The Smart Shopping List

Jayawilal W.A.H

149216C

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the partial fulfillment of the requirements of the Degree of Master of Science in Information Technology.

May 2017

Declaration

I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where explicitly states otherwise by reference or acknowledgment, the work presented is entirely my own.

Name of Researcher	Signature of Researcher
W.A.H Jayawilal	……..
	Date:
Name of Supervisor	Signature of Supervisor
S. C. Premaratne	……..
	Date:

Dedication

"In a walk with nature one receives far more than he seeks"

– John Muir

This thesis is dedicated to my mother, who taught me that even the largest task can be accomplished if it is done one step at a time. You are the reason I am here today.

It is also dedicated to my beloved University of Moratuwa, a place where I learnt so much. This is me giving back for what you have given me.

I also added a quote from John Muir. The quote summed up my longing for the outdoors and the beautiful memories I have while travelling, gazing up at endless plains in southern Utah mountains in an early misty morning.

Lastly, This is for you Kerrie, my longtime friend…

Acknowledgement

Firstly, I wish to give my sincere gratitude to my supervisor Mr. S. C. Premaratne for his sage advice, insightful criticisms, and patient encouragement aided the writing of this thesis in innumerable ways. I could not have imagined having a better mentor and a teacher for this research.

I also would like to thank Prof. Mr. Asoka Karunananda, B.H. Sudantha and all the lecturers of University of Moratuwa, Sri Lanka for their insightful comments and encouragement, but also for the hard questions which incented me to widen my research from various perspectives.

Mentioning my great appreciation to all my friends and batch mates of the M.Sc. in IT degree program for their valuable support, assistance and feedback, Last but not the least, I would like to thank my beloved family for supporting me physically and spiritually throughout writing this thesis and my life in general.

Abstract

Grocery Shopping is considered a tedious and less interesting task by many yet a decisive activity to be undertaken, as it is a vital part in human lifestyle. Different yet simple techniques are used in assisting to carry out these activities, some of them being taking down the items to be purchased in a writing paper or on the personal phone's text program or most commonly, creating a mental list. These traditional ways of handling grocery shopping often suffer from many drawbacks like being forgetful, time consuming and effortful, making them inefficient and unreliable. In terms, this results having time and financial losses which everyone attempts their bests to stay clear of. The outcome of this very study, 'The Smart Shopping List' is a mobile software solution which enables the users to perform their grocery shopping experience with ease overcoming above complications. The Application consists of several modules; Interactive Shopping List where the user can add/remove/cross items, Shop Locator which assist the user to find the ideal supermarket to go to so that most of the items can be bought in one place, Items Recommender powered with Apriori algorithm to remind the user of any possible missing items or items he may be interested in and 'BringMe!', which is a text-to-app feature to share the shopping lists between the users. This system has additional potential to reinforce the information gathered throughout list creation process and the purchases made by the user, in terms of identifying solid patterns by the association algorithm to produce more accurate and personalized projections. Upon the implementation, it was observed that very healthy results were developed from the data mining algorithm from the correlated data that represent an important part of the database, as well as strong usability feedback given by the users showcasing 82% of beta users being benefited from the solution. Above discussed context has been a strong background which reasons for 'The Smart Shopping List' to become a challenging project and as a novel concept not only the in the field of Social lifestyle but also in terms of Data Mining, Geo-Location and mobile technology as well.

Table of Content

List of Figures

List of Tables

Table 6. 1 - Sample Shopping Lists.....43
Table 6. 2 - Rule Selection.....45

Chapter 1
Introduction

1.1 Prolegomena

Software solutions based on mobile platforms have become very popular in today's technology world in a massive scale [1] and are used in vivid range of world domains. Not only mobile applications have become a lifestyle [2] but also the technology related is evolving by the day and demonstrating the promising potential to cope up with most of the human technological necessities. Similarly, data mining is also a broad concept which is employed in almost every field in the world [3], heavily in data sciences. Whilst applying the practicality of data mining on sales and consumer data is prevalent, in identifying purchase and interest patterns, association and association rules, is widely taken into consideration.

Although substantial amount of applications have been produced in various mobile platforms for online grocery shopping [4],[5] and comprehensively data mining techniques have been applied on sales data for interest mining, the requirement has always been present for a solution to enhance static grocery shopping experience of people. Up to the present time people use traditional way of carrying out these activities such as writing on a piece of paper [6] or memorizing items to be bought [7] which is not very reliable, and eventually leads to time and money losses. Coming up with a solution which has a capacity of properly addressing the above issues would be much beneficial for the users, in terms of both financially and lifestyle improvement.

This research presents my work on a mobile-assisted software application namely 'Smart Shopping List' for enabling the users to enrich their grocery shopping experience in to a new level. It allows the users to create shopping lists, find the ideal shop using geolocation services, receive item recommendations filtered using pattern matching recognition with higher level of accuracy using association algorithms, and share shopping lists via text-to-app service. The experimental runs and the comparisons with Weka software have shown promising results in recommending

items to the user. This chapter present the background and motivation of the research, hypothesis, objectives, problem statement, the association based data mining algorithm and the structure of the rest of the thesis.

1.2 Background and Motivation

It is widely acknowledged that the technology is growing every day especially with a substantial concentration on mobility or mobile technology. These innovative and futuristic advancements eagerly seek to undertake every human technological needs in comprehensive manner, to make human lifestyle superior and more convenient.

On the other hand, among the many things people do in their daily lives, grocery shopping is a simple yet a vital task to be done. Despite of how tedious and less desirable it may be, by some means everyone does it. They often find ways of making their grocery shopping more efficient, to certify that there are even various internet sources providing tips and directives in doing so [8] [9]. Whilst mentioning few, one of the common approaches is to take down the items on a piece of paper or a notebook and take it to the store to purchase the items. This is an inefficient practice because finding a pen and a paper at the time of the requirement or in fact to maintain a notebook for grocery shopping listing itself can be time consuming and wearisome . Besides it is very practicable to fail bringing the list to the shop because humans are usually forgetful. This also speaks of another ineffective method; memorizing the items. As mentioned above, human brain can easily forget things as it is the common human nature.

In addition to above common practices, there has been mobile technical solutions which have been found a use for this very objective. Many make use of the native text programs on their mobile phones as well as the custom applications. Withstanding the concept of a mobile phone solution serves a solid purpose over a paper or a notebook in terms of accessibility, portability and maintainability, the current developments still suffer from limitations such as having manually verify the items in the list, not being able to find the store with most of the items prior to leaving home, forgetting specific items to be added to the list, missing out on deals and not being able to instantly share the lists with others.

Not having proper means to cater these problems and limitations, people continually find themselves in distress. They make mistakes, get less-informed which leads them to face time and money losses. Having to go through the hassle of going to the store multiple times or to store-to-store to find an item which is not available can really make the shopping experience more tiresome and distasteful than it already is. If these can be properly taken care of, a considerable amount of time and money can be saved while it certainly can improve people's lifestyle in general.

This has been a strong background to arise an idea of implementing a software concept on a mobile platform to overcome above flaws. Proposed system emerges mobile technology for mobility, Data mining and association rules for interest mining on the actions carried out today and the past, Geolocation technologies for navigational responses and furthermore sharing services. This study focuses on a solution which aims to reduce the time and effort used in the grocery shopping process and making it cost-efficient via mitigating human errors, thereby providing sound value additions to the user's whole grocery shopping experience.

1.3 Problem Statement

Upon conducting a thorough literature review and gathering data and information, it was evident that there are limitations people face when they carry out their grocery shopping activities such as having manually verify the items in the list, not being able to find the idea store beforehand, forgetting specific items, missing out of possible promotions and deals and not being able to instantly share the lists with others. Even though the mobile technology is improving and data mining techniques has been adopted by many subject areas, application of mentioned concepts on mobile assisted grocery shopping has not sufficiently been made in mitigating flows in similar scenarios. It was also observed that significant amount of time and money is wasted and in the end the individuals have poor experiences due to this.

As the current process is manually-driven and consists of many limitations, the need of properly addressing them and solving was tremendously important. A 'smarter' software solution on a mobile platform to effectively maintain shopping lists, analyze previous data and suggest items, provide location services and enable easy sharing,

would be much beneficial for users, as such solution would not only save time and money but also the user experience would greatly improve as a whole.

1.4 Hypothesis

If a mobile software application employed with geolocation services and modern data mining association techniques, namely Apriori algorithm for interest mining can be successfully implemented, users will be able to overcome majority of the difficulties they face during their grocery list creation and shopping, while saving time, effort and money with the addition of an improved user experience.

1.5 Objectives

Problem statement section in this document describes the how the grocery shopping process can be made effective if such a software solution with employed with such services is made possible. I would like to explore further on the application domain and the ability of the using of data mining techniques on interest mining to perform the suggestions required. It can be broken down as follows,

I) To critically study materials on grocery shopping lifestyle, mobile-assisted grocery shopping and data mining techniques for interest mining.

II) To do an in-depth study of data association with a particular emphasis on Apriori algorithm for interest mining.

III) To develop a software solution employed with geolocation services and Apriori algorithm for suggestions.

IV) Evaluate the effectiveness, accuracy and usability of the system.

1.6 Proposed Solution

Smart Shopping List, the proposed software mobile based system would enable the user to create shopping lists, add, remove, edit, cross and uncross items in the list and effectively maintain shopping lists via an interactive workspace. In addition, the users would check the stores nearby using the ‘Nearby’ feature and select a store to verify

the item availability and the projected bill for the available items from the list. Also user can view available discounts and promotions active in the shop. Upon payment with their loyalty card, the users can also verify the status of the items in the list. 'Item Recommender' module analyzes previously items bought and previous shopping lists and suggest items further. 'BringMe!' feature is a text-to-app service which enables the users to share the list with each other. The date mining module of the proposed system performs data association mining using Apriori algorithm on; previous purchased items of the user, previous created shopping lists and previous purchases of the other users. A user centered design is done to capture the cognitive aspects to guarantees the user-friendliness of the system as the users may not be tech-savvy.

1.7 Structure of the Thesis

The rest of the thesis is organized as follows. Chapter 2 critically reviews the literature on Grocery Shopping lifestyle, mobile technology advancement and data association for interest mining and identify the research problems. Chapter 3 is about the data association for interest mining and suggestion. Chapter 4 presents the novel approach to use Apriori algorithm for interest mining. Chapter 5 and Chapter 6 describe the design and implementation respectively. Chapter 7 provides an end-toend run through of the software in practice and Chapter 8 is an evaluation of the effectiveness of the 'Smart Shopping List' and the accuracy of the mining algorithm used with additional parameters. Chapter 9 concludes the research with a note on further work.

1.8 Summary

This chapter gave an overall picture of the entire project presented in this thesis. As such it was discussed in the background and motivation, problem definition, hypothesis, objectives and brief overview of the solution. Next presents a critical review of literature on Grocery Shopping lifestyle, mobile technology advancement and data association for interest mining.

Chapter 2

Overview of Mobile Assisted Grocery Shopping and Applying Data Mining for Purchase Interests

2.1 Introduction

Chapter 1 gave an all-inclusive representation of the overall project described in this thesis. This chapter provides critical review of the literature in relation to grocery shopping lifestyle and the challenges faced and the technological advancement towards mobile assisted grocery shopping. In addition, further assessment has been made on applying data mining techniques with a focus on data association for purchase-interests mining. A review of past researches have been presented for this purpose under the line-up of the following sections, namely early developments, modern developments and challenges. At the end, this chapter defines the research problem as lack of software support to handle static or general grocery shopping and, identifies that a mobile software application employed with Apriori algorithm and Geolocation services can be used to properly address the problem.

2.2 Grocery Shopping and Human Lifestyle

Grocery shopping and human lifestyle has always been an interesting domain basis for many researches as the importance has significantly arisen over past several decades to unveil the correlation between the human behavioral patterns and sales. Some statistical studies are based on data for a specific region while others discus about concept in a broader magnitude. These behavioral analyses reveal fascinating facts which are useful information to identify process oriented requirements and possible improvements which can be made to enrich the user experience.

A survey has been conducted by Bassett R., Beagan B. and Chapman G. observing the connection between family household and grocery store [10]. They discuss about how women are more involved in a grocery shopping process and how shopping lists mask a host of activities and tasks that are undervalued because they are unseen and

unrecognized. Their findings include that most of the families create a shopping list while over half of them wrote a list took it with them on their grocery trips, and some individuals maintain a mental list instead of a physical list. There were also a set which made use of a combination of written list and mental list and also a set of non-list members.

Some of the above findings had also been made accurate a comprehensive research conducted by Tomas and Garaland [7]. The facts that their studies have found that most grocery shoppers used a list on their last shopping trip further prove that most of the consumers prefer maintaining a list as a shopping organizer or a budgeting assist mechanism.

Another very statistical group of analysis have been made by the joint effort of Hartman Group and FMI (Food Marketing Institute) on the grocery shopping trends in the United States. One of their recent report, 'U.S. Grocery Shopping Trends 2014 Overview' provides key insights on American grocery shopping behavior which most of them can be applied in a global scale [11]. The survey indicates that the shoppers are becoming less likely to choose any one store to satisfy all their needs. Shoppers are optimizing their satisfaction store by store and by department. This is a strong point which hints out of a clear deviation from the 'primary store' concept which had been the de facto for last few decades. Shoppers now are considering different options to carry out their grocery shopping rather than sticking to one go-to store. Having a mechanism to find new stores nearby would greatly make benefit to the shoppers and improve the quality of the process as well.

It further finds that women are the major part of the household grocery shopping but men are actively engaged in the process too. Figure 2.1 illustrates, the grocery shopping behavioral patterns and the parameters with their distribution among men and women.

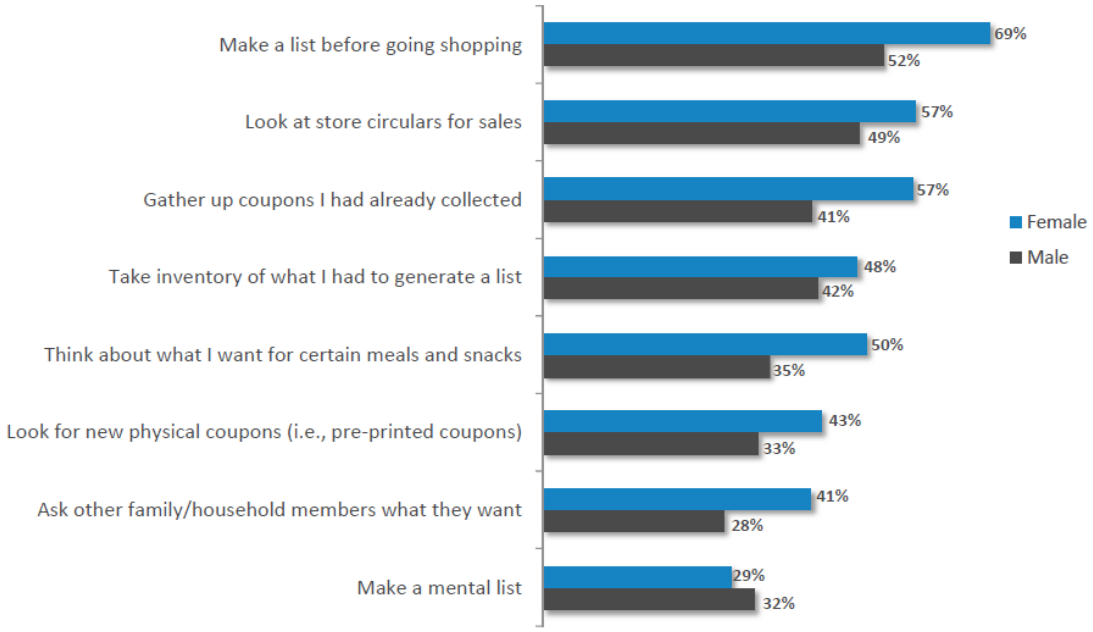


Figure 2. 1 - Shopping Behavioral Patterns Distribution among Sex (Source Hartman Group Shopping Topography Study, 2012)

The study further converses shoppers grocery list making habits. While many shoppers build shopping lists, it was evident that the young adult shoppers wait until the last minute to build their grocery lists. This information clearly demonstrates that if a grocery shopping assisting solution is to be implemented, accessibility and efficiency have to be thoroughly considered as its core features. If some shopper create lists in the last minute and the suggested software application does not allow them to perform their tasks quickly and efficiently with a very accessible manner, the likelihood of the proposed solution becoming successful is slim. Figure 2.2 further indicates the behavioral patterns of the shoppers in making grocery lists against their age.

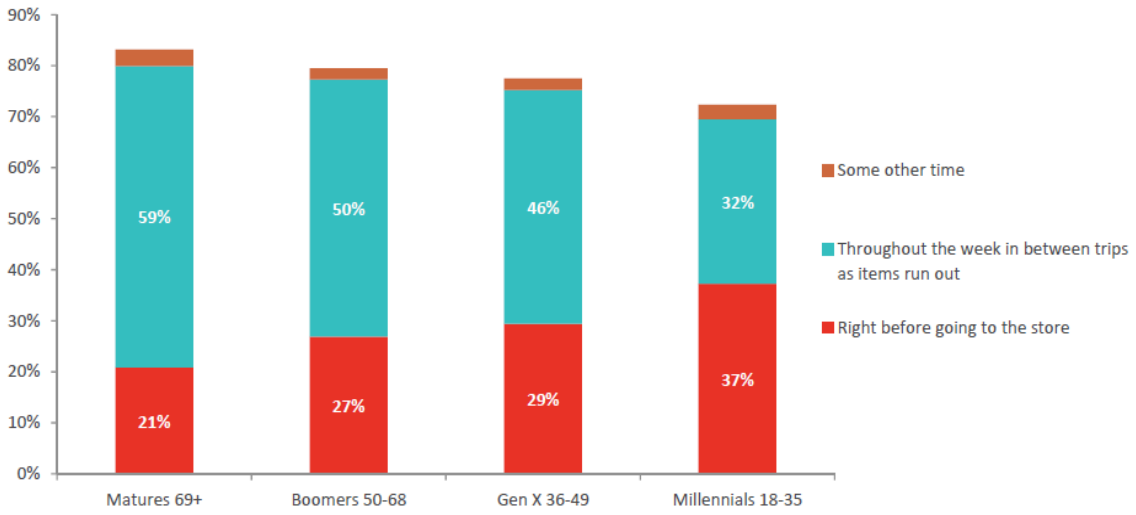


Figure 2. 2 - When Generations Make Their Grocery Lists (Source FMI: U.S. Grocery Shopper Trends 2014)

It was also explored by the same study that Millennials (or Gen Y) not only make grocery lists in the last minute but also are not particularly interested in relying on a fixed set of list items. For younger generations in particular, planning for a shopping trip is much more likely to be about building a meal or other eating occasion rather than stocking up the pantry with a list of basics and trusted items that a meal can be built from later (Figure 2.3).

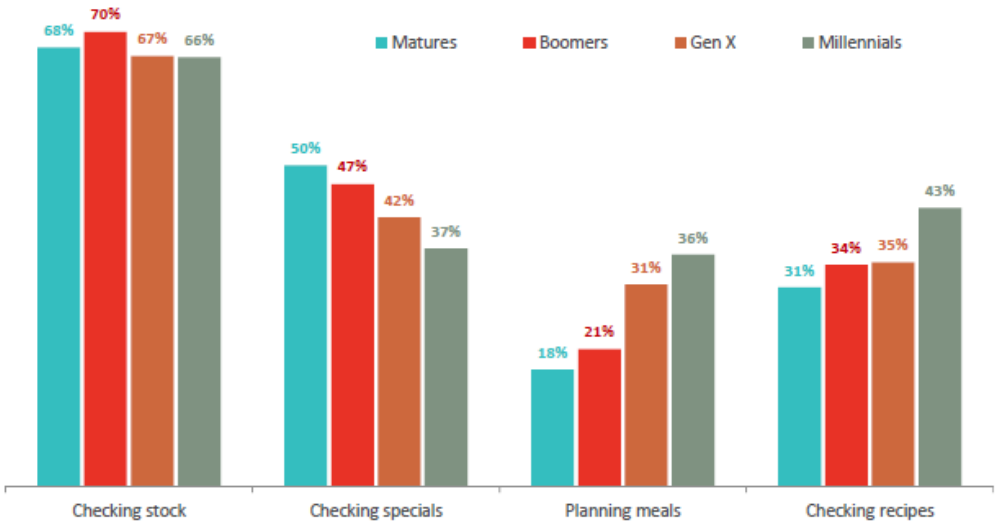


Figure 2. 3 - How Generations Are Inspired to Make Shopping Lists (Source FMI: U.S. Grocery Shopper Trends 2014)

Hence it was apparent that they are willing to broaden their grocery criteria with a variety of items as long as they are useful to their needs. Item suggestion or recommendation assisting method would certainly help in a case similar to this.

Another study by Arnaud, A. and the tem, also shares a similar proposition in terms of the shopping behavior of the millennials [12]. In the beginning, they evaluate the existing materials available on the use of shopping lists and determine that only a handful research has been made especially about millennials despite the importance of the very topic. Their study further reveals that the Gen Y prefers to buy cheaper items but is willing to pay for fresh and quality items. This explains that most individuals may be interested in good deals or promotions available at the store. Arnaud further explains his finding on the reasons for millennials use shopping lists, naming a few; to save time in the store, to save as memory aids, to control the expenditure and as a goal achievement purpose and after he concludes his study.

Reviewing above research materials, several key insights can be extracted which might be of help to identify limitations and for further development of the proposed solution. Table 2.1 shows the digest of the learnings above,

	Observations	**Presumption**
1	Many shoppers make shopping Lists	Since people believe in making a list for their grocery shopping, it is clear that if a tool to be built for this purpose it would be utilized well.
2	People are no longer stuck in only one store to do their grocery shopping	Having a mechanism to find new stores nearby would greatly make benefit to the shoppers and improve the quality of the process as well.
3	Young shoppers make their shopping lists last minute	If a grocery shopping assisting solution is to be implemented, accessibility and ease of use have to be thoroughly considered.
4	Millennials do like to keep their options open for grocery shopping	Item suggestion or recommendation assisting method would certainly help in a case similar to this.

5	Many shoppers use shopping list as a cost management mechanism	Getting to know of how much they will spend at the store in advance would be of great benefit.
6	Millennials and many shoppers in general prefer buying cheaper groceries	Letting the user know of any ongoing discounts, deals and promotions would be ideal

Table 2. 1 - Observations Captured and the Respective Presumptions

2.3 Mobile-Assisted Grocery Shopping

There have been several technological developments in mobile-assisted shopping in past few decades. Today's reach of mobile smartphones and the availability of the mobile development technology have certainly contributed these innovative advancements and keep growing. Below are some notable studies and improvements carried out on mobile-assisted shopping lists creation.

2.3.1 Application I: The Hybrid Shopping List

Heinrichs F., Schreiber D. and Schöning J. under the patronage of Prof. Dr. Antonio Krüger, worked on a project to create a prototype for a hybrid mobile application combining the advantages of paper and electronic shopping lists [13]. An initial study was carried out to understand the creation and usage of paper based shopping lists. They attempt to find the notion behind the concept of natural note taking and the consumer intent in retail. They further develop a model for the consumer shopping intent in retail which is shown in the Figure 2.4,

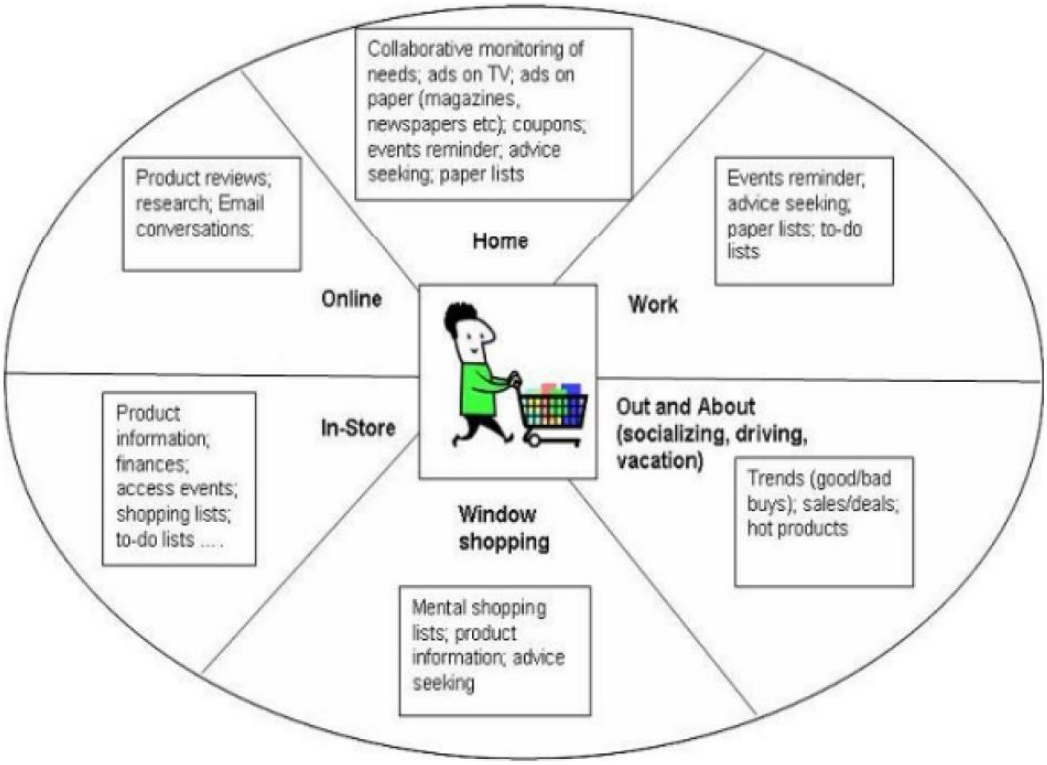


Figure 2. 4 - Multimodal of Shopping Lists and Consumer intent in retail

Then a secondary investigation was made to identify the properties of electronic shopping lists. Findings which were extracted from the initial study, put together with the secondary analysis, a functional matrix was created to design the prototype. Prototype was a system which uses Client-server architecture with a Mobile GUI. The implementation of the prototype was mainly made using Java and Android with a hardware support of Motorola and a Samsung Galaxy S with a Nokia digital pen. Users would write the list using the digital pen and the activity would be captured and be available at the mobile GUI. The project laid paved the way for mobile assisted grocery shopping creation process, however further studies were much required to investigate how many other modalities than pen and paper can be applied and facilitated in the shopping list creation process.

2.3.2 Application II: Intelligent Shopping List

Marcus Liwicki and the three team members (Sandra Thieme, Gerrit Kahl, and Andreas Dengel) developed a system which automatically extracts the intended items for purchasing from a handwritten shopping list [14]. This intelligent shopping list relies on a categorization of the products which is provided by the shopping mall. The system identifies handwritten items in a shopping list by the use of a digital paper. Using the data transmitted to computer, the handwritten data is understood by matching the data against an ontology. Promising results were shown however since the ontology is provided by the mall and very specific and narrow, it has to be enlarged so that the users are able to any items they prefer.

2.3.3 Application III: Multimodal shopping lists

Another interesting development would be the work concluded by Jain and the team regarding a developing a prototype for creating shopping lists from multiple source devices like desktop, smart phones, landline or cell phone in different formats, essentially structured text, audio, still images, video, unstructured text and annotated media [15]. An evaluation was done with 10 participants in two week. Their goal was to further analyze the shopping list creation and management process. Based on their findings, they give the recommendations to develop interactive features for the systems made for managing shopping lists.

2.3.4 Application IV&V: Grocery Retrieval System & Mobile services

Similar to Marcus Liwicki's study, Nurmi and the group introduce a product retrieval system that maps the content of shopping lists written in natural language into the relevant real world products in a supermarket [16]. The system was developed having shopping basket data as the base which they had gathered from a large local Finnish supermarket.

Furthermore the new architecture designed by Wu H. and Natchetoi Y. enabling efficient integration between mobile phone applications and Web Services with the

help of XML compression features [17]. Using this architecture, they have implemented a mobile shopping assistant which has multiple input modes such as camera, voice and Bluetooth. While there still more promising work to be done to further improve the framework, they conclude their study with their plan of releasing the work to public as a generic library.

Interestingly, Most of the applications focused on improving the input method to effectively create shopping list, while none of them have taken look at the border view on the shopping list creation process and, the consumer patterns which were learnt in section 2.2 to further enrich the shopping list creation experience, such as applying data mining techniques for interest mining.

2.4 Data Mining on Purchase Interest and Consumer Buying Patterns

'Studies find that it's part of our psychological makeup to repeat the same things over and over. Essentially humans are very predictable, and stores often take the full advantage of that' [18]. In the controversy over using consumer's previous sales records to predict his buying patterns and using them against him to formulate more sales, data mining has been employed vastly in the retail domain and it is a billion dollar industry today.

Statistical studies often find that there could be a significant relationship between the customer behavior and the items that he purchases. Raicu and the team has done a study to understand the customer preferences on products physical characteristics using data mining [19].The fact that today's high demanding customers and the increasing competitiveness, it requires to have informed application of customer learning and a vivid understanding of the corporate domain, and this laid the very foundation for them to present with a theoretical data mining framework to perform such actions in a systematic and scientific way. The framework mainly consists of two parts; data gathering part to collect relevant and unbiased data and the data mining part to discover the relationships. Data mining is done using 3 approaches; Clustering with K-means, Association via Fuzzy and LSA, and Visualization with MDS. Findings show that the data mining, does offer the industry world with some certainty to leverage sales risks and to focus on targeted marketing to maximize profits.

However Raicu's study mainly focuses more on finding customer requirement pattern for product development eg. Auto industry which opens up the opportunity for more researches to be carried out on finding consumer purchasing trends using their previous sales records.

Additionally, the two Guptas, namely A.K Gupta and Chakit Gupta have conducted another study to identify the importance of the role played by data mining in the field of financial and sales domain [20]. The advantages were clearly demonstrated as data mining can be used to uncover valuable customer related information, like Customer buying tendency, profiles of customers, industry analysis and customer purchasing actions. Their study also addresses the key issues and the obstacles in managing and retaining customer relationships and suggests that a business could build a diverse set of models which can capture and cope up with the different requirements and the preferences of divergent customer types. The Guptas further take their learnings to a profound level to converse about the tactics to be used as data mining techniques to analyze customer data. Classification, Regression, Link Analysis and segmentation were taken in to account. Followed by a case study which differentiate the traditional way of marketing and data mining approach they conclude their study. Guptas's study addresses keenly on the advantages of using data mining techniques over the traditional approaches to increase marginal profits. A broad analysis is done on the subject of applying data mining in to financial statistics and addressing key issues is doing so. As the study was done on a holistic manner on the financial domain, which leaves some research gaps to be fulfilled by future studies to be carried out on domain specific level and strong consumer buying pattern algorithms.

2.5 Association Rule Mining (ARM)

Raorane A. has done a very comprehensive study with Kulkarni R.V. to identify the consumer behavior, his psychological state at the time of purchase and how data mining method can be applied to improve ordinary method [21]. Their argument was despite of how vividly used and popular data mining would be in the retail domain it is critical to select the right mining technique to mine the data. First they carry out a study to understand why a person would buy a specific item and perform a detailed analysis on consumer behavior. The knowledge gathered from the initial study and a

secondary analysis which done to find the most appropriate mining technique to find buying patterns, is put to an application. Association rules are found to be favorable in determining interesting correlations among items in datasets. After a sound example case study with association, they make their conclusion that data mining system are useful to study consumer behavior in supermarkets and retail departmental stores. Also the further add they can be used to find the association of the customers with the different products.

A similar study on evaluating data mining techniques to find purchase items association towards the customer had previously been made by Watada and Yamashiro [22]. Their objective was to improve traditional data mining techniques by applying a few methods including fuzzy clustering and principal component analysis. Improvements were suggested for the many defects included in the conventional techniques. Furthermore, association rule is employed to mine rules for customers using their sales data in a fiber industry.

Above studies clearly emphasize that association rule mining would be the most preferred way of data mining techniques to be applied to uncover the consumer behavior and their correlations to their purchases.

2.6 Apriori Algorithm

While there are several association algorithms are available, selecting an appropriate one for the proposed solution is at utmost importance.

Two Czech researchers namely Turčínek and Turčínková had explored the use of association rules in determining the consumer behavior [23]. They carried out a survey with 1127 individuals which would be a close representation to Czech population to identify problems of shopping for meat products. The objective of the survey was to explore possibilities of data-mining techniques in processing of customer preferences. During the data analysis, they use 2 methods for generating ARMs; Apriori algorithm and FP-grow algorithm which both of them run in Weka software. Results were more favorable towards the Apriori algorithm as it had provided finer data and, the data had to be reduced to extreme values prior to be fed to the FP-Growth since the algorithm works only with binary input data. In the end, they

agree with the very concept of using association rule mining in customer research as it can provide more inside on customer preference.

Similar comparison has been made by B. Yıldız and B. Ergenç in comparing the two association rule mining techniques [24]. They used FP algorithm and Matrix Apriori another enhanced version of basic Apriori algorithm since the native implementations suffers from bottlenecks in its candidate generation phase as it requires multiple passes over the source data. Two case studies were carried out using two datasets per algorithm to see their performance over datasets with different characteristics and the causes for the performance differences. Overall Apriori outperformed FP-Growth in total performance and finding item sets were faster as well.

Adding further value, Prof. Venkatachari has done an in-depth case study of a Mumbai retail store to do a market basket analysis using FP growth and Apriori algorithm [25]. The main objective of the research was is to see how different products in a grocery store inter-relate. The algorithms were used to find frequent items using rapid miner and R programming. Proving the above studies, FP- Growth showed poor performance results in both the tools which bring us to the conclusion that native Apriori algorithm is ideal for identifying customer purchase patterns using sales data in small retail business. Another advantage of Apriori algorithm is that it calculates more sets of frequent items which is very advantages especially when the database is smaller and generating more frequent items is always better if more suggestions to be made.

2.7 Future Challenges

This section briefly summarizes the future work identified by the researchers for their researches which had been previously reviewed above. There are listed below in Table 2.2.

Research	Future Work
Bassett R.(2008)	Suggests further research on the context of grocery list creation and use takes place.
Heinrichs F.(2011)	Improve the application further to focus on the interaction schemes that come digital pens in combination with mobile devices and, to process other input modes apart from pen and paper. eg: Speech
Liwicki M.(2011)	Facilitate the application so that it automatically generates user models of preferred items and used abbreviations.
Jain J.(2009)	Perform a larger survey to uncover more processes that users go through in the shopping list creation process.
Nurmi P.(2008)	Performance improvements for their system by collecting more slang expressions, colloquialisms, abbreviations etc.
Wu F.(2007)	local cache management needs to be more efficient, Pre-fetch data from Web Services, guarantee data consistency and solve data conflicts. Also the solution would be made available as a generic library. This provides mobile application developers an easy access to Web Services.
Turčínek .P(2015)	Plans to Likert scales integration to the solution.
Yıldız B.(2010)	Plan to conduct a study that will help to propose a new association rule mining algorithm that combines the strengths of both the Matrix Apriori and the FP-Growth algorithms.
Raicu D.(2003)	Implementing an algorithm to find segments of people being interested in a certain item is suggested so that it help the managers focus their advertising, promotion so that the time and costs will be significantly reduced.

Table 2. 2 - Domain researches and their future work

2.8 Problem Definition

During the study, it was evident how vital the grocery shopping is in daily life and how the preparation in doing so would save money and time. Also it was learnt that people carry out their grocery shopping in different ways so is their perspective in creating shopping lists. The Literature Review further has identified various requirements and grocery shopping behavioral patterns which is discussed in section 2.2 which could be technically assisted to further improve the quality of grocery shopping list creation process. Table 2.3 below summarizes the presumptions made on the grocery shopping behavior (Table 2.1) and whether users have been satisfied by the key applications discussed in this chapter section 2.3.

	Presumption/ Feature					
Research Prototype	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**
Heinrichs F.(2011)	x		x			
Liwicki M.(2011)	x		x			
Jain J.(2009)	x					
Nurmi P.(2008)	x					
Wu H.(2007)	x		x			x

Table 2. 3 - Feature Coverage of the Research Application Prototypes

According to Table 2.2, Despite of the many solutions are have been implemented for mobile-assisted grocery shopping process which had been already discussed, it is evident that most of the identified the requirements had not been addressed and further development is needed in this regard. Therefore, it was advisable to go for a customized solution of the existing approaches, instead should go for a complete solution which is reliable, user-friendly and easy to use.

The problem is intended to be addressed using Android-based mobile software solution technology. This is mainly because the proposed solution should be accessible and reachable all the time to the user and smart phones serve just the

purpose. In the local context, Android has the larger share in terms of the smart phones and is open-source which is an added advantage. Also Android apps even suitable for seamless integration to large systems backend which the proposed system will comprise of. This makes Android the underlying framework for the solution. A comprehensive outline of the technology behind 'Smart Shopping List' will be discussed in Chapter 3.

2.9 Summary

This chapter presented a comprehensive and critical literature review on consumer grocery shopping behavior, mobile-assisted grocery shopping, applying datamining for mining purchase interests, ARM and association rule techniques. The review further identified the research problem as the inadequate attention to most of the consumer behavioral pattern necessities in the process of shopping list creation. Next chapter will discuss the technology to be used for the solution.

Chapter 3

Technology behind the Smart Shopping List

3.1 Introduction

This chapter presents an all-in-one overview on the technologies adapted in implementing the proposed system ‘Smart Shopping List’. Technology selection was done carefully in a way that they enable us to do the development seamlessly while it does not hinder the usability of the solution. The content of this chapter is lined up under the headings of the technologies used respectively and concludes with a brief summary in the end.

3.2 Smartphone Platform

Implementation of the proposed solution was decided to be done as a mobile application due to many reasons. Due to the sheer behavior of the solution there has been a requirement of a platform which has portability and mobility. The solution has to be available to the user wherever he goes and mobile phone was an ideal answer to this. In addition, Today’s reach of the mobile phones, smart phones in particular is massive and everyone has a smartphone in hand. Smartphone usage is keeps improving by the day and this made smartphone the perfect candidate for it. Figure 3.1 gives a clear idea on what is to come [26].

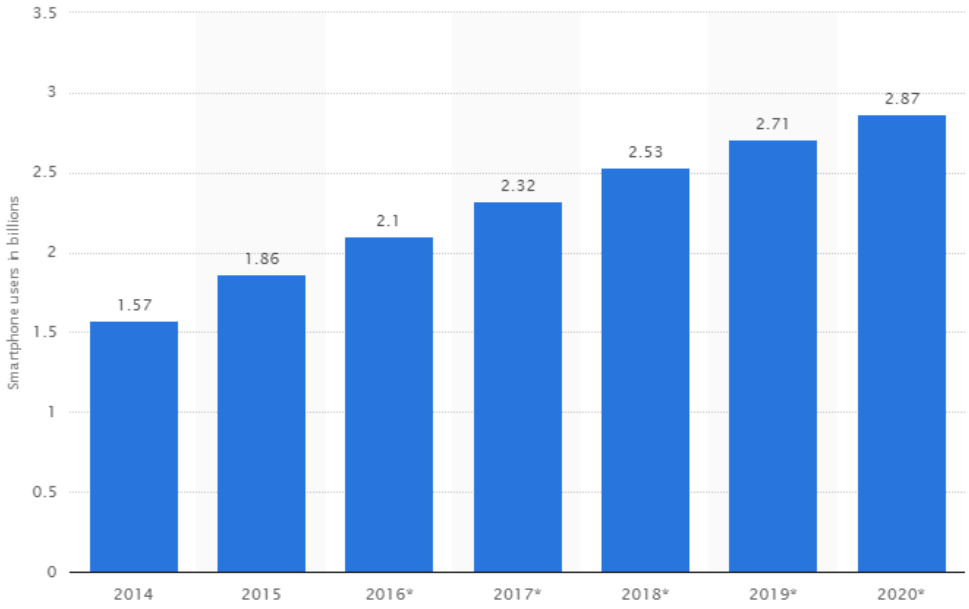


Figure 3. 1 - Number of smartphone users worldwide from 2014 to 2020 in billions (Source www.statista.com)

3.3 Android OS

Android OS as was selected as the platform operating system considering the Android has the largest smartphone distribution worldwide. Figure 3.2 shows the worldwide smartphone OS market share from 2013 to 2016 [27].

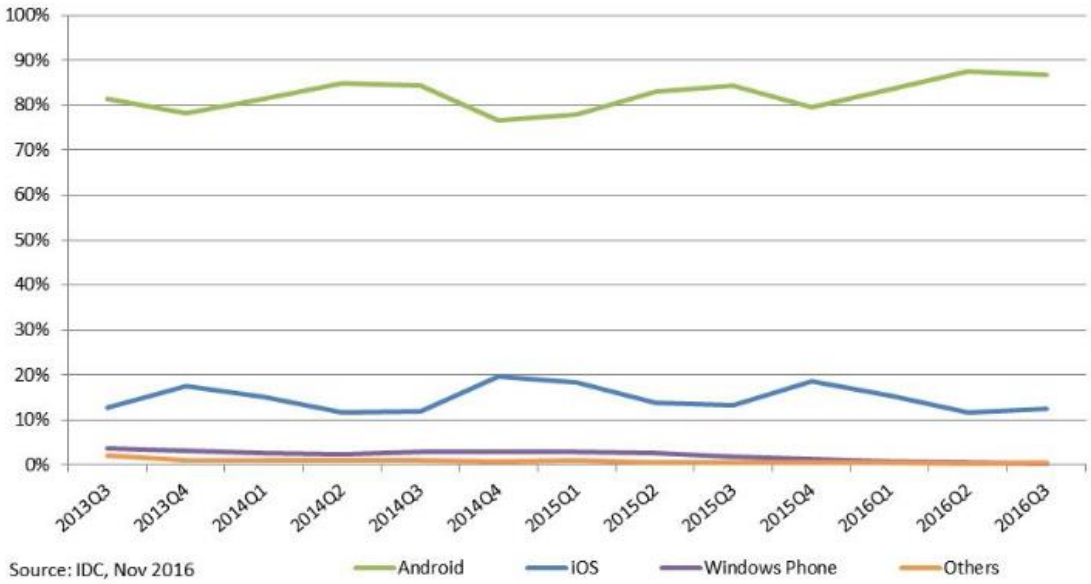


Figure 3. 2 - Worldwide Smartphone OS Market Share in units shipped (Source www.idc.com)

Also in terms of the local distribution of smartphones android hold majority of the market share. Figure 3.3 shows the regional distribution of smartphone OS in 2015 [28]. This makes android OS the ideal platform to develop the proposed solution.

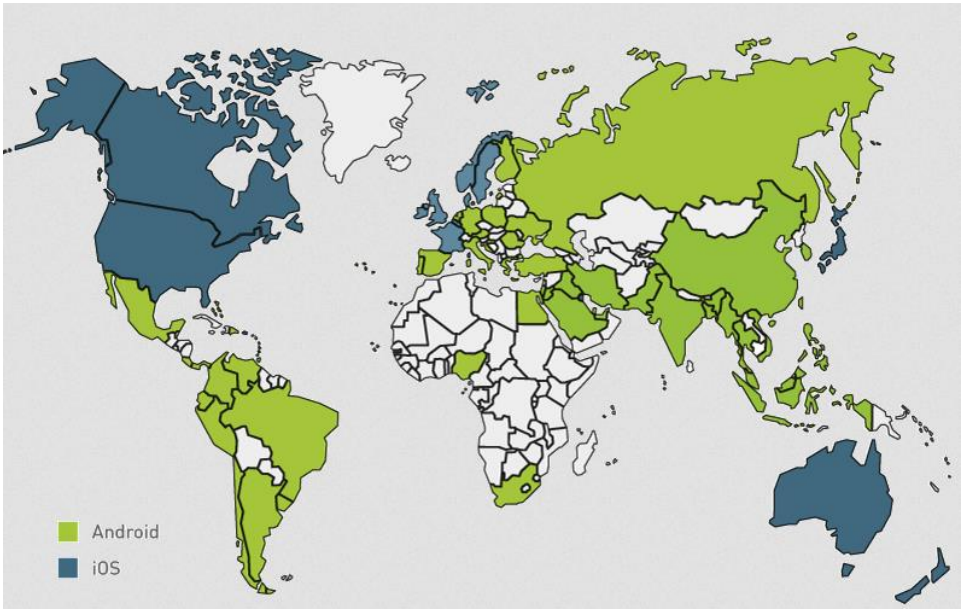


Figure 3. 3 - Regional Distribution of Smartphone OS in 2015 (Source www.deviceatlas.com)

When determining Android version, present android version distribution was considered and android 4.4 'KitKat' was selected as majority of the android users are cumulated after that version [29]. KitKat was a solid build and most of the functionality could be built on top of that. Table 3.1 shows the android version distribution to April 2017.

Table 3. 1 - Android Version Distribution – 2017 (Source developer.android.com)

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.9%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.9%
4.1.x	Jelly Bean	16	3.5%
4.2.x		17	5.1%
4.3		18	1.5%
4.4	KitKat	19	20.0%
5.0	Lollipop	21	9.0%
5.1		22	23.0%
6.0	Marshmallow	23	31.2%
7.0	Nougat	24	4.5%
7.1		25	0.4%

3.4 Apriori Algorithm

The prominent feature of the application is the data mining module, which produces the item suggestions to the user based on his historic data. For this purpose association rule mining technique was used. After a thorough research, it was decided to go with the Apriori algorithm for data mining module due to the simplicity, and better performance and accurate results in terms of retail data mining.

3.5 Android Studio

There were multiple options to choose as the development platform to implement the proposed system. Telerik and Xamarin mobile development kits were considered in the earlier stages due to their significantly easier learning curve. Additionally they both offered hybrid app development (Android, IOS and windows) which might be of a value addition for the future of the application. However with a further study it was discovered that the backend integration with SQL server was slightly difficult and license for the platform with the backend services were very expensive.

Later Android studio was selected as the IDE since it is native for Android developments and provided easy out-of-the-box development experience with its UI and Tools. Development was convenience with Java and integration with MySQL backend was seamless.

3.6 MySQL

The backend was created using a MySQL as it is the go to backend for web and mobile development projects. Setting up the backend services was very convenient. Additionally, MySQL Workbench enabled us to perform some of the MSSQL related tasks to be carried out and make easy migrations to MySQL database.

3.7 PHP and Postman

PHP goes hand in hand with MySQL for the server-side scripting. Developing the server side algorithm was done without much of a hassle. WAMP server provides easy setup of PHP and MySQL services. Postman was used to test out API calls.

3.8 Google Maps API

For geolocation services, Google maps API was selected. Google maps provides extensive and very customizable set of libraries to build location and mapping content. The vast range of customizable options were well sufficient to implement the mapper component of the ‘Smart Shopping List’. An API key is needed to use this libraries which can be obtained by making a free google account or merging with an existing account.

3.9 Summary

This Chapter provided a full overview on the technologies used under the hood of the proposed solutions with proper reasoning. Next comes the Approach chapter where the approach to the proposed solution is discussed.

Chapter 4

The Smart Shopping List – a Novel Approach to Grocery Shopping

4.1 Introduction

After a thorough analysis of the existing literature, the problem was defined in chapter 2. In chapter 3, it was further discussed the technology required to implement the proposed solution to solve the mentioned problem. This chapter presents the suggested novel approach to use Android mobile technology and association rule mining to address the research problem. The approach is detailed under the following headings of hypothesis, input to the system, output of the system, process to convert input to the output, overall features of the system and users.

4.2 Hypothesis

The existing problems individuals go through during their grocery shopping process such as, not having means to properly maintain individual's grocery lists, to find supermarkets nearby quickly, to suggest and remind and missing items, to inform ongoing discounts/promotions, to get projections of final bills and to conveniently share the shopping lists among others, can be properly addressed by introducing an Android mobile software solution powered by association rule mining Apriori algorithm and geolocation services.

4.3 Inputs

Being an Android based mobile application, 'Smart Shopping List' deploys the conventional set of smartphone user inputs. As the primary source, the inputs are mainly given in to the system via the touchpad or the touchscreen of the Android run mobile device through user's touch strokes. Apart from that, text based inputs are

taken from the user heavily in creating shopping lists and suggestion mining. Table 4.1 contains the series direct user inputs taken in to the system by different parts of the system.

Module	Input
My List	Text input (eg: items, item quantity)
Mapper	User's current location (via GPS)
DB module	Shop related data (Items and user sales, simulation used)
BringMe!	SMS text in a given format

Table 4. 1. – Direct inputs to Smart Shopping List

4.4 Outputs

Smart Shopping List generates a number of outputs while it is operation. All of them are in electronic format and can be obtained within the app itself. The following table 4.2 lists out the outputs generated from the system and respective modules.

Module	Output
My List	Finalized/Previous shopping lists
Mapper	Shops nearby to the user
Item Recommender	Item suggestions based on user's history data
Bill calculator	Projected bill
Deals page	Deals/ Promotions

Table 4. 2 – Outputs of Smart Shopping List

4.5 Processes

Smart Shopping List performs several activates in doing its intended tasks. Several processes are involved in doing so. Figure 4.1 illustrates the overall process workflow of the proposed system.

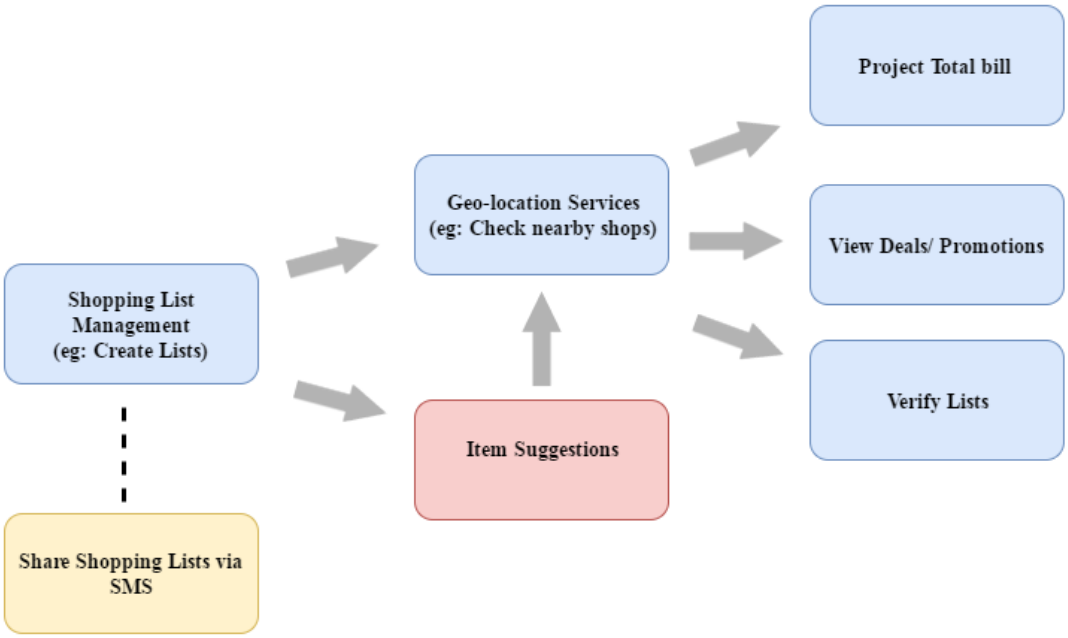


Figure 4. 1 - Process Workflow of Smart Shopping List

4.5.1 Shopping list Management

Here the app enables to the users to create new lists, save lists, update and cross items. This is done on the fully facilitated shopping list creation module. The app creates, stores and update the existing list such as deleting items or editing the quantity as per user's request.

4.5.2 Geolocation Services

Once the system captures the user's current location, it displays the supermarkets nearby to the user including the shops which are registered with the system different to normal shops. Google map API is used to achieve this. This enables the user to perform many other activates such as verify bills, view ongoing deals/promotions.

4.5.3 Item Suggestions

Item Suggestions takes user's historical data in to account and make suggestions based on those. The suggestions are derived based on 3 ways,

- User's previous Shopping lists
- User's previous sales
- Overall sales in the database.

Association rule mining technique is used for this process, namely Apriori algorithm. The algorithm is run considering the previously defined minimum support and confidence.

4.5.4 Bill Projection, View promotions and Verify List

Upon selecting a store via geolocation services, user can view the total projected bill and ongoing deals/ promotions for the particular store. Furthermore the user can verify the purchases made against the items in the shopping list.

4.5.5 Text-to-app imports

The BringMe! Feature captures SMS texts data which is sent in an app-friendly format and extracts the content to import it to the shopping list workspace.

Apart from the above processes the software application performs internal processes such as database manipulations (writes and retrieves) which facilitate the above outputs to the user.

4.6 Features

Proposed system will consist of many features to enhance the usability to give the users a sound user experience. User interface is designed considering HCI concepts so that the design does not deviate from the basic concepts of mobile application design. Shopping list module is a fully-interactive interface which the users can manage their shopping lists with ease.

Since the application is run on mobile platform, to avoid giving heavy overhead and computations to the app, the computations are optimized only to the backend level. eg: the item suggestion algorithm is run at the backend with the given users history data. This enables the application to reduce any memory overloads and produce the outputs fast.

Another key feature would be the application will be free for everyone. The application is developed using open source or low cost methods leaving backend and maintenance the only expenses which can be covered if not made significant revenue by charging a premium from the shops upon registration.

4.7 Users

The application is mainly used by everyone who are interested in making their grocery shopping improved. Shops can register themselves with the system to make visible their item availability to the users, although with the current development this is simulated. Administers are also present to undertake any maintenance related tasks especially at the backend.

4.8 Summary

This chapter gave a thorough representation on the approach to the Smart Shopping List, the proposed solution. The next chapter provides a holistic view of the design of the proposed solution**.**

Chapter 5

Design of the Smart Shopping List

5.1 Introduction

Chapter 4 presented the overall approach to the proposed solution. The purpose of this chapter is to provide a low-level description of the Smart Shopping List system, providing insight into the structure and design of each component. Top level architecture of the design includes major modules namely: My List, Mapper, Item Suggestions and BringMe!. Smart Shopping List has been designed as Android mobile application with a backend with database and web services.

5.2 Top Level Architecture

The proposed solution is broken into two parts: a client-side Android application and a server-side PHP application with MySQL database. The application can further be put in to 2 parts: The functional component (written in Java language) and GUI component (written in XML). The functional part consists of four major components: My List which list management functions are carries out, Mapper where the geolocation services are provided, Item suggestions which is data mining module powered with Apriori algorithm to produce item recommendations and BringMe! Which is a text-to-app feature which enable users to share the shopping lists with others via SMS messages.

The server component of The Smart Shopping List is incorporated of a PHP interface, which manages incoming and outgoing messages and mainly runs the Apriori algorithm, and a MySQL database, which provides centralized storage for application data. The server application receives data from the Android application. This data is then stored in the database and be retrieved upon user's requests.

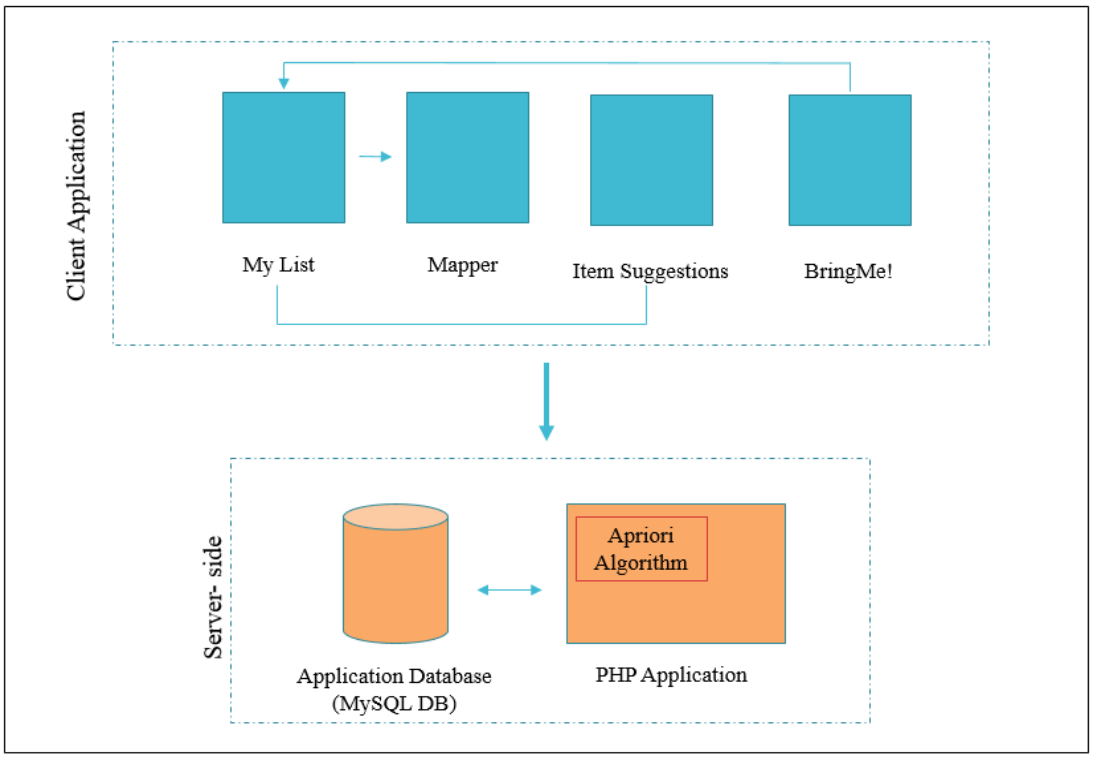


Figure 5. 1 - Top level Design of Smart Shopping List

Figure 5.1 shows the top level architecture of the system. The next described is the roles of each module.

5.2.1 My List

The basis component of the Smart shopping list application is My List. Using My List users can perform their list related management activities such as,

- Create new shopping lists
- Add new items to the current list space
- Remove items from the list space
- Change the quantity of the added items
- Save the current list
- View previously created items
- Edit previously created lists
- Commit/Save changes to old shopping lists.

- Delete previously created lists
- Clear the entire shopping list.

My list has an interactive UI so that the users can perform the intended activities quickly. The Shopping list data changes are automatically saved at the database backend. Item list content in this module will be used for producing item recommendations by the item suggestion module.

5.2.2 Mapper

Mapper component, namely 'Find a store' is responsible for providing geolocation and navigational requirements of the Smart Shopping List. This module captures the user's current location with the user's permission and displays an interactive map component having the user's location as the center. Moreover it displays the nearby supermarkets with the standard icons. Apart from that, if there are any supermarkets which are registered with the system are within the vicinity, they are also displayed in a distinct marker with the total number of items which are available in their stores from the items which are present in the user created list.

Users further can select a registered supermarket to see the item availability and verify the purchased items against the items in the list. Additionally the ongoing deals, promotions of that particular shop can be navigated to as well from the same window to the mobile web browser.

5.2.3 Item Suggestion Module (Association Rule Mining)

This module is part at the client side and other is in the server. The primary function of this component is to analyses historic data and identify association patterns to produce item recommendations. This module processes the following types of historic data for pattern recognition.

- User's created list content
- User's previous sales data
- Entire sales data (Other user's)

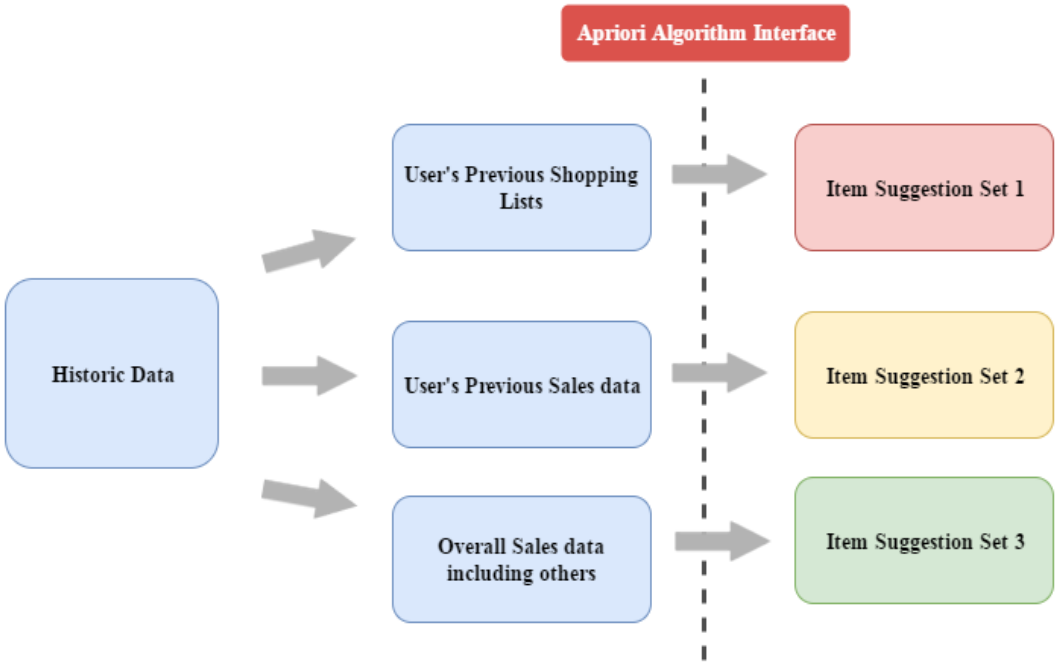


Figure 5. 2 - Item Suggestions Module and the Workflows

Figure 5.2 shows the data workflow of the module and how the 3 types of the data is handled by the algorithm. The component window will have 3 sections; 'Items you may be interested in', 'Items you have previously purchased' and 'People bought this item also bought' for the suggestions generated based on above types of historic data respectively. The client side will pass on the current list content to the server with the user ID while the server side part will retrieve the necessary data from the database and run the Ariori algorithm at the server end. Lastly the generated data will be pass back to the client solution.

5.2.4 BringMe!

BringMe! is a text-to-app feature which allows the users to share their shopping lists via SMS services. An individual may send a Smart Shopping List user a text message in a application friendly format, which the application would read and extract the content and generate the shopping list. The format of the text message is defined as follows.

<BRINGME>Milk,Tea,potatoes,pork ribs

Text starts with a tag <BRINGME>, then the user can include the items needed separated by a comma ‘,’. The person who sends the text may not be a Smart Shopping List user let alone a smartphone user.

5.3 Basic Flow

Having the components identified, it is important to understand the basic flow of the system. Below Figure 5.3 sums up all the roles of the components and the activities which can be carried out by the users using those components.

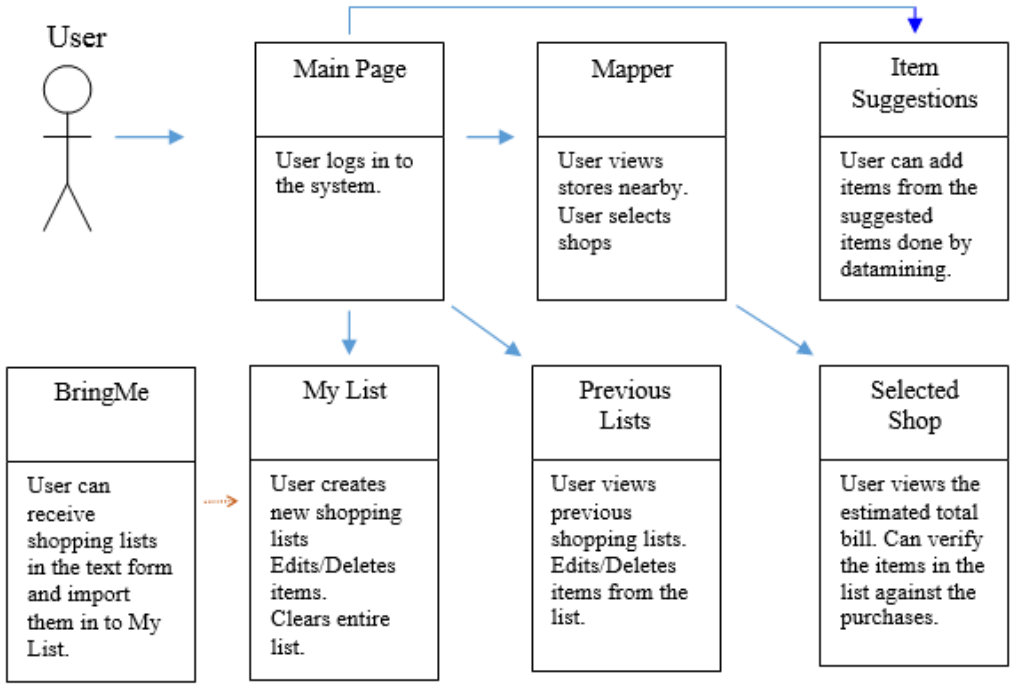


Figure 5. 3 - Activities and Module Roles of the System

5.4 Database Design

The application related data resides in the MySQL database resides in the backend. Data is stored and retrieved via the PHP API calls invoked upon requested by the client application.

5.5 User interface Design

The user interface consists of a set of menus through which the user can interact with and perform his intended functions with ease. Every window would consist of various GUI components, such as buttons, labels, text fields, and list objects. These components will be organized in such a way that the user would be able to quickly grasp the objective of each menu and perform whatever job it is designed for efficiently. As the input method, the user will interact with the UI elements through the Android device's touch screen.

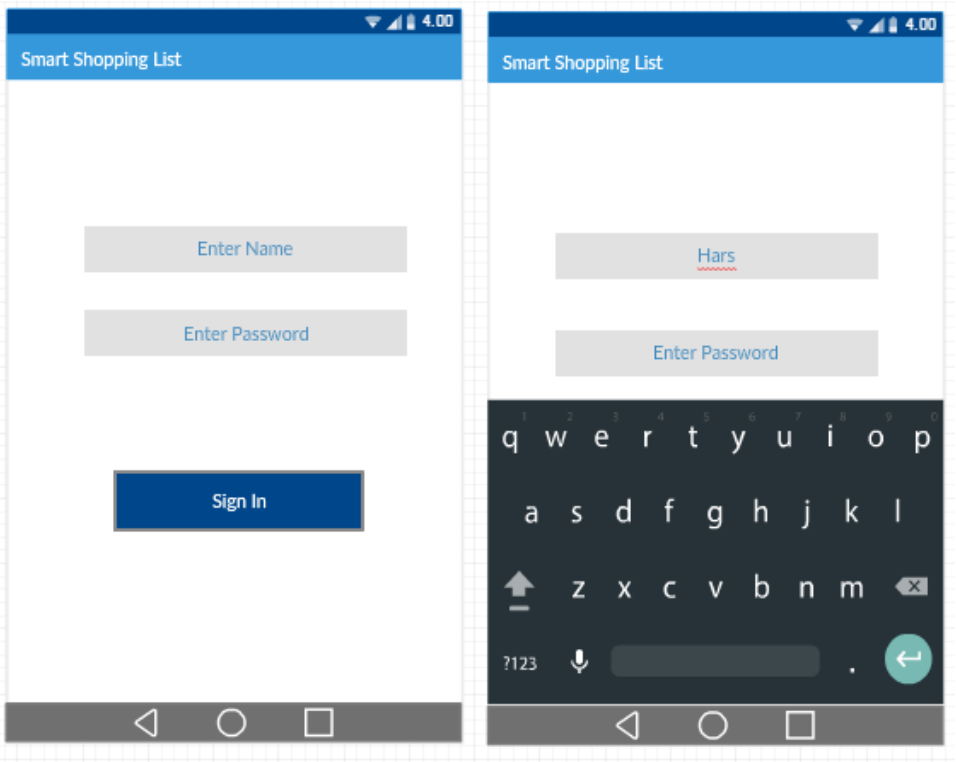


Figure 5. 4 - Login Screen Window Mockup

Figure 5.4 above shows the login page. After the sign in, the user is navigated to the application main screen which the user gets the access to the core application features. Below Figure 5.5 shows the windows of the component My List.

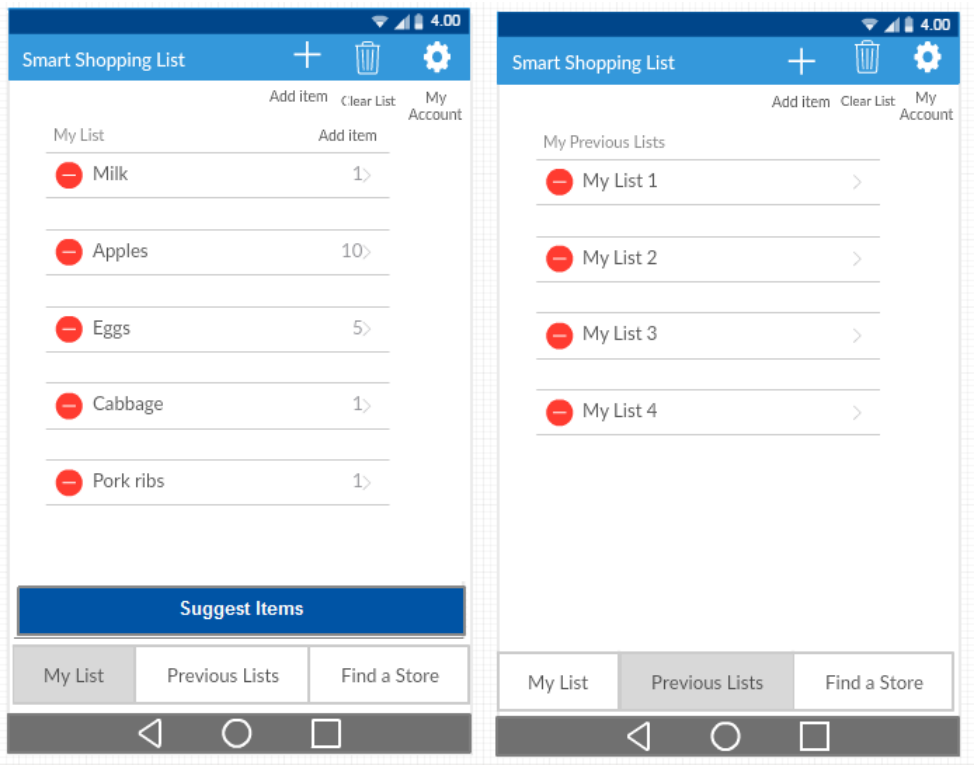


Figure 5. 5 - My List Component Windows Mockup

Item suggest module suggest items to the user analyzing 3 types of historic data discussed in section 5.2.3.Figure 5.6 shows the suggestion window.

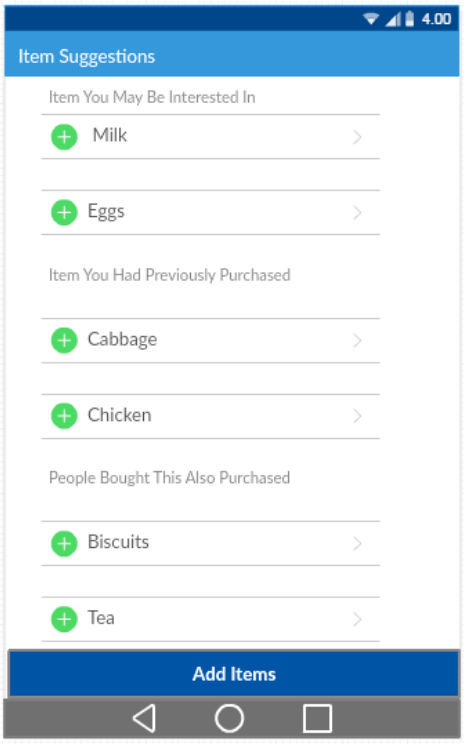


Figure 5. 6 - Item Suggestion Window Mockup

The Mapper component, ‘Find a store’ has a mapping component and the user is able get the geolocation serveries provided the by the application such as: Finding a nearby stores, find stores with their available quantities etc. Upon selecting a store, the user can view the availability of the items, estimated bill and further verify the items in the list against the purchase. Figure 5.7 shows the Mapper component main window and the subsequent window the user gets upon selecting a store by clicking on a marker.

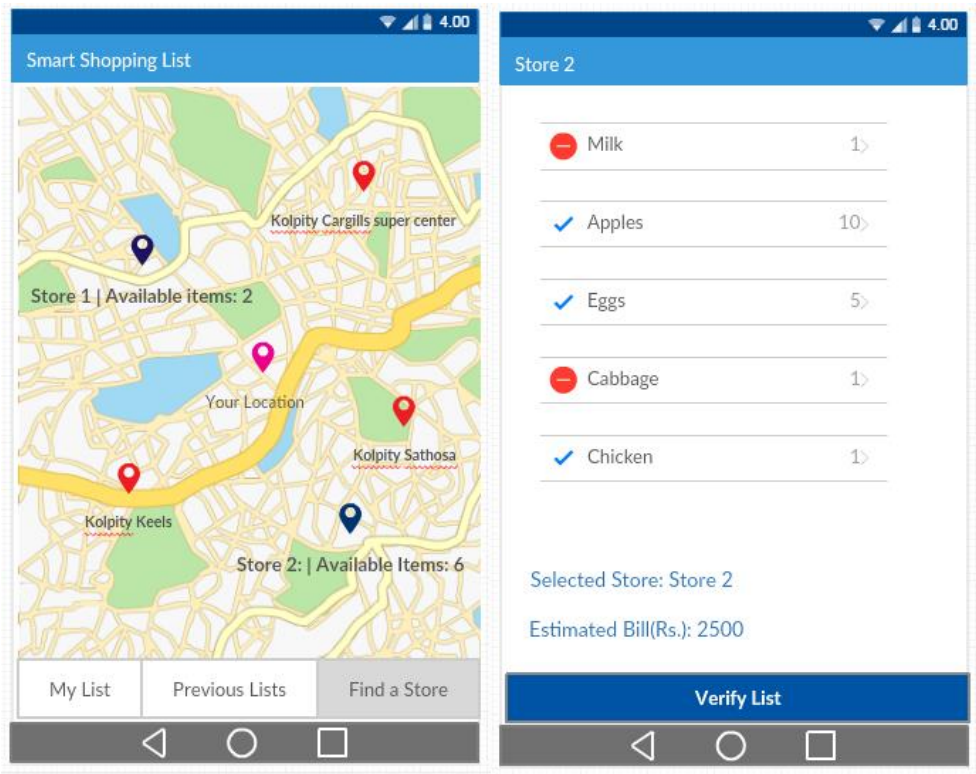


Figure 5. 7 - Mapper Component Windows Mockup

The user design of Smart Shopping List adhere with most of the Ben Shneiderman’s ‘Eight Golden Rules of Interface Design’. Few of them are,

- **Strive for consistency**: all of the major options are displayed at the bottom of the menu (i.e. ‘My List’, ‘Previous Lists’, ‘Find a Store’) and the top tool bar for shopping list management tasks (i.e. ‘Add item’, ‘Clear List’, ‘Settings’)
- **Offer informative feedback**: For many operator actions including updating and deleting items the application provide some feedback.

- **Offer simple error handing:** If fields are not filled out properly when a user presses the submit button, using a dialog box they will be notified.
- **Reduce short-term memory load:** Each menu is designed aiming to do a singular task, this allows the application components easily understood by the user.

5.6 Summary

This chapter gave a very comprehensive report on the design of the Smart Shopping List. The top level architecture, the component separation of the system and the roles of every component, database design were discussed in detail. Finally, the user interface design and how the user interface design principles are applied during the design is also revealed. The next chapter talk through the implementation of the solution.

Chapter 6

Implementation of the Smart Shopping List

6.1 Introduction

Chapter 5 presented the design of Smart Shopping List. This chapter provides and in-depth analysis of the implementation of Smart Shopping. The system has been implemented as an Android mobile application powered by Java with a backend of PHP and MySQL. The algorithms, Hardware, software, pseudo codes, and relevant code segments of the implementations are presented in this chapter.

6.2 System and Mobile Client Application Overview

Smart Shopping List mobile interface has been developed on the Android OS which uses XML as the underlying technology to maintain the graphical properties of elements. The functional side of the application is handled by Java classes which in terms evoke necessary server side PHP application functions and subsequently accesses the MySQL database. Server interaction is of critical note.

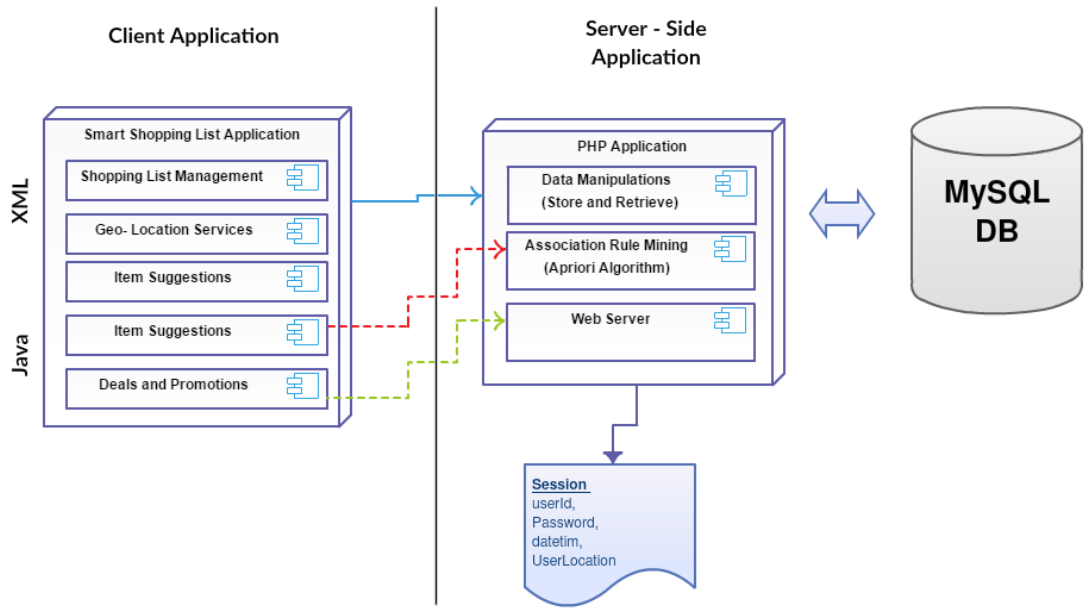


Figure 6. 1 - Deployment Diagram of Smart Shopping List

Figure 6.1 shows the deployment diagram of the application which elaborates the implementation of the overall system and the positions of which each sub components reside in their main component.

As mentioned, Android studio maintains UI elements properties which are kept in XML format separate from the functional part of the application, which is done by Java classes. This adheres to the MVC architecture where view is forged by XML tags, controller is denoted by PHP scripts and underlying java classes, and model is the database resides in the backend. Figure 6.2 and Figure 6.3 show how UI and functions are maintained respectively within the IDE and their contents.

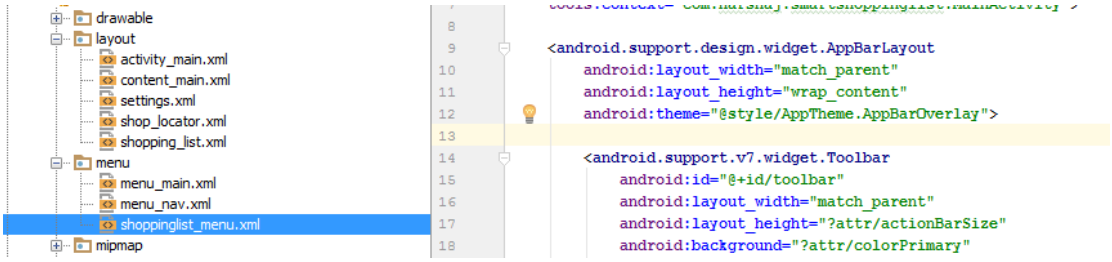


Figure 6. 2 - UI XML

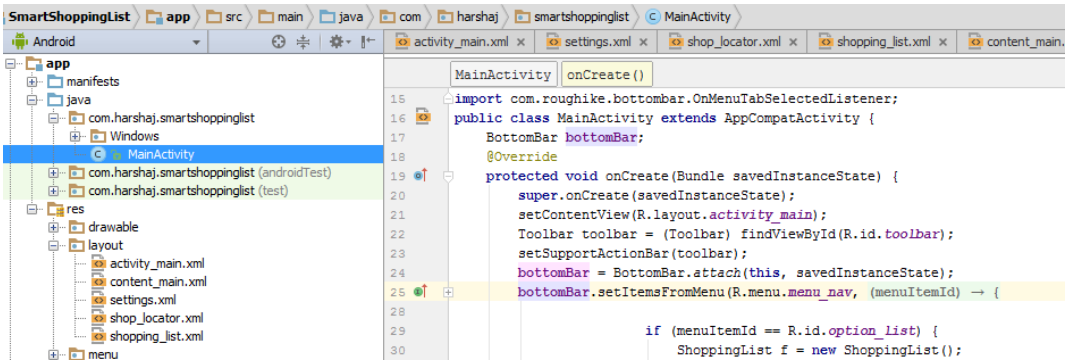


Figure 6. 3 - Underlying Java Classes

When user makes requests via the application front end i.e. user views the previous grocery lists created by him, it will be directed to the remote PHP application by the undying java classes. PHP application then processes the user request whether it is a computation, database related or web related in this case data retrieval from the database. Once the request has been processed and the outcome is present, PHP

application then passes it back to the client application if it needs be. The Java classes then forward the results to the XMLs, populating the GUI with the requested information. Figure 6.4 further shows the flow of how the requests are handled.

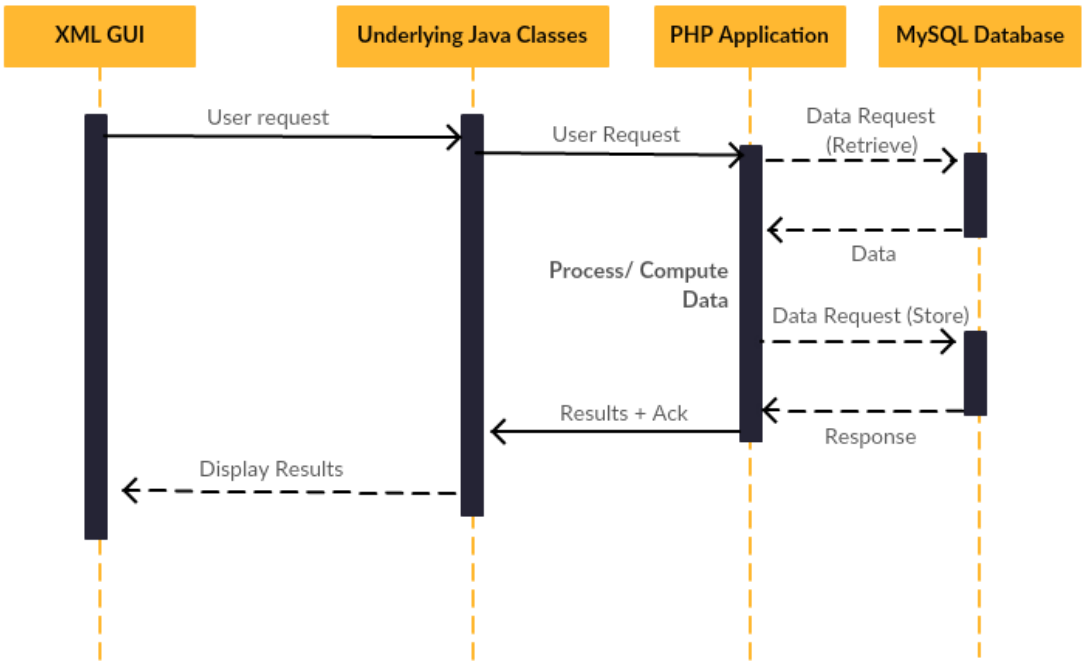


Figure 6. 4 - User Request Handling of Smart Shopping List

Following is a code snippet from DbConnect.php which manages the database connection upon the user request.

```
//connects to the database
    function connect()
    {
        //Including the config.php file to get the database constants
        include_once dirname(__FILE__) . '/config.php';

        //connecting to mysql database
        $this-
>con = new mysqli(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

        //Checking if any error occured while connecting
        if (mysqli_connect_errno()) {
            echo "Failed to connect to MySQL:
" . mysqli_connect_error();
        }

        // returning the connection link
        return $this->con;
    }
```

A local server and a live server were both used as the backend during the implementation. WAMP server was used for local installation to simulate server end with the database.

Since overall implementation notes have been covered, next gives a further dive in to each module to discuss the details at the implementation level. The following sections provide a more detailed description about each component.

6.3 Items Suggestions Module

The prominent module and one of the key features of the smart shopping list is the data mining module, Item Suggestions. As discussed in Chapter 2 and 3, association rule mining techniques are well known for finding hidden patterns among given item sets. Furthermore it was also discovered that the native Apriori algorithm suits best for the needs and the item suggestion algorithm was implemented on that.

The suggestions algorithm uses Apriori logic for identifying frequent item sets. Apriori is a candidate generation and testing approach for finding frequent patterns in datasets. It speaks that if certain item X is appearing in an item set then there is high probability that another item Y is desirable to appear in the same item set. With a given frequency threshold (minimum support), the infrequent item subsets are prune down. It takes the advantage of the fact that any subset of a frequent item set is also be frequent item set and otherwise the algorithm prunes the item set down. The algorithm uses a level –wise search, where k-item sets are used to explore k+1 item sets. Candidate generation is extending the frequent item sets by one item at a time. The following example shows of how Apriori works in finding the frequent item sets. A Set of sample grocery lists is created with several items below in Table 6.1.

List ID	Items
1	Cabbage, Jam, Meat
2	Bread, Jam, Butter
3	Cabbage, Bread, Jam, Butter
4	Bread, Butter

Min support: 2

Table 6. 1 - Sample Shopping Lists

First the algorithm generates the initial set of candidate item sets and calculates their respective candidate support count.

Step 1: Generating 1-itemsets Frequency Pattern

Itemset	Support Count
{Cabbage}	2
{Bread}	3
{Jam}	3
{Meat}	*1*
{Butter}	2

C1

Compare candidate support count with min. support →

Itemset	Support Count
{Cabbage}	2
{Bread}	3
{Jam}	3
{Butter}	2

L1

Provided that the minimum support (minimum frequency threshold) is 2, the algorithm negates any items set which are less frequent than the minimum support.

Step 2: Generating 2-itemsets Frequency Pattern

Itemset	Support Count
{Cabbage, Bread}	*1*
{Cabbage, Jam}	2
{Cabbage, Butter}	*1*
{Bread, Jam}	2
{Bread, Butter}	3
{Jam, Butter}	2

C1

Compare candidate support count with min. support →

Itemset	Support Count
{Cabbage, Jam}	2
{Bread, Jam}	2
{Bread, Butter}	3
{Jam, Butter}	2

L1

Step 3: Generating 2-itemsets Frequency Pattern

Itemset	Support Count
{Cabbage, Bread, Jam}	*1*
{Cabbage, Bread, Butter}	*1*
{Cabbage, Jam, Butter}	*1*
{Bread, Jam, Butter}	2

C1

Compare candidate support count with min. support →

Itemset	Support Count
{Bread, Jam, Butter}	2

L1

This process is continued until an empty candidate or frequent of k-item sets received. In this scenario after the 3rd iteration , the most frequent item set {Bread, Jam, Butter} is derived. Upon finding the most frequent item set, association rule mining is performed using the minimum support specified.

Min. Confidence: 70%

Rule	Support	Confidence
Bread-> Jam & Butter	50%	66.67%
Butter-> Bread & Jam	50%	66.67%
Jam-> Butter & Bread	50%	66.67%
Bread & Jam -> Butter	**50%**	**100.00%**
Butter & Bread -> Jam	50%	66.67%
Jam & Butter -> Bread	**50%**	**100.00%**

Table 6. 2 - Rule Selection

As shown in the Table 6.1 Provided that the minimum confidence is 70% only two rules prove through. When the user requests for suggestions Butter and bread will be suggested given that he has Bread and Jam or Jam and Butter added to his shopping list. This algorithm is run to suggest items based on the 3 types of historical data. When the suggestions are made based on smaller data criteria, higher support is needed to for better mining however for significantly larger data sets smaller support may be used to identify the patterns. This causes the current algorithm to consider more item repeats for suggestions. Following is the pseudocode for the mining algorithm.

```
DECLARE Totalitems []
DECLARE Integer support
DECLARE Candidatesset []
DECLARE Integer MinSup
DECLARE Decimal Confidence
DECLARE Shoppinglist []
DECLARE Decimal InputConfidence
SET InputConfidence to READ input confidence
```

```
SET Totalitems [] to GET items bought from before
FOR each Totalitem in Totalitems []
        DECLARE Candidates []
        SET support to GET number of repetition for the Totalitem in Totalitems []
        SET Candidates [] to Generated candidate items using Totalitems [] and Totalitem
        IF Candidates [] count <= 6
                SET MinSup to 2
        ELSE
                SET MinSup to 10% of Candidates [] count
        END IF
        FOR each Candidate in Candidates
                IF supports >= MinSup
                        Remove Candidate from Candidates []
                ELSE
                        Add Candidate to Candidateset
                END IF
        END LOOP
END LOOP
FOR each Candidateitem in Candidateset
        DECLARE items []
        SET items [] to GET items in Candidateitem
        FOR each item in items
                SET Confidence to Calculate percentage of item repeated in Totalitems []
                IF Confidence >= InputConfidence
                        IF item does not exist in Shoppinglist []
                                Add item to Shoppinglist []
                        END IF
                END IF
        END LOOP
END LOOP
IF Shoppinglist count > 0
        DISPLAY Shoppinglist []
ELSE
        DISPLAY message 'No suggestions found'
END IF
```

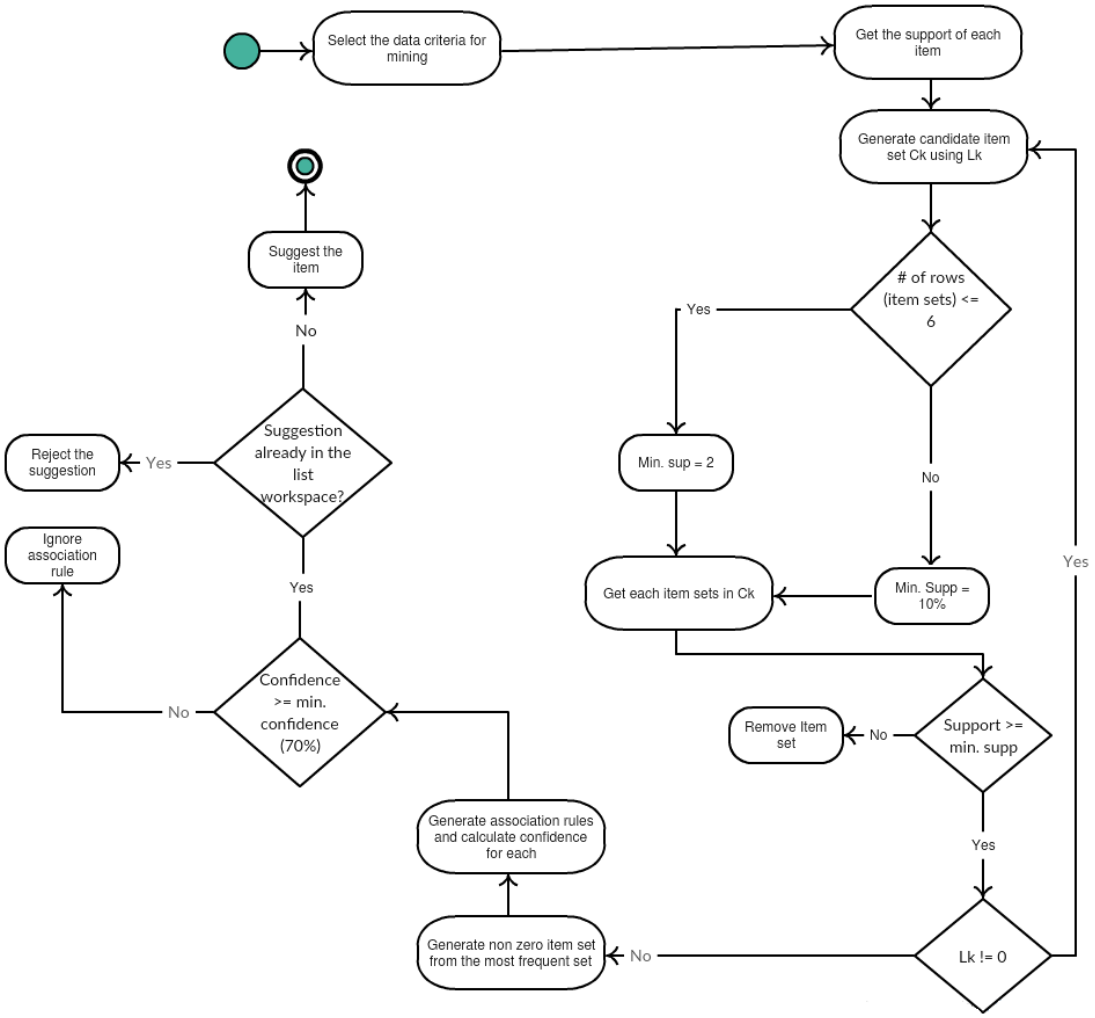


Figure 6. 5 - Logic Diagram for Item Suggestion Module

Figure 6.5 shows the logic behind the item suggestion algorithm for a given instance of data in a grocery list workspace. The similar logic is employed when generating suggestions based on sales data. No data preprocessing such as noise or outlier removal is carried out mainly because two reasons: It is association rule mining is carried out and Apriori algorithm itself gets rid of the infrequent items. The other reason is, since this is a mobile application and used for personal grocery shopping purposes, not very large amount of data generated hence it is vital to use every data that the user generates. Using Figure 6.6 below, it can be seen how the item suggestion algorithm is comprised of the logic mentioned above, in a holistic manner. Based on what the input data is the suggestions are generated accordingly. When the user requests for user suggestions, the current list will be used against the 3 types of data reside in the database. Each Apriori function is responsible for fetching the

necessary data and creating association rules to calculate the confidence and returning the generated date back to the suggestion module at the client side.

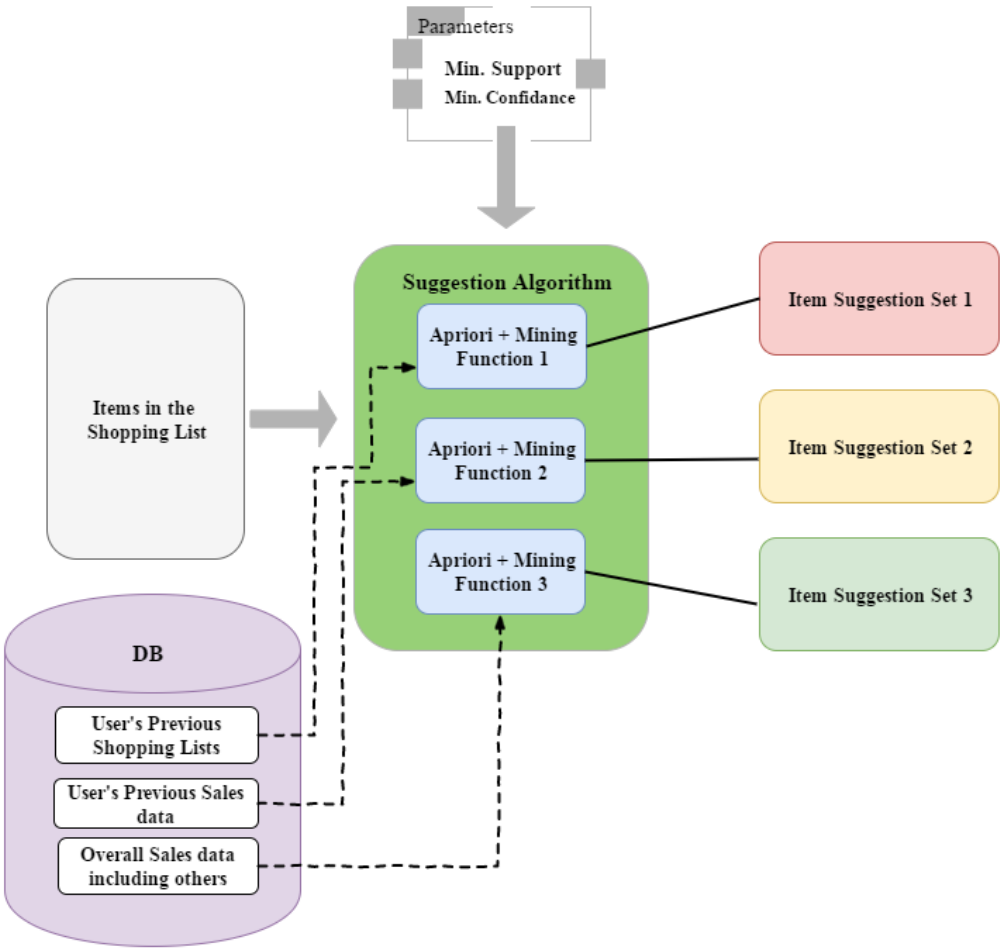


Figure 6. 6 - Algorithmic Layout of the Item Suggestions Module

It was decided to implement data mining functions in the backend due to many reasons.

- Data mining algorithm takes a considerable amount of computation power. Considering this is a mobile solution and the all mobile devices do not perform at the same speed and memory, let alone the mobile devices may be slower than the standalone systems. So the implementation had to be done in a way that it does not hinder the performance of application.
- To minimize the data usages between the client and the server.

- Having too much code written in the client code considerably contributes the size of the application. Smart Shopping List is intended to be efficient and this implementation helps it to be lightweight and efficient.

PHP scripting language is used for this. JSON is used for structuring data to transmit data between the server and the application. Some code snippets for the Apriori function 1 which generate item suggestion based on user's previously created lists is show below.

Fetching list data and putting them in to an array,

```
/*********  Fetching ShoppingList rows from the Database based on
User Id  ********************/

$connect = mysqli_connect(DB_HOST,DB_USER,DB_PASSWORD) or die("Error"
);
$sql = mysqli_select_db($connect,DB_NAME) or die("ERROR");
  $uid = $_POST['uid'];
  $data = mysqli_query($connect,"SELECT * FROM `shoppinglist` WHERE
`UID` = '$uid';");
        while($row = mysqli_fetch_array($data)){
                $n++;

// n is number of rows in the DB for the given user
                $shop = $row['ShoppingListItems'];
                $json_output = json_decode($shop, true);

// Decoding the JSON data of each row
                        foreach($json_output as $s) {
                                $item= $s['item'];
                                array_push($arr, $item);

// Pushing the items of every list in an array
                        }
        }
/********   creating associative array for the items  **********/
        foreach ($arr as $as => $value) {
        $arr1[$value] = 1;
        }

/********   Counting the items  **********/
        foreach($arr1 as $nn => $value){    //
                foreach($arr as $mm){
                        if($mm==$nn){
                                $value = $value+1;
                                array_push($res, $nn);
                        }
                }
        }
```

The algorithm considers 10% as the minimum supports, however if the rows are less than 6, minimum count will be considered as 2.

```
/********  Support count is 10% of the number of rows but if rows
less than 6, it would be 2   **********/
        $min = 0.25*$n;
        $mini = round($min);
                if($n<=6){
                  $mini = 2;
                }else{
                  $mini;
                }

/********  Removing the items from array 'nxt' which have less than
Support count  **********/
        foreach ($nxt as $nx => $val1) {
                if($val1<$min){
                        unset($nxt[$nx]);
                }
        }
```

After the association rules are generated confidence for each is calculated and check against the minimum value. Code piece for calculating confidence,

```
/* Calculating minimum confidence here */
/* minimum confidence = 70% */


foreach ($nxt as $nxxitem => $nxxval) {
foreach ($nxtt as $nxitem => $nxval) {
if (strpos($nxitem,$nxxitem) !== false) {
$g = $nxval/$nxxval;
if($g>0.7){
$fg = explode(',', $nxitem);
$it1 = $fg[0];
$it2 = $fg[1];
if($it1==$nxxitem){
array_push($nxt7,$fg);
}
else{
$hh = array($it2,$it1);
array_push($nxt7,$hh);
 }
 }}}}
```

Sending results back to the Smart Shopping List,

```
/* Encoding the data from array to JSON data */
        print_r(json_encode(array('data'=>$nxt8)));
```

The full source for the item suggestion function 1 is shown in Appendix A. Following Figure 6.7 shows an example of the item suggestion made by the module based on the user's previous lists and previous sales data.

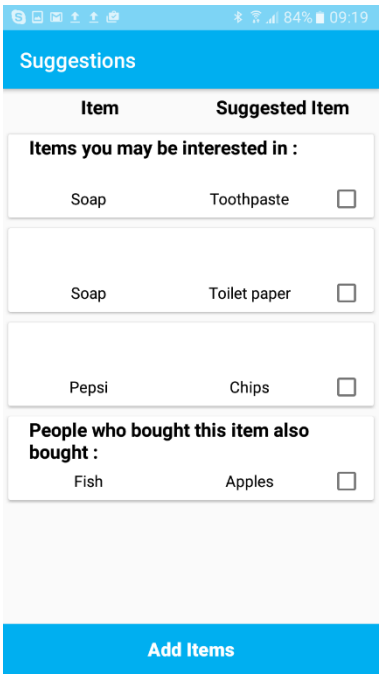


Figure 6. 7 - Item Suggestions Module

6.4 'Find a Store' - Mapper Module

The mapper module provides the location based services to the user which enables him to find nearby stores with ease. Upon the initial run of the Smart Shopping List, the user would get prompted to share the location access with the application in which the module would be ready for use. The varieties of the activities that the user can perform using mapper is listed in the section 5.2.2 and further graphically elaborated with the Figure 5.3. Code snippet for getting user's location permission.

```
public boolean checkLocationPermission(){
    if (ContextCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED) {
        if (ActivityCompat.shouldShowRequestPermissionRationale(this,
                Manifest.permission.ACCESS_FINE_LOCATION)) {
            ActivityCompat.requestPermissions(this,
                    new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
                    MY_PERMISSIONS_REQUEST_LOCATION);
        } else {
            ActivityCompat.requestPermissions(this,
                    new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
```

```
                    MY_PERMISSIONS_REQUEST_LOCATION);
            }
            return false;
        } else {
            return true; }
}
```

The module is powered by the Google Map library. An API key (eg: *AIzaSyCVSoBPvPDx4AeK_QSRfXp0b_HI4l5ATz4*) is needed to make use of the libraries which can be obtained by creating a free google account. The API runs on JavaScript and an extensive amount customizations can be made to the mapping element using this library which helped us to give a personalized look and feel to the component.

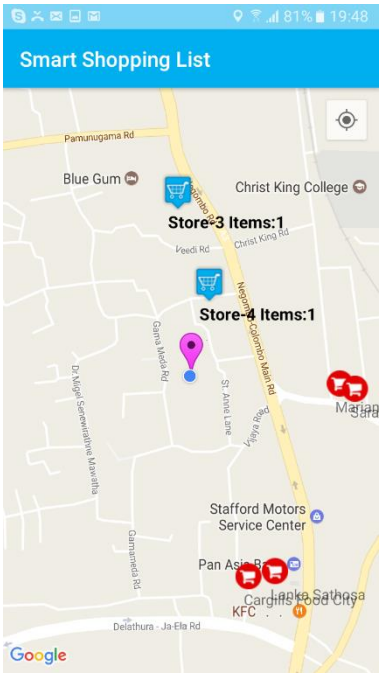


Figure 6. 8 - Screen of the Mapper

Upon activating the Find a Store feature user is navigated to a map element as shown in the Figure 6.8. The supermarkets surrounding the user's current location is shown and the special supermarkets which are registered with the system are shown in a distinct marker as well. If a user has a shopping list created already in the list workspace, the availability of these items are also shown along with the shop's marker with a subtext. For these customizations google maps Marker class was used, which a sample code snippet used for placing a marker is shown below,

```
//Place current location marker

LatLng latLng = new LatLng(Double.valueOf(item1), Double.valueOf(item2));
MarkerOptions markerOptions = new MarkerOptions();
markerOptions.position(latLng);
markerOptions.title("branch:"+bn+"  Found Items:"+qua);

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFac
tory.HUE_GREEN));
mCurrLocationMarker = mMap.addMarker(markerOptions);
mCurrLocationMarker.showInfoWindow();
```

Clicking on a distinct marker unlocks a several shop related activities which the user can perform to further make his grocery shopping process convenient. All the items which are available in the store out from the items in his grocery list would be automatically highlighted and the estimated bill for that particular shop would be displayed. In addition, upon the payment has been made, which the user can further verify (cross) the items automatically off the list. Figure 6.9 shows the shop page that the user is navigated to and shop related activities that the user can perform when the he selects a shop by clicking on a marker.

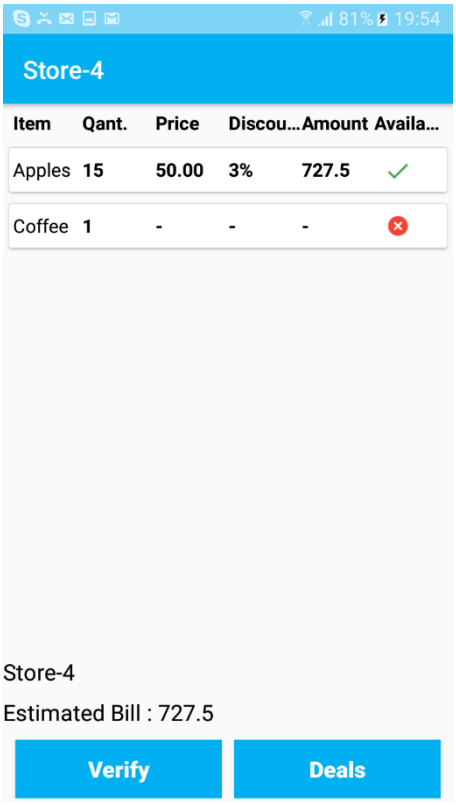


Figure 6. 9 - Store Page of Mapper

After getting in to the store page, the ongoing deals of that particular shop are also shown along with the rest of the information. The URL of the deals page of that particular shops is retrieved from the database and be displayed the mobile web browser.

6.5 My List

My list component is the basic yet the core module which the shopping list functioned are implemented in. User can perform his grocery list management actions using this all interactive, fully-fledged module. Section 5.2.1 provides a range of activities which the user can perform while Figure 5.3 gives a more graphical understanding. Figure 6.10 shows the implementation of the My List component.

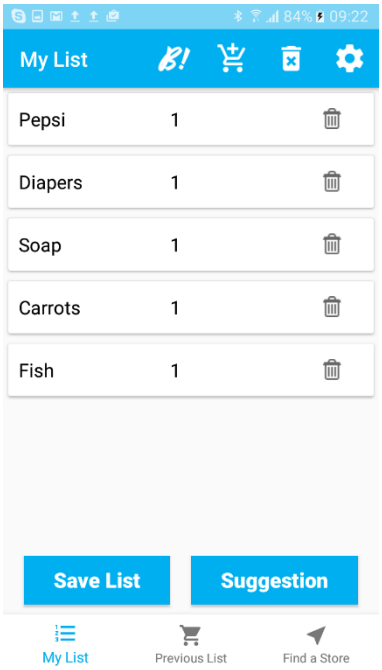


Figure 6. 10 - Screen of the My List

Using 'Previous List' which is a part of the My List component, the user can manage his previously created grocery lists. The same set of management functions available for current lists workspace is also made available here. Figure 6.11 shows the Previous List function and its attributes.

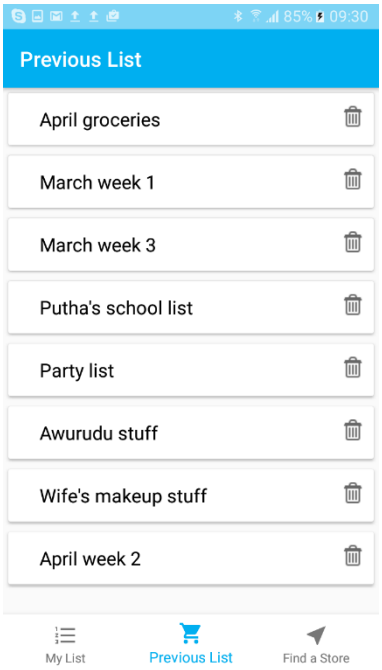


Figure 6. 11 - Screen of the Previous Lists

6.6 BringMe!

The module BringMe! is implemented to allow individuals to make grocery list requests via Short Message Services from the app users. They may send the intended list of items in a text message in a format which the application would understand. When BringMe (B!) is activated, the Smart Shopping List scans the local messages for the messages in the given format (<BRINGME> tag) and lists out the found. Message is broken down in to tokens with the delimiter ‘,’ and the items are extracted and the currently list space is populated. Figure 6.12 shows the B! module content.

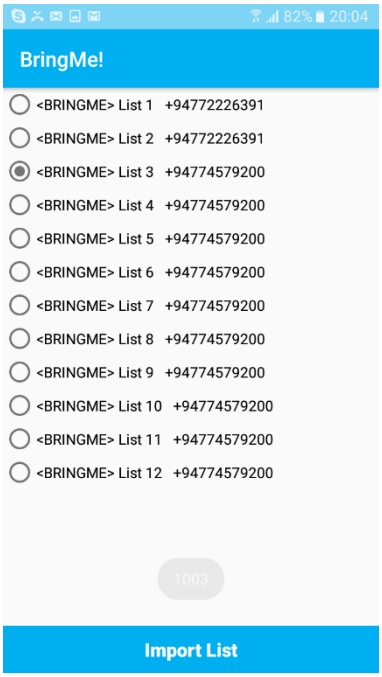


Figure 6. 12 - Screen of BringMe!

6.7 Application Database

This is implemented as a lightweight MYSQL database. Figure 6.13 below shows the simplified database diagram for the database showcasing the core entities of the database. The detailed version of the database diagram and the full schema of the database is available in Appendix B and Appendix C respectively.

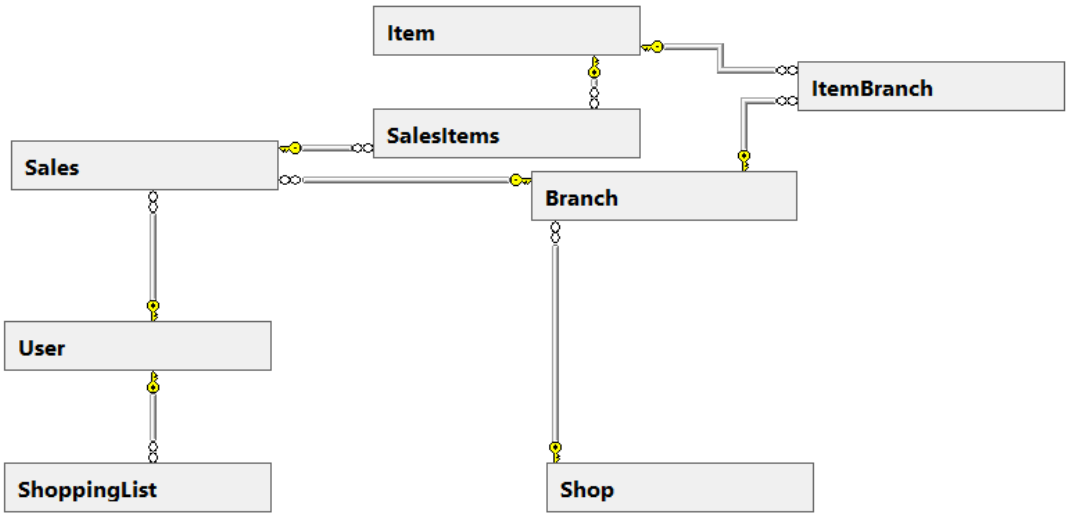


Figure 6. 13 - Database Diagram for Smart Shopping List

6.8 Further Implementation Notes

1. **Use of Fragments**: Several value adding architectural level practices were followed during the implementation of the Smart Shopping List. One of them would be the use of 'fragments' when the module windows are developed. Different from regular view based design, fragments offer backstack and lifecycle features. Although the views are more light weight and simpler to implement they do not perform well with maintaining sessions. Using fragments, the current status of the application can be saved and recreated in a later stage. This allows the application functioned seamlessly even after the application is minimized, closed or even the device is rebooted. Sample code is shown below.

```
Fragment selectedFragment = null;
switch (item.getItemId()) {
    case R.id.option_list:
        selectedFragment = ShoppingList.newInstance();
        FragmentTransaction transaction =
getSupportFragmentManager().beginTransaction();
        transaction.replace(R.id.frame_layout, selectedFragment);
        transaction.commit();
        break;
    case R.id.option_settings:
        selectedFragment = PreviousList.newInstance();
        FragmentTransaction transaction1 =
getSupportFragmentManager().beginTransaction();
        transaction1.replace(R.id.frame_layout, selectedFragment);
        transaction1.commit();
        break;
```

2. **Minimize Data Transmission**: Additionally, significant attention had been given in minimizing the data transfer between the mobile device and the server. Mobile devices are run on mobile data or Wi-Fi and users are usually concerned about the data usage. If the application consumes a significant amount of mobile data, this may become a usability concern. Therefore the software solution was optimized by minimizing the data transfer between the server and the mobile application whenever possible. A good example would be implementing the item suggestion module at the backend so that the only data needs to be transferred between the twos ends would be the list items and the suggested results. If the algorithm to be implemented at the client, extra validations have to be developed and more data has to be fetched from the

database which in terms making the data delivery bulker and the application more process-hogging.

3. **Use of JSON**: In addition, JSON is used to transfer data between the server and the application because it is smaller, faster and lightweight compared to XML. Also JSON can be mapped more easily into object oriented system.

4. **Soft Deletion of Data**: As shown in Appendix B the data in the database is interrelated and deleting some data might cause some of the application features to be dropped. Good example is that deleting a user or an items may directly affect the sales data due to foreign key constraints. To avoid having data conflicts, 'IsActive' ghost boolean field is introduced to every data table (Figure 6.14). This can be set to true or false to enable and disable the record (1-Active, 0- Inactive), without physically deleting the record from the database.

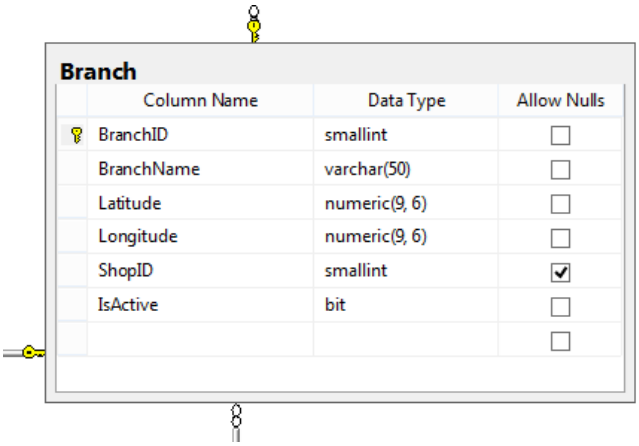


Figure 6. 14 - Soft Deletion using IsActive Field

Following PHP code fragment shows how data is manipulated with IsActive field.

```
if($delete = mysqli_query($connect,"UPDATE `shoppinglist` SET
`IsActive` = '0' WHERE `UID` = '$uid' AND `ShoppingListID` =
'$id';")){
$res['response'] = true;
$res['message'] = "list deleted";
echo json_encode($res);
}else{
$res['response'] = false;
$res['message'] = "list not deleted";
echo json_encode($res);
}
```

6.9 Software Used

The following Software tools, Framework and API have been used in developing and testing the Smart Shopping List.

- Android Studio v2.2.3 was used as the main IDE to develop the client side application. It takes XML to form the graphical and Java language to build the functions.
- Google Map API v3.28 exp for Android was used.
- Wamp server v3.0.6 x64 used to simulate the backend with Apache 2.4.23, PHP 5.6 and MySQL 5.7.14
- SQL Management Studio 2012 was used to create the initial database in MSSQL
- MySQL workbench was used to migrate the MSSQL database to MySQL.
- Notepad++ text editor was used.
- Postman API assist tool.

6.10 Hardware Used

As hardware 2 mobile devices; Samsung Galaxy J7 with Android Marshmallow v6.0.1 and Samsung Galaxy Tab 3v with Android KitKat v4.4 were used to test the application.

6.11 Summary

This chapter presented a comprehensive outline of the implementation of the Smart Shopping List. The System overall implementation and the modules of the system including the data mining module were extensively discussed. The flow charts and the diagrams gave us further support in fully understanding how they work. The Chapter ends with the notes of implementation and the hardware and the software used to develop and test this solution. Next chapter presents the proposed solution in practice.

Chapter 7

Smart Shopping List in Practice

7.1 Introduction

Chapter 6 provided an intense specifications on the implementation of the Smart Shopping List. The application was implemented and has been used for 3 weeks to ensure the functionality is working as expected. This chapter gives an end-to-end practice scenario run of the system which provides the necessary confidence that the application is ready for a formal evaluation.

7.2 Prerequisites

After running the application on the android device, user will encounter with the loading screen of the application and then a login. Upon entering the credentials user can login to the system. Figure 7.1 below shows the sequence.

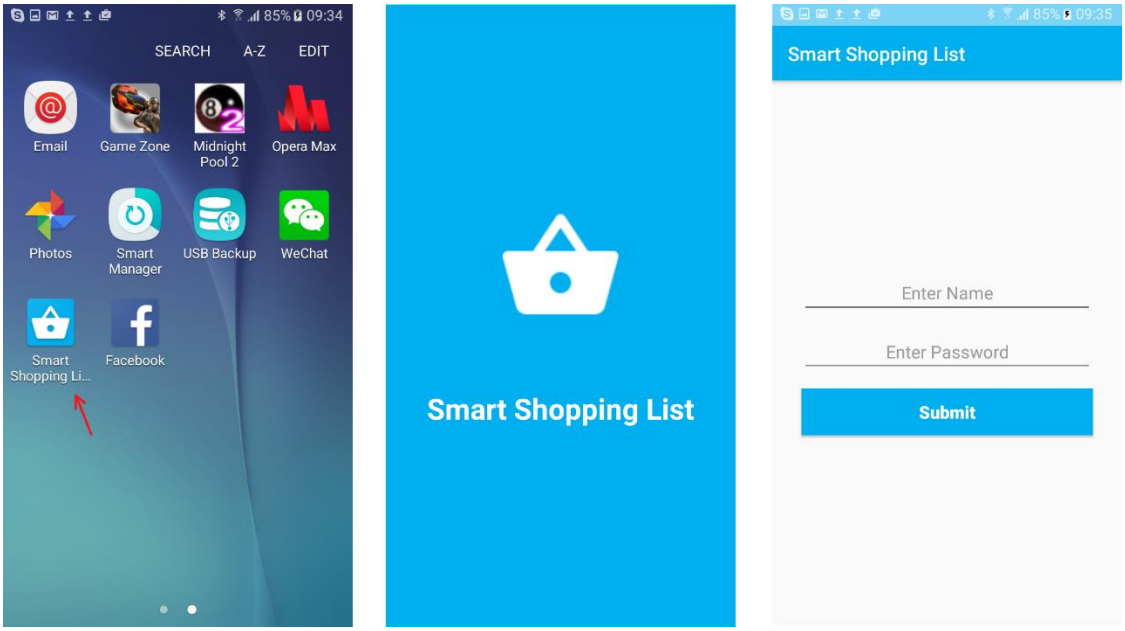


Figure 7. 1 – Smart Shopping List Early Screens.

7.3 My List in Action

User enters the My List home page upon login in. Major Modules can be seen from the home page as well as the My List specific functions. User may add items to the current lists pace using add icon . Upon pressing add, a pop up would show up and allow the user to add the item and the quantity. Figure 7.2 shows home page and adding an item.

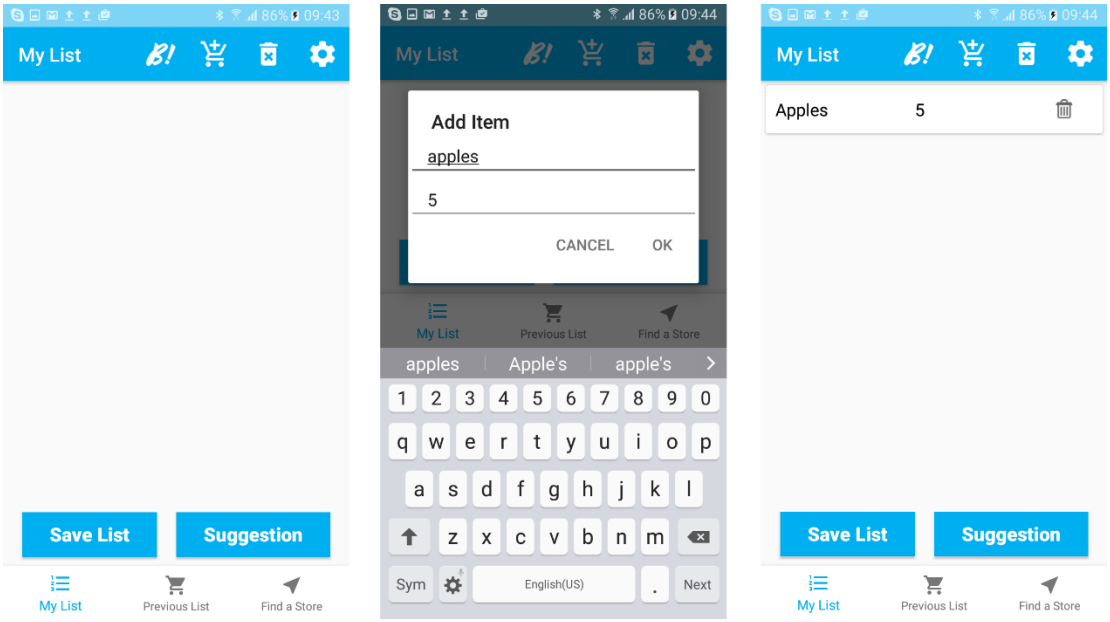


Figure 7. 2 - Smart Shopping My List Screens I

Once the items have been added to the list, user can change the quantity of an added item. This can be done by pressing on an item and changing the quantity in the pop up. Further the user can cross an item from the list by swiping on the item to the right or uncross it back by swiping on it right to the left. Figure 7.3 below shows the screens.

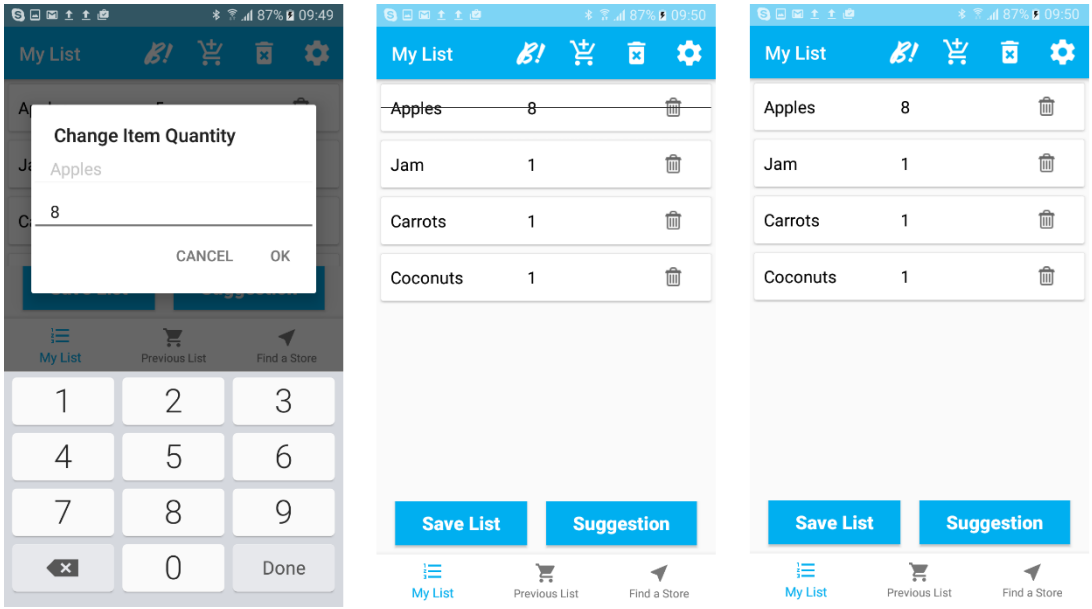


Figure 7. 3 - Smart Shopping My List Screens II

Deletion of items can be done in 2 ways. Specific items can be deleted from the list by using the trash can icon  next to the item. If the user wants to clear the entire list he can use the trash can icon  in the menu bar for that. Figure 7.4 is shown below.

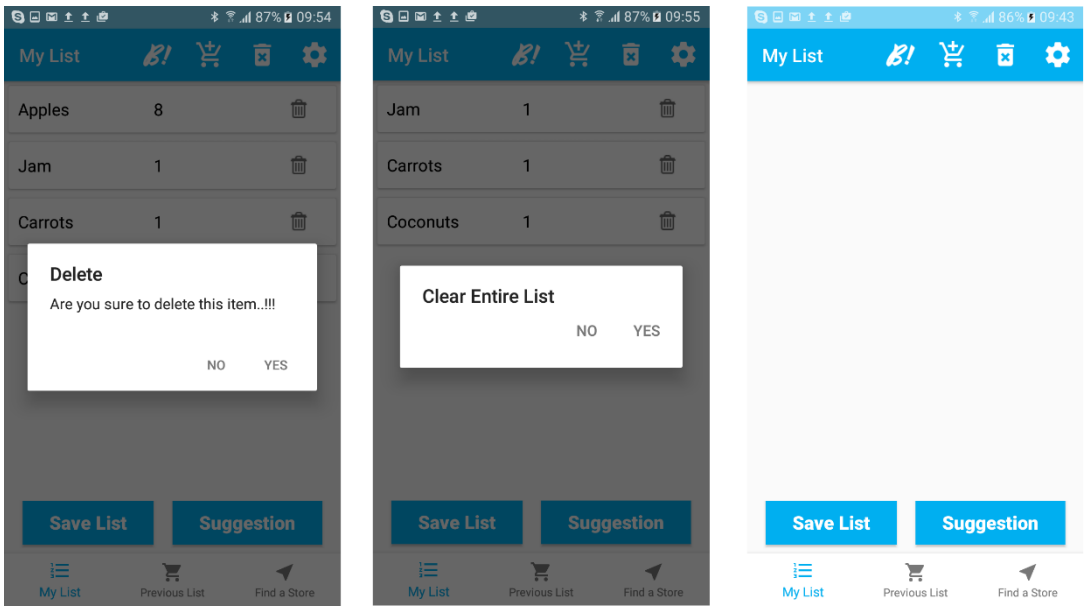
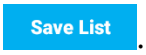


Figure 7. 4 - Smart Shopping My List Screens III

User saves the current list space as a list by clicking Save List button 

. A pop up opens and the user can give a unique name for the list and save his list. Once the list has been saved it can be seen under previous list menu. Save function is shown below with Figure 7.5.

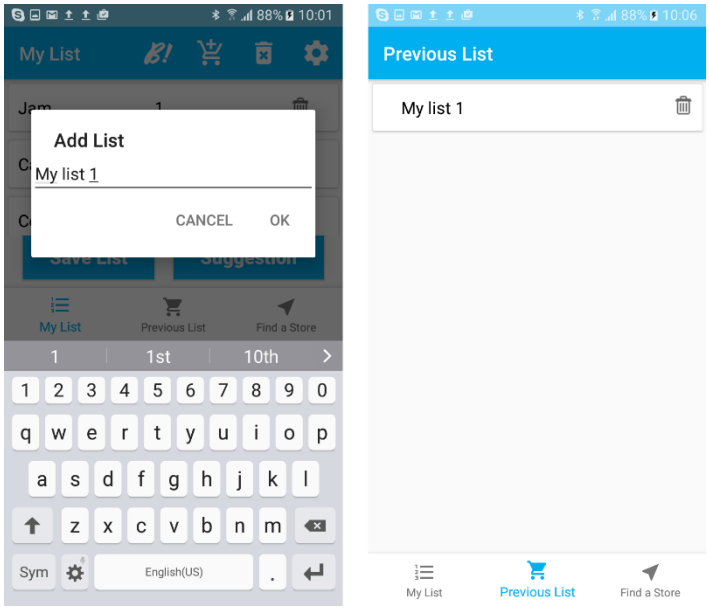


Figure 7. 5 - Smart Shopping My List Screens IV

7.4 Item Suggestions in Action

Once the user has created the grocery list as per his interest. He can further request for item suggestions based on his historic data. The module will produce and display suggestions based on the user's previously lists, previous sales data, and overall sales data for that items in the active grocery list. They are displayed under 'Items You may be Interested in', 'Items You had Previously Purchased' and 'People Bought this Item Also Bought' sections respectively. The items suggested can be added to the list by checking them and pressing Add Items button. Figure 7.6 shows the suggestion in action.

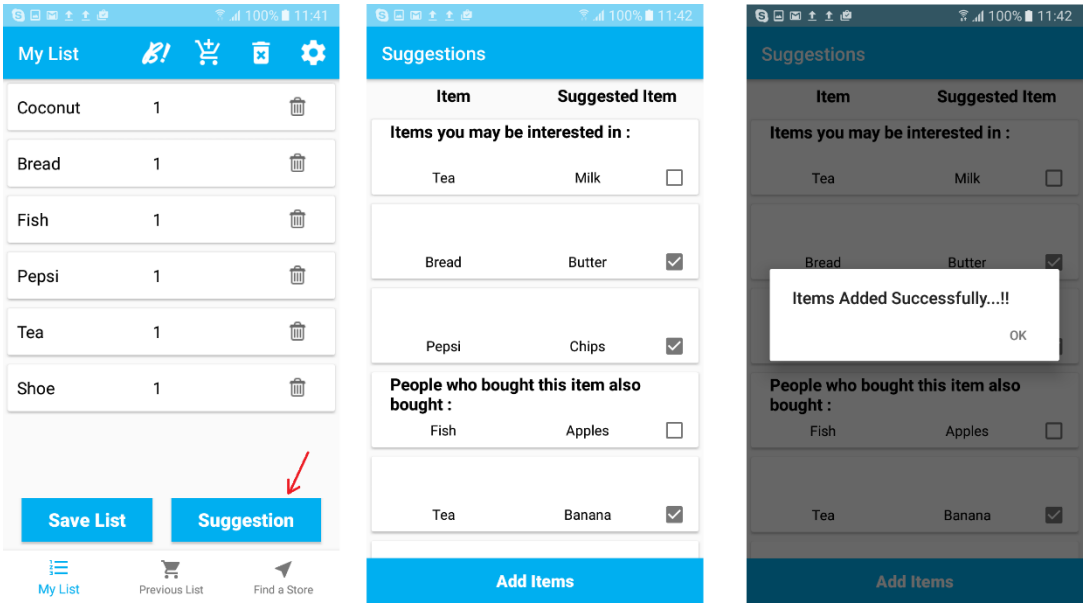


Figure 7. 6 - Suggestion Module Screens

7.5 Previous List in Action

User can open his previous list buy using the Previous list functionality. Upon selecting the menu, a list of user's previously created shopping lists will be displayed. User can select an older shopping list and perform all the functions which are available for the current list and further more delete old grocery lists. Figure 7.7 show the screens of previous lists.

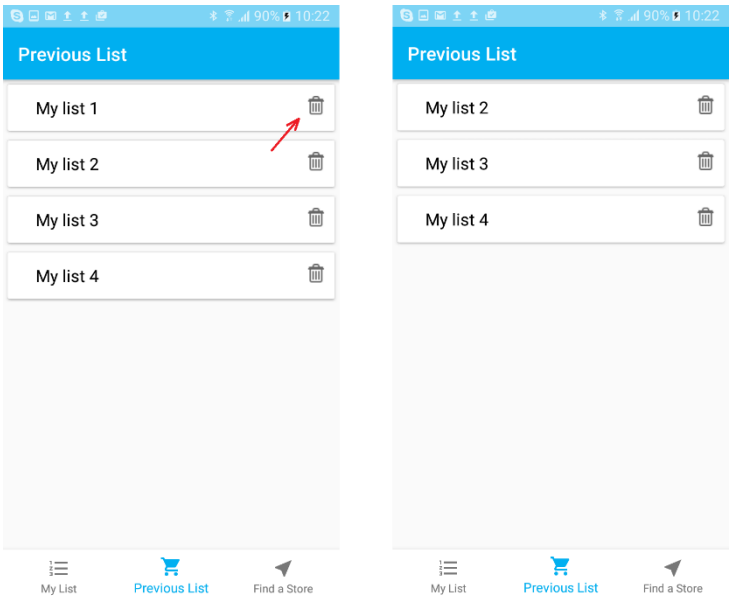


Figure 7. 7 - Smart Shopping Previous List Screens

7.6 Mapper in Action

User can view the nearby supermarkets using the mapper component. Select 'Find a Store' function in the menu and a mapper component will be displayed with user's current location, surrounding supermarkets and the registered supermarkets with their item availability. Selecting an registered supermarket by clicking a marker would bring up the store page which the user can perform store-specific activities including; View detailed items availability, verify the items (auto cross items) once the purchase has been made and view ongoing promotions and deals via web component. Following Figure 7.8 shows the related mapper components screens.

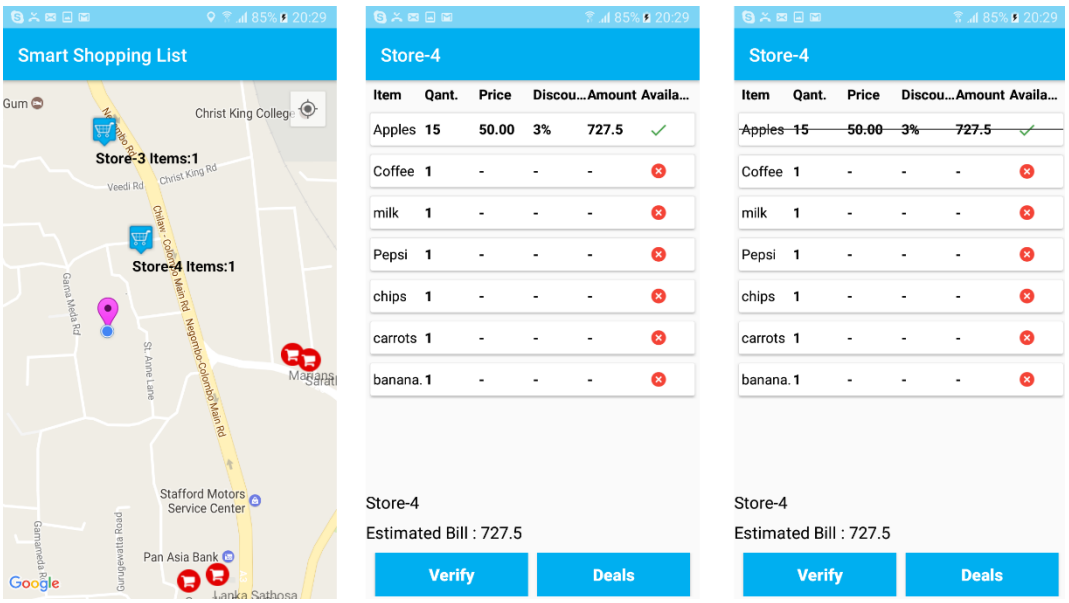


Figure 7. 8 - Smart Shopping Mapper

7.7 BringMe! in Action

User can received grocery item lists via SMS messages. When the user receives an SMS with the tag <BRINGME> with a list of items separated by ','(commas) they can be imported in to the list. Simply select BringMe function, which the app would scan the messages and list the messages in the given format and simply by selecting the message content can be imported. Figure 7.9 shows BringMe! Feature in action.

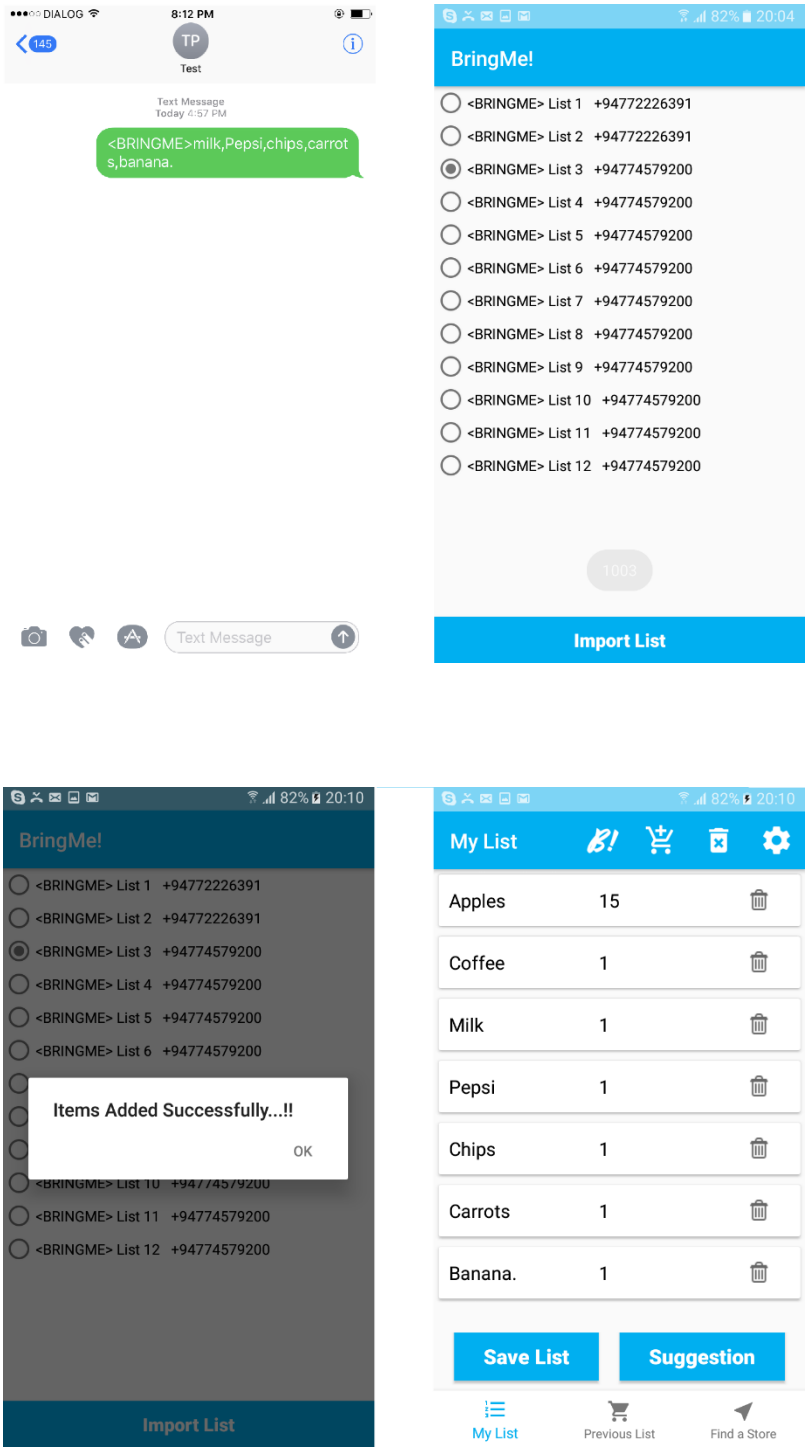


Figure 7. 9 - Smart Shopping BringMe!

7.8 My Account (Settings)

User can logout of the system using the Logout button. Furthermore he can configure his loyalty cards settings in here as well. Figure 7.10 shows the My account page.

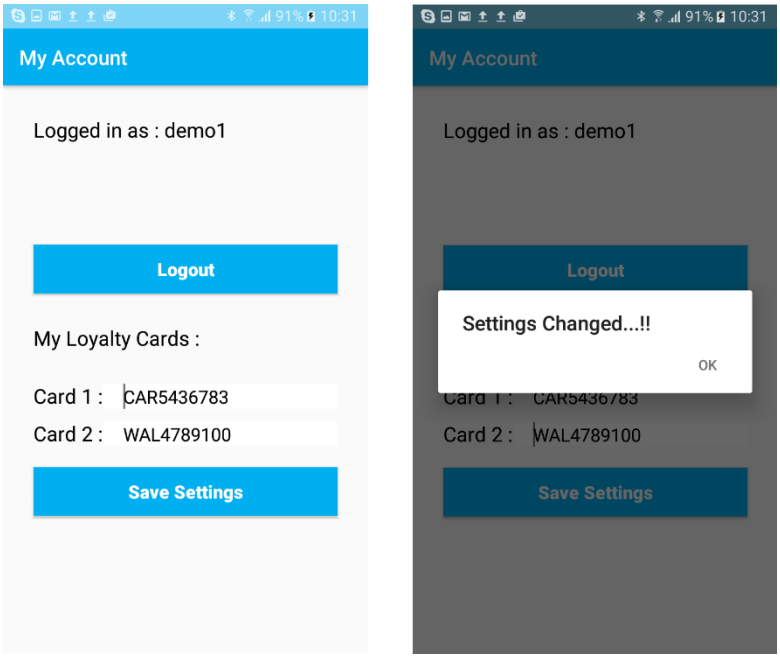


Figure 7. 10 - Smart Shopping My Account

7.9 Summary

This chapter demonstrated an end-to-end usage of the implemented functionalities of the Smart Shopping List. Next chapter is the evaluation chapter which a formal evaluation will be carried out.

Chapter 8

Evaluation

8.1 Introduction

Chapter 7 provided a comprehensive walkthrough of the software. This chapter discusses the evaluation of the approach by a sound assessment of meeting the objectives. In particular, the software prototype was evaluated to analyze whether the hypothesis can be substantiated. The evaluation of the software is done in 3 parts; Algorithm Analysis which is evaluates the outputs of the item mining module, Performance and Security Analysis which addresses the performance, storage and memory management of the solution and Usability Analysis which provides the user feedback on the application and the overall understanding on the application's potential in meeting the its objectives. Here provides the content under algorithmic analysis with test cases, data collection, data analysis and a user feedback on the application itself with the kind of participants, questions being asked and presentation of the results.

8.2 Algorithm Analysis

This analysis is done to find and optimize the item suggestion algorithm to produce desirable results by fine tuning the algorithm parameters; Min. support and Min. confidence. The algorithm is tested for its suggestions made based on user's previously created grocery lists.

8.2.1 Testing Setup

A test criteria of 50 items were considered with 20 grocery lists with 140 item instances. The test cases were designed accordingly by changing values of the algorithm input parameters and observing the outputs.

8.2.2 Test Cases

The following test cases were developed with the given combinations of min. confidence and the min. support and the outputs were observed with a given set of shopping list. In Test cases 5, further comparison analysis was done with Weka software, which a renewed data mining software to evaluate the results of the mining function. Table 8.1 below summarizes the total test cases and the test parameters.

Test Case ID	Test Case Parameters
1	Min sup: 5% Min Confidence: 50%
2	Min sup: 10% Min Confidence: 70%
3	Min sup: 15% Min Confidence: 70%
4	Min sup: 25% Min Confidence: 80%
5	Min sup: 10% Min Confidence: 70% with Weka

Table 8. 1 - Test cases for Algorithm Analysis

8.2.3 Test Case 1 & Results

Test Case Number	Test Case_01
Scenario Objective	Observe the suggestions based on users previous lists made by the Item Suggester for the minimum support of 5% and minimum confidence of 50%.

Precondition

User is logged in to the Smart Shopping List.
User has created 20 previous shopping lists as given.

Input 1

Min. support = 5% (0.05) and Min. confidence = 50% (0.5)
Algorithm is run for identifying patterns for given set of lists.

Expected Outcome 1

Algorithm generates find the frequent item sets and calculates their confidence.

Actual Outcome 1

Algorithm outputs are as follows,
Result 301 generated, first 25 are shown.

```
Eggs,Butter=>0.2,confidence = 1
Eggs,Apples=>0.15,confidence = 0.75
Eggs,Bread=>0.15,confidence = 0.75
Milk,Butter=>0.3,confidence = 0.75
Milk,Tea=>0.25,confidence = 0.625
Milk,Butter=>0.3,confidence = 0.75
Butter,Bread=>0.25,confidence = 0.625
Eggs,Rice=>0.1,confidence = 0.66666666666667
Milk,Rice=>0.15,confidence = 1
Butter,Rice=>0.1,confidence = 0.66666666666667
Rice,Coconut oil=>0.15,confidence = 1
Rice,Tea=>0.1,confidence = 0.66666666666667
Rice,Biscuits=>0.1,confidence = 0.66666666666667
Rice,Chicken=>0.1,confidence = 0.66666666666667
Rice,Potatoes=>0.1,confidence = 0.66666666666667
Rice,Oranges=>0.1,confidence = 0.66666666666667
Rice,Apples=>0.1,confidence = 0.66666666666667
Rice,Bread=>0.1,confidence = 0.66666666666667
Rice,Papadam=>0.1,confidence = 0.66666666666667
Rice,Coke=>0.1,confidence = 0.66666666666667
Milk,Coconut oil=>0.15,confidence = 0.75
Rice,Coconut oil=>0.15,confidence = 0.75
Coconut oil,Chicken=>0.15,confidence = 0.75
Coconut oil,Papadam=>0.15,confidence = 0.75
```

Input 2

User creates the sample lists and requests for suggestions.
Chicken, Coconut oil, Bread, Toothpaste, Chips

Expected Outcome 2

Application generates large set of suggestions based on the given list of items.

Actual Outcome 2

Suggestions are produced and can be added to the list via the front end.

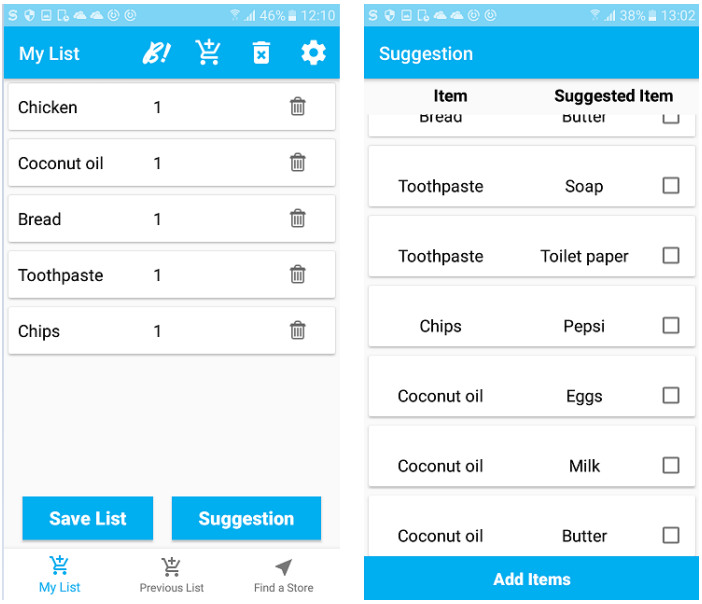


Figure 8. 1 - Test case 1 Suggestions

Observations Made

This configuration produces too many suggestions with unrelated correlations.

8.2.4 Test Case 2 & Results

Test Case Number	Test Case_02
Scenario Objective	Observe the suggestions based on users previous lists made by the Item Suggester for the minimum support of 10% and minimum confidence of 70%.

Precondition

User is logged in to the Smart Shopping List.
User has created 20 previous shopping lists as given.

Input 1

Min. support = 10% (0.1) and Min. confidence = 70% (0.7)
Algorithm is run for identifying patterns for given set of lists.

Expected Outcome 1

Algorithm generates find the frequent item sets and calculates their confidence.

Actual Outcome 1

Algorithm outputs are as follows,
Result 48 generated, first 25 are shown.

```
Eggs,Butter=>0.2,confidence = 1
Eggs,Bread=>0.15,confidence = 0.75
Milk,Butter=>0.3,confidence = 0.75
Milk,Rice=>0.15,confidence = 1
Rice,Coconut oil=>0.15,confidence = 1
Milk,Coconut oil=>0.15,confidence = 0.75
Rice,Coconut oil=>0.15,confidence = 0.75
Coconut oil,Papadam=>0.15,confidence = 0.75
Milk,Tea=>0.25,confidence = 1
Milk,Biscuits=>0.15,confidence = 1
Tea,Biscuits=>0.15,confidence = 1
Biscuits,Bread=>0.15,confidence = 1
Coconut oil,Chicken=>0.15,confidence = 0.75
Milk,Oranges=>0.15,confidence = 1
Tea,Oranges=>0.15,confidence = 1
Eggs,Apples=>0.15,confidence = 0.75
Milk,Apples=>0.15,confidence = 0.75
Butter,Apples=>0.2,confidence = 1
Butter,Bread=>0.25,confidence = 0.83333333333333
Soap,Toothpaste=>0.15,confidence = 1
Soap,Toilet paper=>0.15,confidence = 1
Tea,Bread=>0.15,confidence = 0.75
Chips,Pepsi=>0.15,confidence = 0.75
Toothpaste,Toilet paper=>0.15,confidence = 1
Butter,Jam=>0.15,confidence = 0.75
```

Input 2

User creates the sample lists and requests for suggestions.

Chicken, Coconut oil, Bread, Toothpaste, Chips

Expected Outcome 2

Application generates large set of suggestions based on the given list of items.

Actual Outcome 2

Suggestions are produced and can be added to the list via the front end.

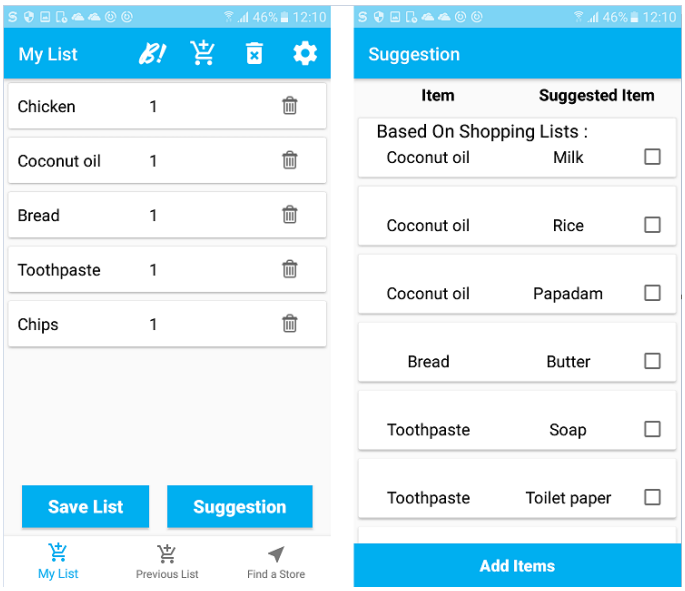


Figure 8. 2 - Test Case 2 Outputs

Observations Made

This configuration produces sufficient amount of suggestions.

8.2.5 Test Case 3 & Results

Test Case Number	Test Case_03
Scenario Objective	Observe the suggestions based on users previous lists made by the Item Suggester for the minimum support of 15% and minimum confidence of 70%.

Precondition

User is logged in to the Smart Shopping List.
User has created 20 previous shopping lists as given.

Input 1

Min. support = 15% (0.15) and Min. confidence = 70% (0.7)
Algorithm is run for identifying patterns for given set of lists.

Expected Outcome 1

Algorithm generates find the frequent item sets and calculates their confidence.

Actual Outcome 1

Algorithm outputs are as follows,
Result 17 generated, all are shown.

```
Eggs,Butter=>0.2,confidence = 1
Milk,Rice=>0.15,confidence = 1
Rice,Coconut oil=>0.15,confidence = 1
Milk,Tea=>0.25,confidence = 1
Milk,Biscuits=>0.15,confidence = 1
Tea,Biscuits=>0.15,confidence = 1
Biscuits,Bread=>0.15,confidence = 1
Milk,Oranges=>0.15,confidence = 1
Tea,Oranges=>0.15,confidence = 1
Butter,Apples=>0.2,confidence = 1
Butter,Bread=>0.25,confidence = 0.83333333333333
Soap,Toothpaste=>0.15,confidence = 1
Soap,Toilet paper=>0.15,confidence = 1
Soap,Toilet paper=>0.15,confidence = 1
Toothpaste,Toilet paper=>0.15,confidence = 1
Coconut oil,Papadam=>0.15,confidence = 1
Chips,Pepsi=>0.15,confidence = 1
```

Input 2

User creates the sample lists and requests for suggestions.
Chicken, Coconut oil, Bread, Toothpaste, Chips

Expected Outcome 2

Application generates large set of suggestions based on the given list of items.

Actual Outcome 2

Suggestions are produced and can be added to the list via the front end.

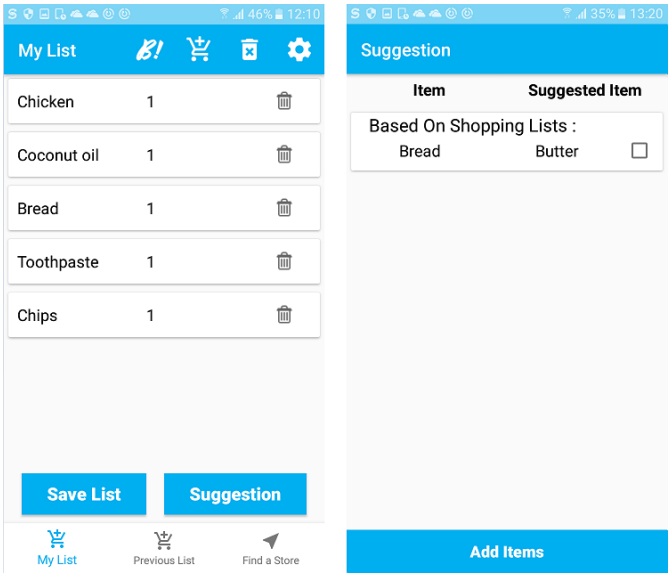


Figure 8. 3 - Test Case 3 Outputs

Observations Made

This configuration filters item sets further however it produces fairly short amount of suggestions.

8.2.6 Test Case 4 & Results

Test Case Number	Test Case_04
Scenario Objective	Observe the suggestions based on users previous lists made by the Item Suggester for the minimum support of 25% and minimum confidence of 80%.

Precondition

User is logged in to the Smart Shopping List.
User has created 20 previous shopping lists as given.

Input 1

Min. support = 25% (0.25) and Min. confidence = 80% (0.8)
Algorithm is run for identifying patterns for given set of lists.

Expected Outcome 1

Algorithm generates find the frequent item sets and calculates their confidence.

Actual Outcome 1

Algorithm outputs are as follows,
Result 2 generated, all are shown.

```
Milk,Tea=>0.25,confidence = 1
Butter,Bread=>0.25,confidence = 0.83333333333333
```

Input 2

User creates the sample lists and requests for suggestions.
Chicken, Coconut oil, Bread, Toothpaste, Chips

Expected Outcome 2

Application generates large set of suggestions based on the given list of items.

Actual Outcome 2

Suggestions are produced and can be added to the list via the front end.
Same suggestion as in Figure 8.3 is available.

Observations Made

Generated suggestions using this configuration is largely insufficient.

8.2.7 Test Case 5 & Results

Test Case Number	Test Case_05
Scenario Objective	Observe the suggestions generated by Weka for the optimum settings for Smart Shopping list (min. support of 10% and min. confidence of 70%.)

Precondition

User has created the ***smartshoppinglist.arff*** file for the above data set for Weka software.
User has fed the created ***smartshoppinglist.arff*** file to the system for association mining.

Input 1

Set the mining configuration as delta = 0.1 minMetric = 0.7
weka.associations.Apriori -N 25 -T 0 -C 0.7 -D 0.1 -U 1.0 -M 0.1 -S -1.0 -c -1
Select Start.

Expected Outcome 1

Weka find frequency patterns and generates the results

Actual Outcome 1

Following results produced by Weka.

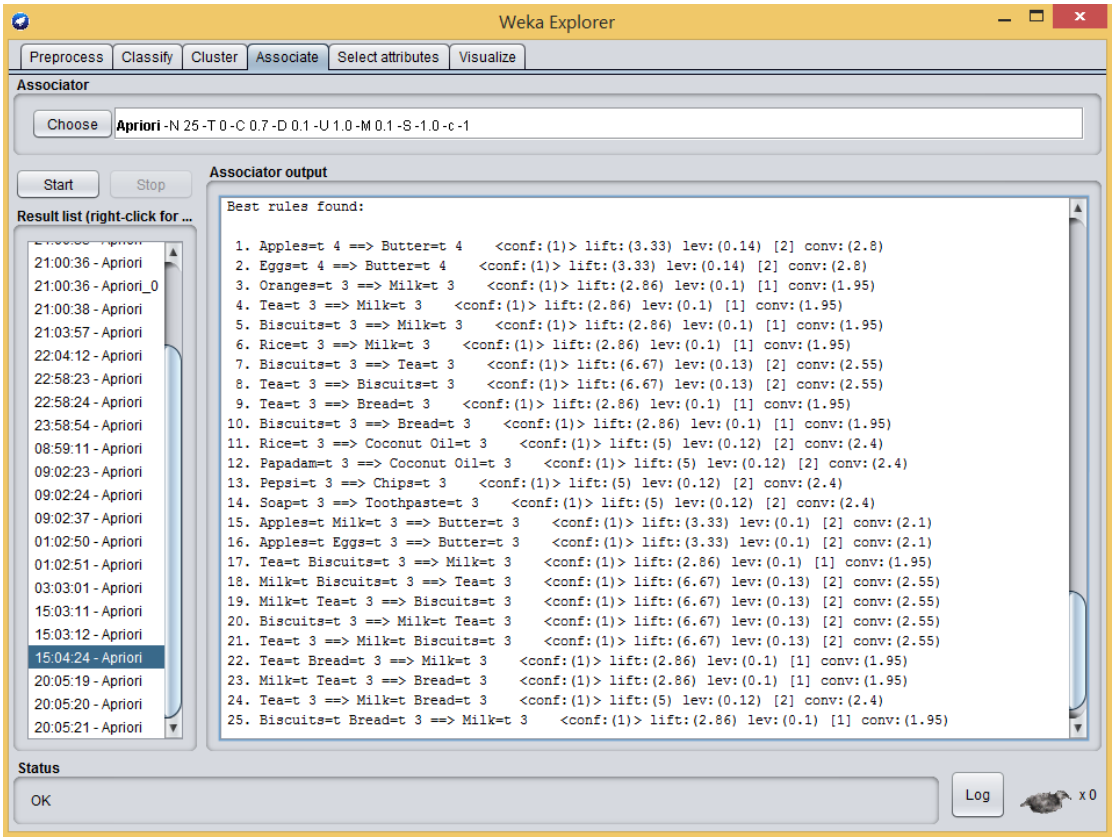


Figure 8. 4 – Results produced by Weka for -C 0.7 -M 0.1

Observations Made

The results generated by Weka and Item Suggestions module share many similarities.

8.2.8 Algorithmic Data Analysis

The above test cases were carried out, results were produced, collected and observations have also been made. The summary of the test results is shown below in Table 8.2.

Test Case ID	Status	Test Method	Tested Release	Comments
Test Case_01	Pass	Manual	SSL v1.0	This configuration produces too many suggestions with unrelated correlations.
Test Case_02	Pass	Manual	SSL v1.0	This configuration produces sufficient amount of suggestions.
Test Case_03	Pass	Manual	SSL v1.0	This configuration filters item sets further however it produces fairly short amount of suggestions.
Test Case_04	Pass	Manual	SSL v1.0	Generated suggestions using this configuration is largely insufficient.
Test Case_05	Pass	Manual	SSL v1.0	The results generated by Weka and Item Suggestions module share many similarities

Table 8. 2 - Test results and Comments

Based on the observations made, it was evident that the most suitable set of configurations for making association rule mining on the data based on the previous lists created by user, have been demonstrated in the Test Case 2. Desirable amount of suggestions were made and application of those rules on the current list space was also at a health level. Considering the results which were generated by other 2 extreme ends were not satisfactory as Test Case 3 and 4 made insufficient amount suggestions while Test Case 1 flooded the application with patterns which the practical aspect of those seemed very scarce i.e {Coconut oil, Butter}. The comparison analysis with Weka Software made in the Test Case 5, further proved that the algorithm functions very well in finding pattern among the item sets. Both algorithms produced very similar if not the same results. It was noticed that 64% (16) of top 25 results by Weka software were available in the top 25 of the generated 48 results by the module.

8.3 Storage, Performance and Security Analysis

Performance and security were considered since the design to the later stages of the solution. Many of the implementation level notes on this have been discussed in the previous chapter. Following Figure 8.5 shows the application properties of a fresh installation on device.

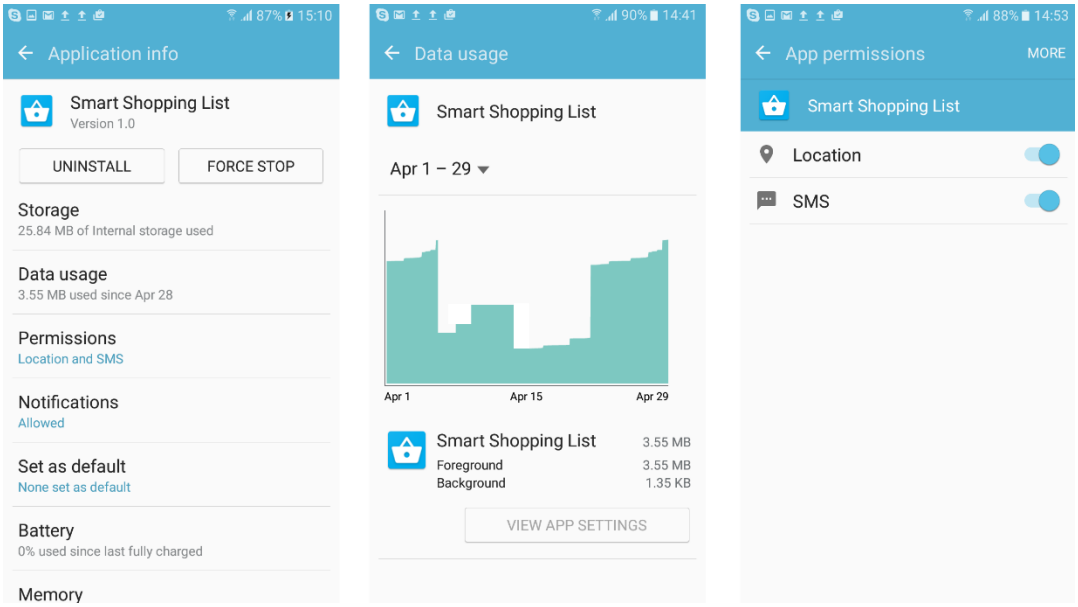


Figure 8. 5 - Application Info

Application weights only roughly around 25MB. This allows the application to be installed on devices without requiring large storage requirements.

Data usage for a session of 30 days with moderate app use was only 3.55MB which is advantageous in terms of mobile data and a good news for the users who have limited data plans. The system design decision taken to make use of server-side query processing and the use of JSON data, undisputedly helped achieving this stats. Average memory usage was also at a healthy level of 75MB.

The Smart Shopping List makes use of few permissions; Location for geolocation services by the mapper and SMS by BringMe! Module. No further permission are required which makes the app desirable for the users who are concerned about their privacy.

8.4 Usability Analysis

A usability analysis for the Smart Shopping List has also been conducted to ensure that the software application is built in a way that the usability of the application has been properly maintained and the users can get their objectives met within the software. This has been achieved by a feedback taken after a 3 weeks of application usage by 15 individuals.

8.4.1 Participants

Seventeen individuals who have android smartphones have been used for the usability evaluation. Participants were picked from multiple age groups, 9 from 22-40 years range with a rough equal distribution of gender and 7 picked from 40-50 years range with a distribution of 5 females and 2 males. 1 female was taken from the age group of 50 to 65. All the participants except the last had an average prior knowledge in mobile applications. Participants who were 40+ were novice to average smartphone users.

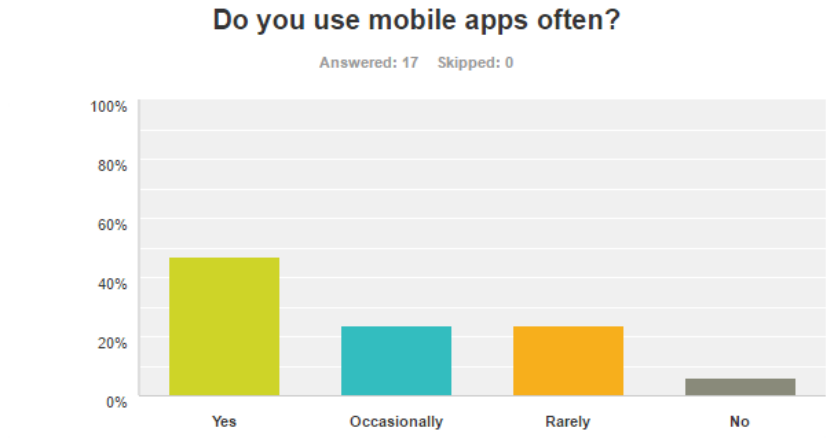
8.4.2 Testing Setup

Participants were given the application and no training or walkthrough was given, having objective of the survey was to observe how an individual would get comfortable with the application interface and its function to perform their intended activities. Learning curve was assessed and overall usage related feedback was taken from the individuals. At the end of the 3rd week an online survey was created and the individuals were invited to share their feedback on the application.

8.4.3 Data Collection

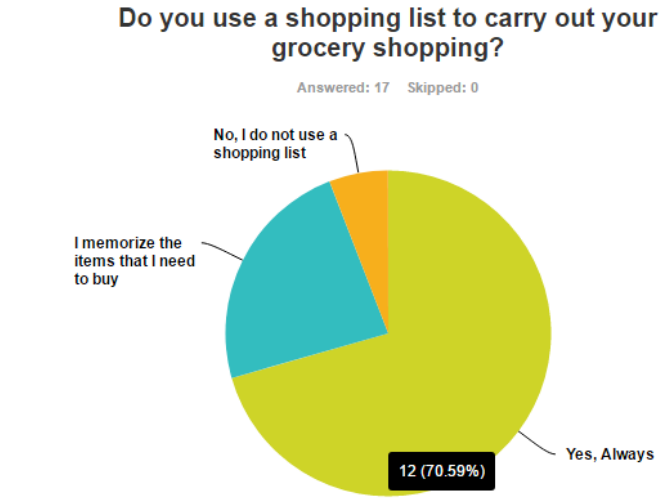
Feedback from the participants are obtained through an online questionnaire which was powered by SurveyMonkey and the links for the survey were shared with the individuals via emails. The layout of the questionnaire used are shown in Appendix D. The following series of figures 8.6 to 8.13 show the data which has been collected and their distribution. Following Figure 8.6 shows the mobile app usage by each

individual with the majority of 47.06% and minority of 5.88% which is the individual who falls in to 50 to 65 category.



Answer Choices	Responses	
Yes	47.06%	8
Occasionally	23.53%	4
Rarely	23.53%	4
No	5.88%	1
Total		17

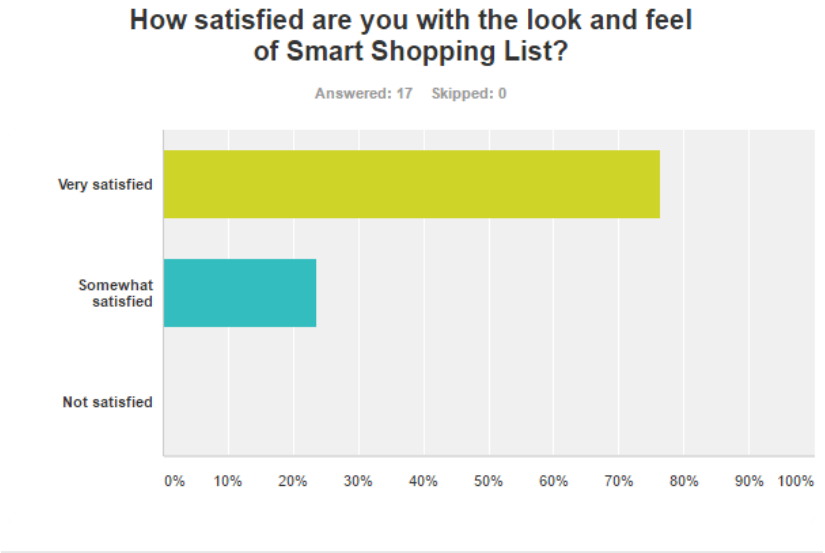
Figure 8. 6 – Usability Feedback I - General Mobile App Usage



Answer Choices	Responses	
Yes, Always	70.59%	12
I memorize the items that I need to buy	23.53%	4
No, I do not use a shopping list	5.88%	1
Total		17

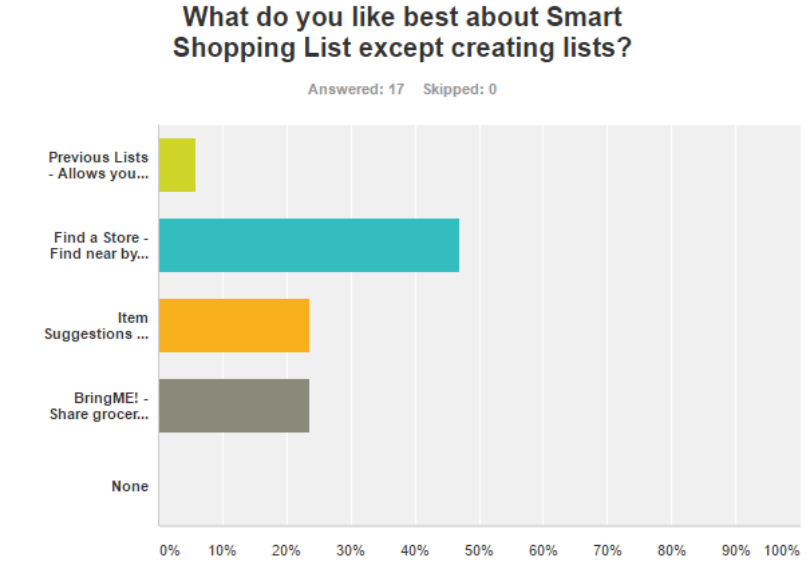
Figure 8. 7 – Usability Feedback II – List Creation Practice

As shown in the Figure 8.7 majority of the participants do create shopping lists for their grocery shopping.



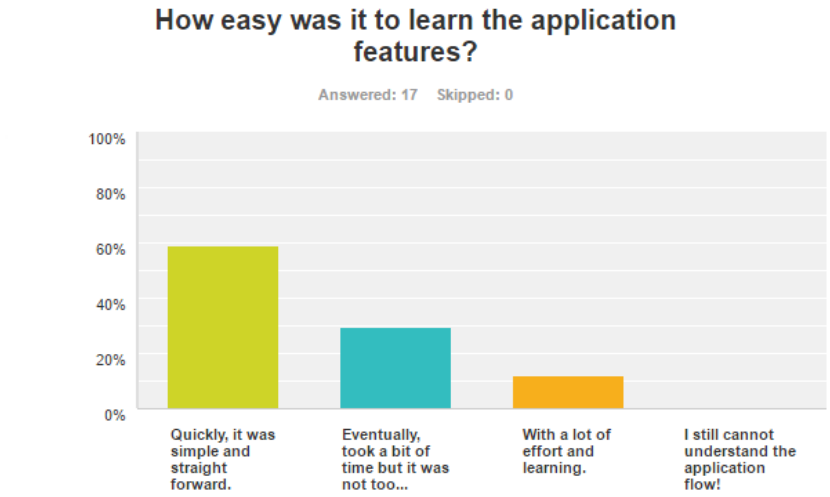
Answer Choices	Responses	
Very satisfied	76.47%	13
Somewhat satisfied	23.53%	4
Not satisfied	0.00%	0
Total		17

Figure 8. 8 – Usability Feedback III – Look and Feel



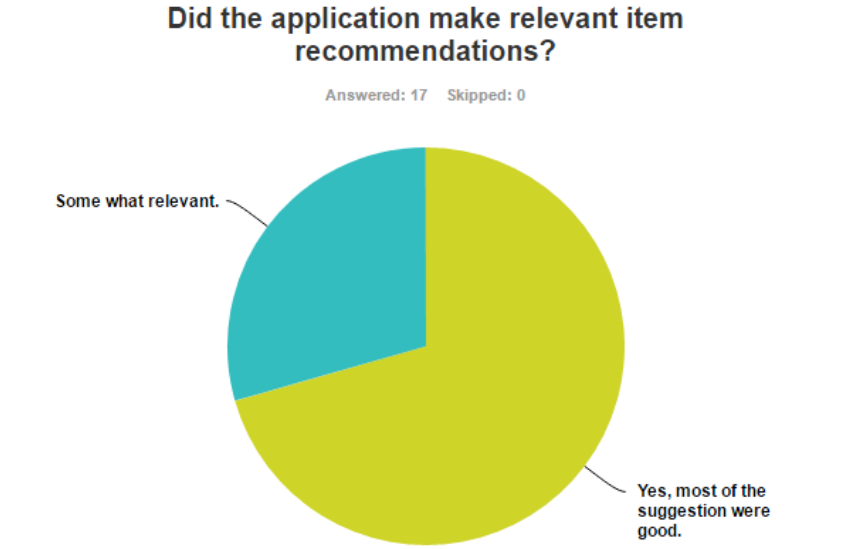
Answer Choices	Responses	
Previous Lists - Allows you to maintain a set of grocery lists	5.88%	1
Find a Store - Find near by supermarkets, Shop specific activities	47.06%	8
Item Suggestions - Recommend items based on your previous data	23.53%	4
BringME! - Share grocery lists via SMS messages	23.53%	4
None	0.00%	0
Total		17

Figure 8. 9 – Usability Feedback IV – Component Preference



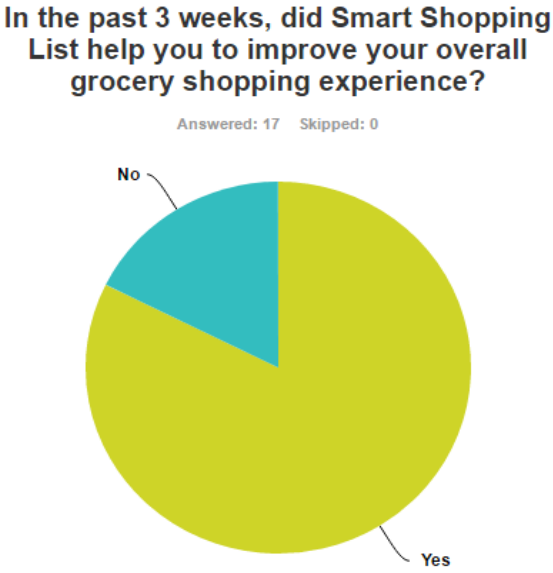
Answer Choices	Responses	
Quickly, it was simple and straight forward.	58.82%	10
Eventually, took a bit of time but it was not too difficult.	29.41%	5
With a lot of effort and learning.	11.76%	2
I still cannot understand the application flow!	0.00%	0
Total		17

Figure 8. 10 – Usability Feedback V – Learning Curve



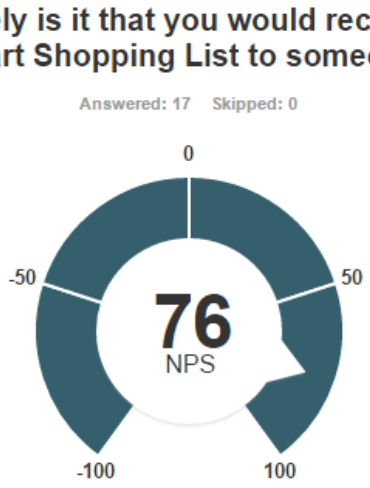
Answer Choices	Responses	
Yes, most of the suggestion were good.	70.59%	12
Some what relevant.	29.41%	5
No, totally incorrect suggestions	0.00%	0
Total		17

Figure 8. 11 – Usability Feedback VI – Suggestions Accuracy



Answer Choices	Responses	
Yes	82.35%	14
No	17.65%	3
Total		17

Figure 8. 12 – Usability Feedback VII – Application's Effectiveness



Detractors (0-6)	Passives (7-8)	Promoters (9-10)	Net Promoter® Score
6% 1	12% 2	82% 14	76

Figure 8. 13 – Usability Feedback VIII – Approval Ratings

Furthermore the participants were asked for their thoughts on the possible improvements of the software. There were many requests to make the application platform-independent more or less to make it available in IOS. Several functional improvements were suggested as well such as, introducing item categories, merge

with online grocery shopping and some cosmetic improvements like making the UI elements bigger. This was requested by the oldest participant.

8.4.4 Overall Data Analysis

A profound analysis with test cases, test results a comparison was done for the algorithm in section 8.2 under algorithm analysis. A Technical analysis on the application's memory management and its performance was also done with a note of the security in the end. Having mentioned that, the usability of the software and the overall usage patterns were uncovered with the usability feedback session. As per Figure 8.6 and 8.7 , having a practice of using of mobile applications and creating grocery lists is a significant factor for a mobile application like this to succeed in providing its services. Over 76% were happy with the design of the application which let us make the direct judgement that the design allowed the users to get their intended activities done smoothly. Results shown Figure 8.8 and 8.10 further support that. The one person who had difficulty understanding the application content was novice smartphone user and was not very confident with mobile applications, although she found the application concept is useful. The mapper component was the most popular module. This suggests that participants would be more benefited from the geolocation services than others however majority voted for this module were millennials. Item suggestions and BringMe were popular among other age groups which proved to be more extensive grocery shoppers. The algorithm is proved to be functioning well and the application provides a great use to majority of the participants to carry out their grocery shopping as seen in Figures 8.11 – 8.13, demonstrating an overall approval net score of 76% with more than 82% positive ratings for a population of 17 participants.

8.5 Summary

This Chapter provided a sound evaluation of the Smart Shopping List. An in-depth analysis on the data mining algorithm was done and an overall evaluation was conducted including the usability of the application in the end. Next chapter gives and overview on the conclusions, limitations and the future work for the proposed solution.

Chapter 9

Conclusion and Further Work

9.1 Introduction

Previous chapters gave a concise understanding of the problem identified and a proposed solution. This chapter concludes the research presented in this thesis. As the hypothesis recollected, Developing a mobile application with the addressed functionalities can significantly improve the quality of the shopping list creation process and grocery shopping in general of an individual. From the evaluation carried out in the chapter 8, It is lucid that more than 82% of a group of 17 people have been benefited by the Smart Shopping List. This chapter submit the overall conclusion, objective-wise achievements and limitations along with suggested further work.

9.2 Overall Conclusions

Grocery shopping is a simple yet a crucial task that everyone all actively participates in the daily lives. Despite of how important the task would be, because of the sheer nature of the of the grocery shopping process, it is widely disliked by many. Many techniques have been adopted by people in terms of improving the quality of the process to make it more efficient and convenient. There have been many technological advancements been made towards enhancing grocery shopping specially in the mobile technology, yet it was evident that there were many issues which were not addressed and improvements to be made. Therefore a comprehensive study was conducted to extract the current issues that people face during their grocery shopping and presented a mobile software solution which addresses them in a proper manner. The approach was evaluated all-inclusive both programmatically and usability. The two main evaluation reveal that; i) The algorithm showed promising results in terms of generating personalized item suggestions, ii) majority including the millennials has been benefited from the proposed solution with a greater approval rate as 82% and a net recommendation score of 76%. The remaining negatives due to, some people do not prefer creating shopping lists in general or they prefer to keep a

mental lists instead of a physical one. Some people also revealed that they do not carry the phone at all times or do not use mobile apps apart from the natives in general. However the majority revealed that the solution assisted them in many ways to enhance their grocery shopping experience. As such overall conclusion supports the hypothesis that novel Smart Shopping List can significantly improve the quality of the lifestyle of a grocery shopper with its many services.

9.3 Objective-wise Conclusions

As objectives were set in the beginning of the study, the assessment of the achievements of them is of critical note. To further recall the set objectives,

I) To critically study materials on grocery shopping lifestyle, mobile-assisted grocery shopping and data mining techniques for interest mining.

II) To do an in-depth study of data association with a particular emphasis on Apriori algorithm for interest mining.

III) To develop a software solution employed with geolocation services and Apriori algorithm for suggestions.

IV) Evaluate the effectiveness, accuracy and usability of the system.

Achievement of objective one is evident from the comprehensive literature review conducted with a multitude of research materials related to grocery shopping lifestyle, mobile-assisted grocery shopping with technological advancements towards the very domain and datamining techniques for user interest mining in the literature review chapter. There six presumptions were constructed which has not been addressed by earlier developments and defined this thesis based on that.

The achievements of objective 2 have also been supported by the literature review chapter. In addition, Chapter 3 presented a description on each technology selected for developing the proposed solution including the data mining algorithm for the suggestion mining.

Achievements in objective 3 are evident from chapters from Approach, Design, Implementation and System in practice which provided the approach to the problem,

the design of the proposed solution, a comprehensive note on the implementation of the system and a walkthrough of the solution in practice respectively.

The objective related to the evaluation of the hypothesis is presented in chapter 8. Complete evaluation was done in 3 aspects; systematic algorithmic analysis on the effectiveness of data mining algorithm with proper test cases and compassion analysis, technical note on application's storage, memory and security management and an overall analysis on the previous evaluation combined with a usability and acceptance analysis. It was revealed that the overall success rate of the project is over 82% with a recommendation net score of 76%.

9.4 Limitations and Future work

Whilst the Smart Shopping List was an overall success, some limitations were identified. As simple and effective the Apriori algorithm is, it inherits weaknesses such as going multiple iterations which can cause to large memory consumptions in larger databases. It is suggested to improve the mining algorithm to overcome this issue by making enhancements, i.e Matrix Apriori, which is a promising improvement to the native Apriori. Algorithm. Furthermore stronger filtrations can be made to strengthen the algorithm by introducing interestingness measure such as Kappa or J-measure. It us further suggested to implement user models of preferred items and used abbreviations so that in a scenario, the application might be able to understand which specific kind of milk (brand, quality) is intended to be bought when the user just writes "milk". Lastly, enabling Smart Shopping List on other mobile platforms, such as growing IOS base and windows phone would also be beneficial for non-android users .

9.5 Summary

This chapter provided the final outcome of the thesis in the sections of overall conclusion and objective-wise conclusions. Chapter ends by addressing the limitations of solutions while pointing out the future work intended. Thus concludes work presented by previous chapters and the thesis itself, 'The Smart Shopping List'.

Appendix A

Item Suggestion Algorithm Pt-1

```
<?php
/* This file is the implementation of Apriori Algorithm based on Shopping
List data to suggest items to users*/

header('content-type:text/json');
include_once dirname(__FILE__).'/config.php';

function test(){
$arr = array();
$arr1 = array();
$res = array();
$nxt = array();
$nxt1 = array();
$nxt2 = array();
$nxt3 = array();
$nxt4 = array();
$nxt5 = array();
$nxt6 = array();
$nxt7 = array();
$nxt8 = array();
$a = 0;
$n=0;
$min = 0;
$rest = 0;

/*********  Fetching ShoppingList rows from the Database based on User Id
 ********************/

$connect = mysqli_connect(DB_HOST,DB_USER,DB_PASSWORD) or die("Error");
$sql = mysqli_select_db($connect,DB_NAME) or die("ERROR");
  $uid = $_POST['uid'];
  $data = mysqli_query($connect,"SELECT * FROM `shoppinglist` WHERE `UID` =
'$uid';");
        while($row = mysqli_fetch_array($data)){
                $n++;        // n is number of rows in the DB for the given
user
                $shop = $row['ShoppingListItems'];
                $json_output = json_decode($shop, true);  // Decoding the
JSON data of each row
                        foreach($json_output as $s) {
                                $item= $s['item'];
                                array_push($arr, $item);          // Pushing
the items of every list in an array
                        }
        }
        /********   creating associative array for the items  **********/
        foreach ($arr as $as => $value) {
        $arr1[$value] = 1;
        }


        /********   Counting the items  **********/
        foreach($arr1 as $nn => $value){    //
                foreach($arr as $mm){
                        if($mm==$nn){
                                $value = $value+1;
                                array_push($res, $nn);
                        }
                }
        }

        $nxt = array_count_values($res);
```

```
        /********  Support count is 10% of the number of rows but if rows
less than 6, it would be 2   **********/
        $min = 0.10*$n;
        $mini = round($min);
                if($n<=6){
                  $mini = 2;
                }else{
                  $mini;
                }

        /********  Removing the items from array 'nxt' which have less than
Support count  **********/
        foreach ($nxt as $nx => $val1) {
                if($val1<$min){
                        unset($nxt[$nx]);
                }
        }

        /******** Creating an Array '$nxt2' having pair of items from
minimum support count table  **********/
        $count = count($nxt);
        $keys = array_keys($nxt);
         for($i=0; $i < count($keys); ++$i) {

                for($j=$i+1; $j< count($keys); ++$j){

                  $temp = array($keys[$i],$keys[$j]);

                  array_push($nxt2, $temp);

                }
        }

      foreach ($nxt2 as $nxt2item) {
        $k = $nxt2item[0];
        $l = $nxt2item[1];
        $uid = $_POST['uid'];

         $data1 = mysqli_query($connect,"SELECT * FROM `shoppinglist` WHERE
`UID` = '$uid';");
                        while($row1 = mysqli_fetch_array($data1)){
                                $shop1 = $row1['ShoppingListItems'];
                                $json_output1 = json_decode($shop1, true);
                                        foreach ($json_output1 as $match) {
                                                $item2 = $match['item'];
                                                $count = 0;
                                                        if($item2==$k){


  foreach ($json_output1 as $titem) {

  $item3 = $titem['item'];

    if($item3==$l){

            $simple = $k.",".$l;

            array_push($nxt5, $simple);

            $nxtt = array_count_values($nxt5);

    }
                                                                        }
                                                                }
                                        }

                        }
            }
```

```
        /* choose pairs only with the min supp */
        foreach ($nxtt as $nxttitem => $nxtval) {
          if($nxtval<$mini){
                unset($nxtt[$nxttitem]);

          }
        }

        /* Calculating confidence.*/
        /* minimum confidence = 70% */
         foreach ($nxt as $nxxitem => $nxxval) {
                 foreach ($nxtt as $nxitem => $nxval) {
                        if (strpos($nxitem,$nxxitem) !== false) {
                                $g = $nxval/$nxxval;
                                        if($g>0.7){
                                          $fg = explode(',', $nxitem);
                                          $it1 = $fg[0];
                                          $it2 = $fg[1];
                                                if($it1==$nxxitem){

  array_push($nxt7,$fg);
                                                  }
                                                  else{

  $hh = array($it2,$it1);

  array_push($nxt7,$hh);
                                                  }
                                        }
                        }
                 }
        }

        /* The selected pairs passing threshold confidence are pushed into
'nxt8' array */
        foreach($nxt7 as $k){
                $catList["item1"] = $k[0];
                $catList["item2"] = $k[1];
                array_push($nxt8, $catList);
        }

        /* Encoding the data from array to JSON data */
        print_r(json_encode(array('data'=>$nxt8)));

}
test();
?>
```

Appendix B

Database Schema

```
-- -----------------------------------------------------------------------------
-- Schema SmartShoppingListApp
-- -----------------------------------------------------------------------------
DROP SCHEMA IF EXISTS `SmartShoppingListApp` ;
CREATE SCHEMA IF NOT EXISTS `SmartShoppingListApp` ;

-- -----------------------------------------------------------------------------
-- Table SmartShoppingListApp.User
-- -----------------------------------------------------------------------------
CREATE TABLE IF NOT EXISTS `SmartShoppingListApp`.`User` (
  `UID` INT NOT NULL AUTO_INCREMENT,
  `UName` VARCHAR(300) NOT NULL,
  `LoyaltyCardNo1` VARCHAR(25) NULL,
  `LoyaltyCardNo2` VARCHAR(25) NULL,
  `IsActive` TINYINT(1) NOT NULL,
  PRIMARY KEY (`UID`));

-- -----------------------------------------------------------------------------
-- Table SmartShoppingListApp.Shop
-- -----------------------------------------------------------------------------
CREATE TABLE IF NOT EXISTS `SmartShoppingListApp`.`Shop` (
  `ShopID` SMALLINT NOT NULL AUTO_INCREMENT,
  `ShopName` VARCHAR(50) NOT NULL,
  `IsActive` TINYINT(1) NOT NULL,
  PRIMARY KEY (`ShopID`),
  UNIQUE INDEX `UQ__Shop__649A7D9636DA1420` (`ShopName` ASC));

-- -----------------------------------------------------------------------------
-- Table SmartShoppingListApp.Item
-- -----------------------------------------------------------------------------
CREATE TABLE IF NOT EXISTS `SmartShoppingListApp`.`Item` (
  `ItemID` INT NOT NULL AUTO_INCREMENT,
  `ItemsName` VARCHAR(200) NOT NULL,
  `ItemPrice` DECIMAL(9,2) NOT NULL,
  `IsActive` TINYINT(1) NOT NULL,
  PRIMARY KEY (`ItemID`));

-- -----------------------------------------------------------------------------
-- Table SmartShoppingListApp.ItemBranch
-- -----------------------------------------------------------------------------
CREATE TABLE IF NOT EXISTS `SmartShoppingListApp`.`ItemBranch` (
  `ItemBranchID` INT NOT NULL AUTO_INCREMENT,
  `ItemID` INT NULL,
  `BranchID` SMALLINT NULL,
  `Discount` TINYINT UNSIGNED NULL,
  PRIMARY KEY (`ItemBranchID`),
  CONSTRAINT `FK__ItemBranc__ItemI__1CF15040`
    FOREIGN KEY (`ItemID`)
    REFERENCES `SmartShoppingListApp`.`Item` (`ItemID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__ItemBranc__Branc__2D27B809`
    FOREIGN KEY (`BranchID`)
    REFERENCES `SmartShoppingListApp`.`Branch` (`BranchID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);

-- -----------------------------------------------------------------------------
-- Table SmartShoppingListApp.Sales
-- -----------------------------------------------------------------------------
CREATE TABLE IF NOT EXISTS `SmartShoppingListApp`.`Sales` (
  `SalesID` INT NOT NULL AUTO_INCREMENT,
  `BranchID` SMALLINT NULL,
```

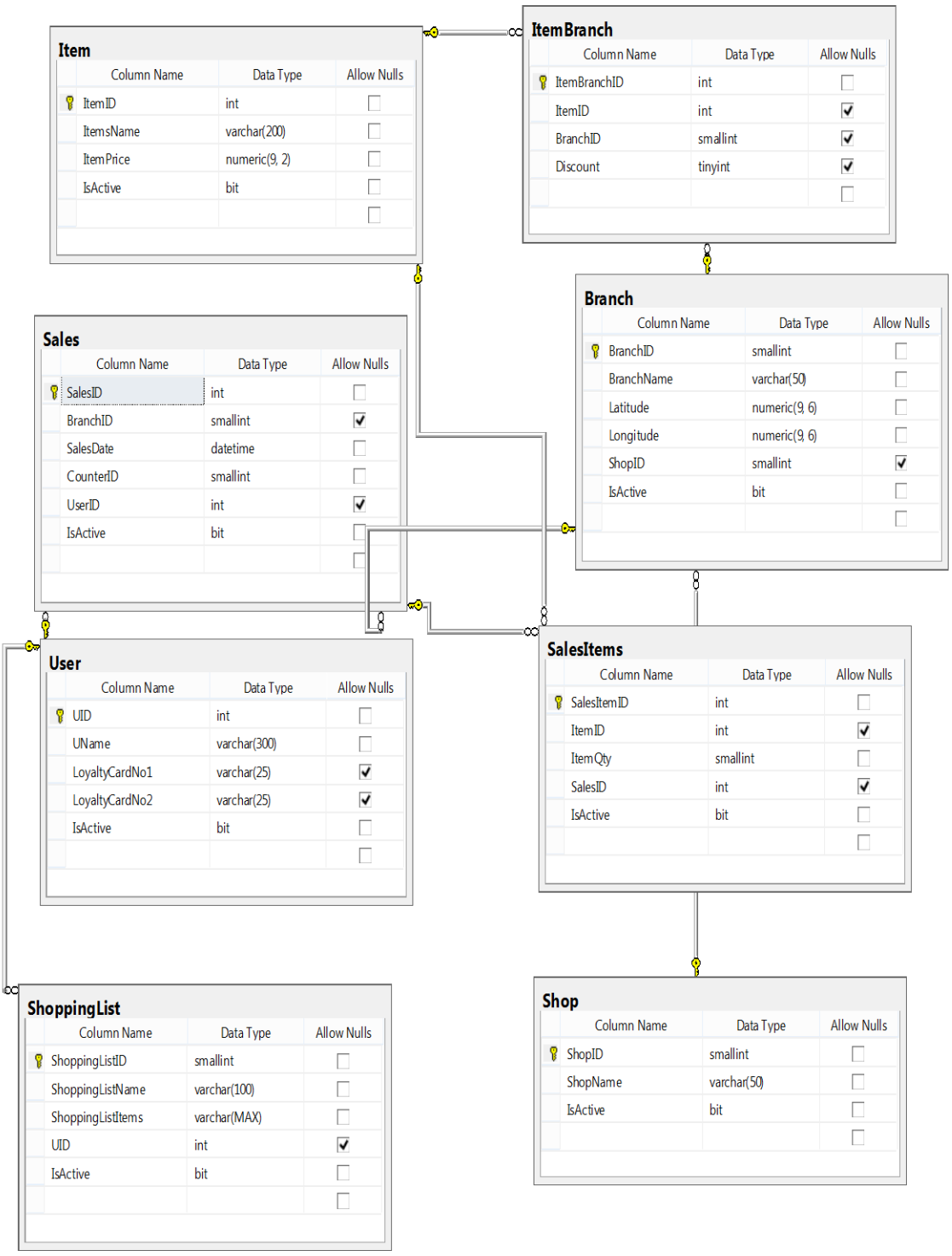
```
  `SalesDate` DATETIME(6) NOT NULL,
  `CounterID` SMALLINT NOT NULL,
  `UserID` INT NULL,
  `IsActive` TINYINT(1) NOT NULL,
  PRIMARY KEY (`SalesID`),
  CONSTRAINT `FK__Sales__UserID__267ABA7A`
    FOREIGN KEY (`UserID`)
    REFERENCES `SmartShoppingListApp`.`User` (`UID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__Sales__BranchID__2C3393D0`
    FOREIGN KEY (`BranchID`)
    REFERENCES `SmartShoppingListApp`.`Branch` (`BranchID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
-- -----------------------------------------------------------------------------
-- Table SmartShoppingListApp.SalesItems
-- -----------------------------------------------------------------------------
CREATE TABLE IF NOT EXISTS `SmartShoppingListApp`.`SalesItems` (
  `SalesItemID` INT NOT NULL AUTO_INCREMENT,
  `ItemID` INT NULL,
  `ItemQty` SMALLINT NOT NULL,
  `SalesID` INT NULL,
  `IsActive` TINYINT(1) NOT NULL,
  PRIMARY KEY (`SalesItemID`),
  CONSTRAINT `FK__SalesItem__ItemI__24927208`
    FOREIGN KEY (`ItemID`)
    REFERENCES `SmartShoppingListApp`.`Item` (`ItemID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__SalesItem__Sales__25869641`
    FOREIGN KEY (`SalesID`)
    REFERENCES `SmartShoppingListApp`.`Sales` (`SalesID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);

-- -----------------------------------------------------------------------------
-- Table SmartShoppingListApp.Branch
-- -----------------------------------------------------------------------------
CREATE TABLE IF NOT EXISTS `SmartShoppingListApp`.`Branch` (
  `BranchID` SMALLINT NOT NULL AUTO_INCREMENT,
  `BranchName` VARCHAR(50) NOT NULL,
  `Latitude` DECIMAL(9,6) NOT NULL,
  `Longitude` DECIMAL(9,6) NOT NULL,
  `ShopID` SMALLINT NULL,
  `IsActive` TINYINT(1) NOT NULL,
  PRIMARY KEY (`BranchID`),
  CONSTRAINT `FK__Branch__ShopID__2B3F6F97`
    FOREIGN KEY (`ShopID`)
    REFERENCES `SmartShoppingListApp`.`Shop` (`ShopID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);

-- -----------------------------------------------------------------------------
-- Table SmartShoppingListApp.ShoppingList
-- -----------------------------------------------------------------------------
CREATE TABLE IF NOT EXISTS `SmartShoppingListApp`.`ShoppingList` (
  `ShoppingListID` SMALLINT NOT NULL AUTO_INCREMENT,
  `ShoppingListName` VARCHAR(100) NOT NULL,
  `ShoppingListItems` LONGTEXT NOT NULL,
  `UID` INT NULL,
  `IsActive` TINYINT(1) NOT NULL,
  PRIMARY KEY (`ShoppingListID`),
  CONSTRAINT `FK__ShoppingLis__UID__300424B4`
    FOREIGN KEY (`UID`)
    REFERENCES `SmartShoppingListApp`.`User` (`UID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
SET FOREIGN_KEY_CHECKS = 1;
```

Appendix C

Database Diagram



Appendix D
Details of Evaluation

The questionnaire is attached with next page in the original format.

References:

[1] "44 facts about mobile apps (yes, 44)," *Calvium*, 05-Aug-2016. .

[2] "Mobile Apps: Saviours for the modern-day lifestyle…," *YourStory.com*, 18-Aug-2015. .

[3] J. S. Deogun, V. V. Raghavan, A. Sarkar, and H. Sever, "Data Mining: Trends in Research and Development," in *Rough Sets and Data Mining*, Springer US, 1997, pp. 9–45.

[4] C. Hand, F. Dall'Olmo Riley, P. Harris, J. Singh, and R. Rettie, "Online grocery shopping: the influence of situational factors," *Eur. J. Mark.*, vol. 43, no. 9/10, pp. 1205–1219, Sep. 2009.

[5] J. M. C. da S. Penim, "Online grocery shopping: an exploratory study of consumer decision making processes," 2013.

[6] S. Spiggle, "Grocery Shopping Lists: What Do Consumers Write?," *ACR North Am. Adv.*, vol. NA-14, 1987.

[7] A. Thomas and R. Garland, "Grocery shopping: list and non-list usage," *Mark. Intell. Plan.*, vol. 22, no. 6, pp. 623–635, Oct. 2004.

[8] Reuters, "Going To The Supermarket? Why Making A Grocery List Is A Must," *Huffington Post*, 07-May-2015.

[9] "7 Ways to Make Grocery Shopping a Less Painful Experience." .

[10] R. Bassett, B. Beagan, and G. E. Chapman, "Grocery lists: connecting family, household and grocery store," *Br. Food J.*, vol. 110, no. 2, pp. 206–217, Feb. 2008.

[11] "FMI | Food Marketing Institute | U.S. Grocery Shopper Trends 2014." [Online]. Available: http://www.fmi.org/research-resources/u-s-grocery-shopper-trends/grocerytrends2014. [Accessed: 18-Feb-2017].

[12] A. Arnaud, A. Kollman, and A. Berndt, "The use of shopping lists by Generation Y consumers in grocery shopping," in *Nordic Retail and Wholesale Conference 2014*, 2014.

[13] F. Heinrichs, D. Schreiber, and J. Schöning, "The Hybrid Shopping List: Bridging the Gap Between Physical and Digital Shopping Lists," in *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, New York, NY, USA, 2011, pp. 251–254.

[14] M. Liwicki, S. Thieme, G. Kahl, and A. Dengel, “An intelligent shopping list-combining digital paper with product ontologies,” *Knowl.-Based Intell. Inf. Eng. Syst.*, pp. 187–194, 2011.

[15] J. Jain, R. Ghosh, and M. Dekhil, “Multimodal Shopping Lists,” in *Proceedings of the 13th International Conference on Human-Computer Interaction. Part II: Novel Interaction Methods and Techniques*, Berlin, Heidelberg, 2009, pp. 39–47.

[16] P. Nurmi, E. Lagerspetz, W. Buntine, P. Floréen, and J. Kukkonen, “Product Retrieval for Grocery Stores,” in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2008, pp. 781–782.

[17] H. Wu and Y. Natchetoi, “Mobile shopping assistant: integration of mobile applications and web services,” in *Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 1259–1260.

[18] “Consumer Behavior Shopping Habits – Marketing Teacher.” [Online]. Available: http://www.marketingteacher.com/consumer-behavior-shopping-habits/. [Accessed: 24-Feb-2016].

[19] D. S. Raicu, “A Data Mining Framework for Target Marketing.,” in *Computers and Their Applications*, 2003, pp. 476–479.

[20] A. K. Gupta and C. Gupta, “Analyzing Customer Behavior using Data Mining Techniques: Optimizing Relationships with Customer,” *Manag. Insight*, vol. 6, no. 1, Jan. 2012.

[21] A. Raorane and R. V. Kulkarni, “Data Mining Techniques: A Source for Consumer Behavior Analysis,” *ArXiv11091202 Cs*, Sep. 2011.

[22] J. Watada and K. Yamashiro, “A Data Mining Approach to Consumer Behavior,” in *First International Conference on Innovative Computing, Information and Control - Volume I (ICICIC’06)*, 2006, vol. 2, pp. 652–655.

[23] P. Turčínek and J. Turčínková, “Exploring Consumer Behavior: Use of Association Rules,” *Acta Univ. Agric. Silvic. Mendel. Brun.*, vol. 63, no. 3, pp. 1031–1042, Jun. 2015.

[24] B. Yıldız and B. Ergenç, “Comparison of two association rule mining algorithms without candidate generation,” 2010.

[25] K. Venkatachari, “MARKET BASKET ANALYSIS USING FP GROWTH AND APRIORI ALGORITHM: A CASE STUDY OF MUMBAI RETAIL STORE,” *BVIMSR’s J. Manag. Res.*, vol. 8, no. 1, p. 49, 2016.

[26] * All products require an annual contract Prices do not include sales tax, “Number of smartphone users worldwide 2014-2020,” *Statista*. [Online]. Available: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/. [Accessed: 21-Mar-2017].

[27] “IDC: Smartphone OS Market Share,” *www.idc.com*. [Online]. Available: https://www.idc.com/promo/smartphone-market-share/os. [Accessed: 21-Mar-2017].

[28] P. P.-24 M. 2016, “Android vs iOS market share in 2015,” *DeviceAtlas*, 24-Mar-2016. [Online]. Available: https://deviceatlas.com/blog/android-vs-ios-market-share-2015. [Accessed: 21-Mar-2017].

[29] “Dashboards | Android Developers.” [Online]. Available: https://developer.android.com/about/dashboards/index.html#Screens. [Accessed: 21-Mar-2017].