COM814: Project 2015-16

Dissertation

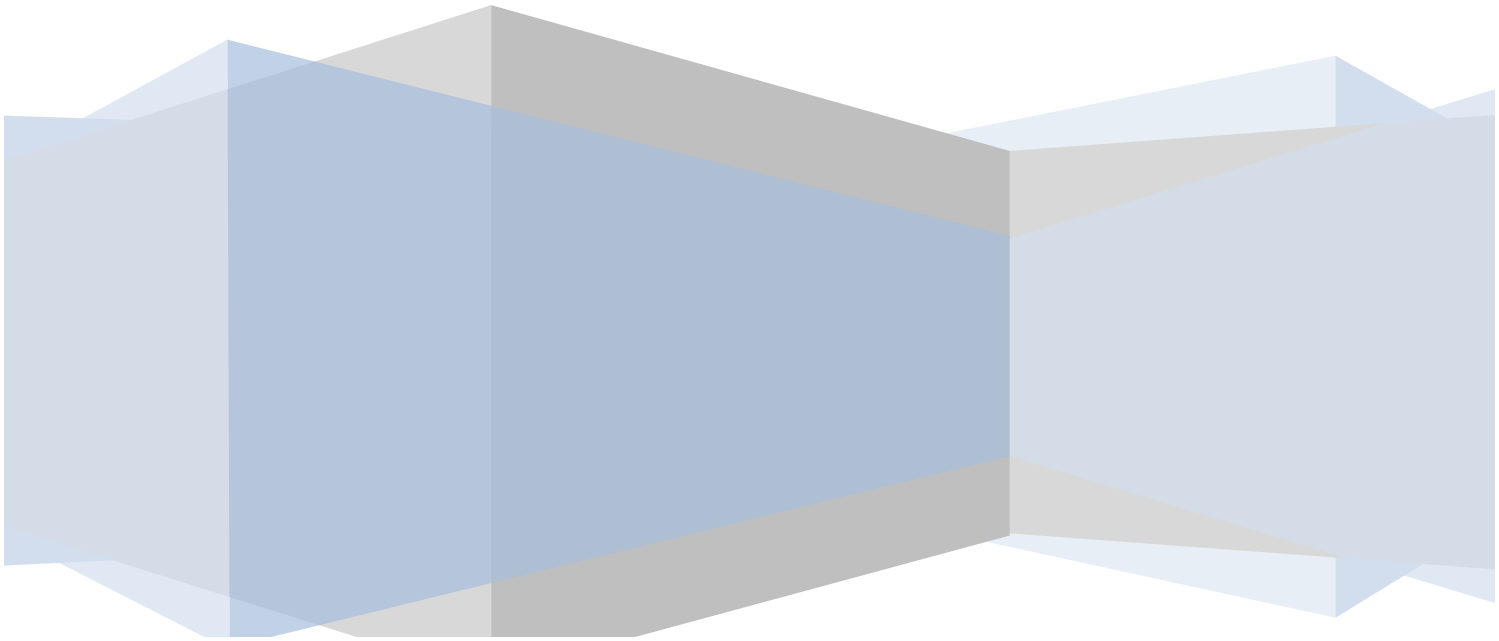
School of Computing & Information Engineering

Katherine McCann B00698218

An Android application to help prevent domestic food waste.

Supervisor Janet Allison
Second Marker Adrian Moore

10th October 2016



Plagiarism Statement

I declare that this is all my own work and does not contain unreferenced material copied from any other source. 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 sanctions 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.

Acknowledgements

Thanks to my supervisor Janet Allison for her time, feedback and assistance throughout this project. Also thanks to Charlie He and Adrian Moore for their help.

Thanks to the community groups who have kindly taken part in my research study and the application testing, and whose feedback was essential to the successful completion of this project.

Thanks to my family who have provided me with continuous support and assistance and to anyone else who has contributed directly or indirectly to the project, in any way.

Table of Contents

Table of Figures

Table of Tables

1.0 Abstract

It is estimated that the UK produces 15 million tonnes of food waste annually—UK households produce almost half of this. Avoidable household food waste has reduced by 21% since 2007 but the average UK household still throws away the equivalent of six meals per week. It is estimated that wasted food could cost each household £250- £400 per year. (Downing et al 2014) Household emissions associated with food waste represent around 14% of domestic emissions for an average household, making it the largest single contributor to total emission, larger than either electricity or gas alone. WRAP estimates that avoidable food waste is associated with of 0.62 tonnes CO2e per household per year, which compares to total emissions associated with consumption of approximately 37 tonnes CO2e. (SHIFTDESIGN 2014) When considering these statistics it is important to note that 1 in 5 people live in poverty in the UK. Many families are in crisis and cannot afford to feed themselves. Today, people are going hungry in their own homes. Rising food and fuel prices, static incomes, high unemployment and changes to benefits are causing many families to struggle to put food on the table. (TRUSSEL TRUST 2014) Food poverty, and a rise in food bank use, has brought the issue of food waste to Parliament's attention. Food waste is also being considered as part of the EU's revised circular economy package, due by the end 2015. This reflects the growing public interest in how much food is wasted and redistributed to those in need. (COMMONS PAPER 2015)

This dissertation describes the work undertaken in the development of a new Domestic Food Waste Manager application, Use Your Noodle, which is a free android application. It can be used to create a shopping list, a mobile list of the user's current groceries as well as a recipe wizard for the user to get inspiration for recipes that use their current groceries as ingredients. Use Your Noodle was developed as an android application for mobile devices by applying current standards for accessibility and usability to guarantee the best user experience for all adult users. Design and user requirements were identified by administering a questionnaire to a variety of community groups. A prototype was developed and the application implemented. The application was finally tested by the focus groups who responded to the initial questionnaire and recommendations for future development of the application were made.

Keywords: recipes, food waste, mobile application, android.

1.1 Introduction

These three issues can give an overview of domestic food waste; households are finding it increasingly more difficult to feed themselves within their budget and there is a large volume of domestic food going to waste; which is contributing to CO2 emissions.

Food waste has a range of social, economic and environmental implications. However, the complexity of the food chain means that sources of food waste can be difficult to identify and quantify accurately. The UK is leading the way in collecting robust food waste data but on-farm data needs to be improved and the Government has commissioned further research in this area. UK waste policy is mainly driven by the EU Waste Framework Directive and food waste is a devolved issue. In Scotland and Northern Ireland, regulations require food waste to be collected separately for anaerobic digestion rather than landfill. In England most food waste is still sent to landfill. (Downing et al 2015)

1.2 Problem Statement

Approximately a third of all food that is produced is wasted, including about 45% of all fruit and vegetables, 35% of fish and seafood, 30% of cereals, 20% of dairy products and 20% of meat. In wealthy countries, like the UK, there are significant levels of unintentional losses which involve food being thrown away by consumers because they have purchased too much, or failed to store it appropriately for future use. (THE GUARDIAN 2015)

The foods most commonly found in British bins are bread, vegetables, fruit and milk. (THE GUARDIAN 2015) Crucially, because a high percentage of food waste is avoidable, it has an unusually high potential for improvement. Equally, research suggests there is a greater appetite and ability for change around food waste, compared to other domestic energy saving activities. (DEFRA 2008)

The decision to design this project to create a mobile application is due to the mobile nature of the use of the application as well as integrating this into a device that most people now use and carry with them daily. This food waste manager app would aim to help in the case of domestic consumers, who unintentionally purchase or prepare too much food, or do not plan or store food in the best possible way for their lifestyle. This mobile application can inform users remotely of the produce that they have available, allow them to input their food purchases, provide recipes for meals that can be created using existing products as well as

targeted advice on food storage to fit in with their lifestyle. They will also be able to track their food spend and food wastage over time to help to inform them of where changes need to be made. This will all take into consideration the need for a healthy and balanced diet on a budget and in an environmentally friendly way.

The problem and solution can be summarised in the following problem statement:

"At a time of economic pressure for all, and within the scope of avoidable behaviours around reducing carbon emissions, a new mobile application would be beneficial for users to manage their food spend and waste. The resources included will aim to help users make informed food purchases, minimise their waste and provide trends for them to get feedback on their food usage/waste over time."

1.3 Aim

To successfully create a Domestic Food Waste Manager mobile application for the wider use of adults that is free.

1.4 Objectives

In order to achieve the aim defined above, the following objectives were decided upon:

1. Conduct background research on the problem and demonstrate why this mobile application constitutes an adequate solution.
2. Examine mobile applications currently available to manage food spend/recipes and identify their pros and cons.
3. Compose and administer a research questionnaire to various community/parents groups to establish what design, features, content and user requirements best suit this new Food Waste Manager application.
4. Create a design centred on adult users that meets the standard requirements for accessibility and usability and implement it.
5. Administer the testing of the application by the focus groups who have pre agreed to when responding to the initial questionnaire.

1.5 Requirements for Development

The app will be designed using an Android platform on IntelliJ IDEA Community Edition. This Java IDE provides all the necessary functional requirements in the app. It is recommended

that this application is used on mobile phone or a tablet. The reason for this is that a mobile device/tablet will already be carried by the user and many features are designed to be used while mobile e.g. while shopping.

The Android architecture includes the database platform SQLite built in to the library layer. Therefore this will be used to provide a database to store product information. The database will facilitate storing the user's individual shopping list, current food inventory and the recipes that they can use and favour.

1.6 Dissertation Outline

Chapter 2 presents a background research on the problem and provides arguments demonstrating why a new Food Waste Manager application constitutes an adequate solution. It concludes with the finding of a comparative analysis conducted on existing Recipe and Grocery Manager Applications, including their pros and cons.

Chapter 3 reports on the research questionnaire administered to the focus groups (consisting of parents and various community groups). It unveils the key findings on the application design, content, features and user requirements. It finally lists the main features and user requirements to be incorporated in the design.

Chapter 4 presents an overview of the application design. It discusses the design methodology followed. It describes the user interface and includes the prototype. It also contains the application map and the design of the relational database.

Chapter 5 reviews the implementation of the design. It presents the software, and programming languages adopted. It explains the processes employed to successfully create the first working version of this new application.

Chapter 6 analyses the test results produced by the developer and the focus groups who responded to the initial research questionnaire and pre agreed to the testing.

Chapter 7 is the conclusion of this dissertation. It summarises the work done and critically assesses the results. It states if the application has achieved to meet the focus groups' recommendations and satisfaction. It pronounces whether the project successfully achieved its aim and objectives. It ends by presenting any additional features and content, and possible improvement that could be made if the project was to be extended.

2.0 Background Review

2.1 Introduction

This chapter begins with an analysis of the scale of the problem of domestic food being unused and thrown away. It identifies possible causes and origins of the problem. It then presents the solution and provides some arguments to support it. It looks at mobile applications currently available for organising food shopping/recipe use and finally reviews the results of a comparative analysis.

To begin the design process for the proposed application, it is vital to acknowledge the current situation and availability of products in today's market. In doing so, ideas can be developed and the key requirements for the application can be measured and prioritised. Reviewing the current situation and analysing the need for specific requirements will form the basis on which the application will be developed. The initial stages of this review involved compiling rich picture diagrams to outline both the current situation and later the proposed solution.

This review will then look in more detail at the current problem, the scale and nature of domestic food waste in the UK finally will consider the target audience the accessibility of the solution as well as the specific content for the application. In relation to the current problem, a number of available mobile applications have been analysed. The effectiveness of each application will be considered and advantages and disadvantages of content and design will be discussed and evaluated. This will allow for ideas to be considered for the development of this new Food Waste Manager app.

2.2 The Problem

2.2.1 Scale of the Issue

Food waste is a major environmental problem both globally and in the UK. The global carbon footprint of food waste per year is 3.3G tonnes CO_2, and if this was compared to the total carbon footprint of whole countries, it would be third after the total carbon footprints of US and China. The greenhouse gases (CO_2, as well as methane CH4 and nitrous oxide N2O) produced in the production, transport, storage and decomposition of food, are significant: food and drink waste in the UK causes 900 million tonnes CO2 per year.

In order to address the problems associated with reducing food waste it is important to identify the sources of the waste.

The Food and Agriculture Organisation of the United Nations uses the following definitions of food loss, food waste and food wastage:

• Food loss: The decrease in edible food mass at the production, post-harvest, processing and distribution stages in the food supply chain. These losses are mainly caused by inefficiencies in the food supply chains, like poor infrastructure and logistics, lack of technology, insufficient skills, knowledge and management capacity of supply chain actors, and no access to markets. In addition, natural disasters play a role. (Downing et al 2015)

• Food waste: Food which is fit for consumption being discarded, usually at retail and consumer level. This is a major problem in industrialised nations, where throwing away is often cheaper than avoiding or re-using and consumers can afford to waste food. Accordingly, food waste is usually avoidable. (Downing et al 2015)

• Food wastage: any food lost by wear or waste. Thus, the wastage is here used to cover both food loss and waste. (Downing et al 2015)

Of the food and drink that is wasted, over half of it is generated in the home (7 million tonnes, or 19% of food bought) and of this, 60% (by weight) is deemed 'avoidable', as it could have been eaten at some point prior to throwing away. It is this domestic, avoidable food waste that this project is focussed on. (SHIFTDESIGN 2014)

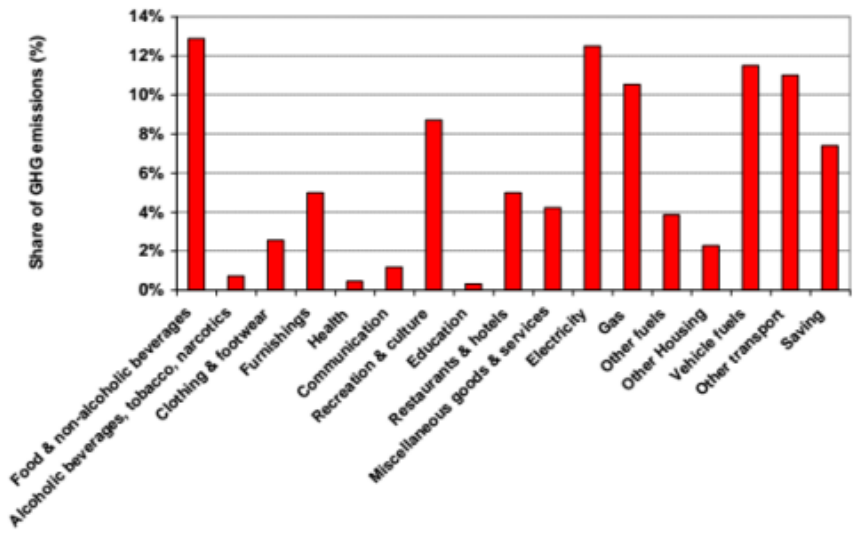


Figure 1.0 The share of total greenhouse gas emissions by category for an average household (SHIFTDESIGN 2014)

Avoidable household food waste has reduced by 21% since 2007 but the average UK household still throws away the equivalent of six meals per week. It is estimated that wasted food could cost each household £250-£400 per year. (SHIFTDESIGN 2014)

The foods we waste the most are fresh vegetables and salad, fresh fruit and bakery items such as bread. (LOVE FOOD HATE WASTE 2015)

There is a case to be made for reducing food waste, as opposed to encouraging alternative methods of disposal beyond landfill, as this has a greater potential impact in carbon reduction. One tonne of food waste prevented leads to 4.02 tonnes of CO_2 avoided, whereas one tonne of food waste diverted from landfill to anaerobic digestion leads to just 0.5 tonnes of CO_2 avoided. (SHIFTDESIGN 2014)

2.2.2 Nature of the Issue

57% of the UK population claim not to waste food (or very little) but in fact as a nation we are throwing away millions of tonnes of food and drink from our homes every year, and most of this is avoidable waste, that could have been eaten. A quarter of the same population think that the food we throw away doesn't cost much, the fact is that food waste is having an economic impact on these households – up to £400 a year for the typical household. (Downing et al 2015)

In order to reduce CO_2 emissions resulting from food waste firstly, the behaviours which lead to food waste and secondly, the context that causes these behaviours, need to be understood. (Downing et al 2015)

Our behaviour (both around food waste and more generally) can be seen to be the result of the myriad of pressures and influences which impact on our lives. The emotional context in which decisions are made and the influence of peers, family members, the media and institutions in our lives is crucial. (SHIFTDESIGN 2014)

The main causes of consumer/household waste include incorrect storage; buying too much, plate scrapings and food preparation waste. In addition, food waste has been attributed to leftovers, confusion over food labelling and Buy One Get One Free (BOGOF) offers. BOGOF offers were thought to contribute to food waste as stated by the House of Lords European Union Committee. The report said retailers were able to "pass on" food waste "from the store to the household" by the use of special offers such as "buy one get one free". (Downing et al 2015)

People do not think a lot about food waste, but they do think about individual activities that reduce and increase food waste, although not primarily for reasons relating to food waste.

Appendix 1 illustrates a rich picture diagram that summarises graphically the problem and presents the different stakeholders.

2.3 Solutions to prevent food waste

It is clear that organising and running a home is fraught with pressures and tensions. There is a clear case for an approach that focuses on alleviating these pressures, with the aim of reducing food waste as a consequence. Any solution should not make food waste an additional thing to worry about, rather should ease existing demands, making the solution genuinely useful/desirable and not requiring the audience to be persuaded to use it

2.3.1 Existing Solutions

There are many ways in which householders try to plan and organise their food shopping, both online and offline.

Offline solutions include spreadsheets, word documents or a paper and pen method to create meal plans that are used to inform shopping lists.

Online solutions include the use of websites and mobile applications (which will be reviewed in greater detail in section 2.4) however these can vary in terms of what functions they can provide and many are sleek recipe databases and will not consider what food product the user already has.

2.3.2 Proposed solution

A new Food Waste Management mobile application that users would access free of charge constitutes an adequate solution based on the following key points: targeted audience, accessibility and cost, internet popularity and specific content.

2.3.3 Targeted Audience

There is no one specific target audience for this app, except to say it will mainly be used by adults, and generally only those who are responsible for food shopping or planning/budgeting.

As the questionnaire results showed, 80% of the respondents aimed to stick to a food budget – and this app can help them with this task. As well as this 90% felt they could benefit from having a food waste manager application (which also mirrors the percentage that responded

that they had a smartphone) and 100% also responded that they felt their friends and family could benefit from the use of one.

This could potentially be a large target audience as in 2014 there were 26.7 million households in the UK. Even if a small percentage of these households were to download the app, this is still a significant volume. (OFFICE OF NATIONAL STATISTICS 2014)

This app is geared towards helping users reduce their food spend as well as their impact on the environment, however it does not have to be aimed at those on a small budget. Reducing your food waste is aligned to food spend but the environmental benefits should resonate with all. Users will need convinced by the features, content and the design as well as the usability that this app could have a potential positive impact on reducing their food spend and waste.

2.3.4 Accessibility and Cost

Having conducted some research into the scale of the problem it is clear that a solution should be created and deployed on a mobile phone platform. Mobile applications that are implemented using standard developing tools are compatible with the main operating systems used on smartphones. Therefore a mobile application offers the users easy access to this new tool from a smartphone or tablet. A mobile application is more suitable than a website as many of the data collected would be while the user is on the go, and are generally accessible to users at all times.

More than three quarters (76%) of UK adults now own a smartphone (GUARDIAN 2015) making this again a very accessible option for adults within the UK.

The current market-leading mobile phone operating system platform is Google's Android offering and research conducted concluded that the Android platform possesses an 80% market share of smartphone operating systems (TECHCRUNCH 2015). Using this it is safe to assume that a solution application would be best created on this operating system as it is the most commonly utilised mobile OS.

All of the food apps discussed above are free to the user, and so to be in line with the market this would be the best fit. However there could be a case to provide in-app purchases for more advanced features in the subsequent releases.

2.3.5 Specific Content

This application is a tool to help users to manage their food waste in 3 distinct ways.

The app can be used to create a shopping list before they leave their homes, ensuring they shop from the list and enabling them to plan before they buy. They can also input the stock items that have at home at this point.

The app will also keep users informed of what they currently have in stock at home whilst they are shopping, so they can ensure they are not buying duplicates and that they will have the space at home to store their food they way they want. The user can also input their newly bought items so that they move from the shopping list to the stock list.

The app will also generate recipes based on any remaining food throughout the week, to prevent the user from having to buy more ingredients, to use what they already have available and prevent food being wasted. These recipes can be rated and then shared with other selected users if the user selects this option.

Appendix 2 illustrates a rich picture diagram that summarises graphically the solution and its impact on users.

2.4 Competitive Analysis

Following the discussion on the food waste problem, a review of some existing food waste mobile applications is necessary before we can consider a solution.

This review was conducted to look at the design and features offered in more detail. They are Love Food Hate Waste App (2016), Yummly App (2016), Food Planner App (2016) and Handpick (2016).

	Love Food Hate Waste	Yummly	Food Planner	Handpick
Downloads (figures from app store)	10,000	1 million	1 million	50,000
Cost – monthly (not including in app purchases)	Free	Free	Free	Free
UK/NI based	✓	✓	✓	x
Food shopping input choice	x	x	x	x

Dynamic pantry	x	✓	x	x
'Smart' recipe advice (Recipes generated from user inputted data)	x	✓ Based on recipes user has liked	x	✓
Ability to share information	✓	✓	✓	✓
Food Spend Trend	x	x	x	x
CO2 emission guidance	x	x	x	x
Meal Planner	✓	x	✓	x
Advertising	x	✓	✓	✓
Shopping list generator	x	✓	✓	✓

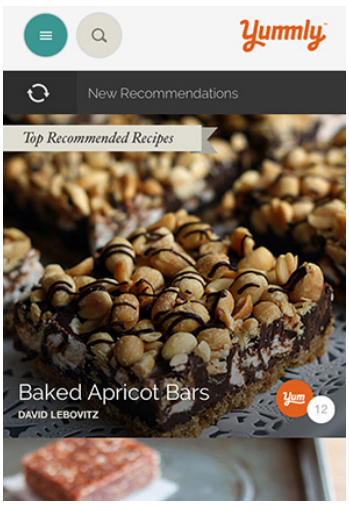
Table 1.0 Overview of competitive analysis

The three most downloaded apps are Yummly and Food Planner and so a more in-depth comparison of these three apps will now be conducted.

2.4.1 Yummly Application Review and Analysis

Yummly is a recipe app that learns your tastes and provides better search results for cooks on the go.

Figure 2.0 Yummly App Icon (screenshot from application)



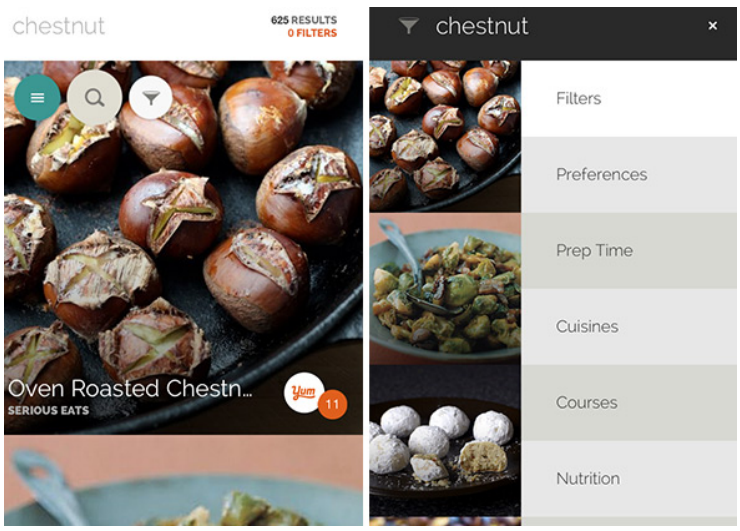


Figure 3.0 Screenshots from Yummly app (screenshot from application)

The Yummly design is very slick and very well designed. The tour is sleek and beautifully designed. With just a few words, the team at Yummly explained exactly what this app does – it is a smart app that records users' likes and tailors search results to their individual tastes.

Applications pros	Application cons
Intuitive and sleek design	Login via social media
Ability to save recipes from varying websites	No ability to prevent food waste generated by recipes/sharing of ingredients for numerous recipes
Ability to share recipes via social media	No food spend trends
Shopping list generated by recipes liked	No storage of food items – shopping list only

Table 2.0 Table showing Yummly app pros and cons

2.4.2 Food Planner Application Review and Analysis

Keep an integrated meal plan, grocery list, inventory and recipe collection that can sync to the cloud across multiple platforms.

Figure 4.0 Food Planner App icon (screenshot from application)

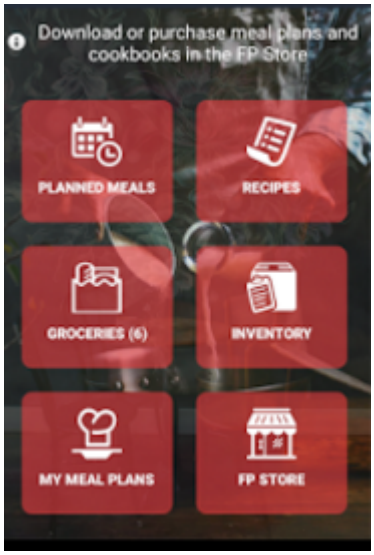


Figure 5.0 Main Menu of Food Planner app (screenshot from application)

Application Pros	Application Cons
Meal Planning function	Design is very cluttered
Shopping list function – generated from recipes	Input must be made manually
Recipes can be added	'Smart' recipes not available
Mobile view of kitchen contents	In-app purchases to remove ads
Syncing ability for recipes and shopping list	Ads used very heavily
Nutritional Information	

Table 3.0 Table showing Food Planner app pros and cons

2.4.3 Key findings

The applications that are currently available seem to either be very sleek recipe databases, where users can bookmark or add recipes to their profile or applications with a very clunky design that allow the user to add more specific information but do not necessarily use this to inform recipes. They are free to access and therefore might display some advertisements in order to make some income.

2.4.9 Recommendations

The following points should be taken into account in the design of this new application:

- The functionality should include a separate space for shopping list (potential items), pantry list (items that user currently has) and a smart recipe generator (based on the pantry list).

- Application should offer free access to all and database to record items and preferred recipes. It should have consistent navigation, as well as an information link and home link on every screen.

- Accessibility and usability should be at the heart of the design with the right type and size of font, minimum scrolling involved, clear and simple instructions and a fun and consistent presentation.

- Efficiency and compatibility would permit the user to load the content and process the data quickly without long delay both on phones and tablets.

2.5 Summary

This chapter has presented the background research on the problem and provided arguments demonstrating why a new Food Waste Manager mobile application constitutes an appropriate solution. It has also reported the results of the comparative analysis conducted on relevant applications currently available on the market.

3.0 Requirements Analysis

3.1 Introduction

This chapter reports on the research questionnaire administered to various community/parenting groups as well as budget food bloggers and unveils their recommendations with regards to the application content, features and requirements. It concludes by listing the functional and non-functional requirements to be incorporated in the design.

3.2 Pre-requirements

For reasons already discussed in the previous section of this dissertation, a mobile application is the most appropriate medium to help householders plan their shopping and use the food they currently have available. Increased availability of mobile phones and tablets at home means that users can easily access this new application. Some primary requirements for this application are already set; it is being designed for android devices and it will be accessible free of charge.

3.3 Requirements research

3.3.1 Research Questionnaire

After researching the current problem, a questionnaire was created to gain a better insight into what the focus group felt were the main problems/requirements to reduce domestic food waste, as well as considering their views on the efficiency of a mobile application to support this. Appendix Three contains the questionnaire and information sheet provided to participants.

A focus group of twenty individuals, each with their own level of food waste awareness, was created. The focus group consisted of three different groups who provided differing ages, abilities and experiences of food waste and budgeting.

The first was 5 website administrators from the website http://fyf20quid.co.uk, which is a website designed to reduce food waste from a food budget perspective who post their weekly food spend and meal planners and provide advice on affordable healthy meals for all. Their main focus is food budgeting and reducing food waste from an economical point of view.

The second group 'Going Green' are an environmental group operating out of the East Belfast Mission, consisting of 6 participants who would be able to provide an environmental input to the design of the project. Going Green have a garden which they use all year round to provide vegetables and herbs for their weekly cooking classes, their main focus is food sustainability and reducing food waste from an environmental point of view.

To address the findings in the research conducted into the nature of food waste a representative focus group of families with children of all ages was deemed necessary. A group of parents from various parenting groups were invited to participate from Ballynafeigh Community Centre; there were 9 participants who agreed to complete the questionnaire.

This focus group will be a key tool in the development of the proposed mobile application and essentially will provide an insight into the current problem as well as requirements and ideas for a proposed solution. The main ideas and concepts gathered from this focus group will be used to form a product backlog and future sprints in the agile methodology used in the development of the application.

The full questionnaire can be viewed in Appendix 3 along with the Information Sheet.

3.3.2 Key findings

Out of the 25 copies handed out, 21 were completed corresponding to an 84% response rate which represents a successful research study and provides an adequate number of answers.

The completed questionnaire data was put into an Excel sheet to be analysed and the key findings are presented in Table 2.

Q1	90% respondents have a smartphone
Q2	80% of respondents aim to stick to a food budget. 50% actively prevent food waste in their home.
Q3	90% respondents use a list when shopping 70% use a list on their smart phone 40% do their food shopping online
Q4	Most used/recommended app/website: Yummly
Q5	90% felt they could benefit from a food waste manager app 100% felt their friends/family could benefit from a food waste manager app
Q6	61% of respondents preferred the name 'Use Your Noodle'
Q7	70-90% of respondents felt that easy input of shopping, mobile view of current kitchen contents and recipes generated from current kitchen contents were Very Important Food storage advice was not felt to be as important with 50% saying Quite important

	72% felt that an easy method of removing items from list was Quite Important 62% felt that food spend trends over time were very important with the rest answering Quite Important Ability to share data was felt to be not as important with 50% replying that it was Quite Important and 30% replying Not Very Important
Q8	100% responded that a manual list would be Very Suitable 30% responded that a list imported from online shopping would be Very Suitable
Q9	30% felt that viewing their food waste by CO2 emissions would be Very Suitable with the other 50% responding Quite Suitable 50% responded that viewing their spend on food waste and trending this data would be Very Suitable 100% responded that tips to help reduce specific items would be Very Suitable
Q10	The following questions related to the features if a user was logged in with a username and password: 50% responded that it would be Very Suitable to view their food shopping habits including spend and favourite items. 400% were interested in sharing this and the CO2 production from their food waste. 30% responded favourably (Very/Quite suitable) to a feature to share and rate recipes
Q11	90% of respondents have offered to test the first version of the app when it is available

Table 4.0 Table showing summary of questionnaire findings

3.3.3 User Stories

These user stories are a representation of the functional requirements which were suggested for the application. Acknowledging these ensures that the developer is fully aware of the criteria required to meet all requirements for the application and also ensures that key requirements are acknowledged. As well as this, it allows individual tasks or features to be separated and managed according to their priority. These user stories then can also to be prioritised and re-prioritised during development.

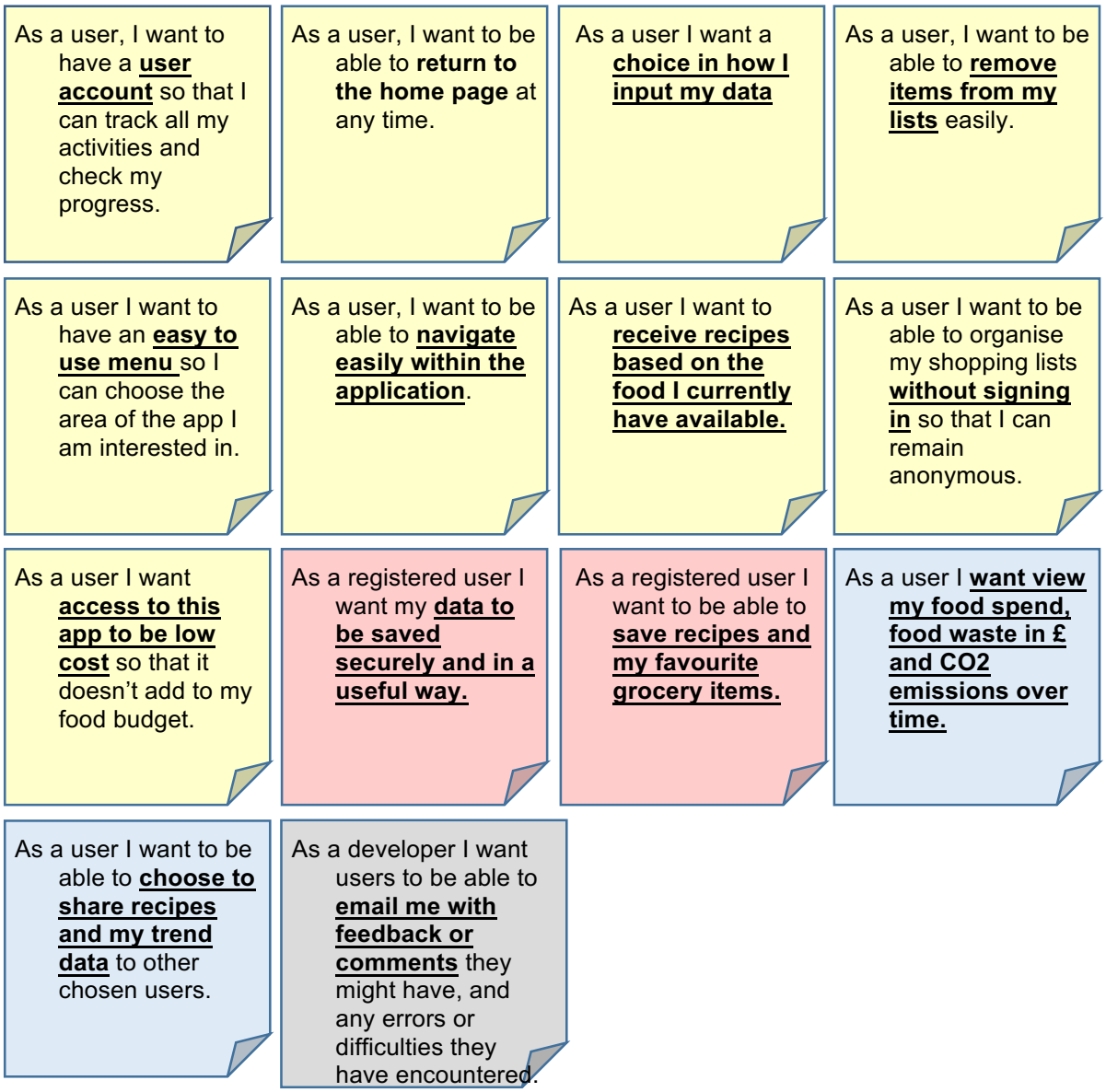


Figure 6.0 User Stories

3.4 System Requirements

Analysis of the current situation and system requirements are essential in the development of a mobile application. Having analysed the current situation, the requirements of this project have been decided. This has allowed for an appropriate methodology to be selected to complete project development- the agile methodology which will be discussed in the design section of the report. Furthermore, analysis of the current situation and feedback from a focus group has allowed for requirements of the application to be developed. These requirements are either functional or non-functional and are of vital importance in the development of the application. This ensures that all end-user suggestions are acknowledged

and also allows for a better understanding of what needs to be included in this type of application in order to make it a success. As the user will more than likely be the only person using their device, and the SQLite database provided by Android Studio is optimised for a single user, the need for a login was deemed unnecessary.

Including Trends analysis was felt unnecessary at this first development, as it would require either some cost information (to display savings in pounds and pence) or a calculation based on the volume and type of food which at this stage was felt to be beyond the scope for this first version of the application.

3.4.1 Functional Requirements

The application must ensure that the application environment is secure and that unauthorised access is controlled.

The application must allow for users to input their data easily.

The application must allow for users to enter/amend data when offline.

The application must provide the user with the ability to hold the amount of data that the user inputs.

The application must allow the user to save their shopping list, their current stock items and generate recipes based on what the user has in stock.

3.4.2 Non-functional Requirements

Consideration of non-functional requirements within the application ensures that a coherent application is developed. These requirements are outlined below.

The application must be compatible with multiple Android devices.

The application must have an attractive design to encourage use.

The application must be intuitive to use.

The application must be designed so separate sections can be used by user independently of the whole design.

Navigation within the application must be consistent to allow for ease of use.

3.5 Summary

This chapter presented the recommendations made by community/parenting groups who responded to the research questionnaire. It also provided the list of functional and non-functional requirements to be incorporated in the application design.

4.0 Design

4.1 Introduction

This chapter presents an overview of the application design. It discusses the design methodology followed. It describes the user interface and includes the prototype. It also discusses the architectural design of the application including the site map and the relational database.

4.2 Design Methodology

This section of the report aims to provide a description of the app interface design and will include digital mock-ups that represent the considerations in the design.

Thumbnails then paper mock-ups of the apps access to functions are initially produced by hand on single-pages. This process allows for quick exploration of the major design components and design space and also first sketches of the page layout.

Once a satisfactory design is achieved on paper, digital mock-ups are necessary for higher-quality representations of the design. They allow for easy arrangement and re- arrangement of page layout and as well as more thorough investigations of font, colour and contrast. Digital mock-ups provide accurate way to preview the colour and contrast issues that might arise in a backlit screen environment.

By ensuring this application is pleasing to the eye and easy to navigate, first time users will leave with a positive impression and a good experience and will most likely return. When an application leaves a bad first impression, it discourages users' consecutive visits. Usability is critical as when users get frustrated, they leave the application and do not return.

Good user interface design (UID) along with the logo and other visual features significantly impacts on the way customers perceive the application. The goal when designing the user interface is to establish accessibility and usability for all the target audiences.

4.3 User Interface Design

The overall design of a mobile application is extremely important. During the design stage, human computer interaction and software engineering issues are considered and solutions are created to ensure that any potential issues are acknowledged and dealt with accordingly. Consideration of these two important design aspects should ensure that a fast, reliable and

visually attractive application is created. Therefore, this design stage is vital as it provides a basis on which the entire application is built.

The user interface is one of the most important parts of the application design. Android Studio provides lots of components for android developers to design graphic user interface.

To begin to design the application's user interface it is important to consider what human computer interaction will be expected with the use of this application as well as the specific considerations along with that.

Ben Shneiderman is an American computer scientist and professor at the University of Maryland Human-Computer Interaction Lab. In his popular book "Designing the User Interface: Strategies for Effective Human-Computer Interaction", Shneiderman reveals his eight golden rules of interface design. These have then be considered when designing templates for user interface best practice.

Strive for consistency by utilising familiar icons, colours, menu hierarchy, call-to-actions, and user flows when designing similar situations and sequence of actions. Consistency plays an important role by helping users become familiar with the digital landscape of your product so they can achieve their goals more easily. (Wong, 2013) This has been considered in relation to the colour scheme and general layout.

Enable frequent users to use shortcuts. With increased use comes the demand for quicker methods of completing tasks. (Wong, 2013) As the user gets more comfortable with the application they would like to reduce the number of interactions they perform. At this stage this requires more consideration to ensure what useful shortcuts could be implemented.

Offer informative feedback. The user should know where they are at and what is going on at all times. For every action there should be appropriate, human-readable feedback within a reasonable amount of time. This has been implemented to show a message displayed to the user to let them know if their item has been inputted correctly into the database or not.

```
if(isInserted=true)
    Toast.makeText(MainActivity.this, "Data Inserted",Toast.LENGTH_LONG).show();
else Toast.makeText(MainActivity.this,"Data not inserted",Toast.LENGTH_LONG).show();
```

Figure 7.0 Error message

Design dialogue to yield closure**.** Don't keep your users guessing. Tell them what their action has led them to. (Wong, 2013) One example from this application is if a user if trying to view all data currently in an empty table in the database a message will display to show that there is no data available to view.

```
public void viewAll(){
    btnViewAll.setOnClickListener(
            (v) -> {
                    Cursor res = myDb.getAllData();
                    if(res.getCount() == 0){
                        showMessage("Error", "No data found");
                        return;
                    }
                    StringBuffer buffer = new StringBuffer();
                    while(res.moveToNext()){
                        buffer.append("Ingredient : "+ res.getString(3) + "\n");
                        buffer.append("Quantity : "+ res.getString(4) + "\n");
                        buffer.append("Unit : "+ res.getString(5) + "\n\n");

                    }//while
                    showMessage("Data", buffer.toString());
            }
```

Figure 8.0 Error handling

Offer simple error handling. Systems should be designed to be as fool-proof as possible, but when unavoidable errors occur, ensure users are provided with simple, intuitive step-by-step instructions to solve the problem as quickly and painlessly as possible. (Wong, 2013) For 'Use Your Noodle', this will include flagging the text fields where the user input is required if these have not been completed with a message to explain what needs to be completed.

```
public void viewAll(){
    btnViewAll.setOnClickListener(
            (v) -> {
                    Cursor res = myDb.getAllData();
                    if(res.getCount() == 0){
                        showMessage("Error", "No data found");
                        return;
```

Figure 9.0 Error handling

Permit easy reversal of actions. Designers should aim to offer users obvious ways to reverse their actions. If a user knows an error can be undone it will make them more comfortable to explore options. (Wong, 2013) Android devices have an inbuilt navigation toolbar (with a back button as well as a home which brings you out of the app and to the main menu of the device) which will be utilised for this application; this will save screen space and also add to the intuitive feel of the application.

4.3.1 Navigation

One of the functional requirements decided upon was the primary navigation based on a main menu and subsequent menus for each section. The use of buttons was preferred to the use of shortcuts and drop down menus to ensure user accessibility was met.

The primary navigation design therefore included four buttons on the main menu, clearly identified with text to facilitate accessibility. To ensure usability for people with disabilities,

buttons were designed to be height of 36dp and touchable targets a minimum height of 48dp. (Material design guidelines 2016)

The secondary navigation included buttons labelled with each function. On each activity there were two buttons on the toolbar – one to bring the user back 'Home' (to the main menu) and the other to provide a popup information menu to provide information to the user.

Storyboard illustrated in Appendix 4 present how the users will navigate throughout the application. They show the navigation elements such as buttons that would allow the user to easily and consistently find their way through the different webpages.

4.3.2 Colour Scheme

A logo has been designed for the 'Use Your Noodle' app. The logo was designed by looking at competitor's/well established brand's logos and understanding the need for clear, simple and friendly design. In my previous research I had established that I wanted to create an app that is not only based on budget food, but more around helping consumers be informed and make smart choices so I wanted a logo that reflected that and had a clear but attractive design. A light blue with black diagram with white font was decided upon as it is clear and will be easy to keep it consistent throughout the user experience.

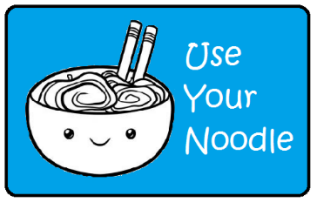


Figure 10.0 Logo design for Use Your Noodle app

Careful consideration of the colour scheme was of key importance in the design of this application and ensured that there was a high level of consistency within the design. The developer decided that the main colours used within the application would be white and blue (#1d3973). This scheme in particular illustrates a clean, fresh and bold design and is attractive from an end user's perspective. The use of this colour scheme will be present throughout the entire application to ensure that the appearance is consistent and attractive.

4.3.3 Touchscreen Considerations

A specific consideration for smartphone applications and their user interface is that most Android devices now have a touchscreen. These offer a way for users to directly manipulate on-screen objects that would have previously been driven through a keyboard, mouse, joystick or other input device. The touchscreen model of direct manipulation has a large impact on the way we think about our user interfaces. It also changes the expectations a user has for an application. Touchscreen devices require us to stop thinking in terms of forms and think about object-oriented user interfaces. Another consideration is that generally a software keyboard is used and this will consume some of the application's screen space. (Clark, 2012)

The first restriction placed on any touchscreen user interface is the size of the user's forefinger – the developer must ensure that there is enough space between objects so that the user only makes contact with the widget or object they are wanting to make contact with. (Clark, 2012) Another impact touchscreens have on user interface design is that an application and all the widgets that it uses must be entirely self-explanatory. It is not possible due to screen size to include a roll-over or tooltip to indicate functionality. The limitations due to screen size must also be considered when designing what information we want to present to the user or ask the user to provide us with, we want to limit this for a streamlined and clutter free application. The first stage of design involved creating a prototype. This ensured that the developer had a thorough understanding of the required functionality and navigation throughout the application. This allowed for designs and colour schemes to be decided at the initial design stage.

Menu Screen

The content within this page, is easily accessible and clear, outlined using four buttons. database.

Figure 11.0 Menu Screen

The consistent colour scheme will be visible within each page and the main content of the application is based around this colour scheme. As well as this, the idea of spacing between content is of vital importance. In particular, the spacing between buttons is an important design aspect of the application. This ensures that the application caters for every-day problems such as the 'Fat Finger Problem' which should be considered in the development of any application. Acknowledging this type of potential issue at the design stage is important as it will enhance the effectiveness of the application and also enhance the appealing nature of the design. Therefore, there must be a focus on the layout of each page within the application as well as the process of navigation throughout each section of the application.

4.4 Architectural Design

The architectural design will be discussed, in brief a summary of the Android architecture, the application architecture and then the structure of the database.

4.4.1 Android Architecture

The app will be designed using an Android platform on IntelliJ IDEA Community Edition. This Java IDE provides all the necessary functional requirements in the app. It is recommended that this application is used on a smartphone device or tablet. The Android architecture includes the database platform SQLite built in to the library layer. Therefore this will be used to provide a database to store product information, as well as recipes that the user can access via a recipe wizard to recommend recipes based on the ingredients they have available.

As well as user interface design, it is also extremely important to focus on the architectural design of an application in the design stage. This ensures that the developer has a good understanding of the software required to complete development. In this chapter the Android architecture will be discussed briefly. The reason for this is that the application will make use of a number of available libraries within the application. Finally, database structure will be acknowledged via database tables and an ER diagram.

In creating a mobile application it is important to acknowledge the current architecture of the platform. The reason for this is because the design of a mobile application should make full use of the available API for the platform, in this case Android. From a developers perspective this will make development must easier as code can be reused within the API libraries, also providing a much more consistent application to run within the chosen platform. In particular the application will make use of the SQLite library, situated within the second layer of the

Android Architecture. The SQLite database “is a useful repository for storage and sharing of application data.” (Tutorials Point, 2014) The application will make use of this resource to store user information and to ensure that personal data is secure within the application. This will enable individual shopping/stock lists, recipes as well as their food waste trends to be saved for the user.

The Application Framework “provides many higher-level services to applications in the form of Java classes.” (Tutorials Point, 2014) These java classes will become very useful as development begins as these can be used to aid development.

The final layer of the architecture is the Application layer. This is where an application is developed, tested, deployed and maintained. In conclusion then, it is evident that there is a wide range of support available within the Android Architecture. From a developer’s perspective this is encouraging and makes the prospect of developing for this platform very appealing.

4.4.2 Application Architecture

Prior to designing a prototype for the proposed application, it was important to ensure there was a clear understanding of the architecture of the application as a whole. This allowed for a clearer understanding of the layout of the application as well as ensuring all pages were considered in the design process.

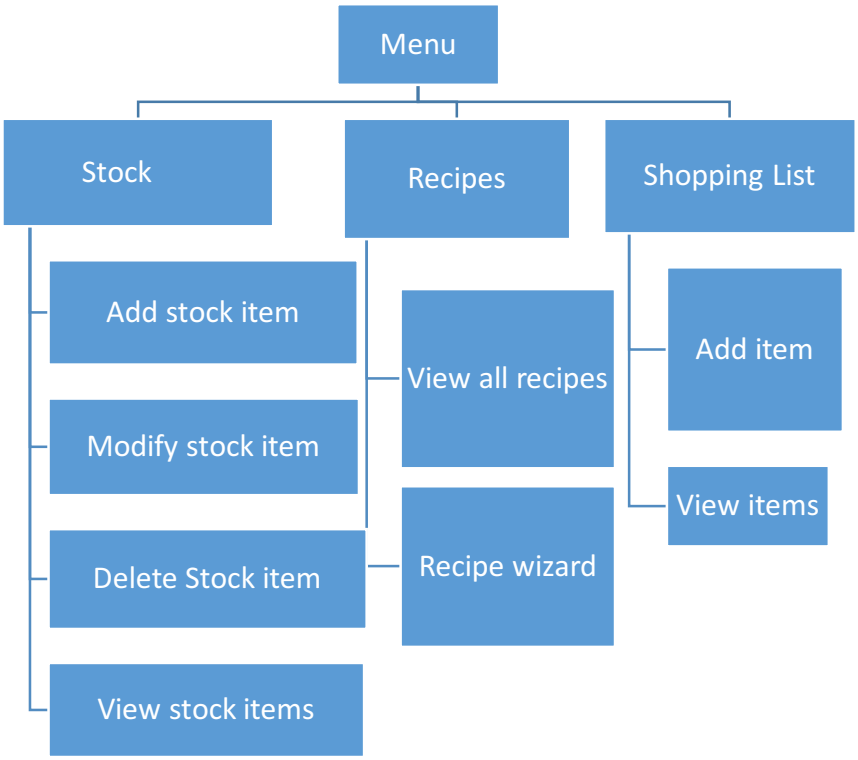


Figure 12.0 Application architecture diagram

4.4.3 Database Design

This app needs to store data, and Android provides a SQLite database to do this. Most database systems need a special database server process in order to work. This is not a requirement for SQLite as a SQLite database is just a file, so when it's not in use, it doesn't use up any processor time. That's important on a mobile device, as they have limited processor time and battery life.

SQLite is optimised for a single user. This app is the only thing that will communicate with the database, and the owner of the mobile device is the only one that will be using the app so currently there is no requirement for the user to have a username and password.

SQLite databases are extremely stable, they can handle database transactions, which means if there is an issue updating several pieces of data SQLite can roll the data back. Also the code that reads and writes the data is written in optimised C code. This is fast and also reduces the amount of processor power it needs. (Griffiths, 2016)

Android automatically creates a folder for each app where the app's database can be stored. Each database consists of two files; database file: this has the same name as the database and is the main SQLite database file. All data will be stored in this file.

The second file is the journal file. This has the same name as the database with a suffix of –journal. This contains all of the changes made to the database. Android can use this to undo (or rollback) your latest changes.

The directory where an app's databases are stored is only readable by the app itself. The database is secured down at the operating system level.

The database has been designed with various integrity rules in mind. The relational integrity has been considered, in a relation no attribute of the primary key can be null. The referential integrity has been considered, if a relation contains a foreign key, the foreign key value must match a candidate key value of some tuple in its home relation. There are also enterprise (or company) constraints – for this application we would not want to provide a recipe that does not have any instructions for example so this would be set to not null etc.

<u>SQLite Classes</u>

These classes have been used to implement the database, examples of their use can be seen.

SQLite Helper - This class is created by extending the SQLiteOpenHelper class. This enables you to create and manage databases.

```java
//created databasehelper class extending SQLiteOpenHelper
public class DatabaseHelper extends SQLiteOpenHelper {
    //declare and initialise database name as final
    public static final String DATABASE_NAME = "database.db";

    //declare and initialise table name
    public static final String TABLE_NAME = "Ingredients";
    public static final String COL_1 = "ingredient_id";
    public static final String COL_2 = "recipe_id";
    public static final String COL_3 = "stock_id";
    public static final String COL_4 = "ingredient_desc";
    public static final String COL_5 = "quantity";
    public static final String COL_6 = "units";
```

```java
public void viewAll(){
    btnViewAll.setOnClickListener(
            (v) -> {
                Cursor res = myDb.getAllData();
                 if(res.getCount() == 0){
                    showMessage("Error", "No data found");
                    return;
                }
```

Figures 13.0 and 14.0 DatabaseHelper and Cursor class– this allows the application to read from and write to the database.

```java
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE " + TABLE_NAME + " (ingredient_id INTEGER PRIMARY KEY AUTOINCREMENT, " +
            "recipe_id INTEGER AUTOINCREMENT, stock_id INTEGER AUTOINCREMENT,ingredient_desc TEXT, quantity INTEGER, units TEXT)");
```

Figure 15.0 SQLite Database - This class gives access to the database.

4.4.5 Relational Database

A relational database was required in order to record and manage all user related information when users choose to login onto the application. The database design process consisted of the following steps:

- Gather all the required user related data.
- Segregate the data by identifying the different entities and their relationships.
- Draw an Entity Relationship Diagram (ERD) to represent the database design and ensure there are no many-to-many relationships.
- Organise the data in tables with all the appropriate attributes, primary and foreign keys.
- Apply normalisation process to ensure a third normal form relational database design is achieved.

Four entities were identified for the application database: Stock, Ingredients, Recipes and ShoppingList.

The database for ‘Use Your Noodle’ application was based on a relational database methodology which defines a set of interconnected data tables for organising and storing

data. In order to design the relational database, an Entity Relationship (E-R) Diagram was drawn to give a visual representation of all the information to be stored within the database and to identify the relationships between the different tables.

The database tables that have been created are listed, along with the E-R diagram. At this stage a user login/registration was found to be unnecessary as mobile devices are generally used only by the owner, and SQLite is optimised for a single user, and so multiple accounts does not seem appropriate at this stage.

Each table in the database was allocated a distinct name and each attribute associated with a particular table was given a name prefixed with a portion of the table name. Each table was assigned a primary key which was used to uniquely identify each of the rows in the table. The primary key was unique and not null, to ensure that there were no duplicate values. As well as primary keys each table was assigned a foreign key which connected one table to another within the database. This is referred to as referential integrity. A lack of referential integrity can lead to the return of incomplete data.

The database consisted of four tables, 'Ingredient', 'Recipe', 'Stock', and 'Shopping List'. The database was arranged in this manner to allow for each 'topic' to be stored separately yet able to interact with each other, when requests for information were made by the user.

Ingredient Table

Name	Type	Description	Value
ingredient_id	INTEGER	Primary Key	Auto-increment
ing_recipe_id	INTEGER		Not null
ing_stock_id	INTEGER		Not null
ing_quantity	INTEGER		DEFAULT(0)
ing_units	TEXT		DEFAULT(0)

Table 5.0 Ingredient Table

Ingredient (ingredient_id, recipe_id*, stock_id*, quantity, units)

Recipe table

Name	Type	Description	Value
recipe_id	INTEGER	Primary Key	Auto-increment
Recipe_name	TEXT		Not null
recipe_instructions	TEXT		Not null

Table 6.0 Recipe Table

Recipe (recipe_id, recipe_instructions)

Stock List table

Name	Type	Description	Value
stock_id	INTEGER	Primary Key	Auto-increment
stock_name	TEXT		DEFAULT(0)
stock_food_type	TEXT		DEFAULT(0)
stock_quantity	INTEGER		DEFAULT(0)
stock_units	TEXT		DEFAULT(0)

Table 7.0 Stock Table

Stock (stock_id, name, food_type, quantity, units)

Shopping_List

Name	Type	Description	Value
sl_item_id	INTEGER	Primary Key	Auto-increment
sl_name	TEXT		Not null
sl_units	TEXT		DEFAULT(0)
sl_quantity	TEXT		DEFAULT(0)

Table 8.0 Shopping List Table

Shopping_list (sl_item_id, sl_units, sl_quantity)

Database E-R Diagram

Figure 16.0 Entity Relationship Diagram

4.5 Summary

The design for the application user interface was first presented in this chapter with some mock-ups. It then described the different elements of the architectural design and their relationship. It concluded with a discussion surrounding the process followed when designing the relational database and finally presented the different relations used for the application data and their attributes.

5.0 Implementation

5.1 Introduction

This chapter first provides details about the hardware and software used in the implementation of the application. It then explains the cycle process adopted to implement the application, which is based on Agile principles. It includes screenshots of Java, XML and SQLite.

5.2 Hardware and Software

A Hewlett Packard 3.07GHz 6GB Windows 10 64-bit laptop computer was used for all the programming and some testing.

Android Studio was used to program in Java, XML and SQLite and testing was conducted with the Android Studio emulator as well as Sony Experia Z5 and a Nexus Tablet.

Adobe Photoshop Creative Suite 6 was used to process and edit the application pictures.

5.3 Cycle Process Implementation

Many Agile principles (Agile 2014) are related to the relationship and communication between developers and customers. Those principles could not be applied directly as our main customers are parents/community group members with whom we cannot have regular communication due to time constraints. Instead, communication with the project supervisor and some focus group participants took place at key moments of the project to review the design and implementation and offer feedback and suggestions that the developer took into account.

The following Agile principles were applied in the implementation of the application:

- Working software is the primary measure of progress: application first implemented with only a few activities then continuously upgraded with additional activities and features meeting more requirements each time.
- Welcome changing requirements, even late in development.

The implementation process was based on the Agile software development model by continuously repeating the following three step cycle:

1. Analysing: identification of requirements or improvements to be implemented were made by prioritisation and estimating the quantity of coding and time required.

2. Developing: developing programs and methods algorithms; coding in Java, XML, SQLite as required; creating/updating database tables and queries; adding pictures and other content.

3. Testing: new implemented programs and features and their interoperability with existing ones were tested.

Table 11 presents a summary of the different cycles that were conducted for the implementation of this application in chronological order.

Cycle	Implementation	Code	Environment
1	Menu activity including header logo, primary navigation, backgrounds, buttons and all textual content.	Java, XML	Android Studio + emulators
2	My Pantry, Shopping List, and Recipe activities created including buttons and all textual content completed.	Java, XML	Android Studio + emulators
3	Database created and four tables with their relationships implemented : ShoppingList, Pantry, Ingredients, Recipes	SQLite, Java	Android Studio with SQLite + emulators
4	Pre-populated database information SQLite statements coded.	SQLite, Java	Android Studio with SQLite + emulators
5	Custom Listview with ASyncTask and query created to display Pantry and ShoppingList items in database to user.	Java, XML	Android Studio + emulators
6	Two methods of data input for Pantry items created to add to database.	Java, XML, SQLite	Android Studio with SQLite + emulators
7	Spinners created to aid with data input. To ensure only certain values can be entered (e.g. units)	Java, XML	Android Studio + emulators
8	Custom ListView amended as ASync task not compatible with creating a clickable Listview	Java, XML	Android Studio + emulators

9	Update queries for stock and shopping list created – including feature to updating quantity of favourite item as well as clickable ListView to update existing items.	Java, XML, SQLite	Android Studio with SQLite + emulators
10	Taskbar navigation with custom information and home button for each activity implemented.	Java, XML	Android Studio + emulators
11	Recipe queries implemented to show all recipes as well as 'smart' recipe query. Automatic entry of needed ingredients to shopping list of selected recipe.	Java, XML, SQLite	Android Studio with SQLite + emulators

Table 9.0 Implementation Cycles Summary

5.3.1 First Cycle

Initially a new Android Studio project was created (including the following files and folders):

The package name for application.

The android version on which the application is created.

AndroidManifest.xml

When an Android Application is created in an IDE, a number of folders are generated to support application development. These include - src, gen, assets and res. However, one of the most important files in an Android project is the AndroidManifest.xml file. The content within the Android manifest file manages key elements of the project including:

During application development the AndroidManifest.xml file must be continually updated. Therefore, it is evident then that managing this file is of key importance while developing any Android project. This provides overall functionality within the application and also allows for the project to be managed within one file. (Felker 2012)

Res folder

The resource folder is a vital project management tool. This allows for values to be stored and resources to be maintained within the appropriate folders. For example, there are four drawable folders contained in this res folder. This allows for images and graphics to be stored and referenced as a property within the project.

Furthermore, the strings.xml file is an important folder situated in the values folder of res. This is an extremely useful file which allows a developer to store, edit and reference string

values outside of the layout pages. Therefore, this strings file will play a key role in the development of the proposed application to ensure the design and efficiency is at an extremely high level. (Felker 2012)

Management of the resource folder will be extremely important throughout project development. This will allow for values and properties to be referenced accordingly and will ensure that the project structure is understandable and as efficient as possible.

MainActivity.java

The functionality of the home class is to provide a menu which allows for users to decide which area of the application they would like to navigate to. This contains four buttons to provide these options. The buttons are concise making navigation to each page easy. Furthermore, the application Logo will be displayed in an ImageView within the Home page. This creates a professional representation of the application itself. This is the activity the home button will return the user to.

The XML coding was realised using previous experience and continuous referring to W3Schools (2014).

5.3.2 Second Cycle

The four activities were created by creating new blank activities, which generates a java file as well as a linked XML file. Following good development practices these were given meaningful names, with Java classes and XML files with corresponding names, and titles and all text was saved in the strings.xml file with the relevant reference.

```
public class MyPantry extends AppCompatActivity {
    //OnCreate method
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_my_pantry);

    }//onCreate
```

Figure 17.0 onCreate of MyPantry activity

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="com.useyournoodle.katherine.useyournoodle.MyPantry"
    android:background="#2787A4">
```

Figure 18.0 XML showing layout for MyPantry activity with background colour

The XML included the blue background colour and were designed for consistent layout using the menu as a template to ensure the style was consistent throughout.

All images for each activity was saved within the res folder and all buttons were implemented within XML with methods added to the onClick, which could then be called in the corresponding Java classes.

```
//reference for button
Button shoppingListButton = (Button) findViewById(R.id.shoppingList_btn);
shoppingListButton.setOnClickListener((v) -> {
        Intent intent = new Intent(getApplicationContext(), ShoppingList.class);
        startActivity(intent);
    });
```

Figure 19.0 Example of Java to set onClick for buttons on Main Menu

This cycle was tested by running the emulator and ensuring that each button brought the developer to the activities created.

5.3.3 Third Cycle

The DatabaseOperations class extends the Java library class SQLiteOpenHelper and is a class designed to create, add, inspect, update and remove data from the SQLite database. This is an extremely important class as this is used to set up the structure of the database in relation to rows, columns and content within the table. If a database has not already been created, this class is used to create a database. However, if the database has already been created the class is used to load the database within the application. It is important to note that queries are not made via this DatabaseOperations class. Rather, the class is used solely to manage the structure and set-up of the database within the application.

There are a number of methods that are very important within this class, onCreate, onUpgrade and onOpen.

```
public class DatabaseOperations extends SQLiteOpenHelper {

    public class MyClass
    {
        Context c;
        public MyClass(Context context) { c= context; }
    }
    //integer to hold database version
    private static final int database_version = 1;
    private static final String DB_NAME = "noodle.db";
    private static String database_Path = "";
    //log tag to identify class
    public static final String TAG = DatabaseOperations.class.getSimpleName();
```

Figure 20.0 DatabaseOperations class showing database version and name variables

Statements were implemented to create each of the four tables with primary and foreign keys specified as well as data type.

```
//Create Table Statements
private static final String CREATE_PANTRY = "CREATE TABLE "
        + StockInfo.STOCK_TABLE_NAME
        + "("+ StockInfo.STOCK_ID+" INTEGER PRIMARY KEY AUTOINCREMENT,"
        + StockInfo.STOCK_NAME+" TEXT NOT NULL,"
        + StockInfo.STOCK_FOOD_TYPE+" TEXT NOT NULL,"
        + StockInfo.STOCK_QUANTITY+" INTEGER,"
        +StockInfo.STOCK_UNITS+" TEXT"+");";
```

Figure 21.0 Example of Create table SQLite statements to create each table

```
@Override
public void onCreate(SQLiteDatabase sdb) {
    //sql statements to create our tables
    sdb.execSQL(CREATE_PANTRY);
    Log.d(TAG,"Pantry table created");
    sdb.execSQL(CREATE_INGREDIENTS);
    Log.d(TAG,"Ingredients table created");
```

Figure 22.0 onCreate for DatabaseOperations showing SQLite statements being executed to create each table

```
@Override
public void onOpen(SQLiteDatabase db){
    super.onOpen(db);
    if(db.isReadOnly()){
        db.setForeignKeyConstraintsEnabled(true);
    }//if
}//onOpen
```

Figure 23.0 onOpen method for DatabaseOperations to ensure foreign keys can be used within our tables

```
@Override
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int oldVersion, int newVersion) {
    sqLiteDatabase.execSQL("DROP TABLE IF EXISTS " + StockInfo.STOCK_TABLE_NAME);
    sqLiteDatabase.execSQL("DROP TABLE IF EXISTS " + IngredientsInfo.INGREDIENTS_TABLE_NAME);
    sqLiteDatabase.execSQL("DROP TABLE IF EXISTS " + RecipesInfo.RECIPE_TABLE_NAME);
    sqLiteDatabase.execSQL("DROP TABLE IF EXISTS " + ShoppingInfo.SHOPPING_TABLE_NAME);
    onCreate(sqLiteDatabase);
}//onUpgrade
```

Figure 24.0 onUpgrade method for DatabaseOperations

The onUpgrade method is only called when the database file exists but the stored version number is lower than requested in constructor.

Individual classes were created to hold the information for each of the four tables, e.g. TableStock. This included String variables to hold the name of the entity and each of the tuples within it.

```
ublic class TableStock {

    //default constructor - prevent accidental creation of class
    public TableStock(){

    }//constructor

    public static abstract class StockInfo  {
        //declare and initialise columns in table

        public static final String STOCK_ID = "stock_id";
        public static final String STOCK_NAME = "item_name";
        public static final String STOCK_FOOD_TYPE = "food_type";
        public static final String STOCK_QUANTITY = "stock_quantity";
        public static final String STOCK_UNITS = "stock_units";

        //declare and initiliase database/table name
        public static final String STOCK_TABLE_NAME = "stock_info";
```

Figure 25.0 Example of Stock Table class to initialise column titles for the Stock Table

This cycle was tested by running the app on the emulator and then exporting the database file from the emulator data. This was then imported into the SQLiteBrowser where it can be viewed. It was possible to see that each of the tables was created, with no data in any table.

5.3.4 Fourth Cycle

Now that the database had been created, String statements were created in appropriate tables and the SQLite commands were written in the DatabaseOperations class to pre-populate the application with recipes and their ingredients.

```
//variables to add an onion to Stock table
public static final String ONION_NAME = "'Onion'";
public static final String ONION_TYPE = "'Vegetable'";
public static final String ONION_QTY = "0";
public static final String ONION_UNITS = "'Whole'";
```

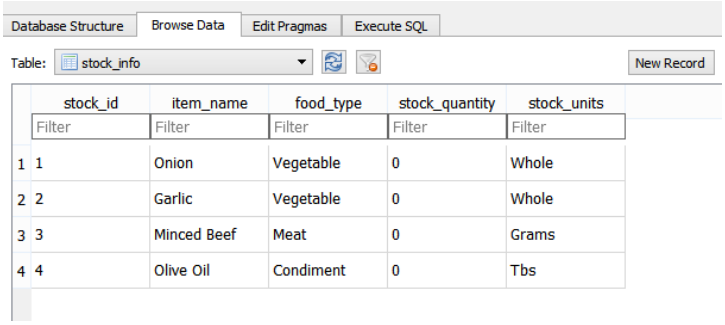
Figure 26.0 Example of String statements created for pre-population in TableStock class

```
sdb.execSQL(INSERT_ONION);
Log.d(TAG, "Onion inserted to stock table");
sdb.execSQL(INSERT_ONION_QTY);
Log.d(TAG, "Onion qty inserted to INGREDIENTS table");
sdb.execSQL(INSERT_GARLIC);
Log.d(TAG, "Garlic inserted to stock table");
```

Figure 27.0 Example of SQLite statements to prepopulate entries to database.

These SQLite statements were executed in the onCreate method within the DatabaseOperations class to ensure there was data in the database as soon as it was created. The order of their insertion was of the utmost importance to ensure that dependent tables had corresponding data in the tables they were dependent on for it to insert correctly.

This cycle was tested by running the app on the emulator, exporting the database file and browsing in SQLiteBrowser application.



	stock_id	item_name	food_type	stock_quantity	stock_units
	Filter	Filter	Filter	Filter	Filter
1	1	Onion	Vegetable	0	Whole
2	2	Garlic	Vegetable	0	Whole
3	3	Minced Beef	Meat	0	Grams
4	4	Olive Oil	Condiment	0	Tbs

Figure 28.0 Example of the SQLiteBrowser showing the pre-populated data inserted into the database

5.3.5 Fifth Cycle

Now that there is content in the database, a function to view this data was implemented. A custom ListView was created to display the data that was now available in the database. This was designed using Android AsyncTaskLoader with ListView and BaseAdapter. AsyncTaskLoader uses AsyncTask to perform work. ListView is a view which shows items in vertically scrolling list. BaseAdapter is a common implementation for other adapters. While

using AsyncTaskLoader we need to extend it and override at least a method i.e. loadInBackground which perform the user task. After loading data we will set data to adapter and finally call notifyDataSetChanged() to reflect list view on UI. (Rai 2015)

android.content.AsyncTaskLoader<D> is a loader that uses AsyncTask to perform the task. It is an abstract class and to use it we need to extend and override its methods. In our example we will iterate a list using ListView. Find some methods that need to be overridden. loadInBackground() : Performs actual task in background and returns the result. onCanceled(D data) : This method is called if task is cancelled before completion. It is used to clean up data post cancellation. cancelLoadInBackground() : We override this method to cancel the background process. If there is no process in background, it is not going to be called. (AsyncTask Android Developers (2016))

5.3.6 Sixth Cycle

Methods for user input of Stock items was designed and implemented. There are two methods – one where the user can select a prepopulated item and update the quantity, and another where the user can enter a brand new item.

The first method was implemented so that the user could update the quantity of a stock item that had been pre-populated in the database with a quantity of zero. These items had to be prepopulated due to the database design, as a stock item has to exist for it to appear as an ingredient for a recipe due to the dependencies of the Ingredients table. The stock items already in the database (only those with a quantity of 0) were to be shown in a spinner (details in Seventh cycle) in this activity and then the user could add the quantity as the item was purchased. A new XML layout and Java Class was created as well as an updateQty method in the DatabaseOperations class to facilitate this.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_add_stock_spinner);
    //Initialise variables to id within layout
    stock_sql_spinner = (Spinner) findViewById(R.id.update_qty_spinner);
    btnUpdateQty = (Button) findViewById(R.id.btn_update_qty);
    inputNewQty = (EditText) findViewById(R.id.input_new_qty);
    //set up onclick listener for button
    btnUpdateQty.setOnClickListener((view) → { updateSpinnerStockQty(); });
```

Figure 29.0 onCreate method of AddStockSpinner Java class showing initialisation of spinners and EditTexts

```
public void updateQty(DatabaseOperations db, int id, String updatedQty){
    SQLiteDatabase sdb = db.getWritableDatabase();
    ContentValues cv = new ContentValues();
    cv.put(StockInfo.STOCK_QUANTITY,updatedQty);
    sdb.update(StockInfo.STOCK_TABLE_NAME,cv,StockInfo.STOCK_ID + " = "+ id,null);
    Log.d(TAG, "New item inserted into Stock table "+ updatedQty + " " + id);
    sdb.close();
}//updateQty
```

Figure 30.0 updateQty method in DatabaseOperations to facilitate updating quantity.

The second method was for the user to be able to input a brand new Stock item (i.e. one that has not been pre-populated by the database. This included 2 EditTexts to facilitate free text as well as two spinners (details in Seventh cycle) to keep the units and food type variables within specified variables.

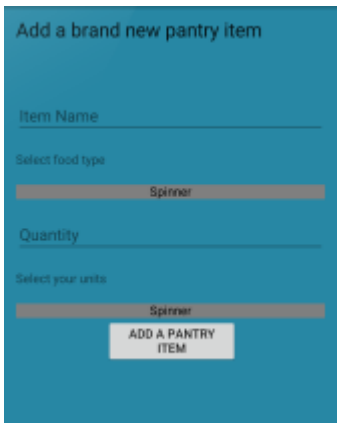


Figure 31.0 Design of addStockProduct layout showing EditTexts and Spinners

```
public void addStockProduct(View view){
    stock_name = STOCK_NAME.getText().toString();
    stock_type = food_type_spinner.getSelectedItem().toString();
    stock_qty = STOCK_QUANTITY.getText().toString();
    stock_units = food_unit_spinner.getSelectedItem().toString();

    DatabaseOperations sqldb = new DatabaseOperations(this);
    sqldb.addStockInfo(sqldb, stock_name,stock_type,stock_qty,stock_units);
    Toast.makeText(getBaseContext(),"Item has been added to your pantry",
            Toast.LENGTH_SHORT).show();
    Intent goMyPantry = new Intent(getApplicationContext(), MyPantry.class);
    startActivity(goMyPantry);

}//addStockProduct
```

Figure 32.0 addStockProduct method showing initialisation of EditTexts and spinners within SaveStockProduct class

This cycle was not able to be tested after this cycle as the Spinners had not been fully implemented.

5.3.7 Seventh Cycle

In each input method it was important to restrict the unit input in particular to ensure that the units inputted would match those in the ingredients table for each recipe. This also reduces user input error and maintains consistency. In order to so this a Spinner was used. A spinner renders a drop down function so that the user can select a specific value. (Android Developers 2016)

For the first input method a Spinner was required to ensure that the user can update the quantity of stock items that have pre-populated in the database with a quantity of zero. The Spinner therefore had to retrieve these items from the Stock Table in the database.

```
private void loadSpinnerSQLData() {
    // database handler
    DatabaseOperations db = new DatabaseOperations(getApplicationContext());
    // Spinner Drop down elements
    List<String> items = db.getAllStockItems();
    // Creating adapter for spinner
    ArrayAdapter<String> dataAdapter = new ArrayAdapter<>(this,
            android.R.layout.simple_spinner_item, items);
    // Drop down layout style - list view with radio button
    dataAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    // attaching data adapter to spinner
    stock_sql_spinner.setAdapter(dataAdapter);
}//loadSpinnerSQLData
```

Figure 33.0 Method to load database items into Spinner

```
//method to retrieve all stock items to populate spinner
public List<String> getAllStockItems(){
    List<String> stock_items = new ArrayList<String>();

    // Select All Query
    String selectQuery = "SELECT  * FROM " + StockInfo.STOCK_TABLE_NAME + " WHERE "+
            StockInfo.STOCK_QUANTITY + " =  0" ;
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cursor = db.rawQuery(selectQuery, null);
    // looping through all rows and adding to list
    if (cursor.moveToFirst()) {
        do {
            stock_items.add(cursor.getString(1));
        } while (cursor.moveToNext());
    }//if
    // closing connection
    cursor.close();
    db.close();
    // returning labels
    return stock_items;
}//getAllStockItems
```

Figure 34.0 DatabaseOperations method to get Stock Table items from database to populate Spinner. Select statement to ensure only items with a quantity of zero are selected.

For the second input method Spinners again were created to limit the units the user could select and the food types they could enter. A list of options was added to the string resource file in an array.

```
<string-array name="food_types">
    <item>Dairy</item>
    <item>Fruit</item>
    <item>Vegetable</item>
    <item>Frozen</item>
    <item>Meat</item>
    <item>Bakery</item>
    <item>Condiment</item>
</string-array>
```

Figure 35.0 String array

```
food_type_spinner = (Spinner)findViewById(R.id.food_type_spinner);
//unit spinner initialised
food_unit_spinner = (Spinner)findViewById(R.id.food_unit_spinner);
//initialise array adapter for both spinners
adapter = ArrayAdapter.createFromResource(this,R.array.food_types,android.R.layout.simple_spinner_item);
unit_adapter = ArrayAdapter.createFromResource(this, R.array.stock_units,android.R.layout.simple_spinner_item);
//set drop down resources for both
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
unit_adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
//set adapters for both
food_unit_spinner.setAdapter(unit_adapter);
food_type_spinner.setAdapter(adapter);
```

Figure 36.0 Initialise Spinners and use adapter to set layout

This cycle (and the previous) was tested by inputting data using both methods using the emulator. The Spinners and their contents were tested as well as ensuring that the data populated the database accurately. This again was checked by accessing the database data via SQLiteBrowser.

5.3.8 Eighth Cycle

This cycle was intended to be used to create clickable ListView items so that when the items were being view in the Stock ListView they could be selected and the quantity updated. It was then discovered that using AsyncTask to create the ListView meant that it was not compatible with implementing a clickable ListView. (AsyncTask Android Developers 2016)

Therefore the fifth cycle had to be amended to remove the AsyncTask function of the existing ListView and recoded.

Three ListViews were created – one to show all recipes, all stock items as well as the Shopping List. ListViews were selected as there is an unknown number of database entries for these tables and so the ListView will expand depending on the number of entries.

Two layouts are created to facilitate a Listview – one with a Listview and the other to set the rows for the ListView to be displayed.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context=".DisplayStock"
    android:background="#2787A4">

    <ListView
        android:layout_height="match_parent"
        android:layout_width="match_parent"
        android:id="@+id/display_stock_listview"
        android:layout_alignParentBottom="true"
        android:layout_alignParentStart="true" />


</RelativeLayout>
```

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="75dp"
    android:background="#2787A4">

    <TextView
        android:layout_marginTop="10dp"
        android:layout_height="match_parent"
        android:layout_width="100dp"
        android:id="@+id/s_name"
        android:layout_alignParentLeft="true"
        android:gravity="center"
        android:textAppearance="?android:textAppearanceLarge"/>

    <TextView
        android:layout_marginTop="10dp"
        android:layout_height="match_parent"
        android:layout_width="80dp"
        android:id="@+id/s_type"
        android:layout_toRightOf="@id/s_name"
        android:gravity="center"
        android:textAppearance="?android:textAppearanceLarge"/>
```

Figures 37.0, 38.0 XML showing layout with ListView and row layout showing each TextView

```
public class DataProvider {

    private String recipe_name;
    private String recipe_instructions;

    public String getRecipe_name() { return recipe_name; }

    public void setRecipe_name(String recipe_name) { this.recipe_name = recipe_name; }

    public String getRecipe_instructions() { return recipe_instructions; }

    public void setRecipe_instructions(String recipe_instructions) {
        this.recipe_instructions = recipe_instructions;
    }

    public DataProvider(String name, String instructions){
        this.recipe_name = name;
        this.recipe_instructions = instructions;

    }//constructor


    }//class
```

```
ublic class DataListActivity extends AppCompatActivity {
    ListView listview;
    SQLiteDatabase sqlDatabase;
    DatabaseOperations dBO;
    Cursor cursor;
    ListDataAdapter recipeListAdapter;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_data_list);
        listview = (ListView) findViewById(R.id.display_recipe_listview);
        recipeListAdapter = new ListDataAdapter(getApplicationContext(),R.layout.activity_data_list);
        listview.setAdapter(recipeListAdapter);
        dBO = new DatabaseOperations(getApplicationContext());
        sqlDatabase = dBO.getReadableDatabase();
        //this will return a cursor object by method in dboperations
        cursor = dBO.getRecipeInfo(sqlDatabase);
```

Figures 39.0, 40.0 A DataProvider and DataListActivity class was created for each ListView

This cycle was tested using the emulator by selecting the appropriate button to display shopping list, recipes or stock items and ensure that everything was in the database was displayed in the ListView

5.3.9 Ninth Cycle

Custom ListView for Stock Items was amended to enable user to click on an item and update the quantity. This included using a bundle to pass the data from the ListView to be viewed on another activity where the quantity can be updated, and the database amended.

```
//Setup onClickItemListener for the items in the listview
listView.setOnItemClickListener((parent, view, position, itemId) -> {
      Intent passToUpdate = new Intent(getApplicationContext(),UpdateStock.class);
       //create bundle object
       Bundle myExtras = new Bundle();
       //
       Stock passingStock;

       passingStock = stockObjects.get(position);

       myExtras.putInt("ID", passingStock.getId());
       myExtras.putString("NAME",passingStock.getName());
       myExtras.putString("TYPE",passingStock.getType());
       myExtras.putInt("QTY", passingStock.getQuantity());
       myExtras.putString("UNITS", passingStock.getUnits());

       passToUpdate.putExtras(myExtras);
       startActivity(passToUpdate);

       Log.d(TAG, "Sending values" +passingStock.getId()+ " " + passingStock.getName()+ " "+ passingStock.getType() + " " +
       passingStock.getQuantity()+" "+ passingStock.getUnits());


   });//setOnItemClickListener
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_update_stock);
    newQty = (EditText)findViewById(R.id.edit_stock_qty);
    updateQty = (Button)findViewById(R.id.edit_qty_btn);


    Bundle incomingBundle = this.getIntent().getExtras();

    passedId = incomingBundle.getInt("ID");
    passedIdDb = (passedId);
    passedName = incomingBundle.getString("NAME");
    passedType = incomingBundle.getString("TYPE");
    passedQty = incomingBundle.getInt("QTY");
    passedUnits = incomingBundle.getString("UNITS");

    Log.d("Update_stock", "Received values " +passedIdDb+" "+ passedName + " " + passedType + " " +
            passedQty + " " + passedUnits);

    text_name = (TextView) findViewById(R.id.update_stock_name);
    text_name.setText(passedName);
    text_type = (TextView) findViewById(R.id.update_stock_type);
    text_type.setText(passedType);
    text_quantity = (TextView) findViewById(R.id.update_stock_qty);
    text_quantity.setText(Integer.toString(passedQty));
    text_units = (TextView) findViewById(R.id.update_stock_units);
    text_units.setText(passedUnits);
```

Figures 41.0, 42.0 Methods to show clickable ListView where Stock Items are sent from the ListView activity to be displayed in UpdateStock activity where user can input new quantity.

This cycle was tested by using the emulator and ensuring that the correct data is sent from the ListView to the UpdateStock activity and a new quantity can be updated and the database updated.

5.3.10 Tenth Cycle

Now that the majority of the activities and much of the functionality had been implemented home navigation and individual buttons were implemented into the toolbar of each activity. (Android Options Menu Example 2016) This helped to aid navigation, to ensure the user could navigate to the main menu at any time (home) and to get information about that specific activity. Integrating these into the toolbar saved space on the activity screen and provided a consistent approach for the user.

```
public boolean onCreateOptionsMenu(Menu menu) {

    getMenuInflater().inflate(R.menu.pantry_ab, menu);
    return true;
}//onCreateOptionsMenu

@Override
public boolean onOptionsItemSelected (MenuItem item){
    switch (item.getItemId()){
        case R.id.main_help:
            //code to run when help button is pressed
            LayoutInflater layoutInflater = (LayoutInflater)getBaseContext()
                    .getSystemService(LAYOUT_INFLATER_SERVICE);

            View popupView = layoutInflater.inflate(R.layout.popup_pantry_info, null);
            final PopupWindow popupWindow = new PopupWindow(
                    popupView,
                    ViewGroup.LayoutParams.WRAP_CONTENT,
                    ViewGroup.LayoutParams.WRAP_CONTENT);

            Button btnDismiss = (Button)popupView.findViewById(R.id.dismiss);
            btnDismiss.setOnClickListener((v) -> {
                    popupWindow.dismiss();
                });
            popupWindow.showAtLocation(popupView, Gravity.CENTER,0,0);
```

Figure 43.0 Method to populate menu for each activity showing information button and home button

This cycle was tested by ensuring that these buttons appear on the taskbar and that they display the information screen for each activity as well as they bring the user to the main menu when the home button is selected.

5.3.11 Eleventh Cycle

The ListView to show all recipes had been implemented previously but now a search facility for recipes was to be implemented. This was facilitated by a Spinner populated with any in stock items (quantity>0) and a Select Statement within the DatabaseOperations that was called when the user selected an item and the search button was selected.

```
public Cursor getSpecificRecipe(String item, SQLiteDatabase sqLiteDatabase){
    String SELECT_STATEMENT = "SELECT " +IngredientsInfo.I_RECIPE_ID +
            " FROM "+ IngredientsInfo.INGREDIENTS_TABLE_NAME
            + " WHERE " + IngredientsInfo.I_STOCK_NAME + " LIKE '" + item + "'";
    Log.d(TAG, "Search for Recipe method " + item);
   Cursor cursor =  sqLiteDatabase.rawQuery(SELECT_STATEMENT,null);

    return cursor;
}//
```

Figure 44.0 method to find recipe with specific ingredient chosen by user

The ingredients needed and current stock quantity of said ingredient was the displayed and the difference between the two (i.e. what the user needs for this recipe) was automatically added to the Shopping List Table.

```
public Cursor getIngStockInfo(int id,SQLiteDatabase db){

    Cursor cursor;
    //to set column names
    String[] projections = {IngredientsInfo.I_STOCK_NAME, IngredientsInfo.ING_QUANTITY,
            StockInfo.STOCK_QUANTITY, IngredientsInfo.ING_UNITS};
    cursor = db.query(IngredientsInfo.INGREDIENTS_TABLE_NAME + " INNER JOIN " +
            StockInfo.STOCK_TABLE_NAME + " ON " + StockInfo.STOCK_TABLE_NAME + "." +
            StockInfo.STOCK_ID + " = " + IngredientsInfo.INGREDIENTS_TABLE_NAME + "."+
            IngredientsInfo.I_STOCK_ID,projections,IngredientsInfo.I_RECIPE_ID + " = "+ id,
            null, null, null, null);
    return cursor;

}//getIngStockInfo
```

Figure 45.0 method to ingredient qty and stock qty for a specific recipe

```
if(cursor.moveToFirst()){
    do{

        String name, total_qty_String;
        int qty, st_qty, total_qty;
        name = cursor.getString(0);
        qty = cursor.getInt(1);
        st_qty = cursor.getInt(2);
        qtyString = Integer.toString(qty);
        st_qtyString = Integer.toString(st_qty);
        units = cursor.getString(3);
        items = new String[]{name, qtyString, st_qtyString, units};
        total_qty = (qty - st_qty);
        total_qty_String = Integer.toString(total_qty);
        IngDataProvider dataProvider = new IngDataProvider(name, qty, st_qty, units);
        //each row of data will be added in the form of the dataprovider object
        listDataAdapter.add(dataProvider);
        dBOperations.addShoppingInfo(dBOperations,name,total_qty_String,units);
        Toast.makeText(getBaseContext(),"Items have been added to your shopping list",
                Toast.LENGTH_SHORT).show();
    //this will return true if there are more entries in cursor
    }while(cursor.moveToNext());
```

Figure 46.0 Steps showing how the cursor returned from above method was dealt with to get the quantity user needs and add it to the Shopping List Table.

This cycle was once again tested using the emulator and searching for recipes based on the in stock items. The database was examined using SQLiteBrowser to ensure the correct shopping list items had been added to the Shopping List Table in the database.

5.4 Summary

This chapter first provided details of the tools used for this implementation such as hardware and software. It then discussed the cycle process used to implement the application, based on Agile methodology principles. It mainly reported in details on the eleven implementation cycles followed to complete the first version of this new application, included screenshots of Java, XML and SQLite codes to illustrate some aspects of the implementation. Some screenshots of the application were finally presented in order to conclude this chapter.

6.0 Testing and Evaluation

6.1 Introduction

This chapter describes the testing performed by the developer following the implementation of the application. It also presents the second research questionnaire administered to the group of community group participants and parents who responded to the initial research questionnaire and pre agreed to the testing. It finally analyses and discusses the results produced during testing to evaluate the application and draw a conclusion in the final chapter.

6.2 In-house Testing

Specific testing has already been performed at the end of every implementation cycle, generally using a PC with emulator and/or Sony Experia z5 and targeting only the newly implemented features. The in-house testing discussed in this section was performed after the last implementation cycle of the application to validate all implemented features. It was performed in a single morning on the same network.

6.2.1 Developer's Testing

Two scenarios, detailed in Appendix 5, were designed to produce a Standard Test Procedure (STP). Appendix 6 presents all the testing criteria with their result. All these criteria were successfully tested first time. None of the eight test conditions needed to be repeated.

6.3 External Testing and Evaluation

A second research questionnaire was designed and provided to the group community group members who responded to the initial research questionnaire by agreeing to test the application when available.

6.3.1 Testing and Evaluation Research Questionnaire

A new research questionnaire was created consisting of eight questions (Appendix 5). Copies of this questionnaire were given to the group 15 from the original focus group (those who had android devices).

Out of the 15 copies handed out, 11were completed corresponding to a 73% response rate in a timely manner which represents a reasonable success.

6.3.2 The Respondent's Testing Equipment

The first question inquired about the hardware and software they used to evaluate the application. All respondents used an Android mobile phone. All respondents provided quite similar and positive results to the remaining seven questions indicating that the use of different devices made no significant difference to the testing outcome. Therefore it can be deducted that the Android functionality was proven.

6.3.3 The Application Appearance and Functionality

Question 2 presented the respondents with nine statements related to the application as a whole. 100% of the respondents agreed or strongly agreed with eight of those statements:

- The application is visually appealing to
- The colour scheme is appropriate
- The text is easily read
- The appearance is simple with uncluttered layout
- The navigation is easy and works well
- All functions work quickly and correctly
- The overall design is satisfying
- The overall functionality is satisfying

6.3.4 The Application Home Menu

Question 3 contained six statements all related to the application home menu. 100% of the respondents found the home menu appearance, layout, content, logo, very suitable or quite suitable. One of the focus group members commented "An attractive menu, easy to understand and navigate".

6.3.5 The Application Navigation

Question 4 proposed five statements surrounding the navigation throughout the application using buttons and clickable layouts. 100% of the respondents found the primary and secondary navigation, using year group and curriculum requirements reusing buttons and sub-menu buttons/clickable list views respectively, very suitable. The Home button and the information buttons were found to be at least quite suitable by all the respondents. The clickable list view, used to navigate to update the Stock quantity, was not as successful with

two respondents finding is very suitable, seven out of eleven respondents finding it suitable, one neither suitable nor unsuitable and the last one stating that it was not very suitable. A new design for this navigation should be implemented in the next version.

6.3.6 Add New Items Function

Question 5 probed the focus group about the add stock item facility. They were asked about the appearance and usability of the buttons and the forms, and the efficiency of the two methods to add pantry items. All of the respondents provided very positive answers by strongly agreeing or agreeing with all the seven statements made. They commented that the simplicity of the forms and the ease of completion was suitable.

6.3.7 ListViews

Question 6 was related to the My Pantry Items ListView that provides users access to display the current stock items the user had. All respondents thought that the appearance, the layout, the header and the textual content were very suitable or quite suitable.

6.3.8 Shopping List Functions

Question 7 dealt with all aspects of the Shopping List activity. Five out of five respondents strongly agreed that the activity page is useful and well laid out. They all agreed that all the functions worked correctly and quickly. Furthermore, they all strongly agreed or agreed that:

- the layout works well
- the text is easily read
- the picture size and appearance are adequate
- the buttons design is adequate
- the overall design and functionality are satisfying

6.3.9 Recipe Functions

The final question provided the focus group with six statements about the Recipes menu. All of the respondents strongly agreed that the layout of main page and the associated pages was adequate and functioning quickly. They all agreed that the content provided was informative and that they all functioned correctly. One of the users found that the recipes page was blank when they searched for a recipe – this was expected and there was a toast added to ensure the user is aware that there are no recipes associated with this stock item.

6.4 Summary

This chapter first described the testing performed in-house and presented the produced results. They established that all the application functionalities were working as required and they also demonstrated that the implementation was successful in achieving functionality across a number of Android mobile phones.

This chapter also presented the second questionnaire research organised with a focus group of community group members and parents who responded to the initial one. It presented the respondent's evaluation results, which were very positive overall and indicative of a successful implementation of their recommendations and associated design requirements.

7.0 Conclusions

7.1 Introduction

This final chapter first summarises the work carried out throughout this project. It then critically assesses the results to demonstrate if the application has met the focus groups' recommendations and satisfaction. It states whether the project successfully achieved its aim and objectives. It concludes by presenting any additional features and content, and possible improvements that could be made if the project was to be extended.

7.2 Project Summary

An initial background research was conducted on the problem of UK domestic food waste and the behaviours that contribute. The research findings allowed initially to determine the impact the problem has on the UK for example, money and produced wasted and its contribution to the environment.

By investigating the possible causes and origins of the problem, it was found that households were the most appropriate target to apply our solution. The existing solutions were researched before proposing a solution to implement in this project: a new Food Waste Management mobile application that users could access free of charge.

The various reasons that made this solution the most adequate were discussed next. Accessibility to mobile devices, in particular mobile phones has significantly increased, and the mobile aspect is critical to the design of this application was also discussed. Encouraging households to plan their weekly shop, give them mobile access to the products they have currently at home and providing recipe suggestions would help to aid householders to achieve a synergy to reduce the amount of food they waste.

A competitive analysis of the Recipe/Food Manager applications currently on the market was then performed. A total of four applications were reviewed to help draw a list of recommendations for the new application that were consequently used in a research questionnaire administered to a number of focus groups/parents. The respondents provided essential feedback and recommendations with regards to the application content, features and requirements. Those two studies were carried out to inform a list of functional and non-functional requirements to be incorporated in the design of the application.

The design of the application was strongly influenced by the Android Developers site that recommends that accessibility and usability be addressed together when developing applications. Following a user-centred design process, the user interface design was focussed on adult users in order to provide them with the most positive user experience possible. Mock-ups of the application were created to help define the user-interface appearance including the colour scheme and the specific font that would be implemented later. The architectural design was also specified including the site map and the relational database.

The application design was then ready to be implemented. The model used to implement the application was based on Agile methodology principles .The full implementation consisted of eleven individual cycles that included three main steps: analysing – developing – testing. All eleven implementations cycles permitted to successfully complete the first version of this new application that was then ready for testing and evaluation.

Together an in-house testing plan and a second external research questionnaire were sufficient to fully evaluate the application. Eight tests were performed in-house by following two Standard Test Procedure scenarios and involving various device/browser combinations. A second research questionnaire consisting of eight questions was given to the original focus groups to conduct scenario based testing to evaluate the application.

7.3 Results Assessment

A list of functional and non-functional requirements had been compiled for the application based on the recommendations made by the focus groups who responded to the first research questionnaire. Every one of those requirements have been successfully implemented in the first version of the application.

The in-house testing results were indicative of having achieved a fully functional application, which likely meet adult user accessibility and usability requirements. The second research questionnaire produced positive and encouraging respondent's evaluation results, which were indicative of a successful implementation of their initial recommendations.

The only limitation from this first version of the application is the restricted number of functions available. It was known from the start of this project that time would only permit to implement enough activities to demonstrate the potential of the application. The final

comment made by one of the members of the focus group (who teaches a budget cookery class) when responding to the second questionnaire summarises this well:

> "It is a very appealing looking application which is already very useful. When completed it should prove to be a wonderful resource to our members."

7.4 Objectives and Achievements

The five objectives defined at the start of this project have been achieved:

1. The background research conducted on the problem has provided enough evidence to successfully demonstrate why this new application was an adequate solution.

2. A competitive analysis was performed by reviewing four applications currently available to manage food spend and recipes. It greatly contributed to the project by generating a list of functional and non-functional requirements for the application.

3. A research questionnaire has been successfully administered to a variety of community group members. Its results helped in establishing what design, features, content and user requirements best suited this new Food Waste Manager application.

4. An appealing and functional design centred on behaviours to reduce domestic food waste has been accomplished, meeting the standard requirements for accessibility and usability. This design was then implemented efficiently to successfully deliver on time an application that passed all in-house testing requirements.

5. A second research questionnaire has been administered to the original focus group who tested the application. Their evaluation feedback indicated that the application has successfully met their recommendations.

7.5 Recommendations

Some recommendations for future improvement and implementations follow:

- Additional activities should be implemented to ensure that all behaviours that contribute to domestic food waste are covered. Providing users feedback, on the food that they waste, and trends on how they have managed to reduce this would be a welcome addition. More variety of style and design should be integrated into these

future activities to avoid repetitive user's interaction and keep them interested and motivated. Also substitutions should be implemented to complement the recipes and promote the creative side of cooking.

- Rating and sharing abilities for the recipes could be added so that users do not get suggested recipes that they did not enjoy again. If they were favoured they could share them on social media.

- Stock items could be inputted using barcode scanner, reducing user manual input. This could include the price of items which could be used when displaying trend analysis.

7.6 Summary

This project aimed at reducing domestic food waste by offering householders free access to a new Food Waste Manager application. An appealing and suitable design was realised based on competitive analysis findings and recommendations from a variety of focus groups. The application was then efficiently implemented, successfully tested in-house and positively evaluated by the focus group members. Overall the project has been a success as all its objectives have been achieved. More functions need to be implemented in order to make this application a favourite with UK householders.

REFERENCES

Agile (2014) *Principles behind the Agile Manifesto*. [Online]
Available from:
http://www.agilemanifesto.org/principles.html
(Accessed 23 September 2014)

Android Options Menu Example (2016) *Android Options Menu Example using getMenuInflater().inflate(), onCreateOptionsMenu and onOptionsItemSelected* [Online]
Available from:
http://www.concretepage.com/android/android-options-menu-example-using-getmenuinflater-inflate-oncreateoptionsmenu-and-onoptionsitemselected
(Accessed 13/08/2016)

Android UI Patterns (2015) *Android UI Patterns* [Online]
Available from:
www..androiduipatterns.com%2F2011%2F12%2Fandroid-back-button-take-two.html&bvm=bv.112064104,bs.1,d.d24&psig=AFQjCNH65O_Qnrh0tG684a7gN2RsH9VPzQ&ust=1453536452981515
(Accessed 05/01/2016)

AsyncTask Android Developers (2016) *AsyncTask* [Online]
Available from:
https://developer.android.com/reference/android/os/AsyncTask.html
(Accessed 10/07/2016)

Clark, Josh (2012), *Designing for touch* [Online]
Available from:
http://www.creativebloq.com/design/designing-touch-2123037
(Accessed 16/06/2016)

DEFRA (2008) A frame-work for pro-environmental behaviours [Online]
Available from:
https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/69277/pb13574-behaviours-report-080110.pdf
(Accessed 12/01/2016)

Downing, Emma. Priestley, Sara. Carr, Wendy. *Food Waste,* PARLIAMENT BRIEFING PAPER Number CBP07045 (2015) [Online]
Available from:
http://researchbriefings.parliament.uk/ResearchBriefing/Summary/SN07045
(Accessed 09/01/2016)

Felker, Donn. (2012) *Android Application Development for Dummies*. New Jersey: John Wiley & Sons Ltd. (pp.80-88)

Love Food Hate Waste (2015) *About food waste* [Online]

Available from:
http://www.lovefoodhatewaste.com/node/2472
(Accessed 10/01/2016)

Office for National Statistics (2014) *Statistical bulletin: Families and Households, 2014* [Online]
Available from:
http://www.ons.gov.uk/ons/rel/family-demography/families-and
households/2014/families-and-households-in-the-uk--2014.html
(Accessed 08/01/2016)

Material design guidelines (2016) Buttons – Components – Material design guidelines [ONLINE] Available from:
https://material.google.com/components/buttons.html#buttons-style
(Accessed 13/06/2016)

Rai, Arvind. (2015). *Android AsyncTaskLoader Example with ListView and BaseAdapter*. [ONLINE] Available at: http://www.concretepage.com/android/android-asynctaskloader-example-with-listview-and-baseadapter.
(Last Accessed 18 August 2016)

SHIFTDESIGN (2014) Domestic Food waste Insights Report. [Online]
Available from:
http://www.shiftdesign.org.uk/content/uploads/2014/09/Shift_Food-Waste-inisghts.pdf
(Last accessed 10/01/2016)

Spinners Android Developers (2016) *Spinner* [Online]
Available from:
https://developer.android.com/reference/android/widget/Spinner.html
(Last Accessed 10/09/2016)

TECHCRUNCH (2014) Lomas, N. Android still growing market share by winning first time [Online] Available from:
http://techcrunch.com/2014/05/06/android-still-growing-market-share-by-winning-first-time-smartphone-users/
(Accessed 20/12/2015)

THE GUARDIAN (2015) Kate Lyons, Glenn Swann and Cath Levett. *Produced but never eaten: a visual guide to food waste.* [Online]
Available from:
http://www.theguardian.com/environment/ng-interactive/2015/aug/12/produced-but-never-eaten-a-visual-guide-to-food-waste
(Last accessed 12/01/2016)

THE GUARDIAN (2015) Arnett George, One in four UK smartphone owners does not make phone calls weekly [Online]
Available from:

http://www.theguardian.com/news/datablog/2015/sep/08/one-in-four-uk-smartphone-weekly-phone-calls
(Last Accessed 05/01/2016)

TUTORIALS POINT (2014) *Android Tutorial Points* [Online]
Available from:
http://www.slideshare.net/bsb_2209/android-tutorial-points
(Accessed 30/12/2015)

TUTSPLUS (2013) Android SDK: Create a barcode Scanner [Online]
Available from:
http://code.tutsplus.com/tutorials/android-sdk-create-a-barcode-reader--mobile-17162
(Accessed 19/12/2015)

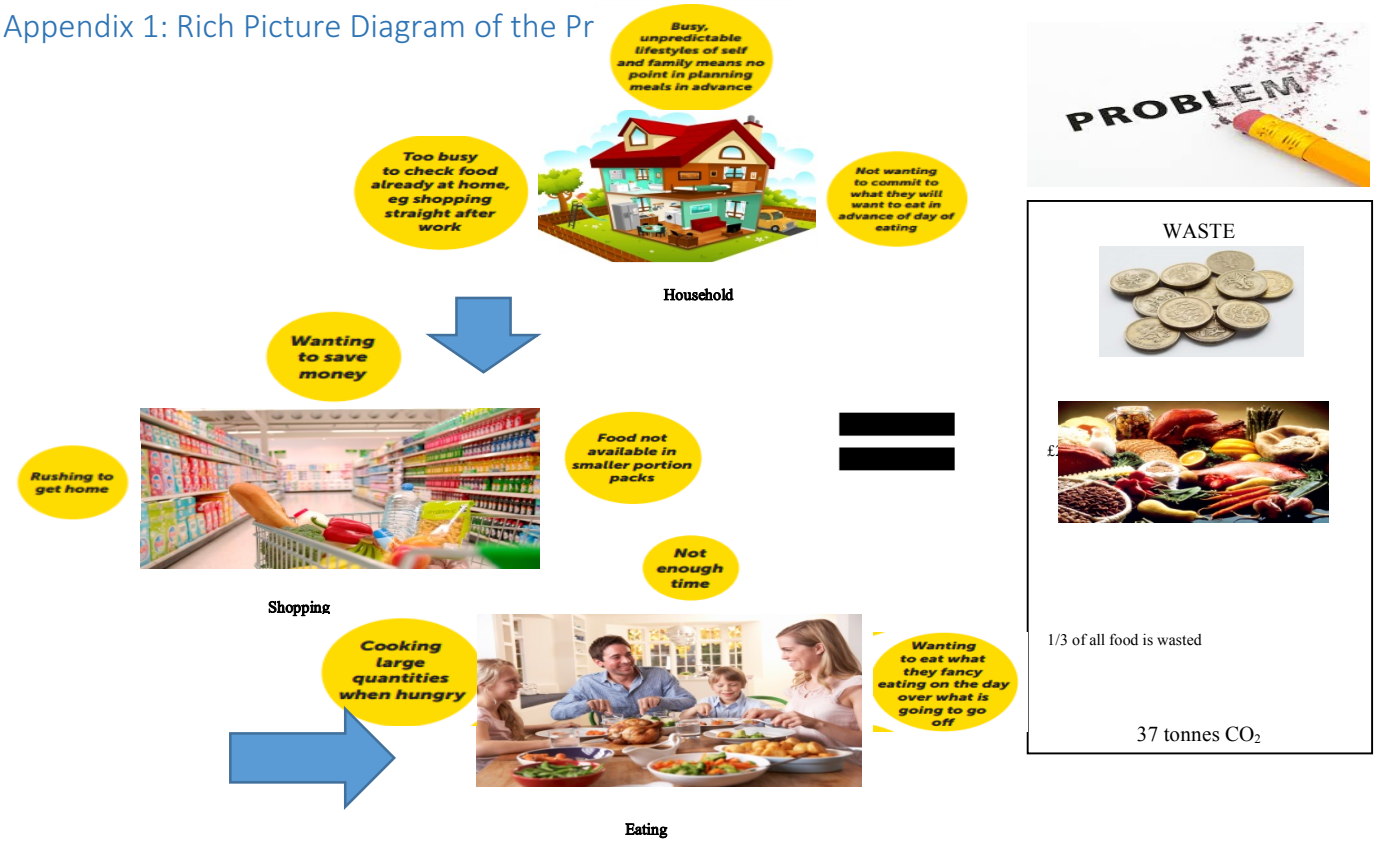
TRUSSELL TRUST (2014) Trussell Trust Foodbank Information Pack [Online]
Available from:
http://www.trusselltrust.org/resources/documents/Press/TT-Foodbank-Information-Pack-2013-14.pdf
(Last Accessed 11/01/2016)

w3schools. 2013. XML Tutorial. [ONLINE] Available from: http://www.w3schools.com/xml/. (Accessed 6 July 2016)

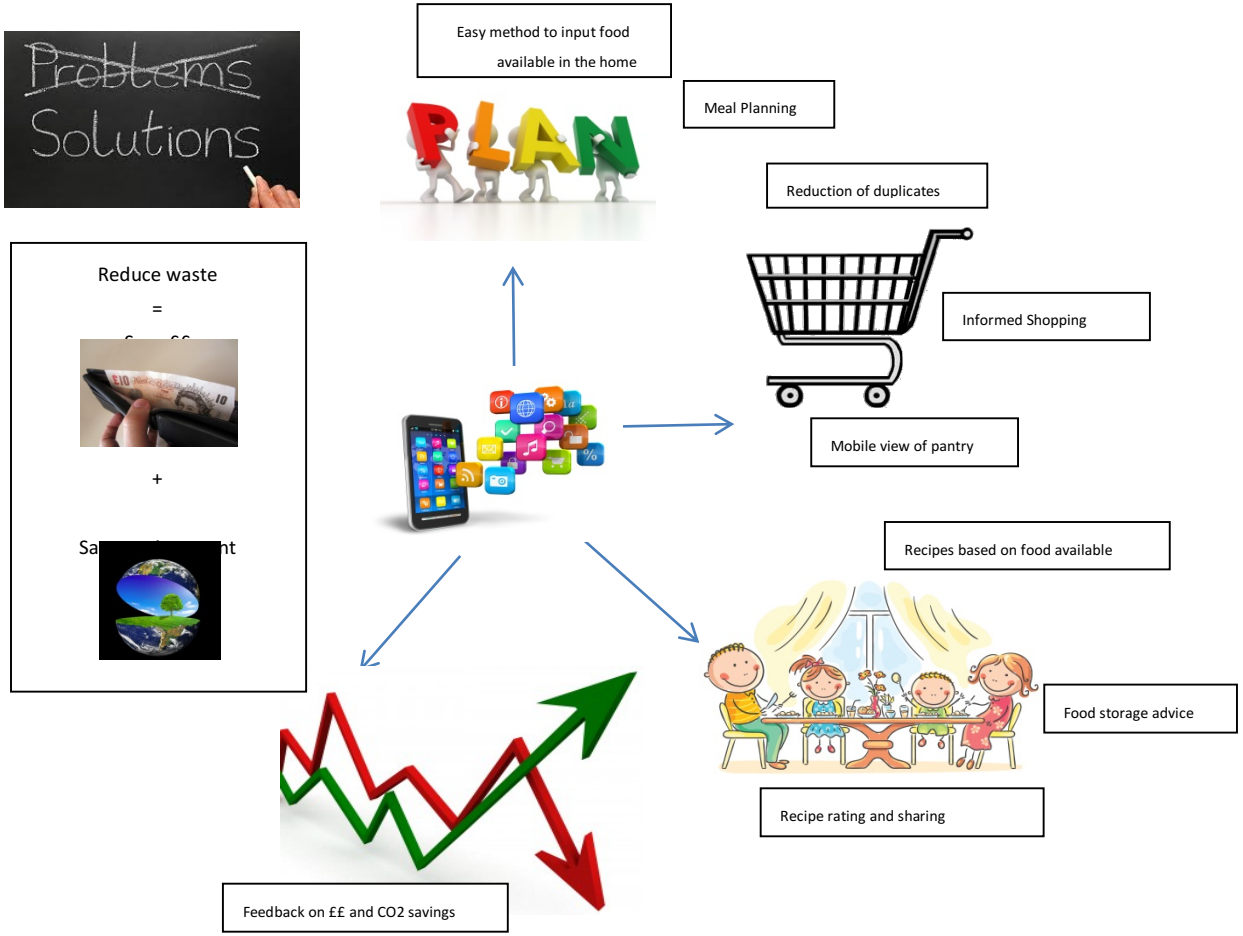
Wong, Euphemia (2013), Shneiderman's Eight Golden Rules Will Help You Design Better Interfaces [Online]
Available from: https://www.interactiondesign.org/literature/article/shneiderman-s-eight-golden-rules-will-help-you-design-better-interfaces
(Accessed 31/05/2016)

Appendices

Appendix 1: Rich Picture Diagram of the Pr



Appendix 2: Rich Picture Diagram of the Solution



Appendix 3 : Focus Group Questionnaire

Information sheet

Project Student: Katherine McCann Email: katherinejmccann@gmail.com Tel: 07749132505	Project Supervisor: Dr Charlie He Email: z.he@ulster.ac.uk Tel: +44 28701 24150

I would like to invite you to take part in a questionnaire aimed at the design, implementation and testing of a new application. This application will provide users with the ability to track their food purchases, have a mobile way to check what's in their fridge and to suggest recipes based on what they have to help prevent domestic food waste.

This questionnaire is being carried out in partial fulfilment of my MSc Professional Software Development course project at University of Ulster School of Computing & Information Engineering. Your involvement would help greatly and would be much appreciated.

This study involves questions relating to the preferred design of this new application and more specifically:

1. The mobile application name and content.
2. The features offered by the mobile application.

Questionnaires are provided to a selection of individuals who are involved in food planning/actively involved in community groups to prevent domestic food waste. It contains 11 questions and should take approximately 20 minutes of your time to complete.

Please be assured that if you agree to partake in this study, under the Freedom of Information Act, you will have the right of access to data which you have contributed. Every effort will be taken to ensure that all personal data, which may identify participants in this study, will be removed from information that will be gathered for the purposes of this research.

Hard copy data will be stored securely only until the successful completion of this programme of study and thereafter safely destroyed. Confidentiality is a vital part of this research. In the final write-up any references or quotations used from questionnaires will be anonymous.

It is emphasised that this research is for the sole purpose of the course of academic study being undertaken by the student at University of Ulster.

Please be aware that participation in this research is NOT compulsory and even if you consent to take part you have the right to withdraw from the study at any time, without prejudice, and request that any information you have contributed be removed from the record.

I would like to thank you for your time in reading this information sheet and would be very grateful if you decide to participate in this study.

Many thanks,

Katherine McCann

Can you please indicate which organisation you are involved with?

_____

Q1 Do you currently have or use a smartphone*? Please delete as appropriate.

Yes/No

*mobile phone with internet access/ability to download apps

Q2 Please indicate how aware of your current food budget and food waste prevention you are:

Please tick <u>one</u> box in each row

	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly Disagree	Don't Know
I aim to stick to a food budget.						

Q3 Please indicate how you currently manage your food shopping habits:

Please tick <u>one</u> box in each row

	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly Disagree	Don't Know
I use a list when shopping to avoid duplicates and manage my purchases.						
I use my smart phone to create a list/help me manage my spending on food.						
I do my food shopping online and use online resources						

to manage my food purchases.						

Q4 Please indicate which of the following (if any) Food Waste/Budgeting apps/websites you have used and which you would recommend?

Please tick <u>all</u> that apply

	Aware of	Used	Would Recommend
Love Food Hate Waste app	☐	☐	☐
How to feed your family for £20 a week (http://fyf20quid.co.uk) website or social media	☐	☐	☐
Yummly App	☐	☐	☐
A girl called Jack website (http://cookingonabootstrap.com/)	☐	☐	☐
Ziplist app	☐	☐	☐
The Resourceful Cook (http://resourcefulcook.com/)	☐	☐	☐
Tesco Real Food (http://realfood.tesco.com/meal-planner.html)	☐	☐	☐
Other (please specify)			_____

Q5 Please indicate the extent to which you agree or disagree with the following statements:

Please tick <u>one</u> box in each row

	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly Disagree	Don't Know
I could benefit from using a food waste/food budget application.	☐	☐	☐	☐	☐	☐
My friends/family could benefit from using a food waste/food budget application.	☐	☐	☐	☐	☐	☐

Mobile Application Name & Content

Q6 Please rate the suitability of the following names for a Food Waste App by choosing from "very suitable" to "not at all suitable":

Please tick <u>one</u> box in each row

	Very suitable	Quite suitable	Neither suitable nor unsuitable	Not very suitable	Not at all suitable	Don't Know
Frugal Foodie	☐	☐	☐	☐	☐	☐
Eat Well Waste Less	☐	☐	☐	☐	☐	☐
Use Your Noodle	☐	☐	☐	☐	☐	☐
Love Your Food	☐	☐	☐	☐	☐	☐
Environmentally App-etising	☐	☐	☐	☐	☐	☐

Are there any other names you would like to suggest: _____

Q7 Please rate the importance of the following content for a Food Waste application by choosing from "very important" to "not at all important":

Please tick <u>one</u> box in each row

	Very important	Quite important	Neither important nor unimportant	Not very important	Not at all important	Don't Know
Easy input of shopping/kitchen contents	☐	☐	☐	☐	☐	☐
Mobile view of current kitchen contents (on smartphone)	☐	☐	☐	☐	☐	☐
Recipes generated based on current food/leftovers available	☐	☐	☐	☐	☐	☐
Food Storage Advice	☐	☐	☐	☐	☐	☐
Easy method of removing items from list when used	☐	☐	☐	☐	☐	☐
Food spend trends over time	☐	☐	☐	☐	☐	☐
Ability to share data over social media – recipes/reduction in spending	☐	☐	☐	☐	☐	☐

Is there any other content you would like to suggest: _____

_____

Q8 Please rate the suitability for food shopping input to the Food Waste app by choosing from “very suitable” to “not at all suitable”:

Please tick <u>one</u> box in each row

	Very suitable	Quite suitable	Neither suitable nor unsuitable	Not very suitable	Not at all suitable	Don’t Know
Manual input – typing into a list	☐	☐	☐	☐	☐	☐
Shopping list import from online shopping	☐	☐	☐	☐	☐	☐

Q9 Please rate the suitability of how you would view food items you have not used?

Please tick <u>one</u> box in each row

	Very suitable	Quite suitable	Neither suitable nor unsuitable	Not very suitable	Not at all suitable	Don’t Know
CO2 produced by waste food and comparisons with NI/UK averages	☐	☐	☐	☐	☐	☐
Spend and trend of what items are wasted	☐	☐	☐	☐	☐	☐
Tips to help you reduce wasting these specific items	☐	☐	☐	☐	☐	☐

Q10 Assuming users have an account and are required to login, please rate the suitability of the following user profile features for a Food Waste Manager app by choosing from “very suitable” to “not at all suitable”:

Please tick <u>one</u> box in each row

	Very suitable	Quite suitable	Neither suitable nor unsuitable	Not very suitable	Not at all suitable	Don’t Know
Recording of Food Shopping habits (to view reports on individual favourites/trends)	☐	☐	☐	☐	☐	☐
Trend/Sharing of food spend (to monitor progress)	☐	☐	☐	☐	☐	☐
Trend/Sharing CO_2 produced from food waste	☐	☐	☐	☐	☐	☐
Favourite Recipes for user	☐	☐	☐	☐	☐	☐
Recipe reviews	☐	☐	☐	☐	☐	☐
Individualised tips to reduce user food waste	☐	☐	☐	☐	☐	☐

Are there any other features you would like to suggest: _____

_____

Mobile Application Testing

Q11 Please indicate if you would be willing to test the mobile application once available?

Please tick <u>one</u> box

	YES	NO
I am willing to test the app when available	☐	☐

☐

* Please note that any name provided will not be transcribed anywhere else. It will only allow me to contact you once the application is available in order to test it. Also all completed forms will be destroyed once the project is completed.

Free Comments

Please include below any comments you wish to make related to the design of this new Food Waste Manager app:

Thank you for participating in this study. Your time and effort is much appreciated.

Form return instructions:

Please return all completed forms to Katherine McCann; katherinejmccann@gmail.com

Appendix 4: Story Board

Algorithm:

SELECT 'Use Your Noodle' app
LOAD

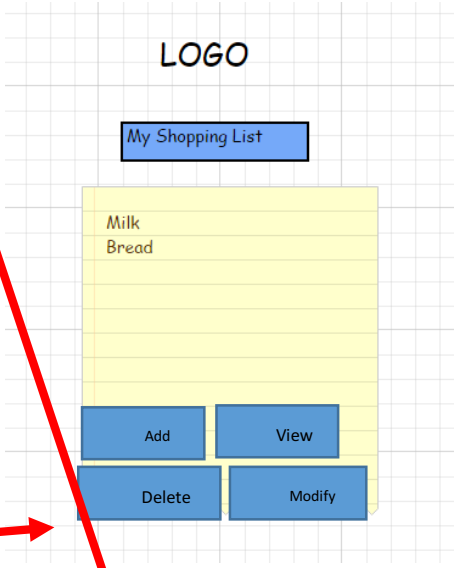
Algorithm B -

IF My Shopping List is selected
LOAD ShoppingList.Java

ELSE IF Pantry is selected
LOAD Pantry.java

ELSE IF Recipes is selected
LOAD Recipe.java

ELSE IF Trends is selected
LOAD Trends.java



Algorithm A -

IF Add Item is selected

RUN addItem()

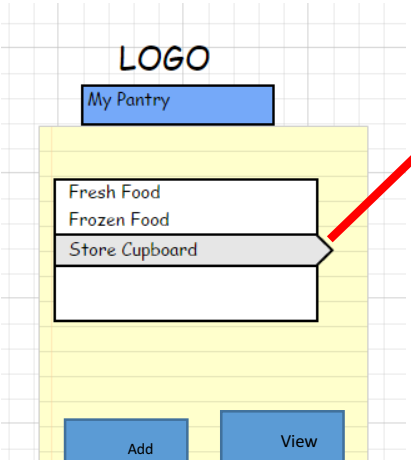
ELSE IF View is selected RUN viewAll()

ELSE IF Modify is selected RUN modifyItem()

ELSE IF Delete is selected

RUN delete()

ELSE IF Home is selected LOAD main_activity.java



Algorithm A -

IF Add Item is selected

RUN addItem()

ELSE IF View is selected RUN viewAll()

ELSE IF Modify is selected RUN modifyItem()

ELSE IF Delete is selected

RUN delete()

ELSE IF Home is selected LOAD main_activity.java

Algorithm A -

IF Recipe by Item is selected

*Select * where ingredient = 'item'*

ELSE IF Inspire Me is selected

LOAD query from database:

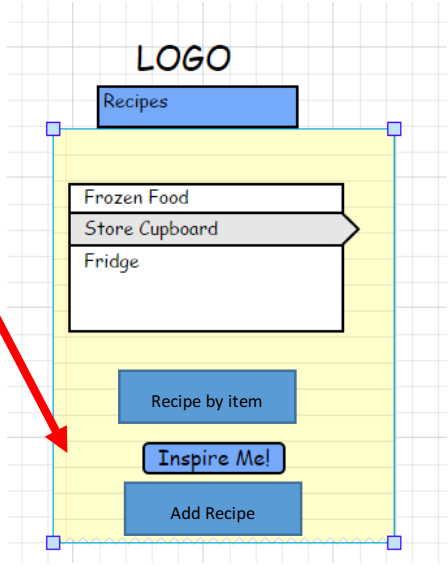
*Select * where recipe_id=(rand.no(no. in database))*

DISPLAY recipe

ELSE IF Add Recipe is selected

RUN Rec_addItem()

ELSE IF Home is selected LOAD



Algorithm A -

IF My Trends is selected

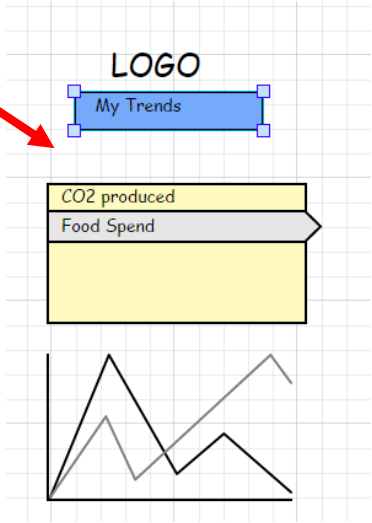
UPDATE calculation table

RUN calculation()

LOAD Calculation from Database

Display Calculation

ELSE IF Home is selected LOAD



Appendix 5: Two Testing Scenarios

Please put a tick or a cross beside each task to state whether it was successful or not. Then please complete the questionnaire below.

Scenario 1

1. Please verify that the application loads correctly on the screen and that you can see the Main Menu page with title and navigation buttons.
2. Click on the button to navigate you to the My Pantry section and confirm that is where you are taken. Click on the View My Pantry button and confirm that it should currently be blank.
3. Click on the home button on the toolbar to return to the Pantry menu. Click on the add Pantry Item button and confirm that that is where you are taken.
4. Select an item from the drop down menu and type in a quantity and click on the button to update the quantity.
5. Click on the View Pantry button to confirm that the item you added can now be viewed.
6. Click on an item in the ListView and enter a new update quantity for this item. Click the back button on the navigation bar to be taken back to the ListView. Check that the new quantity can be seen (or the item should not appear if you entered 0).
7. Click home and then click on the shopping list button. Click to add a new shopping list item – enter details.
8. Click to view the shopping list and check that the shopping list item can be seen. Then click on the home button to return to the main menu.

Scenario 2

Please put a tick or a cross beside each task to state whether it was successful or not. Then please complete the questionnaire below.

1. Please verify that the application loads correctly on the screen and that you can see the Main Menu page with title and navigation buttons.
2. Click on the My Pantry button to navigate to the My Pantry menu. Click to add a pantry item and then click again to add a new pantry item. Complete the edit texts and spinners. Then click on the view pantry to ensure your new item is visible.
3. Click on the My Recipes button to navigate you to the My Recipes section and confirm that is where you are taken. Click on the View Recipes button and confirm that it should you be able to see two recipes.
4. Click back on the navigation bar and return to the recipes section. Click on the search recipes button and select an item to search for a recipe with that ingredient. Click on the show recipes button to display recipe name and instructions.
5. Select to view the ingredients and check that the recipe ingredients and stock quantities are visible.
6. Click on the home button and navigate to the shopping list menu. Select view shopping list and check that new items have been added for that recipe.
7. Click on each of the navigation buttons and then select the information buttons on the toolbar to ensure that the appropriate information pop up appears.
8. Click on each of the navigation buttons and then select the home button on the toolbar to ensure you are brought back to the main menu.

Appendix 6: Developer Testing Results

Scenario	Step	Test Passed	Scenario	Step	Test Passed
1	1	✓	2	1	✓
1	2	✓	2	2	✓
1	3	✓	2	3	✓
1	4	✓	2	4	✓
1	5	✓	2	5	✓
1	6	✓	2	6	✓
1	7	✓	2	7	✓
1	8	✓	2	8	✓
All		✓	All		✓

Appendix 6: Testing and Evaluation Research Questionnaire

Introduction

Q1 Please indicate which hardware you are using to test the application:

Please tick all that apply

Hardware	Android version (if known)
☐ Android Tablet	☐☐
☐ Android Mobile Phone	☐☐

Application Appearance and Functionality

Q2 Considering the application as a whole, please indicate the extent to which you agree or disagree with the following statements:

Please tick one box in each row

	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly Disagree	Don't Know
The application is visually appealing	☐	☐	☐	☐	☐	☐
The colour scheme is appropriate	☐	☐	☐	☐	☐	☐
The text is easily read	☐	☐	☐	☐	☐	☐
The appearance is simple with uncluttered layout	☐	☐	☐	☐	☐	☐
The navigation is easy and works well	☐	☐	☐	☐	☐	☐
All functions work quickly and correctly	☐	☐	☐	☐	☐	☐
The application loading time is adequate	☐	☐	☐	☐	☐	☐
The overall design is satisfying	☐	☐	☐	☐	☐	☐
The overall functionality is satisfying	☐	☐	☐	☐	☐	☐

Application Menu

Q3 Please rate the suitability of the application menu by choosing from “very suitable” to “not at all suitable”:

Please tick <u>one</u> box in each row

	Very suitable	Quite suitable	Neither suitable nor unsuitable	Not very suitable	Not at all suitable	Don’t Know
The menu appearance						
The menu layout						
The toolbar content						
The application logo						
The information displayed						

Are there any comments you wish to make regarding the application menu:

Application Navigation

Q4 Please rate the suitability of the application navigation by choosing from “very suitable” to “not at all suitable”:

Please tick <u>one</u> box in each row

	Very suitable	Quite suitable	Neither suitable nor unsuitable	Not very suitable	Not at all suitable	Don’t Know
Navigation to each screen						
Back button navigation						
Home button						
Information button						
Ease/intuitiveness of navigation						

Are there any comments you wish to make regarding the navigation of the application:

Application Add Item Functions

Q5 Considering the Add Stock/Shopping Item processes, please indicate the extent to which you agree or disagree with the following statements:

Please tick <u>one</u> box in each row

	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly Disagree	Don't Know
The appearance of all buttons is appropriate						
The Add from Favourites form is easy to complete and works well						
The Add New Item form is easy to complete and works well						
The messages displayed are easy to read						
These messages inform the user efficiently						
All functions work quickly and correctly						

Are there any comments you wish to make regarding these processes:

View My Pantry

Q6 Considering the screen to view the pantry items, please rate the suitability of the design by choosing from "very suitable" to "not at all suitable":

Please tick <u>one</u> box in each row

	Very suitable	Quite suitable	Neither suitable nor unsuitable	Not very suitable	Not at all suitable	Don't Know
The appearance						
The layout						
The header content						
The textual content						
The font size						

Are there any comments you wish to make regarding the View my Pantry page:

Application Shopping List

Q7 Considering the Shopping List functions, please indicate the extent to which you agree or disagree with the following statements:

Please tick <u>one</u> box in each row

	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly Disagree	Don't Know
The application is visually appealing	☐	☐	☐	☐	☐	☐
The layout works well	☐	☐	☐	☐	☐	☐
The text is easily read	☐	☐	☐	☐	☐	☐
The picture size and appearance are adequate	☐	☐	☐	☐	☐	☐
The button design is suitable	☐	☐	☐	☐	☐	☐
All functions work correctly	☐	☐	☐	☐	☐	☐
All functions work quickly	☐	☐	☐	☐	☐	☐
The overall design is satisfying	☐	☐	☐	☐	☐	☐
The overall functionality is satisfying	☐	☐	☐	☐	☐	☐

Are there any comments you wish to make regarding this function: _____

_____

Application Recipe Function

Q8 Considering the Recipe menu and the associated functions, please indicate the extent to which you agree or disagree with the following statements:

Please tick <u>one</u> box in each row

	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly Disagree	Don't Know
The Recipes layout works well	☐	☐	☐	☐	☐	☐
The Search Recipe function is understandable	☐	☐	☐	☐	☐	☐
The options (view all, search) available are informative	☐	☐	☐	☐	☐	☐
The layout of the recipe results is adequate	☐	☐	☐	☐	☐	☐
The functionality is working correctly	☐	☐	☐	☐	☐	☐
The functionality is working quickly	☐	☐	☐	☐	☐	☐

Are there any comments you wish to make regarding the Recipe activities:

Free Comments

Please include below any comments you wish to make related to the Use Your Noodle application:

Thank you for participating in this study. Your time and effort is much appreciated.

Form return instructions: Please leave the completed forms at Reception or email to katherinejmccann@gmail.