latter removes the item from the list and returns the user to an updated event view screen containing the up-to-date item wish-list.

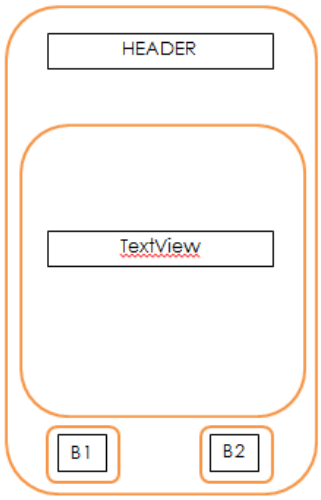
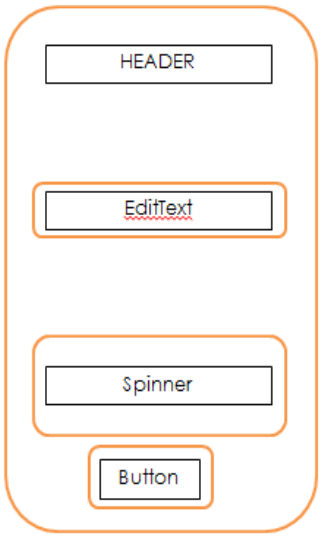


Figure 13 – Item View



In order for there to be items to view, the user must first share these to the application and save them to the desired event. This item share screen contains a header, prompts to define the necessary attributes, an EditText allowing the user to edit the item description, a spinner widget allowing the user to specify which event they are adding the item to and finally a button which saves the item to the corresponding event wish-list.

Figure 14 – Item Share

To allow the user's Facebook friends to interact with a wish-list online, a HTML page has been created in order to support this. This contains the wish-list details in a table holding information about the desired items. The item description is followed by a link that allows a user to reserve an item before an item status field indicating the status of items on the list. Needless to say this updates when an item is reserved, providing contacts with an up-to-date version of the list. Finally the table contains an item source URL that redirects the user to buy the item itself. When a contact selects an item to reserve, this pops up a box prompting the user to enter their name and a message to send to the user, notifying them that an item status has changed.

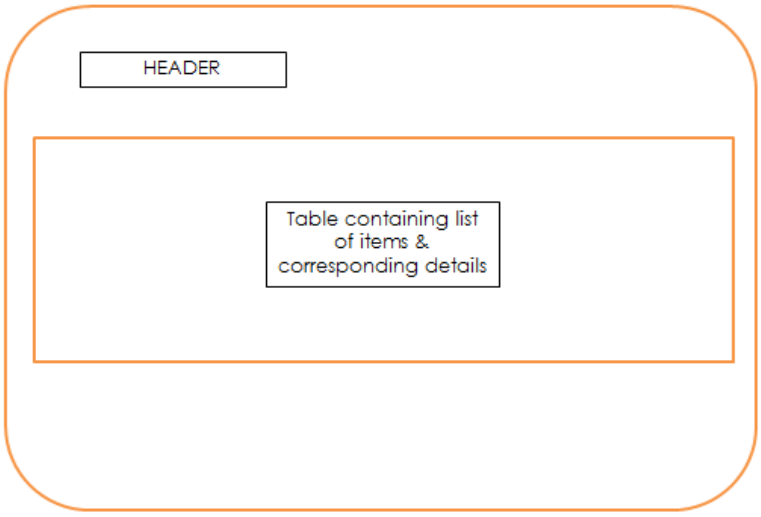


Figure 15 – Wish-list Page

3.4 – Navigation Storyboards

Using the system interface design and application architecture the developer created the following storyboards to demonstrate the functionality of the app.

When the user first opens the app they will be required to login via Facebook. Once successfully done they are returned to the application and faced with a list of their upcoming events.

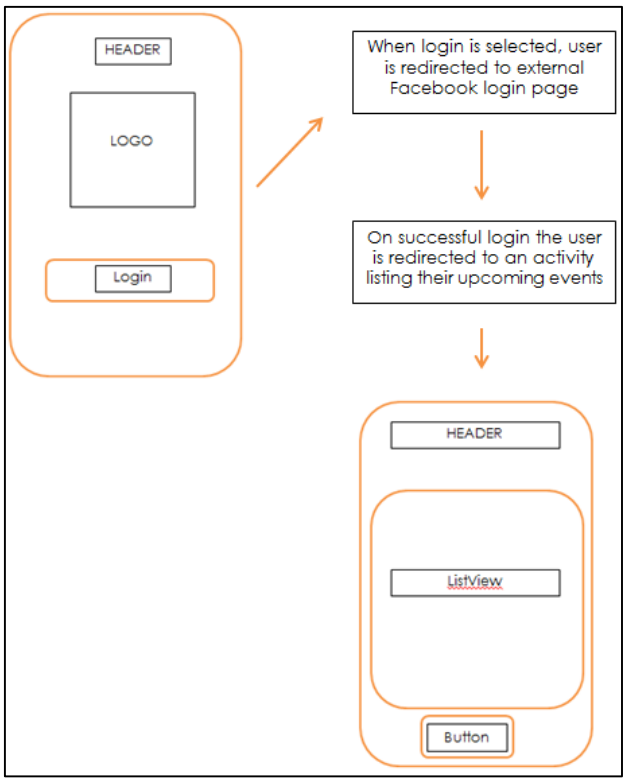


Figure 16 - Login Storyboard

The user then has the option to either create a new event, by clicking the button at the bottom of the screen, or view a specific event by selecting it from within the scroll list of events.

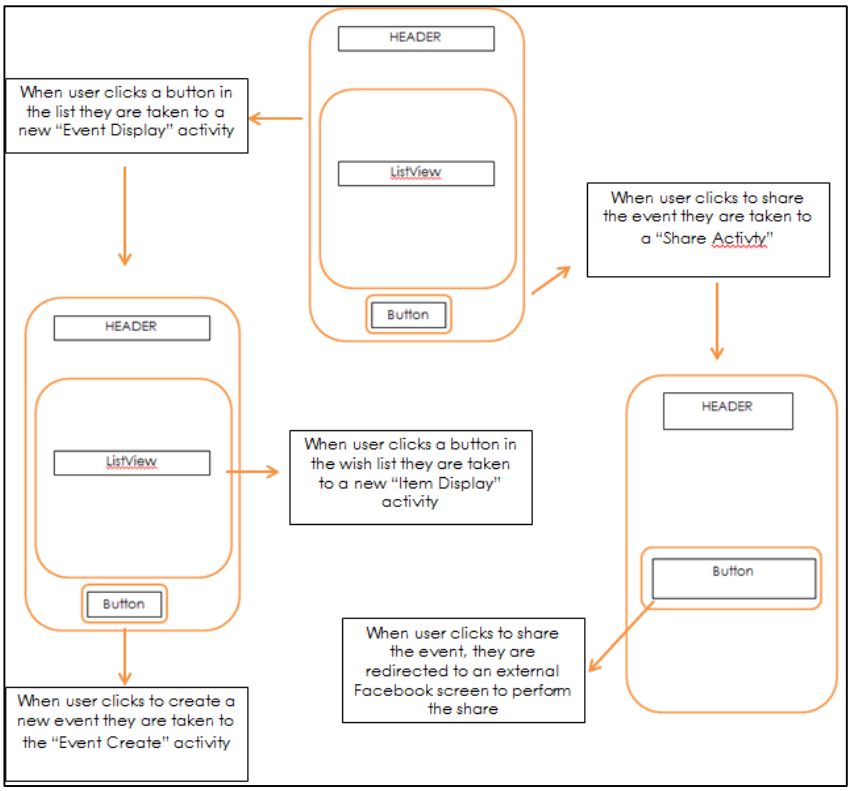


Figure 17 – Navigation post-login

When viewing a particular event the user is faced with the associated wish-list of items – which will be coloured to reflect their current status. The user has the option to view an item in more detail or share the event to their Facebook timeline. In the "Item Display" activity the user can choose to view the item at source in a browser or delete the item from the list if they no longer want it.

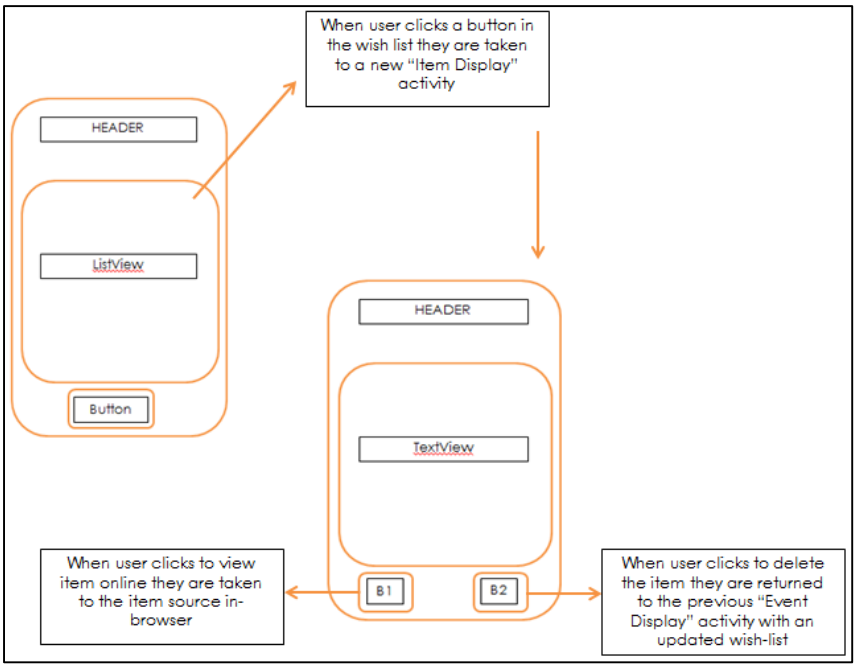


Figure 18 – Item Display Activity

When directed to the "Event Create" activity the user then enters the required event details and saves the event. Once saved, the user is returned to their "Event List" activity and given a notification that the event was successfully saved.

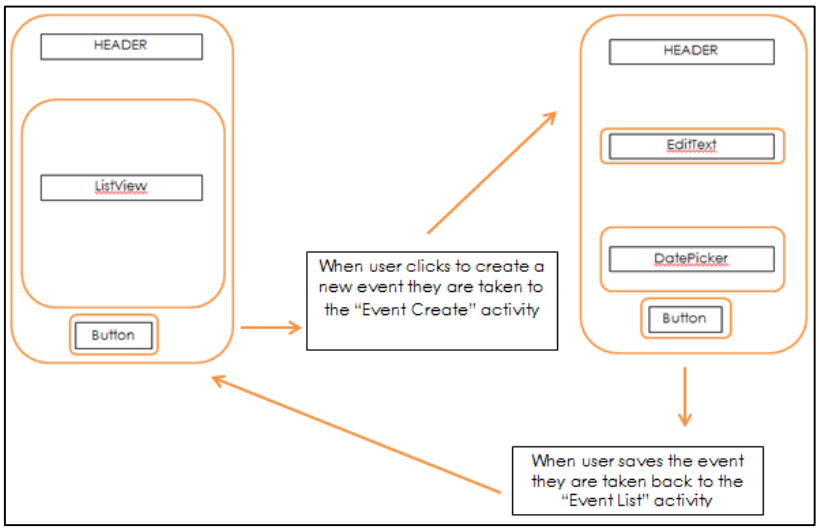
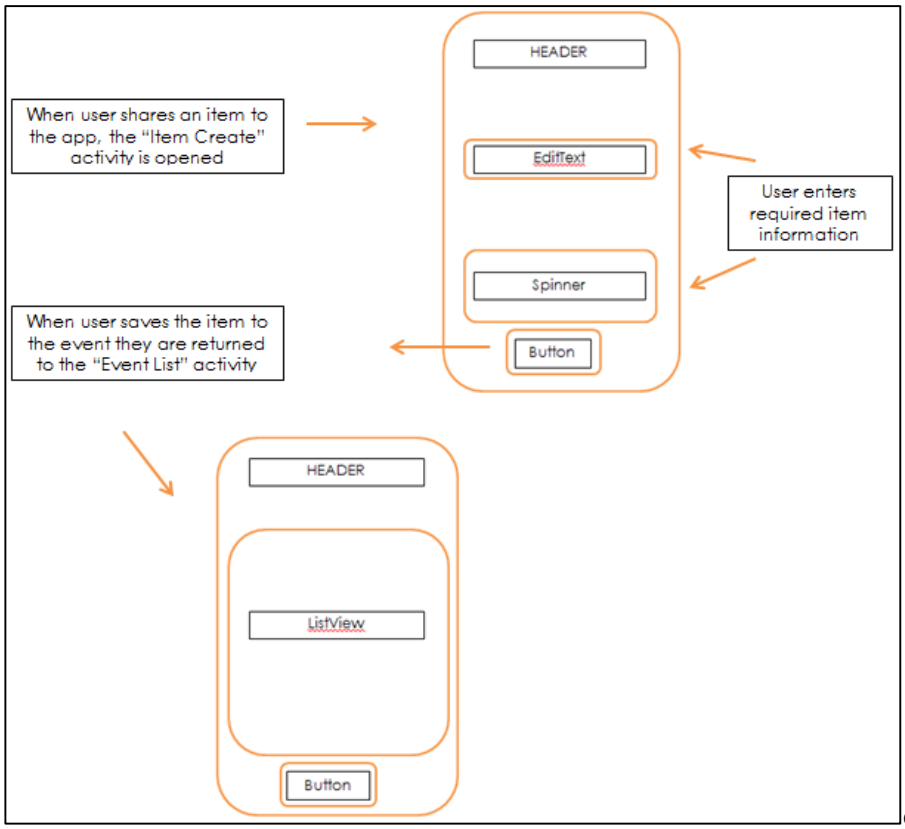


Figure 19 – Event Create Activity

When an item is shared to the application from an external application or browser, the "Item Create" activity is loaded. It retrieves the item description and displays it within an EditText widget, allowing the user to modify it if they want. After selecting the event they want to save it to from the spinner the user is returned to their list of events, whereupon they can select the event to view its updated wish-list.



C

Figure 20 – Item Create Activity

3.6 – Data Design

The application was developed using development tools provided by Google – namely App Engine (Google Cloud Platform, *App Engine*). The developer decided to move forward in this way for a number of reasons –

- Integration between platforms and services ensures that work is easy to share between them
- App Engine provides developers with the ability to easily implement various application programming interfaces (API) – for example the Google Cloud Datastore used to store data for the system at hand
- Using Google's pre-existing infrastructure means time is saved searching for and maintaining host servers, allowing the developer to devote more time to developing the system itself
- Worries about security are diminished thanks to Google's rigorous enforcement of safeguards
- It is free to develop with App Engine; a generous data threshold is reset daily and charges are only incurred in the event of massive data-flow
- Using Google's architecture provides an increased performance and reliability
- Applications are scaled automatically to suit needs so you only pay for what you use

While App Engine offers a number of benefits it is important to note that the developer's decision to use Datastore required becoming familiar with the intricacies of the platform. For instance, Datastore is a NoSQL database and, as such, works slightly differently than a traditional relational database (Google Cloud Platform, *Cloud Datastore*). Due to this, the terminology used by Datastore to describe data and data relationships is different than that used in SQL databases. These have been highlighted in the following table alongside their SQL equivalents for comparative purposes –

	Google Datastore	**Traditional Database**
Object Category	Kind	Table
One Object	Entity	Row
Unique Identifier	Key	Primary Key
Individual Data	Property	Field

Table 4 – Summary of key difference in data terminology (Google Developers, 2013)

As we can see the differences are not wholly bewildering. For example instead of referring to a USER table it will instead be described as a KIND, where one object of this category represents an ENTITY. This entity will have several properties, with its unique identifier described as a key – for example user ID. Given this, the developer has designed a data model to outline the relationship between entities in the proposed system.

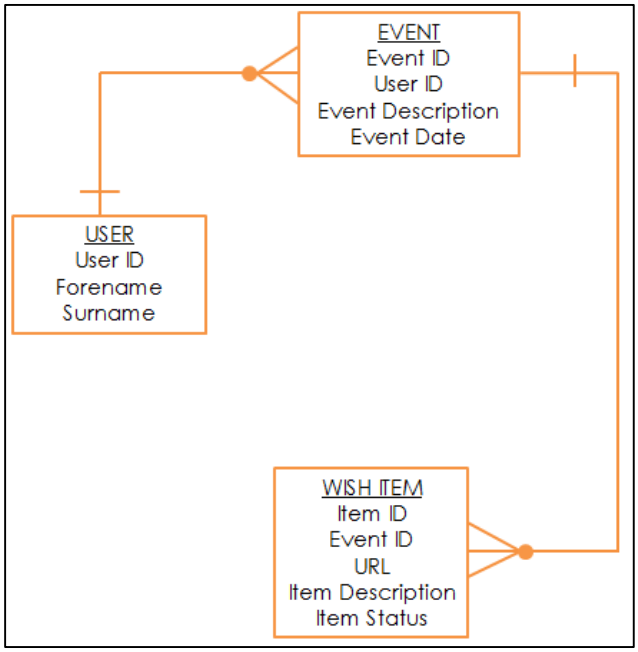


Figure 21 – Data Model

As mentioned above, Datastore uses a key to uniquely identify an entity instead of a primary key. Each of the entities illustrated in the above data model have been described below, alongside their corresponding properties and Java language equivalents.

Entity	Property Name	Type	Java Type
User	User ID	Key	`String`
	User Forename	Property	`String`
	User Surname	Property	`String`
Event	Event ID	Key	`Long`
	User ID	Property	`String`
	Event Description	Property	`String`
	Event Date	Property	`Date`

Item	Item ID	Key	`Long`
	Event ID	Property	`Long`
	Item URL	Property	`String`
	Item Price	Property	`Long`
	Item Description	Property	`String`
	Item Status	Property	`int`

Table 5 – Table outlining entity description

3.7 – Summary

This section of the report detailed both the design of the system architecture and the user interface. This was an important stage before moving on to the implementation of the system in order to allow the developer to clearly understand the workings of the system and how it should appear to users. The developer made reference to Shneiderman's much revered rules of interface design in their attempt to plan an accessible, user-friendly application that would meet the requirements specified in the previous section of this report.

4 – Implementation

4.1 – Introduction

After designing the system in accordance with the system requirements determined from competitive analysis and focus group research the next stage involved the implementation of the system itself. Thanks to the comprehensive design process outlined in the previous section – alongside a thorough understanding of App Engine, Datastore and Android development – the system was efficiently implemented at a steady rate of progress. The following chapter covers this implementation procedure by examining the implementation environment, the implementation process and finally a look at key functionality and Java resources within the system.

4.2 – Implementation Environment

Having initially begun implementation with Android Studio, some early problems with this particular development platform forced the developer to move forward instead with the Eclipse IDE. As mentioned previously, the decision to develop using Google App Engine meant the developer did not have to worry about having to choose a web host for their database thanks to the Datastore API. The application itself was developed in Java on a HP Pavilion laptop with an Intel Core i3-5157U processor, 8GB RAM on the Windows 10 operating system.

4.3 – Implementation Process

The system was developed based on the waterfall model. The developer first determined system requirements by analysing the problem to be solved and identifying the solution to it via thorough market research including competitive analysis and focus group feedback. Following this the developer then carried out a detailed design of the system in response to these requirements before moving on to implement the application.

In order to become adept with using the required tools for developing the system the developer found it essential to complete a number of tutorials. Pertinent documentation and tutorials were found online through a number of developer sites provided by Android, Facebook and Google. These were invaluable as they allowed the developer to enhance their knowledge of the resources needed, including integrating Facebook login to their application, posting to timelines from within the app, establishing communication between the application and Datastore, saving and modifying information within Datastore from the system and enabling the application to receive shares from external sources like the Amazon Shopping app and similar alternatives.

4.4 – Key Functionality & Resources

The creation of logical and efficient Java classes in order to allow desired functionality was pivotal during system development. Equally important was the use and maintenance of existing Android-related resources, most importantly the application manifest, in creating a functional system. What follows is a discussion of key functionality and associated Java resources implemented in the application.

4.4.1 – Manifest & Default Resources

An application manifest is crucial in the successful implementation of a system. The manifest is integral in running applications on the Android platform and must be maintained and updated with care throughout development in order to permit a successful deployment of the system. Some necessary additions made to the manifest during the course of implementation include adding support to enable the use of the Facebook SDK so the developer could integrate necessary login and share functionality, as well as setting up the application to receive shares from external browsers and applications. These are discussed in more detail in the relevant sections of this report and are shown alongside the updates made.

Other important sections within the project structure include the `src` folder. This contains Java classes holding the source code required to allow the system to perform required tasks. Another important directory is the project's `res` folder. This contains objects used primarily in the creation and upkeep of Android application activities, for example the `layout` sub-

directory contains all activity screens used within the system, while the `drawable` folder contains images used in the system. There is also a `strings` xml file which is used to store default text, such as button text and prompts used in the app.

4.4.2 – Entities & Endpoints

After creating a new project in Eclipse, the first thing implemented by the developer was a series of Java classes representing entities. These are simply Java object classes containing the basic structure for entities to be used within the system. These classes are important because they are used to generate Google Cloud Endpoints – which are required for storing entities within Datastore. For example, the event entity class contains variables used to define the structure of an event entity as outlined in the data design section of this report. Additionally, the class contains get/set methods in order to set the information for new events and retrieve the corresponding details for existing ones. Each entity class follows this similar structure, and in order to generate a corresponding Cloud Endpoint Class the developer simply right-clicks on the class in the project explorer and selects the "generate" option as illustrated in the following image –

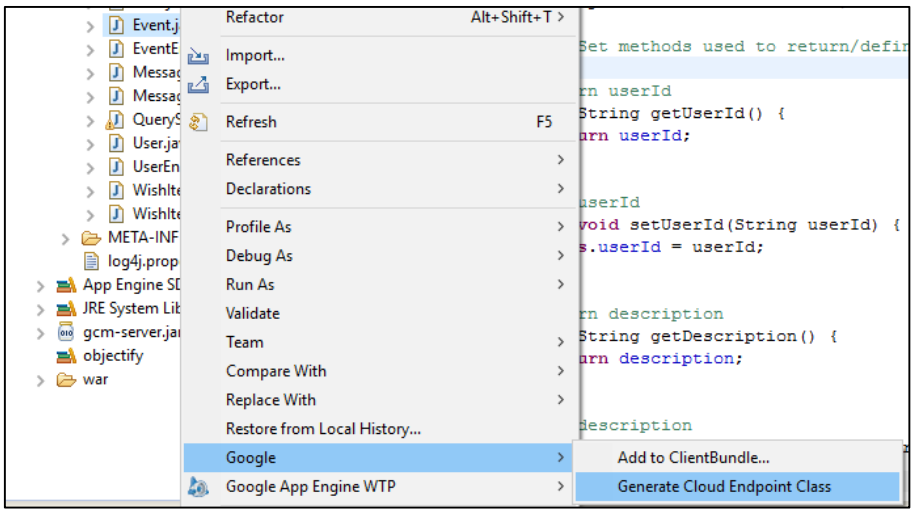


Figure 22 – Generating an Endpoint Class

The generation of an Endpoint Class creates a new Java class containing auto-generated code which supports operations that can be performed on entities within Datastore. Once an entity class has been created and finalised a corresponding endpoint class must be generated to ensure that Datastore can handle incoming queries on stored entities. The creation of entities and endpoint classes in this instance represents the successful implementation of a data storage platform with which the system can communicate with by requesting existing information or sending new data to it.

4.4.3 – Creating an Event Entity

Having created the entity class and generated the corresponding endpoint it was now possible for the developer to begin the process of allowing users to create entities. To create an event the user is simply prompted to enter necessary details and, once the information has been entered, the user creates this event by pressing a button which executes a method called

`addEvent()`. This method calls a subclass which allows communication between the application and Datastore to occur.

In this subclass the work is done which allows the newly created event entity to be saved in Datastore. This is done asynchronously in the background in order to minimise resource use on the device. The following code shows the initialisation of a new endpoint builder which will allow the user to store this event within Datastore. It then initialises a HTTP request to support communication between the app and Datastore, before creating a new endpoint to hold the information of the new event.

```
// Class used to asynchronously create & store the new event
public class EventEndpointsTask extends AsyncTask<Context, Integer, Long> {

    // Execute in the background
    protected Long doInBackground(Context... contexts) {

        // Create a new endpointBuilder that prepares for a new entity
        Eventendpoint.Builder endpointBuilder = new Eventendpoint.Builder(
                AndroidHttp.newCompatibleTransport(), new JacksonFactory(),
                new HttpRequestInitializer() {

                    public void initialize(HttpRequest httpRequest) {

                    } // initialise

                }); // endpointBuilder

        // Build the new event
        Eventendpoint endpoint = CloudEndpointUtils.updateBuilder(
                endpointBuilder).build();
```

Figure 23 – Code showing preparation process for new event entity

In order to save the event created by the user, the details must be assigned to a new event entity. This is done by creating a new instance of the Event class and setting the various details. In order to check for a successful execution the developer assigned the new event ID to a variable and logged this. The application context is then updated before an intent is used to pass the user to a new activity showing a list of their events, as below –

```
        try {

            // Set the event details
            Event event = new Event().setDescription(editTextDescription
                    .getText().toString());
            event.setEventDate(new DateTime(
                    getDateFromDatePicker(datePicker)));
            event.setUserId(((WishListApplication) getApplication())
                    .getFacebookId());
            Event result = endpoint.insertEvent(event).execute();

            // Log the event id
            Log.d("", "Event ID is " + result.getId());

            ((WishListApplication) getApplication())
                    .setEventIdentifier(result.getId());

            // Send user to the EventListActivity to view their updated event list
            Intent intent = new Intent(EventCreateActivity.this,
                    com.uniwishlist.EventListActivity.class);

            // Start the intent
            startActivity(intent);

        } catch (Exception e) {

            e.printStackTrace();

        } // try/catch

        return (long) 0;

    } // doInBackground

} // EventEndpointsTask
```

Figure 24 – Setting the event details & saving in Datastore

Set methods are used to assign the correct event details from the information passed via the corresponding widgets. For example, the event description is set by passing the value entered in the editTextDescription widget as a string. An additional method was created to return the date from the date picker widget and enable it for use.

4.4.4 – The User Entity

While not explicitly carried out by the user, the creation of a user entity is integral to the functionality of the system. The entity is structured to support a user ID, their forename, surname and their email address. These details are assigned within the application whenever the user logs in via their Facebook account. As described in the section detailing Facebook functionality, the Graph API is used to access this user information from their Facebook profile.

In the system, the user's ID is assigned the value of their Facebook ID upon login. By integrating this existing ID into the system, this ensures that the ID is unique to a particular user and is linked to their Facebook account. After the user has successful logged in, a method is used to assign their details to a JSONObject – passed as a parameter in this method header – by using get methods to retrieve the desired information. Following this, these details are added to the global application object –

```
// Here we have all the details to
// create a new event
// We then add the user's name and
// Facebook ID to the global
// application object
((WishListApplication) getApplication()).setFacebookId(object.getString("id"));
((WishListApplication) getApplication()).setFacebookName(object.getString("name"));
```

Figure 25 – assigning user details

4.4.5 – Sharing an Item

Another key function of the system is the ability to create an item entity and save it to a specified event wish-list. The first step in allowing this was to add intent filters to the application manifest in order to support shares from external applications and browsers.

```
<intent-filter>
    <action android:name="android.intent.action.SEND" />

    <category android:name="android.intent.category.DEFAULT" />

    <data android:mimeType="image/*" />
</intent-filter>
<intent-filter>
    <action android:name="android.intent.action.SEND" />

    <category android:name="android.intent.category.DEFAULT" />

    <data android:mimeType="text/plain" />
</intent-filter>
<intent-filter>
    <action android:name="android.intent.action.SEND_MULTIPLE" />

    <category android:name="android.intent.category.DEFAULT" />

    <data android:mimeType="image/*" />
</intent-filter>
```

Figure 26 – Manifest share update

The filter works by first preparing the application to receive a share intent from external sources. The category is set to default in order to send the user to the correct activity via an implicit intent. Finally we prepare the application to receive the data. When an item is shared, the user is passed to an activity which allows them to edit the item details and save it to an event of their choice.

4.4.6 – Creating an Item Entity

Having been passed to this new activity the corresponding Java class must deal with the type of share that has been made to the app. This is done at the beginning of the class via the following code –

```
// Get intent, action and MIME type to deal with external share
Intent intent = getIntent();
String action = intent.getAction();
String type = intent.getType();

// Handle the share
if (Intent.ACTION_SEND.equals(action) && type != null) {

    if ("text/plain".equals(type)) {

        // Handle text being sent by passing it to method below
        handleSendText(intent);

    }// if

} // if
```

Figure 27 – Handling a share

A new item entity is created and saved to Datastore in much the same way as an event entity. Via the use of a background endpoints task, a new item entity is created and the details provided by the user are assigned to this new item before being saved. Additional background work is done here in order to provide the user with a selection of events to save the item to. This is done by querying Datastore to return a list of events with a user ID matching that of

the user currently logged in. These events are saved into an ArrayList before being added to the spinner widget. Once the item has been shared, the description has been modified to suit and an event has been picked, the details are assigned to the item via an endpoint builder and it is saved within Datastore.

4.4.7 – Facebook

Before being able to implement Facebook-related functionality a number of steps had to be taken to ensure it could be done so effectively. The developer was first required to register as a Facebook developer and download the necessary SDK containing required support libraries. Following this an application ID was required for use within Facebook as well as a key hash that linked the application with the developer profile.

4.4.8 – Facebook Login

Having completed initial setup of the application with the Facebook developer site the developer began implementing Facebook login within their application. A tutorial provided on the site outlined the process and required updating the application manifest to support this functionality. The modification is seen below

```
<activity
    android:name="com.facebook.FacebookActivity"
    android:configChanges="keyboard|keyboardHidden|screenLayout|screenSize|orientation"
    android:label="@string/app_name"
    android:theme="@android:style/Theme.Translucent.NoTitleBar" />
```

Figure 28 – Manifest update to support Facebook login

Subsequently the developer created a new activity allowing Facebook login that made use of several classes from the SDK, including AccessToken – used to make Graph API requests in order to retrieve information from a user's Facebook profile. When a user successfully signs in via Facebook, a new GraphRequest is executed using their access token. Their Facebook user ID is requested and is used to initialise the ID of the user entity to this same unique value. This is done within the following onSuccess method from the activity's Java class –

```
public void onSuccess(LoginResult loginResult) {

    // Initialise accessToken with the current token
    accessToken = loginResult.getAccessToken();

    // Initialise a new GraphRequest to request details of
    // the user logging in
    GraphRequest request = GraphRequest.newMeRequest(
            accessToken,
            new GraphRequest.GraphJSONObjectCallback() {

                // Execute after a successfully completed
                // request
                @Override
                public void onCompleted(JSONObject object,
                        GraphResponse response) {

                    try {

                        info.setText("User Name: "
                                + object.getString("name")
                                + "\nUser ID: "
                                + object.getString("id")
                                + "\nAuth Token: "
                                + accessToken.getToken());

                        // We then add the user's name and
                        // Facebook ID to the global
                        // application object
                        ((WishListApplication) getApplication())
                                .setFacebookId(object
                                        .getString("id"));
                        ((WishListApplication) getApplication())
                                .setFacebookName(object
                                        .getString("name"));
```

Figure 29 – onSuccess method for successful login

4.4.9 – Facebook Timeline Share

Another key piece of functionality involving the Facebook SDK allows a user to share an event wish-list to their Facebook timeline. In order to support this, another modification was made to the manifest –

```
<provider
    android:name="com.facebook.FacebookContentProvider"
    android:authorities="com.facebook.app.FacebookContentProvider1005742186188729"
    android:exported="true" />
```

Figure 30 – Manifest update for Facebook Share

Another Java class was created containing the code to perform the timeline share operation. Whenever a user selects an event to share to their timeline the details of this event are passed via an intent to this share activity. The event details are loaded into a Facebook ShareDialog, which is implemented by the SDK. When the user presses the button to share an event wish-list to their timeline, the following method executes –

```
// Method executes a share to user's timeline
public void doMyShare(View view) {

    // If the shareDialog can show the content...
    if (ShareDialog.canShow(ShareLinkContent.class)) {

        // ... set the content
        ShareLinkContent linkContent = new ShareLinkContent.Builder()

                .setContentTitle(description)

                .setContentDescription("Click here to see my wish list")

                .setContentUrl(

                        Uri.parse("http://uni-wish-list.appspot.com/list.html?eventId="
                                + id)).build();

        // Show linkContent in shareDialog
        shareDialog.show(linkContent);

    } // if

} // doMyShare
```

Figure 31 – Facebook timeline share

This code populates a Facebook ShareDialog with the necessary event information, at which point the user then specifies who can see the post before sharing it to their timeline.

4.4.10 – Servlets & HTML Page

In order to allow contacts to view a wish list online the developer was required to create two servlets – a QueryServlet and a ReservationServlet – and a HTML page to display the corresponding information in an easy to understand manner.

The QueryServlet uses HTTP requests to query information from Datastore and retrieve necessary information. The servlet is passed an event ID which is used to request the items associated with this event from Datastore. This is done by the servlet as it loops through a full list of items within Datastore, checks for a corresponding event ID before placing the matching items into a JSONArray. JavaScript contained within the HTML page used to view the wish-list presents these items as a readable series of desired gifts instead of confusing JSON objects which are meaningless to end users.

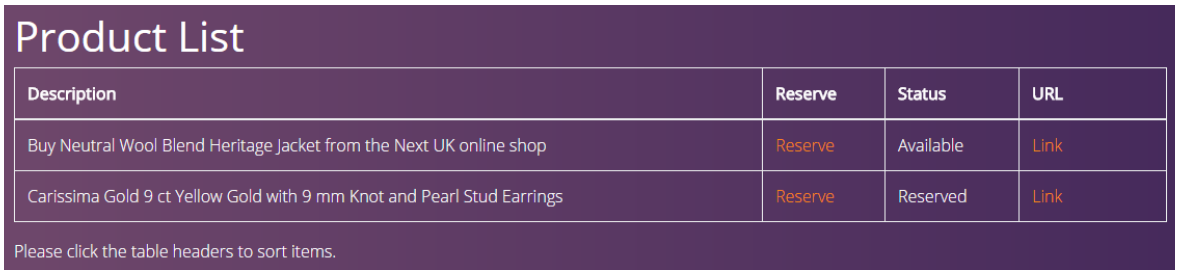


Figure 32 – HTML view of a wish-list online

The ReservationServlet works in a similar manner, while adding the functionality that allows contacts to reserve an item from the list. Once the list has been returned and formatted within the HTML, contacts can select an item to reserve. The ReservationServlet then queries Datastore to update the

status of an item which has been reserved. This in turn updates the list and allows those who have access to it to see which items have been reserved.

4.4.11 – Google Cloud Messaging

In an attempt to provide meaningful feedback and enhance the personal experience of using the system, the developer implemented the use of the Google Cloud Messaging service to send notifications to item recipients. When an item is reserved via the HTML page by a contact, they are prompted to enter their name and a message to send to the recipient.

Figure 33 – Item reserve dialog

This is done by manipulating the auto-generated GCMIntentService class and provides instantaneous feedback to the recipient via a notification from the app.

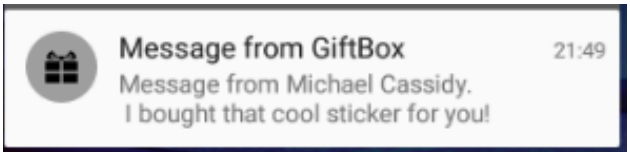


Figure 34 – Application notification

4.5 – Summary

The implementation of the system was the most time-consuming process as the developer needed to follow a number of online tutorials in order to become familiar with the tools being used. In particular, becoming familiar with Datastore and performing operations using this platform was difficult at times, however being able to develop and host the data storage and system together using secure, existing architecture provided by Google meant that this was worth it overall.

This section has analysed the implementation process of the system, beginning with the environment and methodology used before going on to look at some of the key functionality and resources used within the application.

5 – Testing & Evaluation

5.1 – Introduction

A pivotal stage during the creation and implementation of a software system, testing offered the developer the opportunity to ensure that the functionality of the system at hand matched the requirements set out in the corresponding section of this report. The following chapter outlines the steps taken during testing of the app – both in-house and end user testing – before analysing the results of these tests as a means of evaluating the system as a whole.

5.2 – Developer Testing

White-box testing was carried out by the developer in-house throughout the development of the system. This testing method allowed the user to ensure that the system was functioning correctly at various levels of development. Testing was carried out individually on new pieces of functionality – both before and after they were implemented within the system – in order to ensure they worked correctly separately before being integrated within the system. This led to various versions of the system being developed, each containing a new item of functionality that worked as desired.

Developer testing was predominantly carried out using a Sony Xperia SP smartphone as well as a Sony Xperia Z2 smartphone. Following the completion of the system a full system test was carried out. Known as black-box testing, this is done to certify that the full system functioned as desired. The results of the final system test are shown below –

Test Criteria	**Sony Xperia SP**	**Sony Xperia Z2**
Login	✔	✔
Create Event	✔	✔
Share event to Facebook Timeline	✔	✔
Share an item from external shopping application	✔	✔
Edit item details	✔	✔
Save the item to the event	✔	✔
Share an item from external browser	✔	✔
Edit item details	✔	✔
Save the item to the event	✔	✔
View wish-list online	✔	✔
Reserve an item	✔	✔
Delete an item	✔	✔
View updated wish-list	✔	✔

Table 6 – Developer testing results

5.3 – User Testing

Following a full self-evaluation of the system the developer felt it important to carry out external testing of the system with members of the focus group who had previously indicated their willingness and ability to partake in the testing procedure. This testing method does not require the tester to know the inner workings of the system so applies well to this case. The methodology was beneficial because it allowed testing to be carried out from a new perspective and ensured that a level of independent objectivity was achieved. In order to do this the developer created a brief test scenario to be completed by the users – with a template included in Appendix B of this report – as well as several feedback questions aimed at revealing how effectively the system responded to initial demands and requirements.

Of all tests carried out, functionality was a success for each test in each case. This was reassuring for the developer as it meant code had been implemented correctly and was functioning as desired. Following the test scenario users were asked to indicate how easy they found it to use the application. As the results show, the users found the app easy to navigate. This was encouraging for the developer as the design of an easy-to-use application was one of the biggest challenges to be overcome during the development of the system.

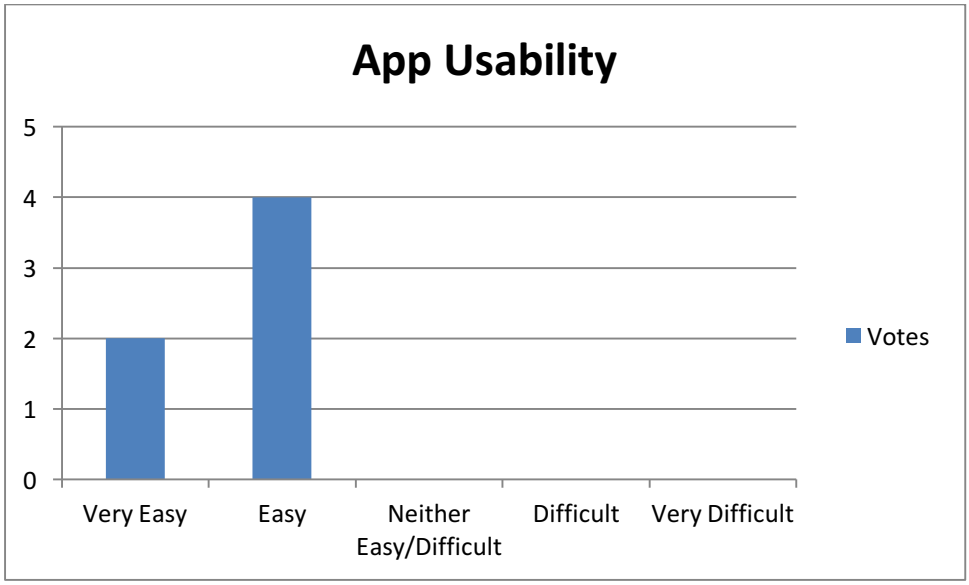


Figure 35 – App usability rating

Following this the users were asked to rate the accessibility of the system. Similarly the responses indicate that they found the system very accessible, with only one response indicating a sense of ambivalence.

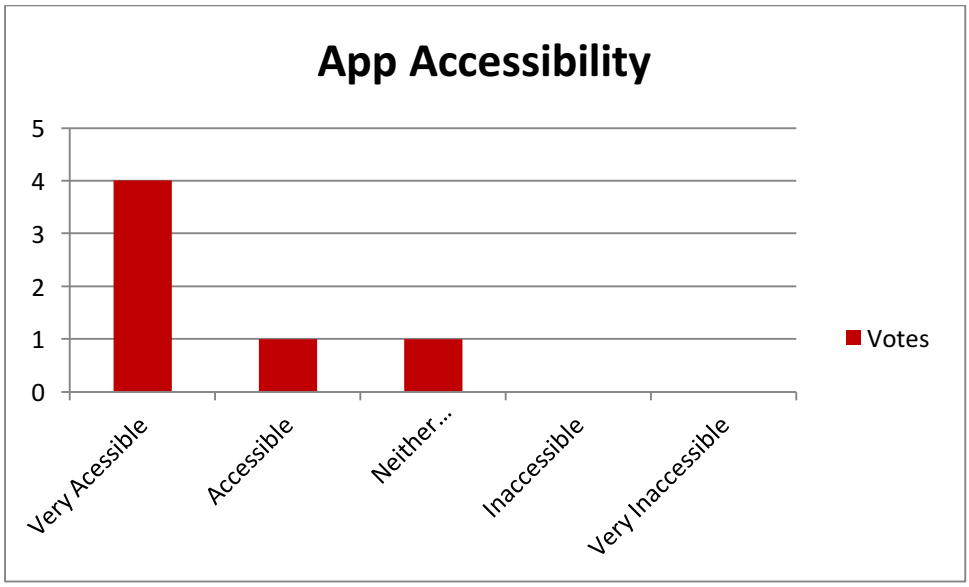


Figure 36 – App accessibility

App accessibility is another important consideration when developing mobile applications and the feedback indicates that the developer was successful in delivering an accessible app that is relatively easy to use.

The focus group was then asked to indicate how appealing they found the appearance of the system. Again positive results indicate that users considered the app to be highly appealing, confirming that the developer was successful in their attempts to provide an intuitive and user-friendly experience while using the system.

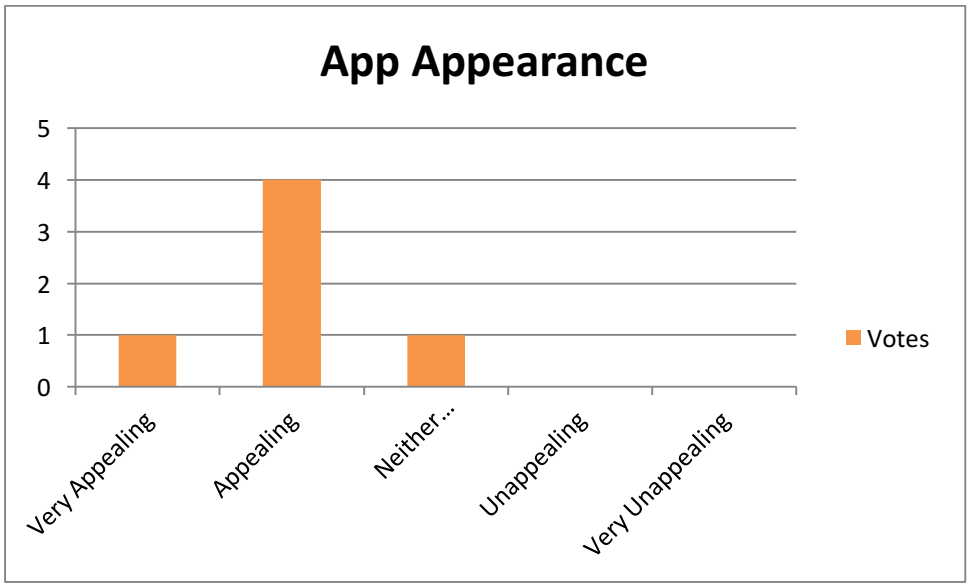


Figure 37 – App appearance

In order to gauge how the focus group felt about the system in its current form they were asked to rate the likelihood that they would personally download and use the system as it was presented to them.

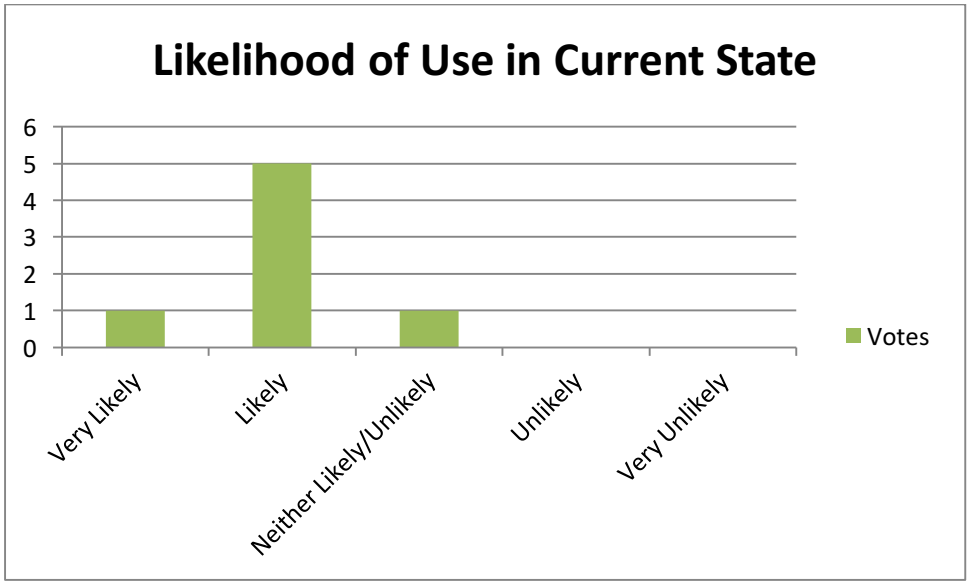


Figure 38 – Likelihood of use in current state

As the preceding graph illustrates the focus group indicated they would be very keen to use the app in its current form. When asked to explain their choice, the general consensus was that the app provided an intuitive and accessible way of managing wish-lists for upcoming events. Referring back to the first questionnaire completed for the analysis stage, comments reflected that the system achieves the goals that had been previously set out for it. This indicates a level of user satisfaction and fulfilment of previously established goals that is encouraging to the developer as it shows how initial aims have been achieved.

Similarly users were then asked to indicate how likely they would be to recommend the app for use to others in its current form. Again, feedback was overwhelmingly positive and gives the developer a sense of satisfaction for having developed an effective system that meets the needs of the target audience.

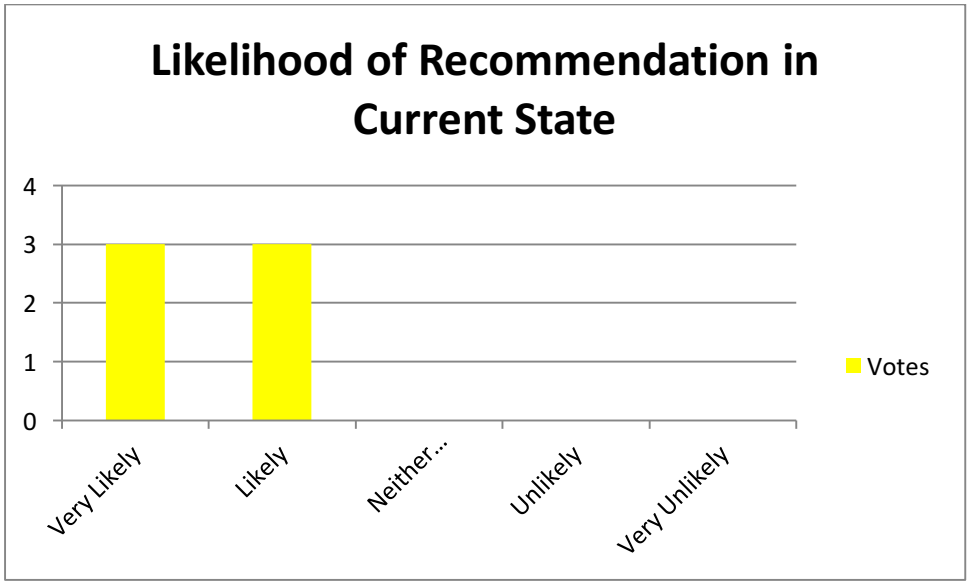


Figure 39 – Likelihood of recommendation

Users were finally asked for any additional comments or items of functionality they would like to see in future development of the system. Some responses included developing the system for integration with other social media platforms or devices, implementing the use of barcode scanning and even developing the application for standalone use. While this is only a brief summary of the suggestions, they have been developed further at a section later in the report dealing with recommendations for future development.

5.4 – Summary

This section has detailed the testing process for the system. It began by looking at testing carried out by the developer – ensuring functionality operated as required and testing on multiple devices – as well as analysing feedback and responses from external focus group testing. The irrefutably positive feedback gained from this process was encouraging for the developer as it allowed them to receive constructive real-world feedback

from target users. The positive reception of the system also highlighted the successful completion of objectives outlined at the beginning of the report – namely designing an intuitive, accessible system that allowed users to create and manage event wish-lists with ease and share them with their circle of contacts.

6 – Conclusions and Recommendations

6.1 – Introduction

This final section of the dissertation is concerned with reviewing the system as a whole and analysing its success in relation to the objectives specified during the analysis stage of the project. A brief summary of the project development will be followed by an assessment of how successful the developer has been in achieving these objectives. Finally the developer will provide some recommendations for further development if the system was to be enhanced via further development.

6.2 – Project Summary

Initial research was carried out during the analysis phase of this report aimed at looking at the current situation surrounding preparing for upcoming celebratory events. The developer found a number of difficulties faced by both parties concerned with upcoming events – namely gift recipients at the centre of festivities and those buying for them. This found that buyers have difficulties when choosing items to buy and deciding where to buy them from – online or on the high street – among other stresses such as whether or not the recipient will like the item and ensuring they don't already have it. Recipients on the other hand also found it difficult in providing gift ideas to their circle of friends and family.

After researching the current problem and carrying out further analysis via feedback gained from a focus group, the developer began to design a system to solve the problem at the heart of this report – one that would allow

recipients to generate a wish-list of items for an upcoming event that enables those they share it with to see what they want, which would also update to reflect the status of items on the list. The design of the application was informed both by this feedback and by previously described guidelines set out by Ben Shneiderman which promote the development of intuitive, accessible and user-friendly applications.

Following the design phase the developer began implementing the system itself. This began by completing tutorials online and reading relevant documentation that allowed the developer to become adept with the development tools described throughout this report. The system was implemented at a steady rate of progress, resulting in the development of a prototype which was brought back to willing members of the focus group for testing. Having carried this out, feedback was taken on board and used to inform the latter section of this report. The system was scrutinised against its original aims and a number of conclusions were drawn up, with recommendations found at a later point of this chapter.

Thanks to effective time management the various stages of the project occurred in a manner that allowed any unforeseen errors to be dealt with accordingly. This was the case when initial attempts to store entities within Datastore became problematic when trying to query them back for display within the system. The developer was able to pinpoint the problem – arising due to the use of generated entity keys – and solve it by simply adding verification code to an auto-generated class that resolved the error in a timely manner.

6.3 – Objectives & Achievements Revisited

Assessing the completed system against the initial aims and objectives is a key aspect of project management as it allows the developer to show how requirements have been met and, if necessary, what changes have been made during the process.

Analysis of the problem carried out by the developer during the initial research enabled them to clearly understand the current situation. Personal experience as well as feedback received from a focus group highlighted some of the main problems faced by those planning for upcoming celebratory events. This resulted in the development of a problem statement which formed the core focus of the project.

Market analysis was carried out in an attempt to identify contemporary solutions to the problem and any gaps in the market that could easily be filled. This was done by selecting several well-known wish-list organisers and carrying out a competitive appraisal of each in order to determine both their positives and their shortcomings as a way of informing the solution to be developed as part of this process.

Following competitive research, a focus group was used for the developer to receive feedback from real-world users about their difficulties preparing for upcoming events and what they are looking for in a wish-list organiser. This resulted in the development of key system requirements used to shape the development of the system. A prototype system was subsequently developed and tested which was taken back to several members of the focus group for external testing and further feedback. After

having been tested in a simulated scenario the system was evaluated against the original project aims and the results of this and accompanying feedback was used to inform the content of this chapter.

Given this, alongside the fulsome feedback from the focus group, the developer can say that the project safely achieved the aims and objectives set out at the beginning of this report.

6.4 – Recommendations

During development of the system it became clear to the developer that there were a number of potential features for the application which could be implemented during a further period of expansion. A list of these follows –

- Barcode scanning – this would be a useful feature for use when items are found in isolated or bespoke shops that are difficult to access or offer no online source, or even if users come across an item they would like while on their travels. Users would simply scan the barcode and this would enable them to save the item to an event in much the same way as the current share function.
- Integration with other social media platforms – this would allow users to share their event wish-lists with friends and family via different networks, perhaps beneficial if they have contact with someone not using a particular network.
- Development for other devices – this would increase the potential market share for the application by allowing users with various devices to download and use the app. This could be done either by cross-

platform development or – while time consuming and expensive – develop the application for various target platforms.

- Ability to send personalised messages of thanks – while a generic notification of thanks appears when a user reserves an item, members of the focus group felt it would be a pleasant feature to support the use of personalised messages of thanks to gift-buyers.
- Development as a standalone app – the suggestion to develop the system as a standalone application would be beneficial in allowing those without Facebook to use the app and is a valid option that would require an alternative method of user accounts not linked to social media.

6.5 – Summary

Given the conclusions, recommendations and feedback presented in the report, the developer is pleased with the system which has been developed. The objectives outlined at the beginning of the report have been met. Project development was managed in such a way as to allow various deadlines and targets to be achieved and for the project as a whole to be completed in a timely manner that allowed for the resolution of any problems along the way.

References

Android Developers (N/A) *App Manifest*, available online at <https://developer.android.com/guide/topics/manifest/manifest-intro.html> [accessed 6/1/16]

Chaffey, Dave (2016) *Mobile Marketing Statistics compilation*, available online at <http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/> [accessed 7/6/2016]

eMarketer (2015) *UK Retail Ecommerce Sales to Reach £60 Billion This Year*, available online at <http://www.emarketer.com/Article/UK-Retail-Ecommerce-Sales-Reach-60-Billion-This-Year/1012963> [accessed 7/6/2016]

Google Cloud Platform (N/A) *App Engine*, available online at < https://cloud.google.com/appengine/> [accessed 11/1/16]

Google Cloud Platform (N/A) *Cloud Datastore*, available online at < https://cloud.google.com/datastore/> [accessed 11/1/16]

Google Developers (2013) *Datastore Introduction*, available online at < https://www.youtube.com/watch?v=fQazhzcC-rg> [accessed 11/1/16]

Office for National Statistics (2015) *5 facts about online retail sales in the UK*, available online at <http://webarchive.nationalarchives.gov.uk/20160105160709/http://www.ons.gov.uk/ons/rel/rdit2/e-commerce-and-internet-use/5-facts-relating-to-web-sales/sty-5-facts.html> [accessed 7/6/2016]

Shneiderman, Ben (2010) *The Eight Golden Rules of Interface Design*, available online at <http://www.cs.umd.edu/~ben/goldenrules.html> [accessed 11/1/2016]

Appendix A – Focus Group Questionnaire Template

Gift Wish-List App Questionnaire

As society becomes ever more mobile and the international community becomes increasingly globalised it is now common to find that families and groups of friends find themselves scattered across the globe. Consequently, at times of celebration it can be difficult to coordinate the purchase of gifts for those at the centre of festivities, while avoiding the infamous "ten toaster problem" – when several individuals of good intention inadvertently purchase the same item.

To solve this, an application is being developed – with the aim of easing the often stressful process of gift purchasing – which will allow recipients to generate a wish list of items which can be seen by specified contacts, who can then use it to determine what gift to buy, safe in the knowledge they are not duplicating the selection of another purchaser.

The following questionnaire is intended to gather feedback from potential real-world users in order to aid the development of this application. Thank you for your time in taking part in this process.

Q1. Please indicate how difficult you currently find it

a) As a guest, organising your gift-purchasing habits (i.e. buying in time, within budget)

	Extremely	Somewhat	Neither	Somewhat	Extremely	
Difficult	☐	☐	☐	☐	☐	**Easy**

b) As a recipient, trying to provide gift ideas for an upcoming event

	Extremely	Somewhat	Neither	Somewhat	Extremely	
Difficult	☐	☐	☐	☐	☐	**Easy**

Q2. Please indicate how beneficial a mobile gift wish-list app would be for you

a) As a guest purchasing an item for an upcoming event

Very Beneficial	Beneficial	No Difference	Not Beneficial	Pointless
☐	☐	☐	☐	☐

b) As a recipient aiming to generate a list of desired items

Very Beneficial	Beneficial	No Difference	Not Beneficial	Pointless
☐	☐	☐	☐	☐

Q3. Please rank the following problems associated with gift-buying based on personal experience *(1 being the biggest concern, 9 being the smallest concern)*

Not knowing what to buy ☐
Not knowing what the recipient wants ☐
Not knowing where to buy from ☐
Budget management ☐
Buying on time ☐
Worrying an item might already be purchased ☐
Worrying the recipient already has the item ☐
Worrying the recipient won't like the item ☐
Getting the best deal ☐
Other (please specify)

_____

_____

Q4. Please indicate which of the following gift wish-list services you are aware of and, if applicable, those which you have used and would recommend *(Tick all that apply)*

	Aware Of	Have Used	Would Recommend
Gift Planner	☐	☐	☐
Giftster	☐	☐	☐
iWishFor	☐	☐	☐
John Lewis: The Gift List	☐	☐	☐
mGifts	☐	☐	☐
Santa's Bag	☐	☐	☐

Other (please specify)

_____

_____

Q5. Please rate the importance of the following features for a gift wish-list app from 1 to 10 *(1 being most important, 10 being least important)*

Social Media Integration	☐	Search Filters	☐
Event Countdown	☐	Price Comparison	☐
App price	☐	Gift-sharing	☐
Ease of Use	☐	Item Status	☐
Event variety	☐	Device Compatibility	☐

Other (please specify)

_____

_____

Q6. Given that user accounts will be required to access and use the app please rate the importance of the following user account features from 1 to 10 *(1 being most important, 10 being least important)*

Purchase History	☐	Multi-payment Platforms	☐
Suggested Items	☐	Security	☐
Popular Items	☐	Budget Limit	☐
Loyalty Rewards	☐	Customisation	☐
Notifications	☐	Charitable Donations	☐

Other (please specify)

_____

_____

Q7. Please indicate the most suitable name for a gift wish-list app from the following options from 1 to 5 *(1 being most suitable, 5 being least suitable)*

"Get Me This" ☐

"GiftBox" ☐

"Gifter" ☐

"I Want That One!" ☐

"To Buy or Not To Buy" ☐

Other (please specify)

_____

_____

Q8. In the interest of maximising readability please indicate which of the following font styles seems most appropriate for an app

Sample Text	–	Century Gothic	☐
Sample Text	–	Arial	☐
Sample Text	–	Times New Roman	☐
Sample Text	–	**Impact**	☐
Sample Text	–	Comic Sans MS	☐

Q9. Please indicate if you would be willing to take part in a testing session if required *(Tick One)*

I am willing to be part of the testing process ☐

I am not willing to be part of the testing process ☐

Please leave your name for contact purposes if you are willing to participate in testing

_____

Please feel free to supply any additional comments, suggestions or recommendations with relevance to the gift wish-list app in question

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Appendix B - Focus Group Testing Form Template

System Test Scenario

Having previously indicated your willingness to take part in the testing procedure for the gift wish-list application, the following test scenario and accompanying feedback questions have been developed in order to receive constructive feedback on the developed system.

I would ask that you follow the subsequent steps in order and indicate whether or not you were able to successfully complete the tasks specified. Following this it would be of great appreciation to the developer that you answer the provided feedback questions as honestly as possible in order to give an accurate representation of your feedback for the system.

Thank you again for taking part in the earlier analysis questionnaire as well as this latest phase in the project – your time and cooperation are very much appreciated.

Test No.	Test Criteria	Success (Y/N)
1	Login via Facebook	
2	Create a new event	
3	Share the event to Facebook timeline	
4	Share an item from an external shopping application	
5	Edit the item details	
6	Save the item to the event	
7	Share an item from a shopping site via an external browser	
8	Edit the item details	
9	Save the item to the event	
10	View the wish-list online	
11	Delete an item from the wish-list	
12	Reserve an item from the wish-list	
13	View the updated wish-list	

Feedback Questions

Q1. Please indicate how easy you found it to use the app (1 being very easy, 5 being not easy at all)

1 2 3 4 5

Q2. Please indicate the accessibility of the app (1 being very accessible, 5 being not accessible at all)

1 2 3 4 5

Q3. Please rate the appearance of the app (1 being very appealing, 5 being not appealing at all)

1 2 3 4 5

Q4a. Please rate the likelihood that you would download and use the app in its current form? (1 being very likely, 5 being not at all likely)

1 2 3 4 5

Q4b. Would you mind briefly explaining your choice?

Q5a. Please rate how likely you would recommend the app in its current form (1 being very useful, 5 being not useful at all)

1 2 3 4 5

Q5b. Would you mind briefly explaining your choice?

Q6. For future reference, are there any additional features that you would like to see implemented in the system?

Q7. If you have any additional comments please feel free to record them below