**CONSUMER SHOPPING APPS DECISION SUPPORT TOOL**

BY

ANG SINYEE

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology

(Perak Campus)

Oct 2015

**UNIVERSITI TUNKU ABDUL RAHMAN**

# REPORT STATUS DECLARATION FORM

**Title**: **CONSUMER SHOPPING APPS DECISION SUPPORT TOOL**

**Academic Session**: _____

I _____

**(CAPITAL LETTER)**

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

_____ _____

(Author's signature) (Supervisor's signature)

**Address**:

_____

_____ _____

_____ Supervisor's name

**Date**: _____ **Date**: _____

## DECLARATION OF ORIGINALITY

I declare that this report entitled "**CONSUMER SHOPPING APPS DECISION SUPPORT TOOL**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature :_____

Name : ANG SINYEE

Date :_____

## ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere thanks and appreciation to my supervisors, Mr. Ku Ching Soon who has given me this bright opportunity to engage in this project. I am enthusiastically appreciative to him as he guides me and supervision motivates me from the beginning of the project to the conclusion of the project. His guidance and wisdom have expands my knowledge and widen my experience in developing a system. He is always willing to spare his time for his students and me throughout the development process even though he is very busy. A million thanks to you.

Besides, I would like to thanks my friends who are willing to share their ideas and opinions with me. I won't forget the contribution from them as their opinions help me in improvise the project. Their supports means a lot to me during the FYP project development.

Last but not least. I would like to express my appreciation to my parents and sisters for their dedication and the many years of supports during my studies.

## ABSTRACT

Saving money and spending wisely is a concern for most of the consumers, especially when the GST has been implemented in Malaysia. This system will be useful when anyone feeling uncertain whether the product is worth-buying or not. Currently there are a few mobile applications that is used to compare prices in other country. However, none of it is fully utilized by every sellers and consumers in Malaysia due to dissatisfactory and inconvenience, hence it is difficult to fully integrate in our daily life. Throughout some review on other similar category application, most application having great idea but all module is stand alone, and isolated, it did not provide enough interactivity with user. Therefore, we aim to develop an application which are widely used by consumer in Malaysia with combination of multiple functions within single mobile applications which is Conawa application, this include all the idea of be a smart consumer. Through this, it will assist consumers to be aware of the information that will lead them be a smart consumers. This application can help consumer to aware on the information that will lead them be a smart consumer. The system will apply some algorithm on input data processing for both location data, human input data such as semantic search, and image data. Phased development approach will use to be employed in this project. This application is expected to utilize by people from low and middle tier social rank such as youngster, housewife and old folks. The final delivery system will provide the convenience way to let user have more involvement in the system. Besides that, this can ease users to get sufficient information regarding latest products, cheapest prices, freebies, vouchers, and location either in offline or online mode. They can share this information with the people around them as well. Through application, user will be able to tell that the petrol spent for wandering around is whether worth or just compromised the price with distance. Based on the wish list, the system is able to tell consumers the total cost (total product cost + total petrol cost) incurred for each local shop/ outlets around them and it can provide a route suggestion for consumers.

## TABLE OF CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF CHARTS

## LIST OF CODE SEGMENTS

## LIST OF ABBREVIATIONS

ADB Android Debug Bridge

ADT Android Development Tools

API Application Programming Interface

CONAWA Consumer Awareness Application

ERD Entity relationship diagram

FYP Final Year Project

GPS Global Positioning System

GUI Graphical User Interface

IDE Integrated Development Environment

JSON JavaScript Object Notation

KM Kilometer(s)

OCR Optical character recognition

OS Operating System

PC Personal Computer

PNG Portable Network Graphics

SDK Software Development Kit

SDLC System Development Life Cycle

UI User Interface

URL Uniform Resource Locator

UTAR Universiti Tunku Abdul Rahman

XML Extensible Markup Language

# CHAPTER 1 INTRODUCTION

## 1.1 Project Background

Computer and application/ software were created in the early 80's and 90's respectively. Both of them work together to provide a graphical user interface to ease the users in their daily operations. That was the time when the World Wide Web (www) started to emerge. The applications started to evolve from desktop into mobile phones in the last 20's and work together with the Internet.

Mobile devices have transformed the way we live and assisted human with their daily work since the introduction of the IBM Simon in 1992 such as food delivery system, alarm, virtual wallet and presentation. The evolution of smartphone started of as a touch-based computer that was embedded with an operating system, internet access, third party applications and many more. Mobile applications are meant to be portable and convenient due to their small size and these have made the mobile application design more challenging in order to deliver similar information and hence optimizing the performance with desktop applications.

Nowadays, smartphone has dominated the whole IT gadgets market. This shows that mobile applications play an important role and are highly demanded by users. Most mobile applications are published on more than one target in order to generate more revenue:

i. Apple iOS apps published on Apps Store
ii. Android apps published under google play store and amazon
iii. Xiaomi apps published under mi market
iv. Windows Phone apps published on Windows store

According to App Annie Index Market Q3 2014 report (2014), Google Play's downloads outnumbered iOS app store by up to 60% in the past several quarters with different kind of apps like social media apps, video and audio apps, business apps and etc. iOS, Android and Windows currently have a variety of available APIs which can be used by developers for free. There are also a number of available functions that enable the development of this system.

Many businesses are aware of the benefit that can get from mobile apps. Whether it is with ads or in-app purchases, this can create a stream of income. The most obvious benefit that the business units can gain is resource-saving either in term of time, cost, or marketing their products.

In Malaysia the prices of almost everything went up tremendously since the beginning of this year. The inflation effect undoubtedly increased the financial burden of middle and lower income families. Therefore, it is vital that they save money for bad times. They have to spend wisely by comparing prices and quality when buying from any outlet especially with the implementation of GST (Government Service Tax) in Malaysia. However, this may not be a considering factor for those higher income groups.

Nowadays, everyone is learning to be a smart consumer. Basically, retail shops have their own supporters or revenue from local consumers. In order to be a smart consumer, to know whether what have be spent are of worth and of the best deal, one has to visit retailer outlets or online shops to compare prices which could be very troublesome and time consuming. In order to attract more customers, all the retail shops compete with one another. Eventually they won't know the standard price for certain product if it is sold with far lower price in other parts or state.

When we think of handy and portable devices, smartphone is the first thing that pops out. Currently, a few mobile applications came out with products price comparison applications in the market, but none of them is fully utilized by most sellers and consumers due to the terrible user experience, standalone isolation module and not user friendly business process design. By utilizing power and features of mobile apps (e.g.: push notification), we aim to develop an application which can be widely used by consumers. It can assist those seller and consumers in Malaysia with the combination of multiple functions within single mobile applications which is Conawa application, this include all the idea of be a smart consumer. The idea of application name 'Conawa' came from '**con**sumer **awa**reness'.

The ultimate objective of this application is good things come in a tiny package. It can bring about convenience to both sellers and consumers with high usability and durability

than common apps in the market which stay supported for three years. This application can assist consumers to be smart consumers. For instance, it provides the convenience to let user get involved in the system. Besides, this can also enable users to get sufficient information effortlessly regarding the latest products, cheapest prices, freebies, vouchers, and location information. In addition, they can share this information with the people around them as well. The system will be able to help consumer to make better decisions and evaluate whether a specific product is worth buying.

Apart from that, the system can provide a route suggestion for consumers. Based on the wish list, the system is able to tell consumers the total cost (total product cost + total petrol cost) incurred for each local shop/ outlets around them.

The semantic search will be integrated in this system for product search query. There are five aspects of semantic search matching which is:

– Form: NY -> NY
– Phrase: “hot dog” -> “hot dog”
– Sense: utube -> youtube
– Topic: Microsoft Office -> Microsoft, PowerPoint, Word, Excel…
– Structure: how far is earth from sun **means** distance between earth and sun
– Chinese / English Translation

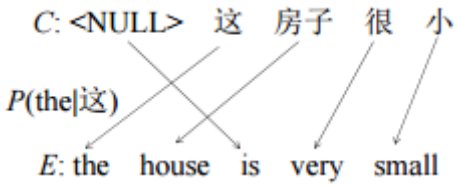



Figure 1-1 Chinese to English Translation

All the aspects mentioned above should be catered in the system.

.

## 1.2 Problem Statement and Motivation

### (I) Energy and time consuming

In Malaysia, we all suffer from price inflation every month. Consumer will only be able to discover the actual price and events such as freebies giveaway, limited time promotion product by going to the retailer personally and survey the price one by one. This is a tedious work especially during special celebrations. When festive season is around the corner, we will go shopping. The things to buy will appear to be in larger quantities as compared to non-festive seasons. Consumer will notice the price difference when they purchase in large quantities compared to small amounts. Thus, most consumers will suffer from wandering around few different stores to compare prices. This can be very tiring and time-consuming task as they spend a lot of time to gain those beneficial information in short time. Sometimes, we will need to know the price per unit in order to know whether the product is worth buying or not. In addition, the petrol spent for this will instead be a liability to us, consumers.

### (II) Lack of Proper Platform

The platforms for both consumers and seller / retail shop / café business to advertise or gain consumer product or food and beverage. Even though we have platforms like social media (e.g.: Facebook) but the information is limited due to no proper administration and organization. It spreads around so we won't be able to get accurate and latest information as is not often up-to-date. There are existing consumer related system or application for shopping purposes but most of them just focus on a few specific price sources that have cooperation with the system or the application normally owned by the retailer shop themselves. It is hassle for user to refer to hard copy brochures and sometimes brochures aren't send to houses.

### (III) Inefficient in noting down wish list

User always can't efficiently note down wish list in most android application such as S memo. Very often, user will note down the things they wish to buy, this will be easier for them to recall what they lacking. However, most of the traditional memo or notes applications are just like white papers, you can write anything you

like on it, but it is unorganized way. Besides that, they may need to write down full product name if the wish list will be sent to third person otherwise he or she may not understand what you write if you just write in short form . It will be better if there is a suggestion list for them to choose the product name or just like what they did in real life, pull the product into shopping cart.

**(IV) Isolation Of module**

Most of the existing system are having these problem. The navigation and business process design is not user friendly as it is hard to learn, hard to use, hard to remember the operation if it is not used for some time, the system does not increase the efficiency of performing task even after using the system. If you wish that the user won't uninstall right after download the apps, the user experience is the prior consideration. The application can be an unattractive feature but if the user experience (UX) design is bad, the application would not be welcomed by user. The existing system makes all the modules look like a standalone system, no interaction at all, It looks like the separated system are purposely brought together. As example, after feature A , supposing feature B will be used following by feature A, however most existing system will prompt user to go back to main menu to choose feature B option manually instead of making feature B as same business process path with feature A. The system flow design should be designed in a way that the related module should put together, make them have interaction and correct branching.

**(V) Input information and limited information**

Many existing system hardcode the product name and product type, thus very limited product or product assortment (quantity type, quality type, size) is available in application database. Users are not allowed to request for new item if the item is not in existing system. Besides, the input is all character-based, users are required to enter every single words (product name, product location etc.) manually to the textbox, and no suggestion list is provide. In addition, the current search engine in system is searching by keywords. Currently, most app are using keywords search which is very troublesome and not user friendly. The Semantic search and barcode scanner features should be provides, the search results is

produce by evaluating the search phrase (Chinese or English) and finding the most relevant results in database.

**(VI) Uncertainty on store choice**

The consumers are allowed to add products into to-buy wish list whenever the product they have used up. However, consumers always hesitate whether which supermarket can let them spent the least by buying the same amount of product. For normal consumer, when they see local shop A having cheaper price for product A, whereas local shop B having cheaper price for product B and so on. In this scenario, they will either buy everything together in shop A if the shop nearest to their current location or buy the product one by one in each different shop. The first alternative, the total petrol spent probably is lesser however the total price to pay for all to buy product will be higher than the second alternative which requires travelling to several different places to gather or buy all the product they want. In fact, both of the solutions above are inefficient. They may either spent too high in total product price or spent too much on petrol.

## 1.3 Project Objective

There are several objective in this project, consumer awareness system:

**(I) Able Provide Ideal Store Suggestion based on user location**

Through Fussy calculator, user can make better decisions on which item to purchase based on the unit price and price comparison calculated by the system. This is very common for many users who are wandering around the store and not sure which item is worth to buy. From this system, user will be able to tell that the petrol spent for wandering around is whether worth or just compromised the price with distance.

**(II) To get Product Unit Price so can assist user in decision making**

The semantic search is searching based on meaning. It allow user to search for product name that having similar meaning or similar type. The search results will be produced by evaluating the search phrase and finding the most relevant results in database. By this approach, user do not need to type the exact keywords in order to get information. Suggestion list is provided together with meaning search to

increase the user experience.

**(III) To ease users to get accurate and timely product information**

By using this system, user will be able to get accurate and timely information instead of outdated information regarding latest products, best deals and location and navigation path to the specific retailer outlets nearby. If the user wish to purchase the item in certain store/location, but does not know the direction to the store, the user can use this feature to navigate themselves to the store. Furthermore, user can view the brochures just by single touch.

**(IV) Able create favorite list and cart list**

The wish list can be created by manually enter the product name, follow by a semantic search suggestion list will automatically list out to allow user to choose the desired product name or press add to wish list button when inside product detail page. Later, the system will provide the most suitable store suggestion based on location distance and price of product.

**(V) Nearby store and location navigation**

User can search for items available in nearby store based on the maximum distance willing to travel by user. User can capture the item name as input, the current location of the user, and the maximum distance user willing to travel. Besides that, the system can provide a convenient way for users to get the shortest path to reach the store. From here, the user can use this feature to navigate themselves to the store. This will be very useful for those users who are new to the environment, e.g.: worker/ staff outstation supporting, family member come to visit their child or relatives, travel to certain location , those user who saw any store promotion information, time sales or other people introduce them the new store . They wish to purchase the item in certain store/location, but does not know the direction to the store or those user want to know which path is shorter in order to make sure the path they drive is shortest and the petrol spent will be as little as possible.

**(VI) Sent product detail or user custom favorite list through social media or SMS**

Sent out particular product detail to the contact number through contact picker

or social media. This can let other friend or family member know if there is product promotion. Furthermore, user can sent the favorite list to user's family and request third person help them in buying grocery if user is busy at particular time.

## 1.4 Project Scope

This project aims to develop a mobile application in android platform from front end to back end that can be served as an information sharing platform mainly for both online and offline consumers and seller. The web service and database should deploy on cloud instead of personal PC. The application will apply some algorithms for semantic search engine, translation for search engine (meaning search) and barcode scanner which work with barcode database will be required in this project too. The interface should be designed in a professional and user friendly-like feel to attract users. If time permits, this application will be developed to be supported by IOS and Windows platform as well. The project will mainly cover the consumer module but not the store representative module. The store representative module will be put in future enhancement, thus all the product information will be pump into database manually instead of through product upload portal by store representative.

The scope for this project as below:

**(I) Store's catalogue**

Consumer can view most of the hypermarket catalogue main page in single interface. After pressing the catalogue picture consumer can view all pages in the certain catalogue by swiping. They can access it in a single application. The consumer can download the catalogue for offline viewing purpose.

**(II) Share new Item**

User can share the item which does not exist in specific store/ outlet. The details include product description, image, price, and promotion end date if applicable. Shared item can be search by the users.

**(III) Share Price**

User can share the item price when they see the item on the store if the specific item record already exist.

**(IV) Search Item by semantic search algorithm and Intelligent Search Item based on nearby location**

Users can search for item available in nearby store. Conawa will allow users to capture the item name as input, the current location of the user, and the maximum distance user willing to travel. Based on these information, Conawa will list down all the matching items.

**(V) GPS navigation to desire store and single map that show all available store around user location**

If the users wish to purchase the item in certain store/location, but does not know the direction to the store, the users can use this feature to navigate themselves to the store.

**(VI) Offline shopping list**

Conawa provides the convenience for consumers to record down the items that they wish to purchase. User can "check/tick" on the items that have been purchased and keep track on the remaining items. Users can estimate how much they will be spending.

**(VII) Unit Calculator**

It is common to see product that has different packaging with different price. For example, a 12 pieces KitKat chocolate cost RM9 while 20 pieces cost RM14. Which packaging is worth buying? Punch in the item price and quantity to see the result instantly instead of hesitating on which to buy.

**(VIII) Ideal Store Suggestion**

Conawa application also will assist users in store choice decision making with ideal store suggestions with best value prices store. The offline items wish list will be used as inputs to this module. The module can suggest the top 5 store

that user can buy the product. The store will list down and rank by least spending until most spending required.

The application database is for data storing and centralization. At the end of project I, an android based prototype of the consumer awareness system is expected to be delivered out which able to perform the basic authorization and some of the module as mentioned above, whereas all the module mentioned above in Project II. The Android devices may require connection to server thus 3 tier architecture will be applied.

The application front end user interface will be coded by using java, xml with Photoshop together and SQLite as local database, whereas JSP+ servlet will be use to code the web service and SQL Server will serve as remote database. The system allow to install on android device only.

### 1.5 Impact, Significance and Contribution

This application can assist consumers to be aware of the information that will lead them be a smart consumers. For instance, it provides the convenient way to let users have more involvement in the system. On the other hand, this will be a tiring and time-consuming task as they spend a lot of time to gain those beneficial information in short time. The proposed features can assist consumers to get timely information instead of outdated information regarding latest products, best deals and location and navigation path to the specific retailer outlets nearby.

Normally consumers will like to know the price per unit in order to know the product is worth buying or not. Thus, this application can assist users to make better decision on which item to purchase based on the unit price and price comparison calculated by the system. This is very common for many users to wander around the store and hesitate whether which item is worth to buy. From this system, user will be able to tell that the petrol spent for wondering around is whether worth or just compromised the price with distance. This will be an effective efficient way as it will suggest the best store choice to consumers so that the consumers can spent the least by buying same amount of product in terms on time, distance, product price and petrol spending.

Very often, users will like to note down the things they wish to buy, this will be easier for them to recall what they are lacking. This application solves the problem in traditional memo or notes application in more organized way by auto barcode scanning, suggestion list and user can even navigate to the location that sell the product directly through a single touch. This definitely will increase consumer shopping experience and perhaps this will be an enjoyable experience after using the system. Moreover, user can immediately scan the barcode of product and add into to-buy-wish list whenever the product they use has been used up.

In order to further enhance the user experience, suggestion list is provided together with meaning search on user input queries, so user do not need to type the exact keywords. The nearby search features will be integrate in the system to filter out all irrelevant result, user can adjust the result that display limited to nearby store only.

In short, with the final deliverables of this system, the system will be able to provide useful information for consumer and shorten their time in obtaining those product information and assist them on product choice decision making. This will be able to bring convenience and assist consumers in decision making. At the same time, the retailers will be able to attract more customers. For the sake of minimize the spending to get the maximum benefit, consumers can utilize the application in their daily life for purchasing product or get any ideal suggestion from the system.

## 1.6 Chapter Summary

This report will be divide into five (5) parts and each part will discuss about different things that can describe the system from different aspects. The report basically starts with an abstract which acts as formal summary of the system and report. It briefly describes the main points of the application, problem statement, methodology, approach used, the system obtain and also the conclusion. By reading through abstract, readers can get the rough idea about the system, so they can decide whether this is interesting enough for them continue to proceed into the content of report.

It will be divided into 5 parts which show in table below:

Table 1-1 Chapters Summary

| Chapter | Title |
|---|---|
| 1 | Introduction |
| 2 | Literature Review |
| 3 | System Methodology |
| 4 | Prototype Design |
| 5 | Conclusion |

In Chapter 1, the introduction states the problem statements, background, motivations and objectives of the project in detail. With such information, readers can understand the problem encountered previously, the motivation of this system development and the objective such development with respect to the background of the project.

In Chapter 2, the literature review, is divided into four (4) part reviews. Comparison of related works are done in order to show the research and study that have been done with relation to this topic. It involves facts finding, online research from journal paper, product, marketing report, and review how other people's system work. This gives a clear idea of the topic and thorough understanding from previous developers or existing system. This also portrays the understanding on the common system development that we have gained and also the necessary concepts that should be added into the application.

In Chapter 3, it discusses about the system analysis. This chapter mainly explains the software methodology that will be used in the system development, project timeline and how the system works in high level formal UML format such as use cases, activity diagram, class diagram and object diagram. It gives the reader a clearer picture on how the user interacts with the system, the behavior of system and some technique that will be applied in this system. Besides that, software and hardware requirement, system requirement and architectural design are also included in this chapter.

In Chapter 4, the prototype design is shown at here. This includes the user interface design, icon design and logo design. Besides that, the business flow design, ERD design are shown in this chapter too.

Chapter 5 is the conclusion of the overall system report. Furthermore, the findings from this research and in-depth discussion will be summarize in this final chapter too.

# CHAPTER 2 LITERATURE REVIEWS

## 2.1 Chapter Overview

In order to solve problem statement, the literature review is separated into four (4) parts, first part is about the mobile application based review and second part is the semantic text search based review. The second part of literature review is arranged by the year of the journal published to show the evolution of semantic search. This chapter will also compare the strengths and weaknesses among existing systems. The third part of review is to understand consumer psychology on store choices and the fourth part will review the solid techniques in place that can create a profitable app.

## 2.2 Consumer Applications

Other Consumer Application that download of playstore are analyze in order to understand their strength and weakness.

### 2.2.1 Android Mobile App : RedLaser

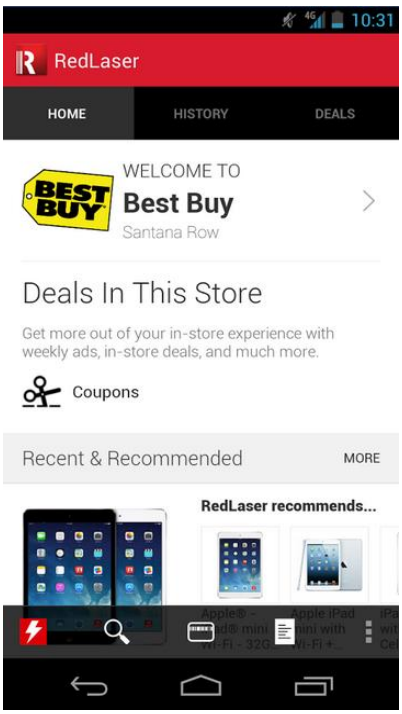



Figure 2-1 Screenshot of RedLaser

**Introduction**

RedLaser is a mobile application developed by ebay. RedLaser's concept is simple. RedLaser allow you to scan a barcode and it will gives you several places online to find that item, perhaps at a lower price. The application has simple yet great user interface that emphasize on monotone which is red, white and black color.

**Pros**

It allows user to scan the barcode by UPC, EAN and UPC-E barcode, these 3 barcode standard is commonly use nowadays. It find the product using online google search engine and pull out the information from google and amazon. The price automatically detect your location and show you the appropriate currency.

The barcode string will be search from google and amazon to provide other competing online shops. The results pages are well organized It is helpful to survey the online store price was the best deal or not. The result show price as well as product details, reviews, and suggested complementary products. In addition, the History section will keep track of all of their recent searches, making it easy for user to recall or re-open information about products of interest. Products you scan will be add to your list of history items for future viewing. The application allow you to email any of your scanned list to yourself or your friend. When a grocery item barcode is scanned, you can expand the detail tab to view nutrition and allergen information using the built in browser such as Facebook application (open website without exit the application and move to another application).

**Cons**

On the bad side, the application unable to scan prices quickly if at all and need to enter manually. Besides that, it will direct you to EBAY mostly (99%). Very often, the Scanner not able to scan the barcode if the product material is with round or shiny surface. It rarely display the lists of competitors who have similar item and their prices but on rare occasion it always pulls the information up which the price is not correct. The application not showing local store, only online information are showing.

### 2.2.2 Android Mobile App : Smoopa Shopping

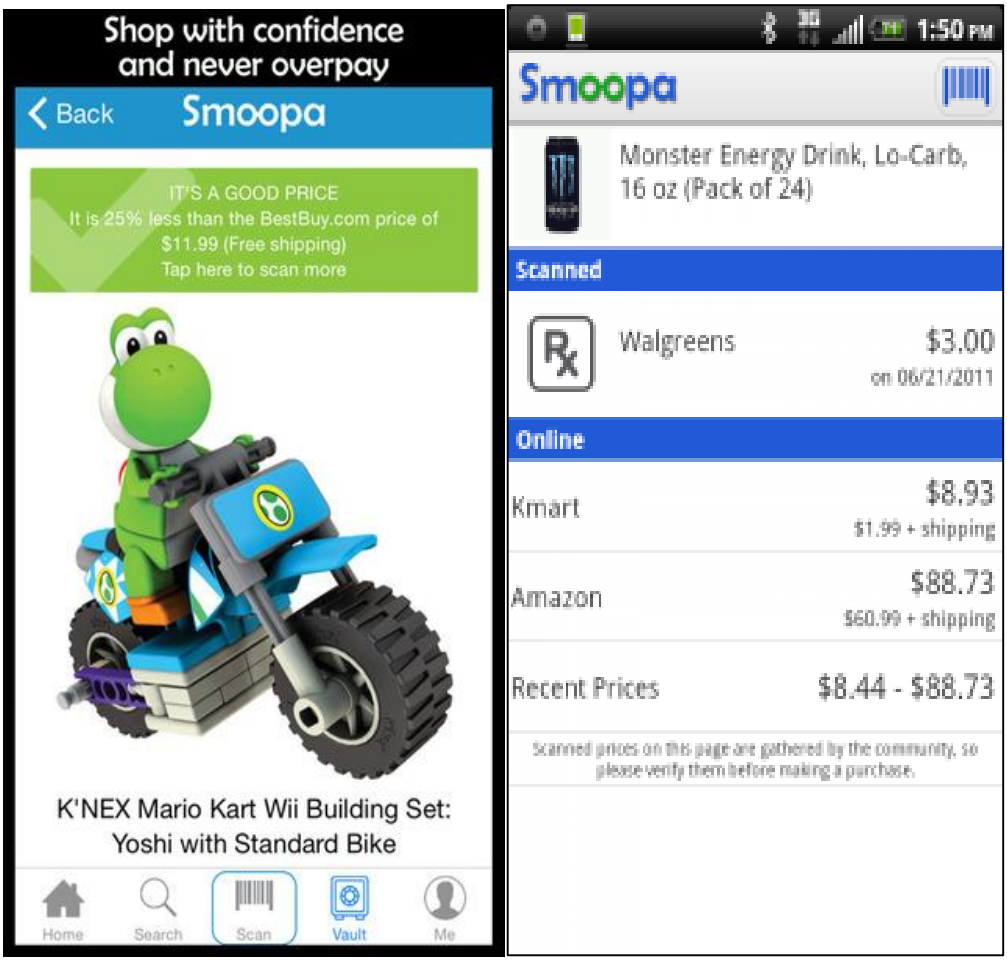



Figure 2-2 Screenshot of Smoopa

**Introduction**

This application launch at 2011 by an ex-googler. The application provide a clean and easy learn and use interface. This app that lets you compare prices from 17 major retailers of US, customize own price alerts notification, and create wishlists of items sold in this 17 major retailers.

**Pros**

The apps allow you to set the alert notification, so whenever if an item falls into your desired price range, it will send push notification when you device in online mode. The price are allow to send to social media such as twitter, facebook and whatsapp, and they can see where to get the best deal on those items.

Like redlaser, this app share similar concept with the first app that had review, redlaser. It use the device camera to scan the product barcodes and it will pull up price comparisons, however this utilized power of social network, it does allow user to update the price. This application does not include amazon prices due to some legal issue.

Every time consumer scans an item and enter price data from physical store, there's a random chance they will be rewarded with credits of around 50 cents in credit. These credit can later use to redeem as gift cards, rebate voucher and do charitable donations. If consumer find a better price online through the app, they can buy the item directly through app and waiting for cash on delivery service at home. The app will redirect the request list to the retailer. Smoopa earns affiliate revenue when consumers introduce other user sign up and share reasonable price or buy items through the app. They can also earn revenue through targeted advertising.

**Cons**

The major bad side of this application is the application no option for consumer to search by product name or did not provide any suggestion list, only barcode input are allow. It is inconvenient for those product with no barcode or the barcode had been removed. In addition, this application only useful for buying electronics since no grocery shopping were include. The product are limited to the prices of products that are in their database, no new product are allow to add or request.

### 2.2.3 Android Mobile App : ScanLife

**Introduction**

According to google play, this apps founded at 2012 and last enhancement was August 17, 2015. The founder of scanlife is Scanbuy, Inc which is specialized in mobile Website builder, surveys, mobile Web-pages, specialized branding, designer QR Codes, and others. This application allow consumer to scan product only for UPC barcode, and search result from stores like Amazon, Toys R Us, Macy's, Best Buy, Walmart, Target, and purchase right from your phone (depending on location).

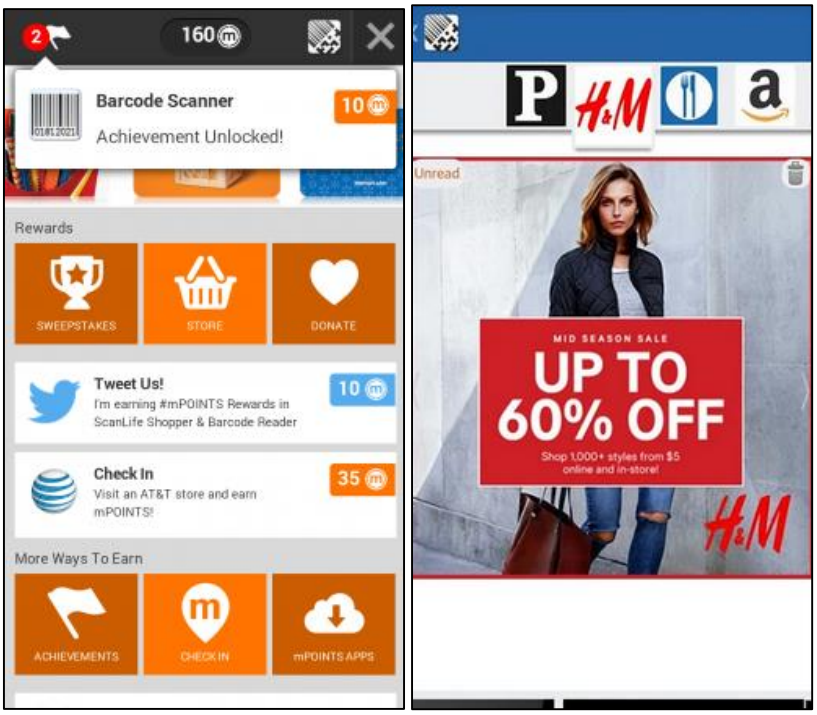



Figure 2-3 Screenshot of ScanLife

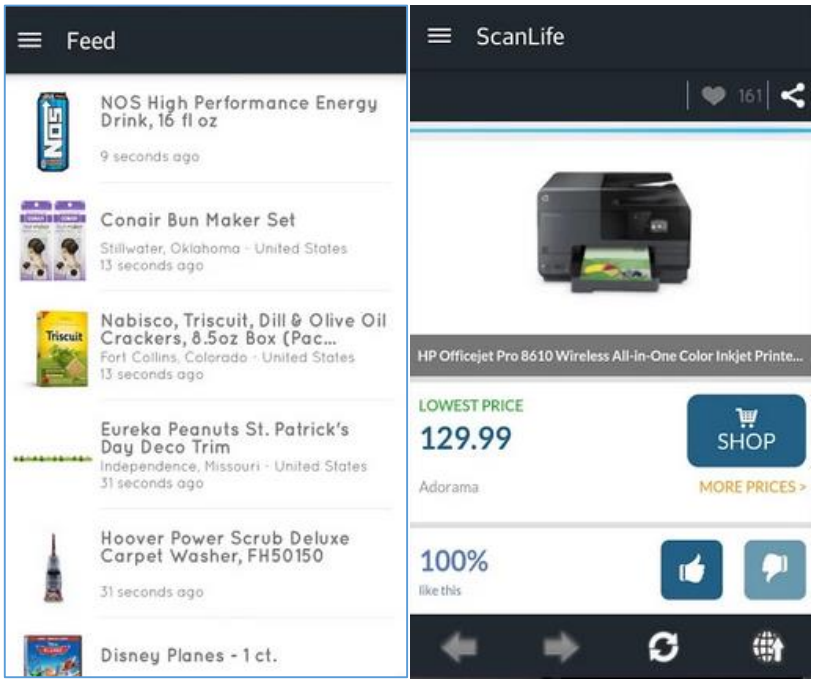



Figure 2-4 Screenshot of ScanLife Product Detail

**Pros**

Besides that, consumer can check the product reviews and see what other consumer's comment and feedback after navigate to the product detail page. The app can works as traditional QR code scanner and QR code creator. It is popular due to affiliated with credit points, the consumer can earn credit point rewards as scanning QR codes and barcodes and the credit earned can use for real gift cards redemption later on (US only).

All scanning history will be saved automatically by default or the user can choose to save into favorites list. Moreover, you can purchase right from your phone by cash on delivery. The apps provides Syncing with Facebook to save the previous scanning history. After review, the impression this app gives me is it has an extremely fast barcode scanner with good image processing optimization. The good things about this app is there are plenty of information about scanned products such as pricing, reviews, online stores, etc. Consumer can save plenty of money in stored, the product information will retrieved and online price comparison can be made in the store just by scanning the products barcode. From here, the consumer can know the online and local product price

Furthermore, sometime certain instant coupons will be offer in the apps and the system will display other relevant suggest product for any product you view previously. Eventually you can order your desire product online from competitors by the scanning features.

**Cons**

Disadvantage of this app is there most of the Items either were not found or QR codes no longer support and it only support UPC barcode. In addition, there were plenty product that failed to display competitive shop and relevant coupon. Besides that, the apps just include limited store, for example Amazon, Toys R Us, Macy's, Best Buy, Walmart, Target.

### 2.2.4 Comparison of Application

Table 2-1 Strength and Weakness between Existing Systems

| Features / Application | **Strength** | **Weakness** |
|---|---|---|
| **RedLaser** | 1. Email scanned list to mailbox.<br>2. Information and price google and amazon<br>3. Auto detect location and show appropriate currency.<br>4. Show product details, reviews, viewing nutrition and suggested products. | 1. Pull out EBAY price mostly.<br>2. Scanner struggles with round or shiny items.<br>3. Only online information.<br>4. Limited product in database, cannot request new item.<br>5. No using GPS location based |
| **Smoopa** | 1. Customize alert notification based on desired price range.<br>2. Price are allow sent to social media ( eg: facebook ).<br>3. Allow user to share and update price.<br>4. Chance to reward with credits (US only).<br>5. Buy the item directly through app. | 1. Cannot search by product name, only barcode are allow<br>2. Electronic gadgets only.<br>3. Limited product in their database, no way request new item.<br>4. QR code are not allow.<br>5. No using GPS location based |
| **ScanLife** | 1. Barcode code scanner and QR code<br>2. Affiliated with credit points, US only).<br>3. Scanning history can choose put in wishlist<br>4. Some Instant coupons will be offer. | 1. Most of the Items were not found.<br>2. Not having timely information, wrong prices.<br>3. User cannot have involvement to the system such as sharing.<br>4. QR codes no longer works |

Table 2-2 Comparison of Existing Application with Proposed System

| Application / Features | **RedLaser** | **Smoopa** | **ScanLife** | **Conawa (Proposed)** |
|---|---|---|---|---|
| Auto detect location | YES | N/A | N/A | YES |
| Online store or Local store | Online only | Online & Local Electronic Gadgets | Online and Local | Local store |
| Sent to social network | N/A | YES | N/A | YES |
| Share Price | YES | YES | N/A | YES |
| Share Item | N/A | N/A | N/A | YES |
| Search nearby store (GPS) | N/A | N/A | N/A | YES |
| Semantic Search | N/A | N/A | N/A | YES |
| Search By Text Input (eg: product name) | YES | N/A | YES | YES |
| Barcode scan search are allow | YES | YES | YES | YES |
| Request new item | N/A | N/A | N/A | YES |
| Fussy Calculator | N/A | N/A | N/A | YES |
| Suggestion Store | N/A | N/A | N/A | YES |
| WishList | N/A | N/A | YES | YES |
| Email scanned list | YES | N/A | N/A | N/A |
| Price alert notification | N/A | YES | N/A | N/A |
| View product nutrition detail information | YES | N/A | N/A | N/A |
| Reward with credits | N/A | YES | YES | N/A |
| Offer Coupons | N/A | N/A | YES | N/A |
| Buy product directly from apps | N/A | YES | N/A | N/A |

### 2.3 Semantic Search Technology

The challenge of develop a semantic search engines is how to create a query that can understand by computer. Before process the queries, the queries must be convert into the format that the computer can understand their semantic meaning. For human, we express them in natural language sentence, we can understand the semantic meaning easily. However for system, it is pretty hard to process the natural language due to the complexity and ambiguous (Bäurle, F, July 2011). There are three paper review here and the review summary below is arrange according to their date created and published.

| | | |
|---|---|---|
| yutube | yuotube | yuo tube |
| ytube | youtubr | yu tube |
| youtubo | youtuber | youtubecom |
| youtube om | youtube music videos | youtube videos |
| youtube | youtube com | youtube co |
| youtub com | you tube music videos | yout tube |
| youtub | you tube com yourtube | your tube |
| you tube | you tub | you tube video clips |
| you tube videos | www you tube com | wwww youtube com |
| www youtube | www youtube com | www youtube co |
| yotube | www you tube | www utube com |
| ww youtube com | www utube | www u tube |
| utube videos | utube com | utube |
| u tube com | utub | u tube videos |
| u tube | my tube | toutube |
| outube | our tube | toutube |

Figure 2-5 List of Mismatch Text Search

| | |
|---|---|
| “how far” earth sun | average distance from the earth to the sun |
| “how far” sun | how far away is the sun from earth |
| average distance earth sun | average distance from earth to sun |
| how far from earth to sun | distance from earth to the sun |
| distance from sun to earth | distance between earth and the sun |
| distance between earth & sun | distance between earth and sun |
| how far earth is from the sun | distance from the earth to the sun |
| distance between earth sun | distance from the sun to the earth |
| distance of earth from sun | distance from the sun to earth |
| “how far” sun earth | how far away is the sun from the earth |
| how far earth from sun | distance between sun and earth |
| how far from earth is the sun | how far from the earth to the sun |
| distance from sun to the earth | |

Figure 2-6 List Of Mismatch Sentence Search

### 2.3.1 Broccoli: Semantic Full-Text Search at your Fingertips

This paper *Broccoli: Semantic Full-Text Search at your Fingertips.*(Hannah, B , 2012 ) discuss about the fast and easy use search engine which is doing the semantic full-text search. This semantic full text search combine the traditional full text search and Chinese/ English translation in a knowledge based databases. Find contextual co-occurrences of the words from the full-text part of the query with entities matching the ontology part of the query. This is handle by decomposing each sentence into contexts, meaning that the parts of the sentence that "belong" together. The search operates on four kinds of objects: ordinary words (e.g., edible), classes (e.g., plants), instances (e.g., Broccoli), and relations (e.g., occurs-with or native-to). Queries are trees, where nodes are arbitrary bags of these objects, and arcs are relations.

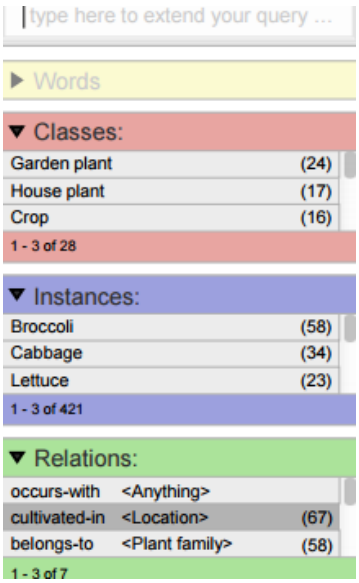



Figure 2-7 Screenshot of Broccoli Website

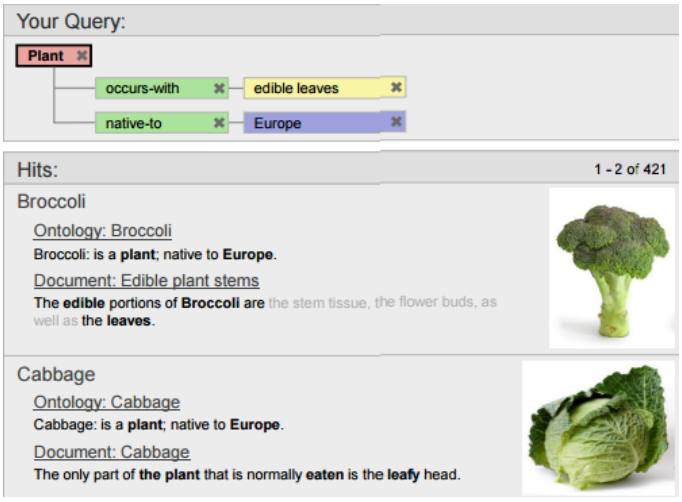



Figure 2-8 Screenshot of Broccoli Website Query Tree & Hits Result

The box on the top left with pink color indicate the current query as a tree. There is always one node as the root of the tree. The large box below shows the hits grouped by instance (of the class from the root node) and ranked by relevance (if Broccoli is among the hits, rank it first).Below show the ontology and the full text is provided. For the latter, a whole sentence is shown, with parts outside of the matching context greyed out. The hits partition show the success matching for the root node parent class and indent query tree (query class belongs to the plant in this example) and grouped by possible classis.

To solve the traditional keywords search query, and ranked the query result y some according to most relevant result to the query. For example, consider the query plants with edible leaves and native to Europe. This is pretty difficult if applying keywords search since there are no similar wordings in there.

The impressive part of this paper was the new index, it use inverted lists for prefixes instead of words, this increase the performance for query suggestions. This inverted list is based on context lists and it enables fast query suggestions. The context list for a prex contains one index item per occurrence of a word starting with that prex.

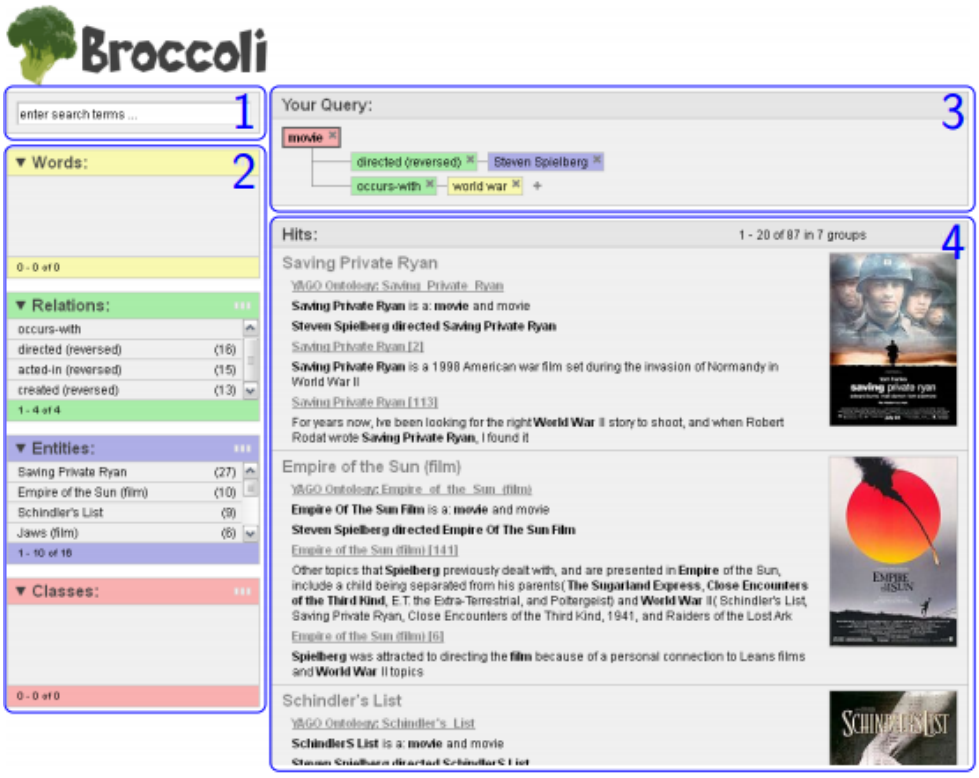



Figure 2-9 Broccoli Search String: Movie

When the user enters search query into search box, the system will sends query to back end and retrieve the relevant proposed words without refreshing the page. The automation of user interface not only restricted to display of proposal but also include right partition hits. The suggestion classes at the boxes below the search input can be clicked and the words will be added to the query. The query tree will show at the top right as the query gets refined.

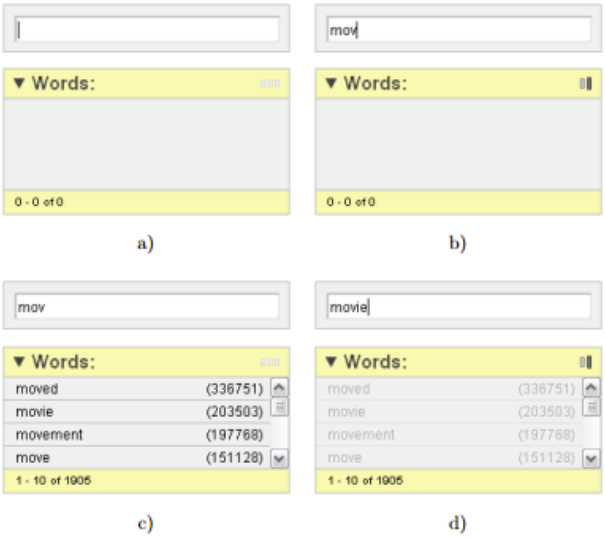



Figure 2-10 Step Of Handling user input

Above depicted the step of how system handle user input:

A) Initial state of the input field.

B) After something is type in the input field, the interface will refresh automatically and load the related similar words. At the same time, the loading icon will start loading activity while processing query.

C) Related similar words are successfully retrieved and display on the screen

D) If wish to search for other new words, just type another new words into the input field, the system will automatically start process the query and updates the results without refresh the page or open another new page.

### 2.3.2 ESTER: Efficient Search on Text, Entities, and Relations

According to founder of ESTER Holger, B, (*Holger, B, 2007)*, ESTER combines full text and ontology search in the engine. The search engine using only 4 basic operations which is prefix, joins,entity recognition, and ontological knowledge. It has a very fast query processing speed.

ESTER applied the English Wikipedia combined with YAGO ontology. The search text box automatically reacts to user input without press a search button. After it get the input, it will detect sematic classes which the input belongs to.

The last prefix of query will be check whether match any class name that exist in system after every keystroke. If more than one matches class, it display a list of suggestion that propose for all matching classes. By clicking any of the class in suggestion list, the system will detect the prefix in the query and replace it with relevant class name. The results of the current query will display at the right hand side of ESTER system. The results will be automatically update when the query string is changed. The system can interpret the keywords that enter by user whether is semantic classes or simple word occurrences.

The left hand partition display the classes or entities results that exist in their database for the last keyword that was added to the query string. Thus, this makes the result looks unorganized because if the users want to propose for other parts of query, the user need to manually change the sequence of the keywords in the query string.


Figure 2-11 Screenshot of Ester Wikipedia

The above example, query “beatles musician” searching the English Wikipedia in above example. The list of completions and hits is updated automatically without search button. The number in parentheses after each completion is the number of hits that would be obtained for that particular completion.

The quality of the search results provided by ESTER, or any other semantic search engine of a similar kind, depends on three main factors:

(1) The quality of the ontology

This is the successor of YAGO. YAGO had proved that by extrapolation empirical studies from human assessment on a sample, that 95% of YAGO's facts are correct

(2) The quality of the entity recognizer

This experiment take 10% of the words or phrases which the corresponding entities are known (because they link to some Wikipedia page), and the percentage of frequency entities recognized correctly was measured. This paper compare their implementation (ESTER) against two simple baselines: the naive scheme that assigns every word to the most common sense that has been encountered for that word in the training set (TOP), and the scheme that assigns every word to a random sense (RANDOM).

| Scheme | all words | 2 senses | 3 senses | $\geq$ 4 senses |
|---|---|---|---|---|
| ESTER | 93.4% | 88.2% | 84.4% | 80.3% |
| TOP | 91.9% | 83.5% | 77.2% | 77.6% |
| RANDOM | 71.5% | 50.2% | 33.4% | 14.0% |

Figure 2-12 Percentage of Accuracy by 3 scheme

For the Wikipedia collection combined with the YAGO ontology, ESTER can process a complex queries in a fraction of a second, with an index size of only about 4 GB.

### 2.3.3 SUSI: Wikipedia Search Using Semantic Index Annotations

SUSI a successor of Ester claim by (Buchhold, B., 2010) in *SUSI: Wikipedia Search Using Semantic Index Annotations*. This approach also using the same combination, basically it combined full-text and ontology search, and. Susi is similar with Ester proposed system but it eliminate some unnecessary operation and optimized the required operations to prefix search only, so it had better performance than Ester. SUSI restricting the relations to the classes' taxonomy and ontology entities. Besides that, instead of treating the whole Wikipedia article as a single document, SUSI improve the precision by treats each single sentence in any article as a separate document

Susi having a user interface which is pretty similar with Ester. Both Ester and Susi not guarantee accuracy on those complex queries that need to combine the entities relations and full-text.

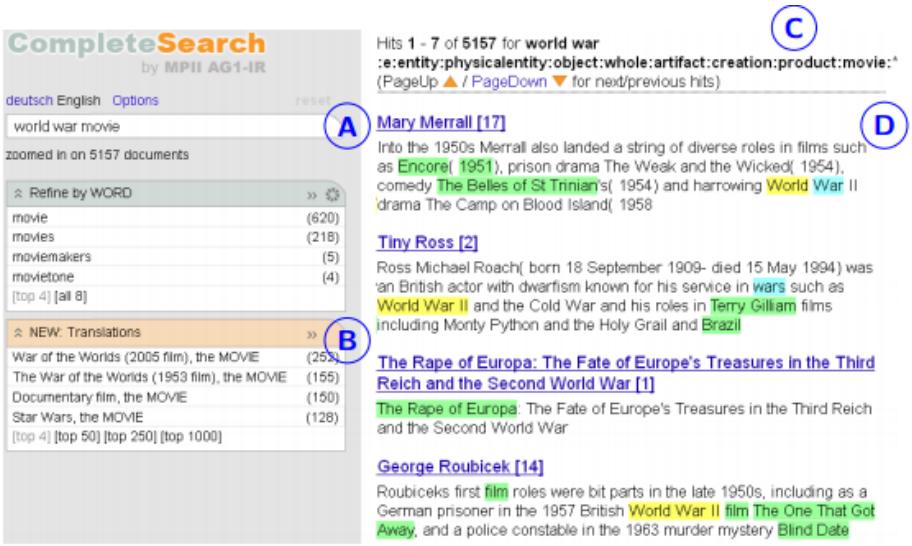



Figure 2-13 Screenshot of SUSI

"A" is the input field. After input for last word, classes are automatically detected. The "B" boxes list the suggestion of classes from last prefix of query string. "C" indicate the query string that generate by system and used that query string to retrieve information. As we can see, the query string represented by very long and complicated words. For instance, the "C" long words can be used to derive the Classes and Entities of query string. "D" mark the result that hits the prefix classes.

### 2.3.4 Comparison of Semantic Search Technology

ESTER (2007) and SUSI (2010) combine full-text and ontology search. The approach used in SUSI is quite similar to the ideas of ESTER. This two search algorithm/ technique support fast query processing and provide an intuitive and interactive user interface. ESTER is using prefix search and joins, but SUSI improve the ESTER algorithm with prefix search only. The ESTER output always next to an occurrence of an entity. However if only prefix search is using only instead of combination of prefix search and joins, this end up will blows up the index size, therefore ESTER improve their algorithm by uses joins to avoid this problem. The SUSI, which is the successor of ESTER solve the index explode problem by using some special techniques and this

techniques reduce the blowup index size problem which commonly encountered by semantic queries. This make SUSI having faster performance than ESTER. The special technique use by SUSI is restricting the relations to the classes' taxonomy and ontology entities. In terms on precision, SUSI is better than ESTER too. SUSI achieved this by treats each single sentence in any article as a separate document instead of whole article as single document.

Broccoli is created by a group of professor in Department of Computer Science University of Freiburg which lead by Dr. Hannah Bast in 2012, which is done after ESTER and SUSI. Similar with ESTER and SUSI, broccoli combine full-text and ontology search in their searching algorithm. The broccoli improve upon ESTER since elements from the ESTER ontology are not recognized in their contexts, which mean it does not consider contexts but purely syntactic proximity of words / entities. Besides that, Ester's simplistic user interface was ok for queries with one relation, but practically unusable for more complex queries.

As conclusion, ESTER and SUSI have used co-occurrence in a whole paragraph or sentence, or based on word proximity, both of these give poor results. Broccoli provide the indexing search that support query times of roughly 100 milliseconds or less.

## 2.4 Factors that Affect Store Choice

This part of review is to understand consumer psychology on store choice. In order to get the idea of how to make the application be able to automatically provide suggestion on the ideal store to consumer so they can spend the least cost to buy the product, we review a journal that mentioned about measurement between store choice and distance, price, etc. factors.

### 2.4.1 How the Measurement of Store Choice Behaviour Moderates the Relationship between Distance and Store Choice Behaviour

This is a Study of the Interface Between Retailers and Consumers (Hansen,T and Cumberland,F and Hans, S, Solgaard., 2004). This article mainly discuss that store location is a factor that will influences consumer store choice decision. In this article, author mentioned that, commonly for consumer to choose between grocery stores,

consumers may make an assessment on several criteria such as product quality, product price, assortment, convenient location, distance between consumer's home to store, customer service, store atmosphere, and pleasantness of shopping. However, the most critical one is price and distance as shown in figure above. Both of these cost should fall within consumer's budget or so called usual resource limit.

This author carry out the research above by empirical research which include 631 consumers and 58 helpers which is marketing or business bachelor holder as data collectors.

The consumers are request to provide response to the following aspect:

Quality: Offers good quality grocery and fresh grocery products.
Atmosphere: Has a good in-store atmosphere and clean environment with friendly and helpful staff.
Price level: Offers low prices or always have special offers.
Assortment: Provide more choice on grocery products and often released new product.

Distance was measured by using the cognitive physical distance and time distance from a consumer's home to the outlets that visit most often.

The studies show that the main criteria that affect consumer store choice is physical location and time distance, product price and service output. The results show distance will be largely affect the choice behavior as the frequency go to the store and the time required. Thus the author came up the model below:

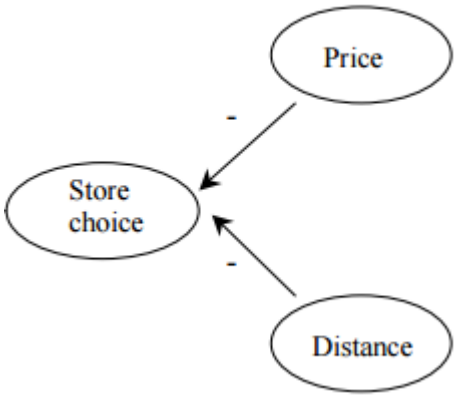



Figure 2-14 Model of consumer store choice

## 2.5 Mobile Apps Revenue

Although there is plenty of money to be made from mobile apps, it doesn't come easy. In this part we will review on the solid strategy in place in order to create a profitable app.

### 2.5.1 Source of Application Revenue

According to think gaming apps sales data analysis from US, free apps can generate alot of revenue. As an example, accoding to the sales data analysis done by thinkgaming.com from US, Clash of Clans (COC) is a free app. The revenue getting by COC per day is around $1,009,463. Although there is plenty of money to be made from mobile apps, it doesn't come easy. We must have a solid strategy to create a successful apps that can gain profit. Some technique to get the revenue and make money for our app are listed below.

**Advertisement**

Some external advertisement will be integrate with CONAWA in some way such as:

**(I)** Banner : Appear at the top or bottom of app screen and can prompt users to do some action such as application installation or view online shopping website's product.

**(II)** Full-page ads: Full-page ads that cover whole background in app at natural breaks or transition points.

**(III)** Video ads : The ads will be play automatically and users are allow to skip the video after 5 seconds, such as YouTube.

All above three ways can be achieved by using a third party advertisement agent webservice that will auto provide different category of advertisement based on developer desired.

**Cooperation**

Cooperate with retailer shop or restaurant (eg: café ) to give the voucher, so can achieve win-win situation on all 3 party ( CONAWA, business seller, consumer ). The voucher image will be given by the retailer shop or restaurant. Whenever any consumer make purchase based on the voucher, some commission will be shared to us.

**In-apps purchase**

In-App Purchase will be implemented into the CONAWA app. The way it works is whenever the user wishes to have some gift redemption by using points earning, 1-5% of the points redeem that earn by the user will be taken away by system, and the taken away points will be some kind of revenue income for the system.

# CHAPTER 3 METHODOLOGY

## 3.1 Chapter Overview

Methodology and software with hardware technology that involved in the proposed system will be discuss and justify in this chapter. User, hardware and software requirement will be discuss in formal way to ease reader when reading. It explains how the system works through modelling the requirement according to UML standard such as several high level designs: use case diagram and activity diagram to describe how the system interact with user., class diagram and object diagram to show the class and object that required in the system. Architectural design, and the language and scripting use in each 3 tier software architecture are discuss in this chapter in detail.

## 3.2 System Development

System Development Life Cycle (SDLC) is the most common process to ensure that software system meet the requirements and develop in consistent way. Very often, we need to follow the formal SDLC methodology in order to ensure consistency in development and be more able to estimate the resource required. The software development methodology that is used for this project is throwaway prototype methodology.

### 3.2.1 Development Methodology

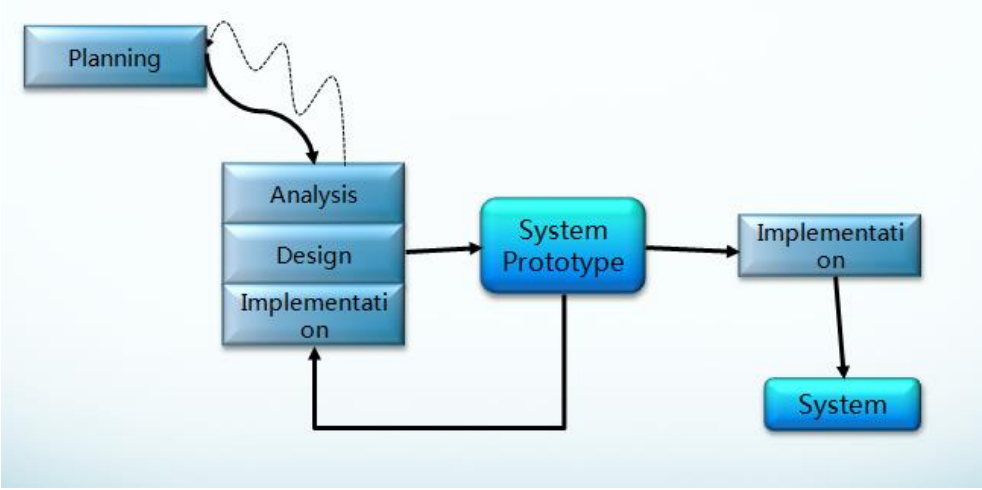



Figure 3-1 Evolutionary Prototype

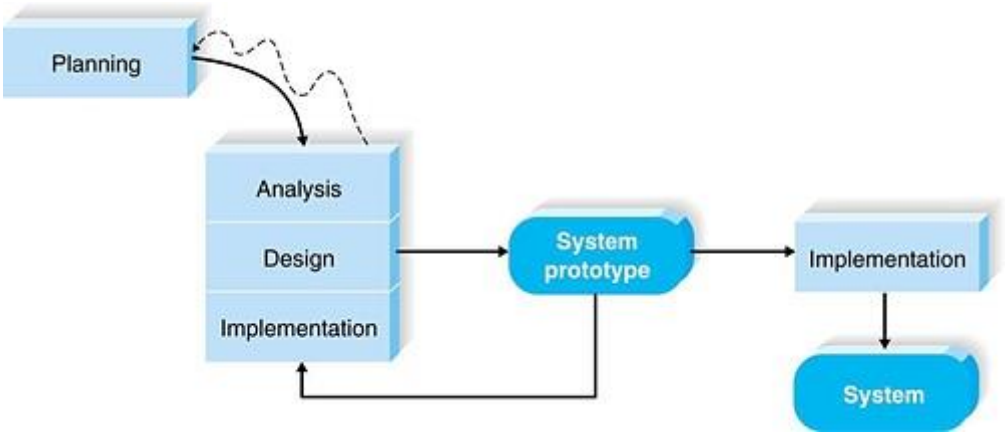



Figure 3-2 Throwaway Prototyping

In nowadays everyone are strive  for the time required of put the product into market, the software should be out the door faster.  Software cannot take years to develop, the faster the development process, the product can put in market earlier, and thus more benefit can be earned. There are many type of system development methodology model can be used to develop a system such as Rapid Application Development (RAD), Agile, Prototyping, Waterfall and so on. According to Radack (n.d.), the most effective way is to put everything into the development process which starts with the analysis, design, implementation and system prototype and continues until disposal of the system.


Figure 3-3 SDLC model

We choose Throwaway Prototype methodology as our system development methodology. This is iterative and incremental model. The difference between throwaway and evolutionary prototype is whether we build the full system on the prototype or whether we throw it away when complete the prototype development and completely re-code in order to build the new full system. In Throwaway Prototype, only capture the essence and reuse the essence part in porotype development phase. Both categorize as Rapid Application Development (RAD) since it able to see the outcome early and welcome the changing of requirement. After system planning phase, this model constantly develop a system after undergoing 3 main phases which are system prototype analysis phase, design phase and system prototype implementation phase. This 3 process will be keep iterate and we will let user to try out the system prototype then collect feedback from them. This is to ensure all the system requirement are achievable, is workable and the system requirement had validate by using the system prototype.

This methodology eventually breaks the requirements into series of cycle and built the prototype sequentially through these series of different version of CONAWA app. The prototype analysis phase then leads into prototype design and implementation. This mean the prototype can be built sequentially so that the future version prototype will begin only after the preceding prototype version has been approved, implemented and fulfill requirement. We start out with Version 1 in first iteration that does all the things abstract module that must be in the application, after collect feedback from user then follow it with Version 2 in second iteration that adds more features, enhancement, and so on. The requirement will be rate by priority matrix (high, middle, low). High priority requirements will be put in first iteration and so on. After the features in prototype agree by the user and confirm stable, we will throw the prototype and rebuild a complete system integrated by using top down approach to build the complete system so it can save time and cost of debugging.

After throw away the prototype, the system environment and configuration setting will be setup in this phase, then part of the user interface code that done in the prototype will be reuse and refine in system implementation. The business flow and the database mapping, create table script and stored procedure will be develop at here too. Before

we setup the connectivity and create web service, all the data will be hardcoded model such as hardcode the data in list or collection. After the business logic, database table created and UI are done, we will start to setup network connectivity and complete web service in order to let business logic to communicate will database or external web server. At this moment, we will keep pump data from front end into database and see whether the data communication and passing is successful. Later on, all the module will be integrate by top down approach over big bang approach because if we integrate everything together like big bang approach, it is harder to do debugging and detect the defects.

The biggest advantage of using this approach is it welcome the changes of requirements when prototype development phase, lets us adjust the features easier that need to be added in future to cater more requirements or remove unrealistic, unachievable, unpractical requirement. It would not cost much resources to change the requirement since this is just a static prototype with no dynamic function. Unlike some other product development methods such as traditional SDLC waterfall approach, all design is done at the beginning of the project, and then changes are harder to make when the requirements change. This can ease us evaluate the proposed system through the several cycle of prototype version before implementing the whole business flow into real proposed system.

Since this project is an individual project instead of group project so this methodology provide a lot advantage for individual project. It makes budgets easier to cope. We haven't done such an huge individual project like this before, we does not have a lot budget in this project so by this approach we can ensures that the cost of development is lower than what it would have been if the whole problem was tackled together. Below show the detail break down in each phase:

**Planning**

On the planning phase, existing problems faced by the consumer are identified and the system idea will be gathered. The development of project scope to outline the project's deliverables and define the goals and target user. Later on, project objectives, and time required from requirement gathering until complete of whole system are being identify

in this process too. Project methodology, objectives, both internal and external resources required and time required are being identify in this process too.

**Analysis Phase**

After determine the software and hardware requirements for the CONAWA project, the software requirement will separate into functional and non-functional requirements and the requirement will write in formal format according to ISO900 standard. Functional requirements is to define the features or service that this system shall provide whereas non-functional requirement is the constraints of the services or features that list in functional requirement such as performance constraints, security constraints or operational and cultural and political Requirements constraints on this system. Static analysis review testing are done base on system requirement to remove the unnecessary, unrealistic, conflict, ambiguous requirement.

Based on the data and requirement that has been gather, we analyze the existing problem faced previously and develop a solution from it. Later on, list all the solution and the requirements and modelling the requirement according to UML standard such as use case diagram, activity diagram, class diagram and object diagram that will discuss in Chapter 3. Project methodology and both internal and external resources required are being identify in this phase. Timeline Gantt chart are develop in order to monitor and track the project progress.

**Design and prototype implementation phase**

Design and prototype implementation phase will be conducted after thorough planning and analysis, which will develop prototype for this system. Some of designing examples are Entity relationship diagram design, related module design icon and logo design, business logic design, and system interface design. We have to finish all the requirement in the different iterative of the system lifecycle until the requirement are clearly understood.

There are always defects and bugs in the design, so will refine it and do more testing such as dynamic testing and static testing. The details of the internal design such as connectivity, integration, web service, database connectivity and external aspects like

performance and security can be ignored at this stage. The several version of CONAWA Prototype is developed in this stage, where the very basic requirements are showcased and user interfaces are provided.

These features may not exactly work in the same manner internally in the actual software developed. The prototype will be create and let the users use and evaluate it. Based on their feedback, we will decide whether is it necessary to continue develop another improvised prototype or stop the prototype that will satisfy their requirements. Basically, the iterative will keep repeating until CONAWA app prototype is thought to be completed and finalized by the developers or users.

**Implementation**

After prototype version finalized, integration technique design, test case design and full system interface design will be carry out at this phase. The prototype will be throw away and the process of implementation full system include non-functional requirement will be carry out, database create table and insert table script, and interface and business logic will be code in the system. Part of the user interface code that done in the prototype will be reuse in system implementation and continue refine.

After the business logic, database table created and UI are done, we will start to setup network connectivity and complete web service in order to let business logic to communicate will database or external web server. Integration and system testing will be performed on the system to detect and remove the defects in the system that will cause failure that deviate from our expected result.

### 3.2.2 System Development Methodology Flow Chart

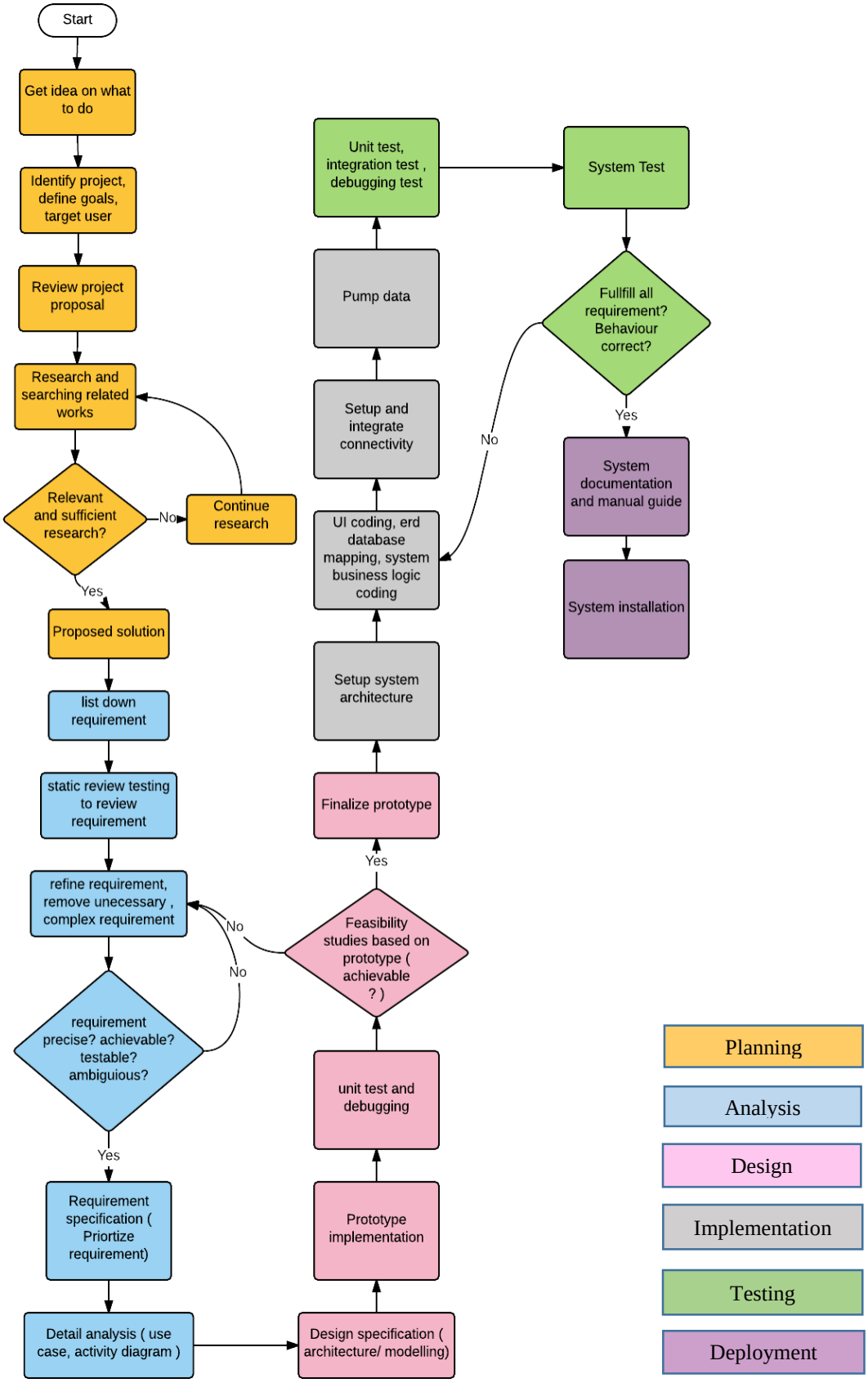



Figure 3-4 Flowchart of System development

### 3.3 Proposed Solution

The proposed solution is to develop an android mobile application (if time permits, hybrid apps will be develop for both iOS and windows platform too) which can be widely used by consumer and retailer outlets. It can assist those business seller and consumers in Malaysia with combination of multiple functions within single mobile applications. The solution allows user to know the best deals in different store there are located near them quickly instead of blind search and wondering around shop.

The several user category as shown in picture below such as consumer, administrator and retailer store will interact with system directly by using smart device. The smart device will use the integrate GPS sensor to retrieve information from database through web service that host in web server. Finally, the information that retrieve from database will sent back to user device.

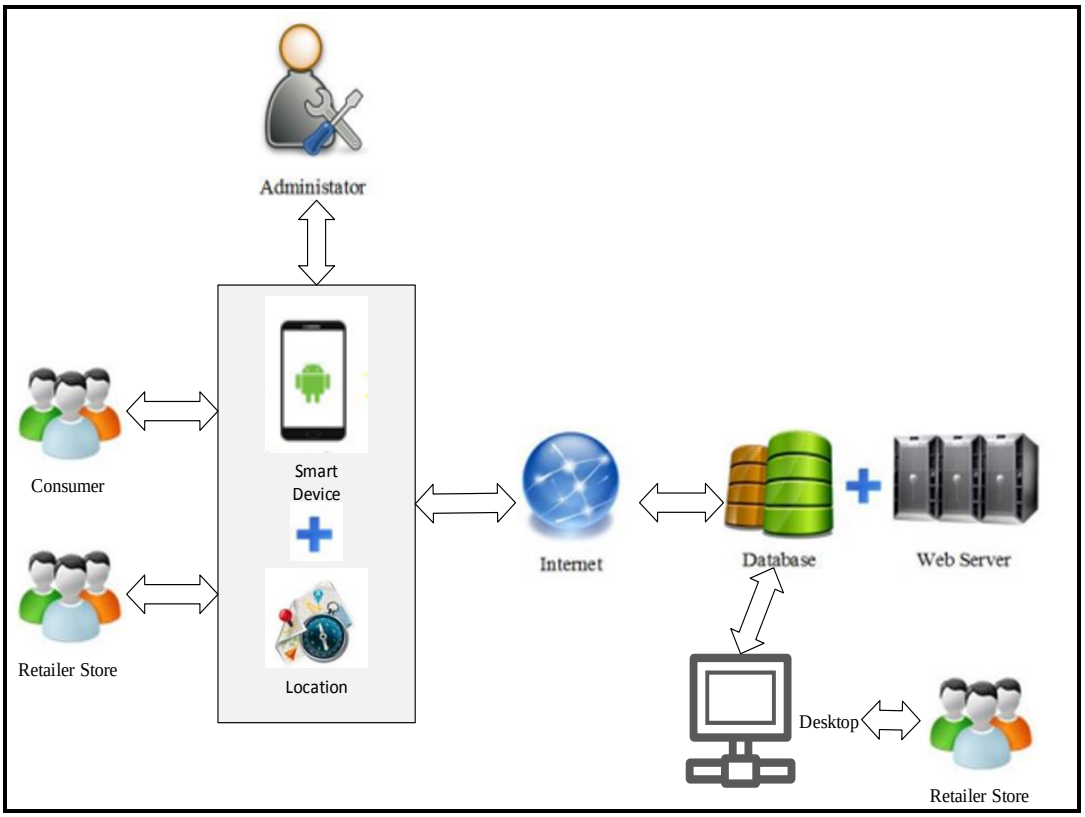



Figure 3-5 Block Diagram of Proposed system

### 3.3.1 Proposed System Features

1) Store's catalogue
2) Share new Item
3) Share Price
4) Search Item by semantic search algorithm and Intelligent Search Item based on nearby location
5) GPS navigation to desire store and single map that show all available store around user location
6) Offline shopping list
7) Check In Widget
8) Share To Social Media
9) Fussy Calculator
10) Ideal Store Suggestion

### 3.3.2 Semantic Search Flow

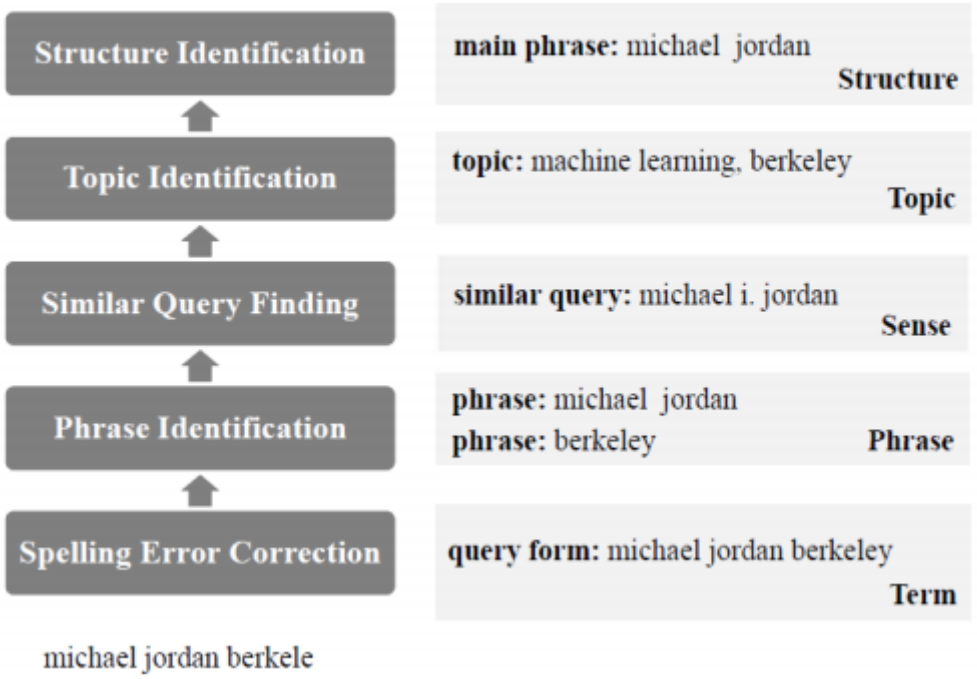



Figure 3-6 Semantic Search Overall Flow

There are mainly 2 step in semantic search:

i. Matching -> Matching between one query and another query.
   - Challenge-> Mismatch

ii. Ranking -> Ranking a list of documents

**(I)** Correct ranking on the top

Different search engines are designed dedicated for different area or different purpose (e.g.: websites, google). For our system, we will focus on ontologies of consumer product.

### 3.3.3 Ideal store suggestion

The common fuel type used by Malaysian is RON 95 and RON 97. According to Mileage Comparison between RON 95 and RON 97 in Constant Speed Test (A.E. Md. Saad, Z. Saad, A.K. Ahmad, S. Abdullah, F. Ab. Razak, 2011), we will need to accelerate to reach higher speed, thus when a car is accelerating, the fuel consumption is relatively higher. This is because the car required more fue in order to accelerate. However if we can maintain constant speed (either traffic free or straight road with no holes on it), the car consumption will be at the lowest because combustion is complete by now.

According to the car specification which has been published in the website of The Malaysian Automotive Portal (Autoworld), after include air conditioning and other factors such as wind resistance, the average for normal distant is 15kms to 16kms per liter while distant or long journey at constant speed of 70 -90 km/h consumes the average of 17-18kilometers per liter since you maintaining constant velocity for manual car, and 15.9 kilometers per liter for auto car. Below show the experiment result done by the 4 researcher from Universiti Teknologi Mara (UiTM) MALAYSIA.

| Speed (km/h) | FC of RON 95 (liter/h) | Mileage of RON 95 (km/liter) | FC of RON 97 (liter/h) | Mileage of RON 97 (km/liter) | Average Mileage per liter for Both RONs (km/liter) |
|---|---|---|---|---|---|
| 30 | 2.2 | 13.6 | 1.9 | 15.8 | 14.7 |
| 40 | 2.7 | 15.1 | 2.3 | 17.4 | 16.3 |
| 50 | 3.1 | 16.1 | 2.6 | 19.1 | 17.6 |
| 60 | 3.6 | 16.7 | 3.1 | 19.4 | 18.1 |
| 70 | 4.0 | 17.5 | 3.5 | 20.0 | 18.8 |
| 80 | 4.8 | 16.7 | 4.4 | 18.2 | 17.5 |
| **90** | **5.9** | **15.3** | **5.4** | **16.7** | **16.0** |
| 100 | 7.0 | 14.3 | 6.4 | 15.6 | 15.0 |
| 110 | 8.6 | 12.8 | 8.2 | 13.5 | 13.2 |

Table 3-1 Average Car Fuel Consumption in km per hour

From both the Malaysia car specifications which had been published and have the research done, the average fuel spend per km is 0.055L – 0.075L.The fuel spend will also varies according to the car brand, and car engine volume (in unit cc).

However in our application we just need to consider the relative price of fuel spend per km for each location, so the variables such as car type and engine volume and so on is constant variable, just the location is manipulate variable. The price of petrol spend is the responding variable, it just give rough price and not an exact one since we more interest on calculating which outlet location or which path having lesser fuel consumption.

Thus the diagram below depict the ideal store module flow:

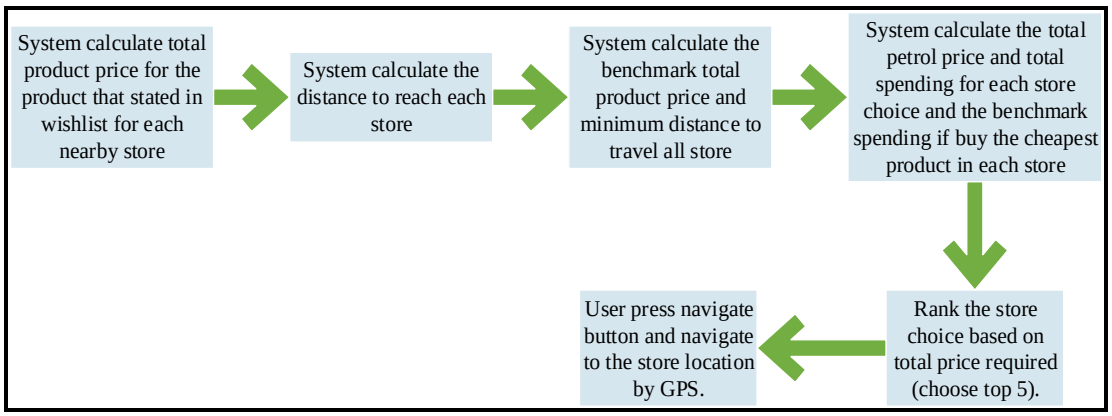



Figure 3-7 Ideal Store Suggestion Flowchart

Step 1 – System calculate total product price for the product that stated in wishlist for each store nearby.

Table 3-2 Total Product Price Conceptual Table

| Store | Total Product Price |
|---|---|
| Giant, Kampar | RM 104.99 |
| Tesco, Kampar | RM 107.99 |
| Econsave, Kampar | RM 102.99 |
| Tenaga Cergas, Kampar | RM 110.99 |

Step 2 – System calculate the distance to reach each store.

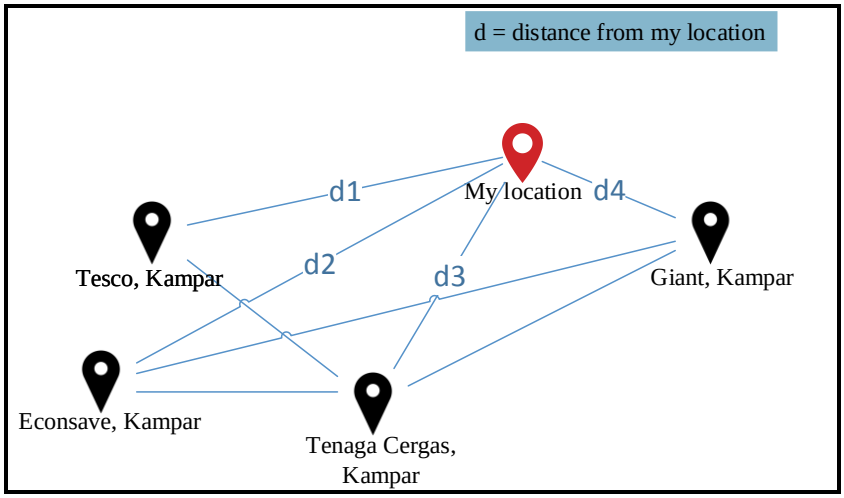



Figure 3-8 Distance of Each Store

Table 3-3 Total Product Price and Total Distance for Each Store Conceptual Table

| Store | Total Product Price | Total Distance ( km) |
|---|---|---|
| Giant, Kampar | RM 104.99 | d4 |
| Tesco, Kampar | RM 107.99 | d1 |
| Econsave, Kampar | RM 102.99 | d2 |
| Tenaga Cergas, Kampar | RM 110.99 | d3 |

Step 3 – System calculate the benchmark total product price and minimum distance to travel all store.

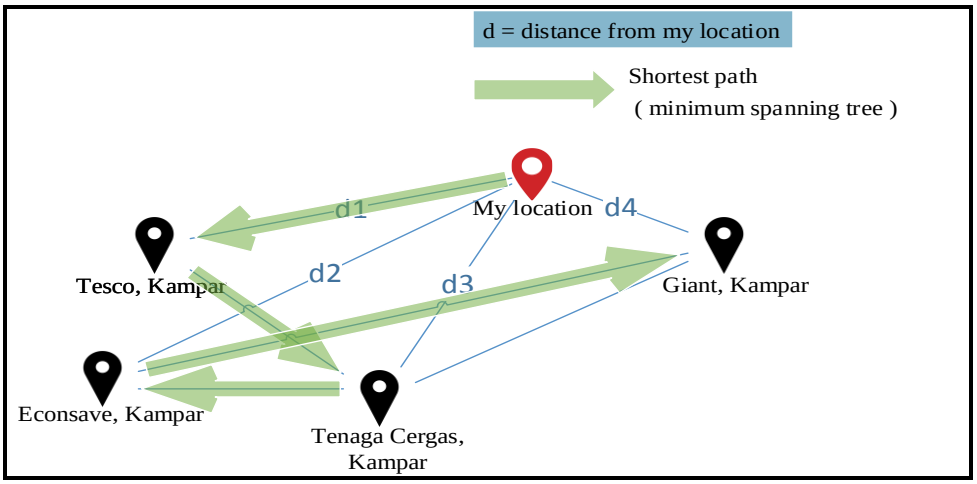



Figure 3-9 Minimum distance required Travel to all Store

Table 3-4 Total Product Price and Total Distance for Benchmark Measure

| Store | Total Product Price | Total Distance ( km) |
|---|---|---|
| Giant, Kampar | RM 104.99 | d4 |
| Tesco, Kampar | RM 107.99 | d1 |
| Econsave, Kampar | RM 102.99 | d2 |
| Tenaga Cergas, Kampar | RM 110.99 | d3 |
| **Giant, Kampar**<br>**+Tesco, Kampar**<br>**+ Econsave, Kampar**<br>**+ Tenaga Cergas, Kampar** | **RM 97.99** | **d5 ( shortest path )** |

Step 4 – System calculate the total petrol price and total spending for each store choice and the benchmark spending if buy the cheapest product in each store ( travel to all store).

Table 3-5 Total Spending for Each Store

| Store | Total Product Price(RM) | Total Distance ( km) | Total Petrol Price(RM) | Total Price Required(RM) |
|---|---|---|---|---|
| Giant, Kampar | RM 104.99 | d4 | p4 | RM 104.99+ p4 |
| Tesco, Kampar | RM 107.99 | d1 | p1 | RM 107.99+ p1 |
| Econsave, Kampar | RM 102.99 | d2 | p2 | RM 102.99+ p2 |
| Tenaga Cergas, Kampar | RM 110.99 | d3 | p3 | RM 110.99+ p3 |
| **Giant, Kampar +Tesco, Kampar + Econsave, Kampar + Tenaga Cergas, Kampar** | **RM 97.99** | **d5 ( shortest path )** | **p5** | **RM 97.99+ p5** |

Step 5 – Rank the store choice based on total price required (choose top 5).

Step 6 – User press navigate button and navigate to the store location by GPS.

### 3.4 System Planning

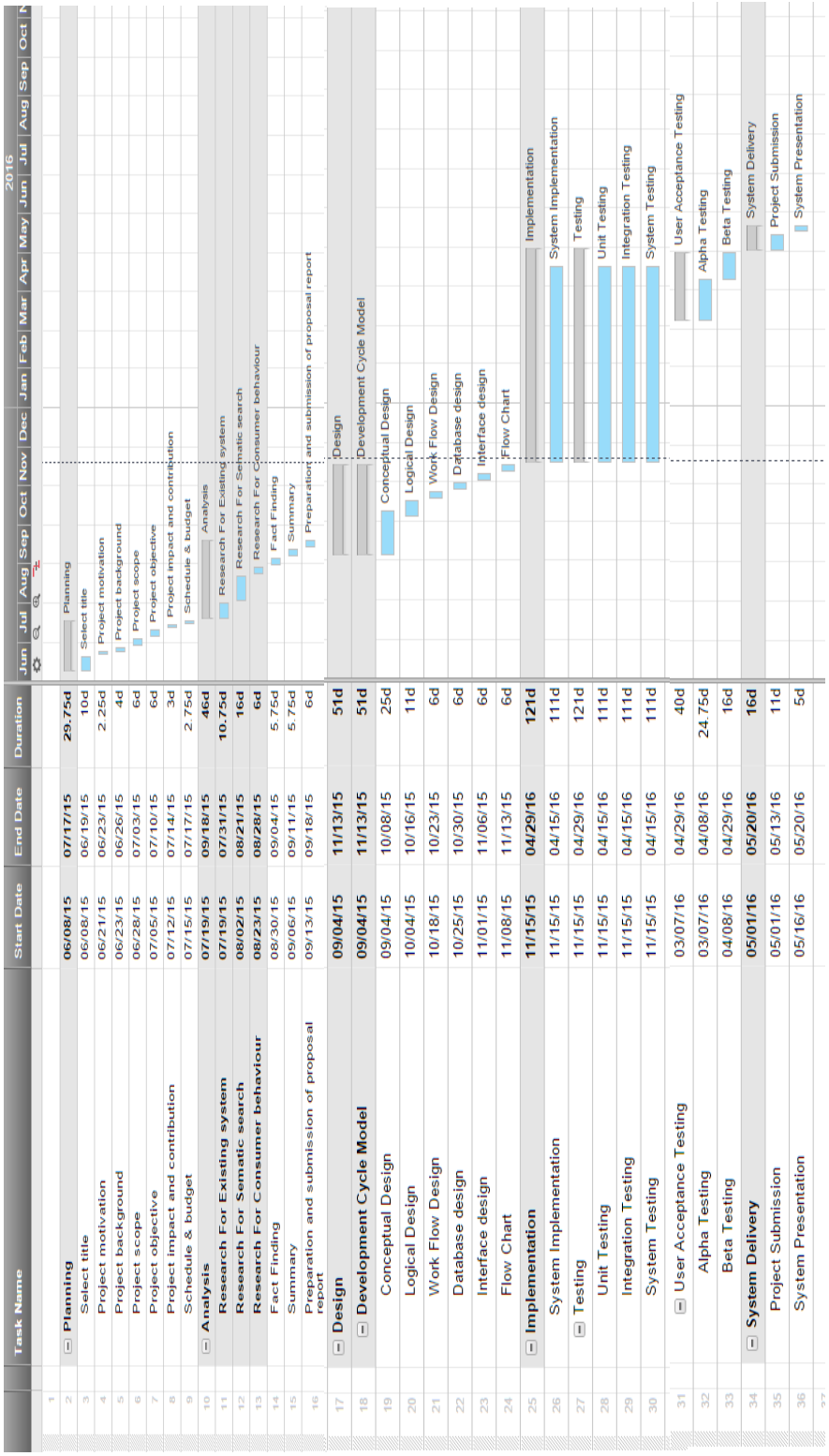
In the system planning phase, existing problems faced by the consumers are identified and the system idea will be gathered. The reviews of several consumer application, semantic technology, human store choice behavior and mobile application marketing technique will be done. The scope has been developed in order to outline the project's deliverables and define the goals and target user. From these information, solutions are proposed. Later on, project objectives, and time required from requirement gathering until complete of whole system are being identify in this process too. Through reviews, basic requirements of the consumer decision making system is collected and thus, provide a clearer picture of developing a solution to solve the existing problem by adding certain services in system.

### 3.4.1 Project Timeline

As shown in the Gantt chart below, 1 year will be used to complete this project. This project consist of 5 main phases which are Planning, Analysis, system design, Implementation, System Delivery respectively. The first two month is mainly used to perform the planning of system, followed by 2 and half month for the system requirement analysis. For the following 2 and a half month, the system design is proposed to continue with the whole development process. Then, 5 and a half month will be used to develop the system. At the same time of development, the testing phase will be carried out throughout the whole process in parallel way in order to reduce cost of testing. Finally, the final month is used for documentation and the collection of feedback.

The Documentation of Final Year Project will be done in this short semester which is three and half week. The Documentation includes of the planning, analysis and design of the system prototype. Later on, the prototype of the proposed project will start developing from the fourth week until the 7th week.

In next January long semester, the full project implementation will start in week 1 and continue until week 9. After the system deployment, the system will enter maintenance phase and will collect feedback from user and use for future enhancement.




| | Task Name | Start Date | End Date | Duration |
|---|---|---|---|---|
| 1 | | | | |
| 2 | **Planning** | **06/08/15** | **07/17/15** | **29.75d** |
| 3 | Select title | 06/08/15 | 06/19/15 | 10d |
| 4 | Project motivation | 06/21/15 | 06/23/15 | 2.25d |
| 5 | Project background | 06/23/15 | 06/26/15 | 4d |
| 6 | Project scope | 06/28/15 | 07/03/15 | 6d |
| 7 | Project objective | 07/05/15 | 07/10/15 | 6d |
| 8 | Project impact and contribution | 07/12/15 | 07/14/15 | 3d |
| 9 | Schedule & budget | 07/15/15 | 07/17/15 | 2.75d |
| 10 | **Analysis** | **07/19/15** | **09/18/15** | **46d** |
| 11 | **Research For Existing system** | **07/19/15** | **07/31/15** | **10.75d** |
| 12 | **Research For Sematic search** | **08/02/15** | **08/21/15** | **16d** |
| 13 | **Research For Consumer behaviour** | **08/23/15** | **08/28/15** | **6d** |
| 14 | Fact Finding | 08/30/15 | 09/04/15 | 5.75d |
| 15 | Summary | 09/06/15 | 09/11/15 | 5.75d |
| 16 | Preparation and submission of proposal report | 09/13/15 | 09/18/15 | 6d |
| 17 | **Design** | **09/04/15** | **11/13/15** | **51d** |
| 18 | **Development Cycle Model** | **09/04/15** | **11/13/15** | **51d** |
| 19 | Conceptual Design | 09/04/15 | 10/08/15 | 25d |
| 20 | Logical Design | 10/04/15 | 10/16/15 | 11d |
| 21 | Work Flow Design | 10/18/15 | 10/23/15 | 6d |
| 22 | Database design | 10/25/15 | 10/30/15 | 6d |
| 23 | Interface design | 11/01/15 | 11/06/15 | 6d |
| 24 | Flow Chart | 11/08/15 | 11/13/15 | 6d |
| 25 | **Implementation** | **11/15/15** | **04/29/16** | **121d** |
| 26 | System Implementation | 11/15/15 | 04/15/16 | 111d |
| 27 | **Testing** | 11/15/15 | 04/29/16 | 121d |
| 28 | Unit Testing | 11/15/15 | 04/15/16 | 111d |
| 29 | Integration Testing | 11/15/15 | 04/15/16 | 111d |
| 30 | System Testing | 11/15/15 | 04/15/16 | 111d |
| 31 | **User Acceptance Testing** | 03/07/16 | 04/29/16 | 40d |
| 32 | Alpha Testing | 03/07/16 | 04/08/16 | 24.75d |
| 33 | Beta Testing | 04/08/16 | 04/29/16 | 16d |
| 34 | **System Delivery** | **05/01/16** | **05/20/16** | **16d** |
| 35 | Project Submission | 05/01/16 | 05/13/16 | 11d |
| 36 | System Presentation | 05/16/16 | 05/20/16 | 5d |
| 37 | | | | |

Figure 3-10 Gantt chart

### 3.5 System analysis

In this part, user requirements are gathered by interviews and performing several research by observation. Later on, user requirements will be refined into system requirements and will be documented. This is a formal collection of information that organizes the requirements of a system. The hardware and software requirement will be listed in this phase too.

#### 3.5.1 Fact Finding

Fact finding is a process where the related information, user and system requirement, will be gathered in the planning phase through several techniques. The several techniques that used is listed below:

i. Observation

Through observation, we can observe how the existing system is being used by users. Apart from that, we went to several hypermarket to do some observation such as consumer behavior, and deduce out the problems faced by consumers. From such way, we get a clearer picture on the process of how consumers use the system, their action and their psychology.

ii. Research

We did a lot research on the Internet and that enabled to let us explore new and useful things. The data and facts that are relevant to consumer psychology and theirs needs were collected. By knowing how other related work done in this area and the previous existing problem, we can prevent some of their mistakes in future builds. We can do some enhancements based on the existing proposed system too.

iii. Survey

We created online survey through google form and distribute through email and social media. The question include multiple choice and single choice question. There are 20 people response to this survey and analysis were done based on the survey result.

### 3.5.2 Survey Analysis

The survey produced 20 responses during 3 days it was open in Google Form. Respondents are not required to write their name on this questionnaire, so their responses will never be linked to them personally. Below show all the question had been asked and the response analysis.

First Question: Do you prefer shopping outlet with cheaper average price compare to other store?

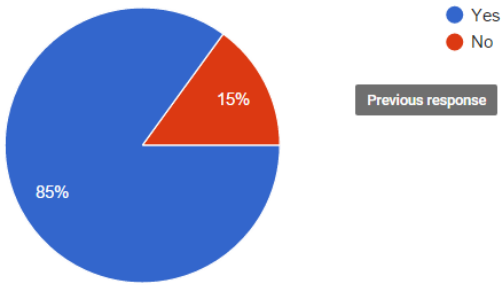



Chart 3-1 Cheaper Store Choice Pie Chart

Figure above show that among all the respondents, 17 or 85% of them stated that they prefer the store that sell product cheaper generally whereas 15% of them does not prefer to this statement. Basically this obviously depict that, in this GST price inflation period, everyone will try their possible to get cheaper product as possible.

Second Question: What is the aspect you will consider first for shopping store choice?

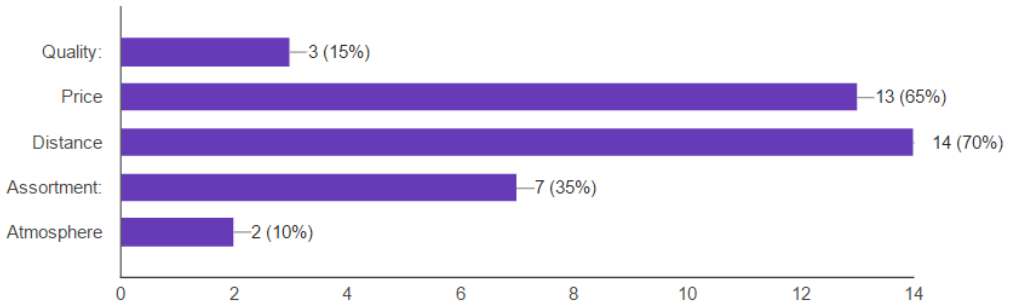



Chart 3-2 Aspect Considered in their store choice

According to figure above, this most aspect that user think that is important is Distance which is total 70% of them or 14 people. This question is multiple choice thus the result

is more than 100%. It is quite surprising that distance is a more important aspect than price that user will take into consideration for store choice. However the difference not that much, price aspect is second higher aspect. In short most people think that price and distance is the most important aspect in store choice.

Third Question: Before you decide to buy a product, will you do comparison or survey on different store?

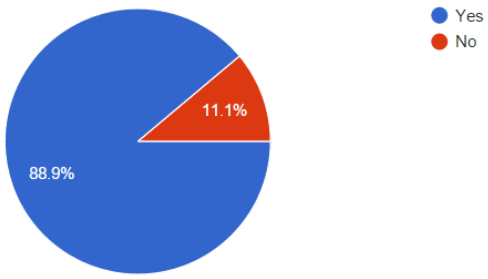



Chart 3-3 Survey on different store to get product with best price

From the chart in above, it shows that only 11% of respondents said that they do not care the price in different store or do not have interest to compare and survey other market before decide to buy certain product. This conclude that majority of consumers will do some survey before buy any product, the survey include self-approach to store or reading store brochure.

Fourth Question: How many outlet you will conduct the price survey before decide to buy any daily product? (E.g. Drinks, biscuit, cheese)

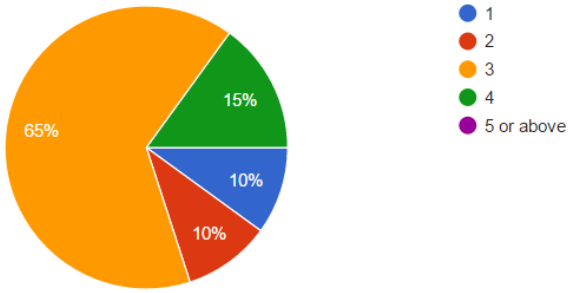



Chart 3-4 Number of Outlet Conduct Price Survey

From the result of survey, there are 65% of respondents agree that 3 store survey is the ideal number. This may due to nowadays majority do not really have time to do survey on too many store but survey before buying product still a must for them.

Fifth Question: Will you carry out price comparison on product with different packaging but same product, same product but different brand or different brand similar packaging before decide to buy any product?

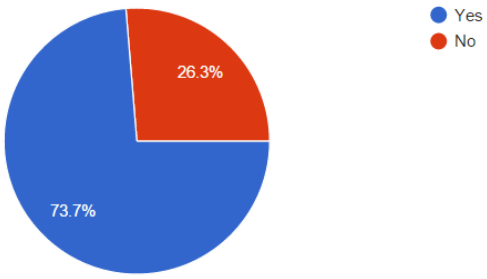



Chart 3-5 Unit Price Comparison Pie Chart

The pie chart above show the result of 20 respondents that answer to this particular question. As overall, 73% of the respondents will compare the product price when buying in terms on brand and packaging. Sometime they may need to compromise the unpopular brand and packaging together with unit price.

Sixth Question: Would you like to have a tools to assist you to decide on the stores is worth to go? (Taking distance and cheaper product price into consideration).

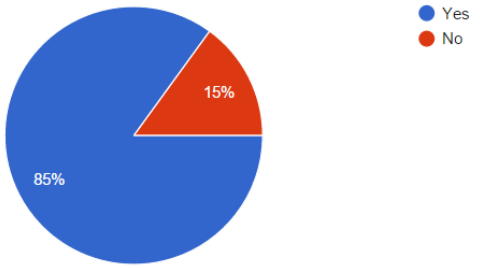



Chart 3-6 Wish To having own decision supporting tools

Seventh Question: Will you like to have a tools to support you in planning shopping route and navigation path?

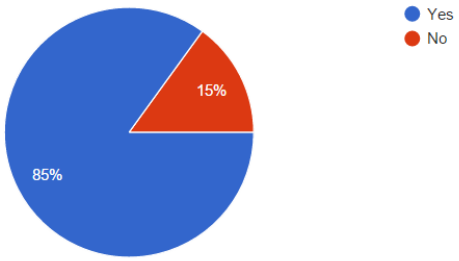



Chart 3-7 Like to try on new apps that can support them in purchase planning

Among all the respondents, 85% of them stated that they are having interest in trying the new apps and supporting us to invent the new idea to assist them in decision support making.

### 3.5.2 Stakeholder/ Actor

Consumer – Those who are interest to use this system help them in buying product

Retailer shop/ outlets representative – Official representative of retailer shop and outlets. They are allow to upload official information to system. Besides that, those retailer shop or restaurant (e.g.: café) representative that cooperate with CONAWA to give the voucher are in this category.

Developer – Planning, analyze, evaluate and develop the system of this system project.

Administrator – Monitor unusual behavior of system user such as spam, reporting and approve all the user request and updating.

### 3.5.3 Requirement Terminology

According to RFC 2119 and ISO900, English is imprecise, in order to ensure consistency of requirement, the verb should and shall will use in requirement.

**a.** "Shall" dictates what must be done. This is mandatory

**b.** "Should" dictates optional or desirable but not mandatory

### 3.5.4 System Requirement Overview

This part discuss about the features, services and constraints that the system should or shall provide. It separate into functional and non-functional requirement. Functional requirement mean the services that the system should or shall provide, how the system should behave to user inputs and how the user interact with system whereas non-functional requirement is the constraints on the services or features offered by the system.

### 3.5.5 Functional Requirement

**Share Item module**

I. **Share Item**

The system shall allow consumer to share the item which not exist in specific store/ outlet. The details include product description, image, price, and promotion end date if applicable. Shared item can be enter by the consumer manually or barcode scanning together with auto location detection, then provide location suggestion list.

II. **Share Price**

The system shall allow the consumer to share the item price if the specific item record already exist.

**Search Item Based on Nearby Store**

**a) Intelligent Search by Nearby Store**

The system shall allow consumer to search for item available in nearby store. This shall allow user capture the item name as input, the current location of the user (detect by location sensor and provide location suggestion list), and the maximum distance user willing to travel. Based on these information, system will list down all the matching item. The system should sort the list based on the lowest price or the nearest store (default will be the last updated time).

b) **Search Item by Semantic Search**

The system shall allow consumer to do product semantic search, which mean search for product name that having similar meaning/ different language or similar type.

**GPS Navigation to Desire Store**

The system shall provide the navigation service to navigate consumer to the store. This feature is especially useful when the user wish to purchase the item in certain store/location, but does not know the direction to the store.

**Store's Catalogue**

The system shall allow consumer to view most of the hypermarket catalogue main page in single interface. After pressing the catalogue picture, the system shall display all pages in the certain catalogue by swiping. The consumer can download the catalogue for offline viewing purpose.

**Fussy Calculator Module**

The system shall provide consumer to calculate the product unit price. It is common to see same product that has different packaging with different price. For example, a 12 pieces KitKat chocolate cost RM9 while 20 pieces cost RM14. Which packaging is worth buying?

**Shopping Wish list**

The system shall provide the convenience for consumer to record down the item that they wish to purchase. Consumer can add item, delete item and "check/tick" on the items that have been purchase and keep track on the remaining items. Users can estimate how much they will be spending.

**a. Ideal Store suggestion**

The system shall provide user the total cost will spend (petrol cost and total product cost) based on user market choice. The system should display the cost that will spend for each store and the benchmark cost that will spend if user go to every store ( buy product 1,2 in store A, product 3,5 in store B and product 4,6 in store C)

**User Login**

The system shall let member login to the system validate the password and username.

### 3.5.6 Non-Functional Requirement

**Operational Requirements**

3.1. The system should operate in most major android mobile device.

3.2. The system should operate on screen regardless which resolution.

**Performance Requirements**

a. The system should having responds time (interaction activities such as button clicking) within 0.5 - 4 second (depends on device data bandwidth).
b. The system should be able to detect concurrent action of different user.
c. The system should refresh information when the time scrolling down the page, the scroll page is already in top of the page but still continue scroll down.

**Security Requirements**

a. The system should only let administrator to approve the requesting item by user.
b. The system should not allow time interval of consumer share price or share item smaller than 1 minute.

**Cultural and Political Requirements**

a. Ringgit Malaysia (RM) is the only currency used in this system.
b. English is the main display language used in this system however Chinese input are allow in this system too.

**Usability Requirement**

a. The system should have a friendly user interface which is easy to learn, self-explanatory, have good utility and to let user easy to remember the step to perform certain tasks,

### 3.5.7 Software Requirement

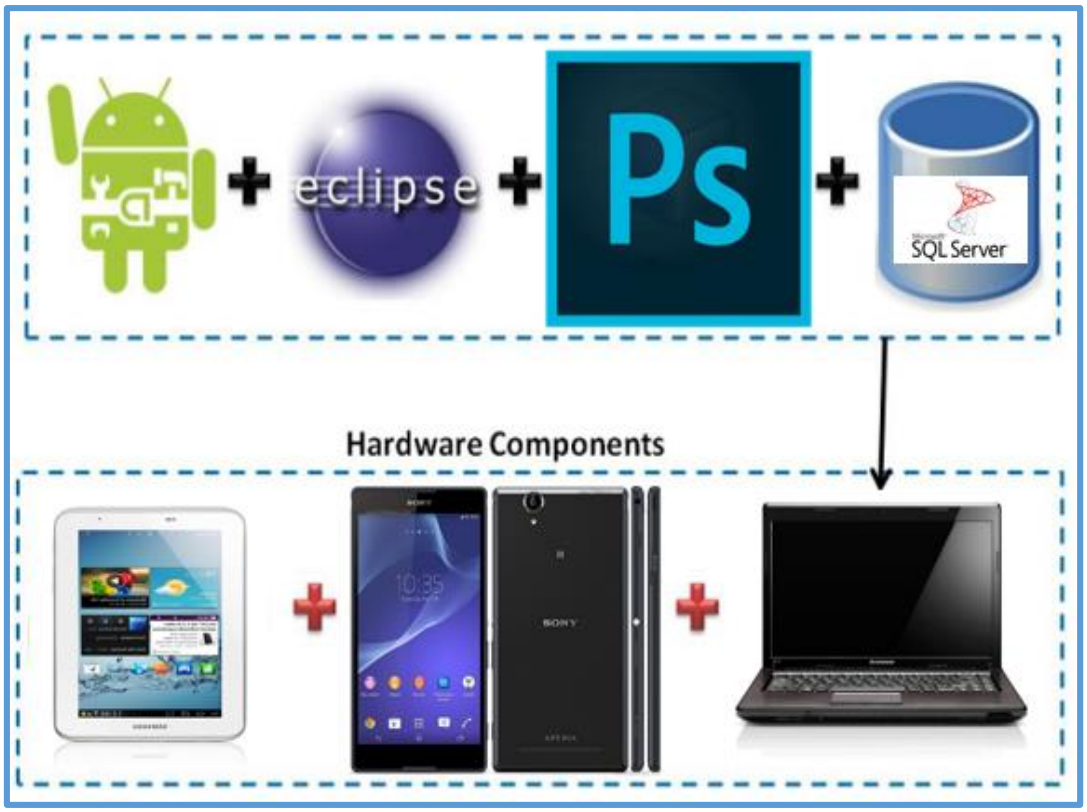



Figure 3-11 System Design Specification

There are several piece of software and technologies have been used to assist the development of the system. They are divided into 5 categories, which are stated below:

**(I)** Design and Development Tools

- **Visual Paradigm Community Edition**

  Visual Paradigm is a software and system development tool commonly used for system modelling. It support the modelling of business processes and work flow with diagrams such as use case diagram, class diagram, sequence diagram and package diagram.

- **Microsoft Project 2013**

  It is a project management solution that manage project and resources and analyze project information so can always keep in track and aware of the milestones.

- **Eclipse**

  Eclipse Kepler is an IDE that can support many language. It mainly use in whole development of system. All the technical coding part are done in here.

- **Adobe Photoshop**

  This is the photo editing software used to edit images or create the new images.

**(II) Database**

- **SqLite**

  SQLite is a software library that use in android application to implements a self-contained, server less, zero-configuration, transactional SQL database engine. SQLite is the most widely deployed SQL database engine in the world.

- **Microsoft SQL server**

  Microsoft SQL server is a Relational Database Management System (RDBMS) serve as main centralized data storage during system development and deployment. It is used to create database solutions to store the data. Stored procedure will be wrote too to support every transaction of program.

**(III) Operating System**

- Microsoft Windows 8.0

  Windows 8.0 is the latest updates from Microsoft. It will be used during the system development by which diagramming and programming processes will be done in this OS. Using this OS to develop the system gives more or less the same performances for the system but it gives a better interface for user to use.

### 3.5.9 Hardware Requirements

**a) Android Platform Tablet and Smartphone (device with different screen resolution)**

An environment use to test and debugging the application. Application can install on the device directly from the compiler with Android bridge driver. Different screen resolution of device and different OEM device will be required to do system testing, this include XiaoMi phone, Samsung phone, Samsung Tablet, Sony Phone etc.

**b) Hardware Specification**

Figure 3-12 Development Environment Device

Table 3-6 Hardware Specification

| | |
|---|---|
| **Processor** | Intel Core i5-2450M Processor (2.5 GHz, 3MB Cache, up to 3.3GHz) |
| **RAM** | 6.00 GB |
| **Operating System** | Windows 8.1 Pro |
| **HDD** | 720GB SATA 5400RPM |
| **Battery** | 6-cell Li-ion battery |
| **USB Port** | 3 |

## 3.7 Analysis UML Diagram

### 3.7.1 Use Case Diagram

There are 10 modules are cover in project scope. The administrator module not under scope, but in order to have better understand about how system interact, we add the system admin actor and the management module. Basically the module cover include Log in, registration, search item, share item, share price, unit price calculator, catalogue, favorite list, cart list, Desktop check in widget, GPS navigation and ideal store suggestion. Each circle below depict single module or combination of module.

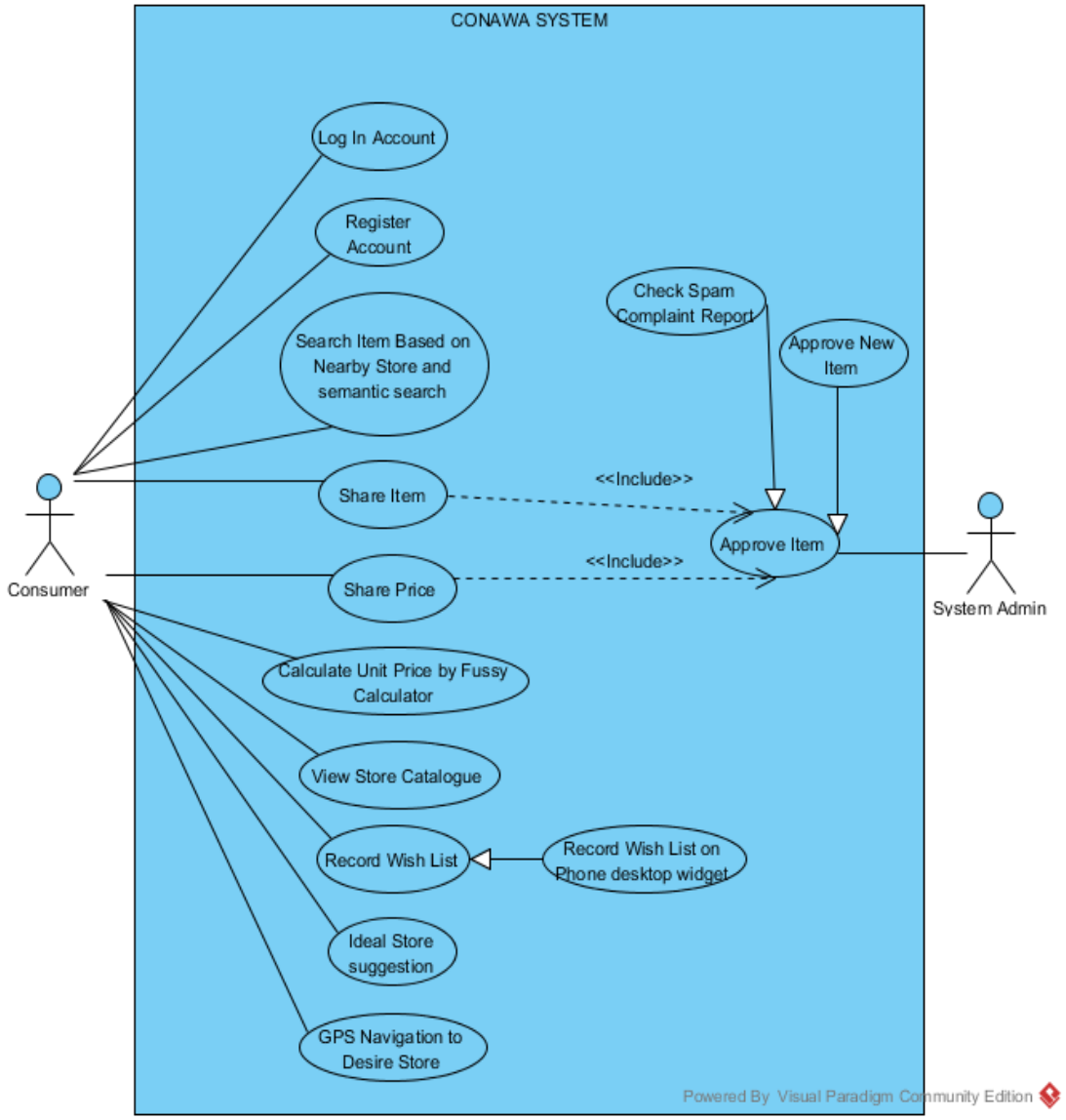



Figure 3-13 Use Case Diagram

### 3.7.2 Activity Diagram

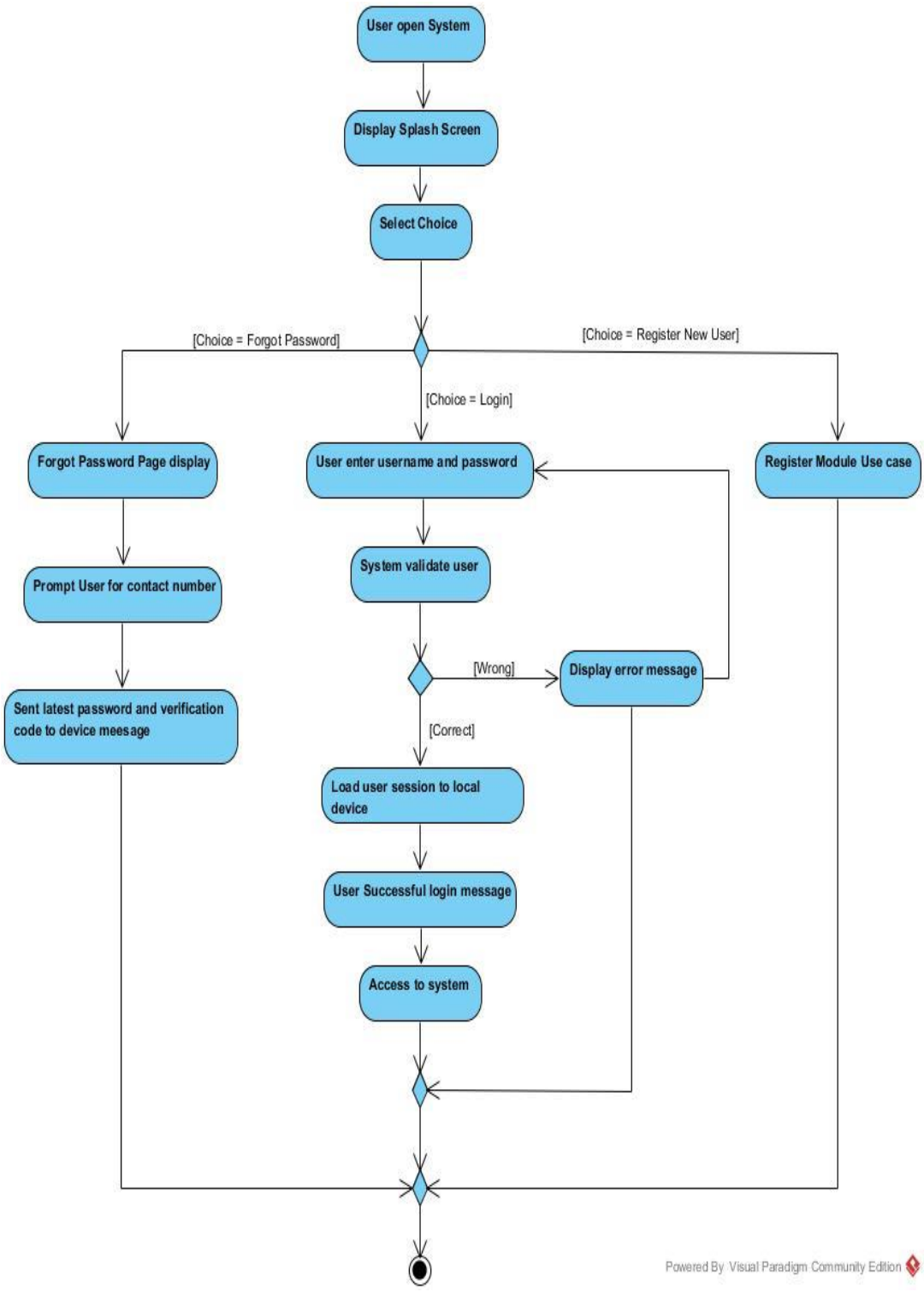



Figure 3-14 Activity Diagram of Login System

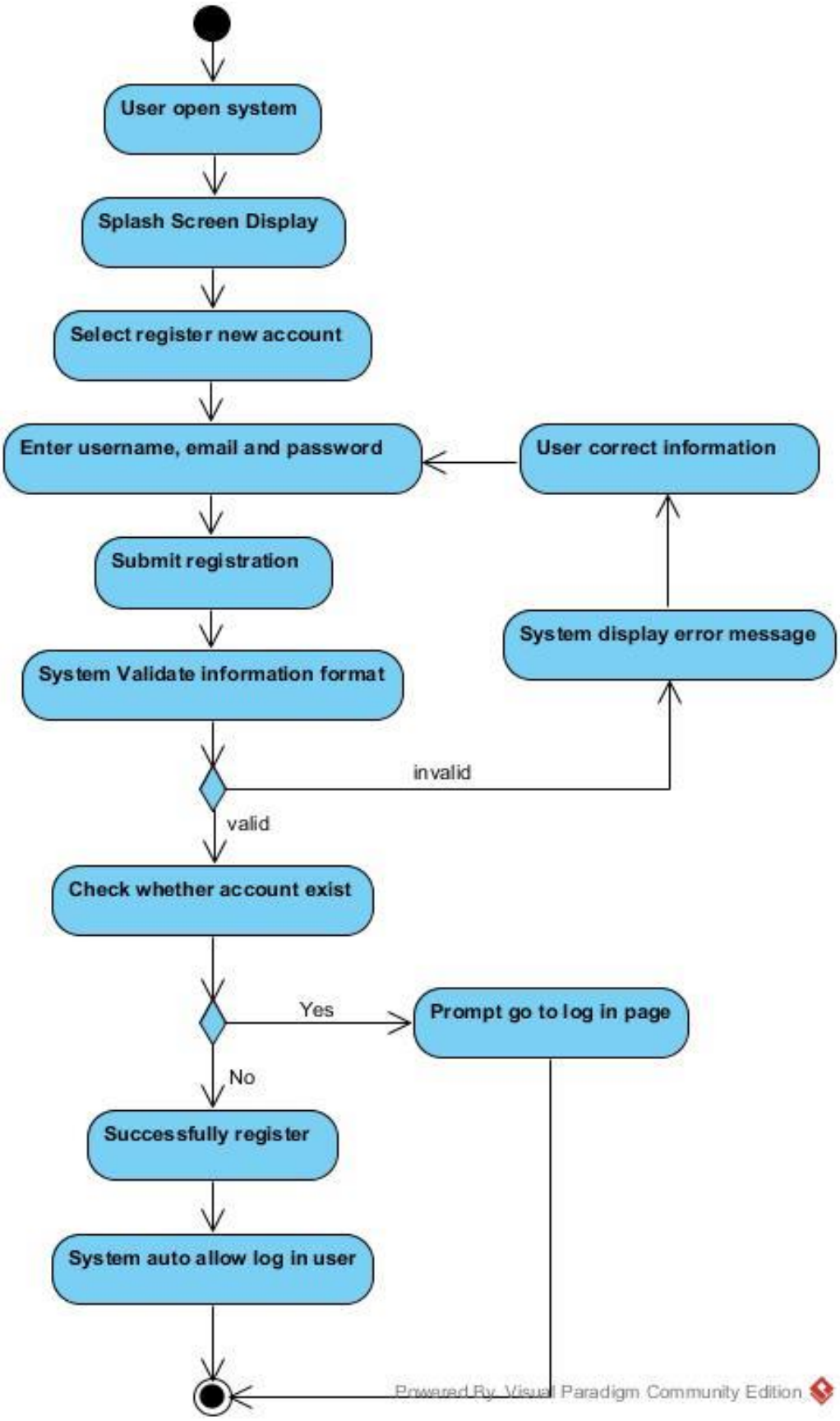



Figure 3-15 Activity Diagram of Registration Module

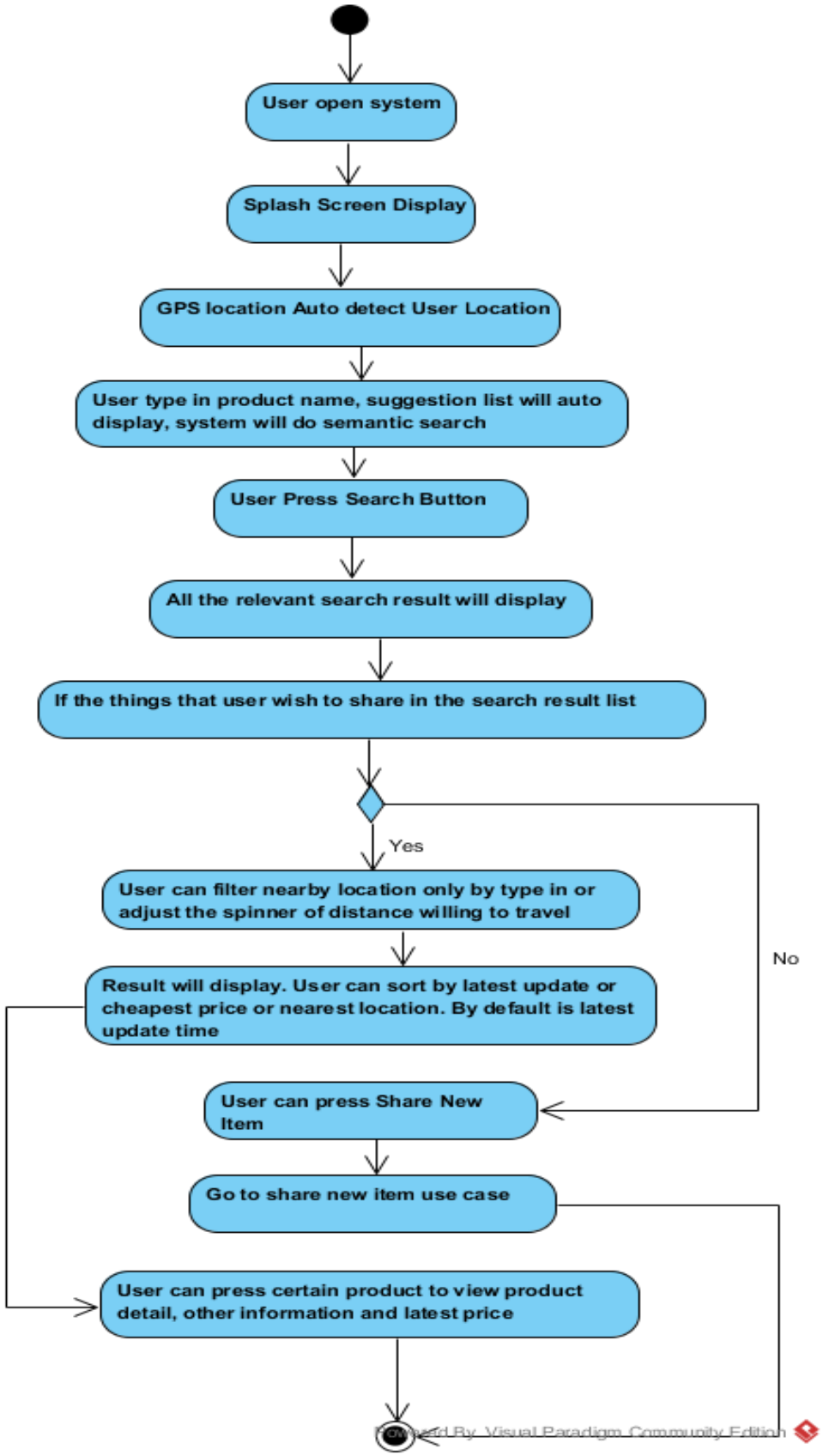



Figure 3-16 Activity Diagram of Search Module

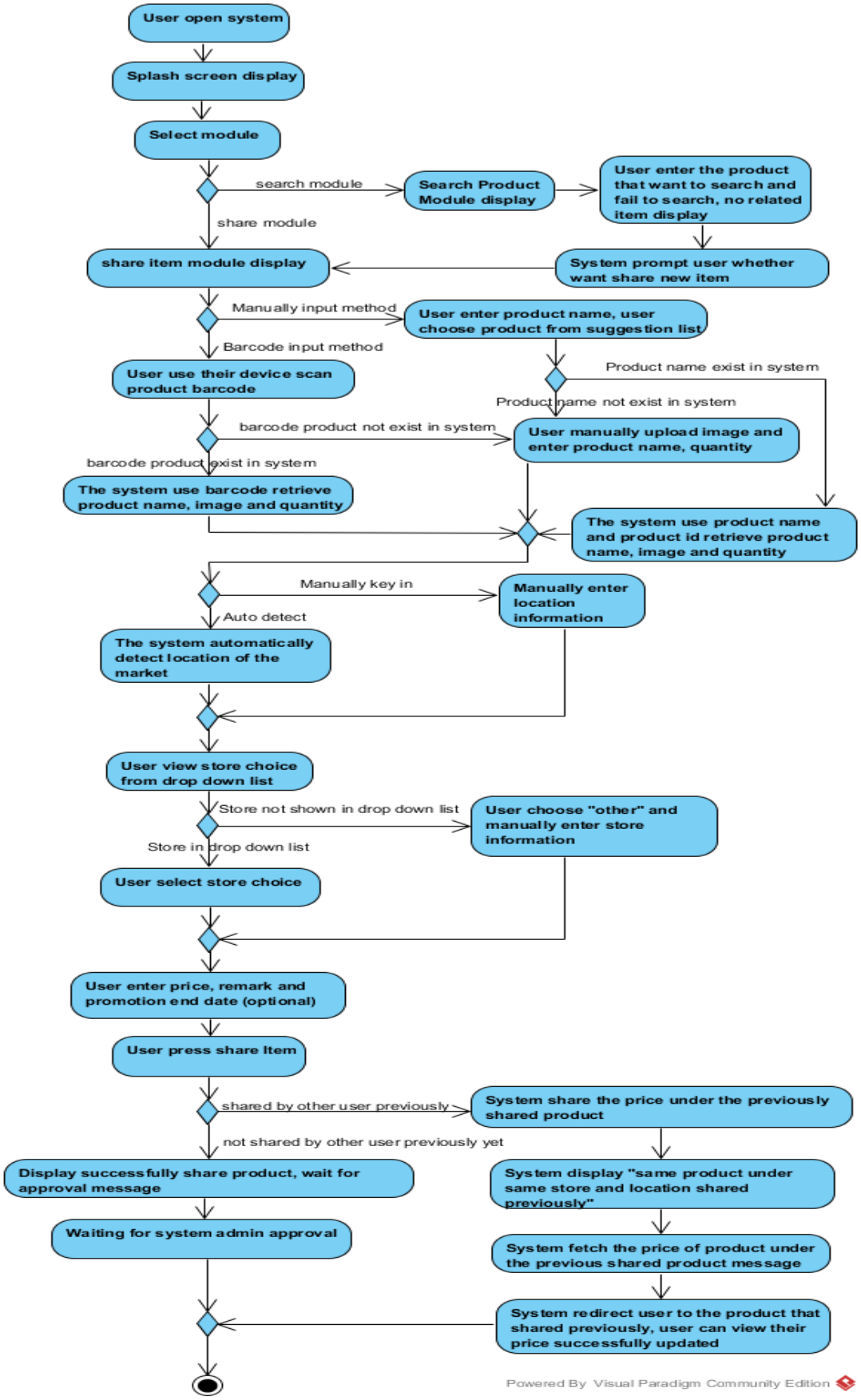



Figure 3-17 Activity Diagram of Share Item Module

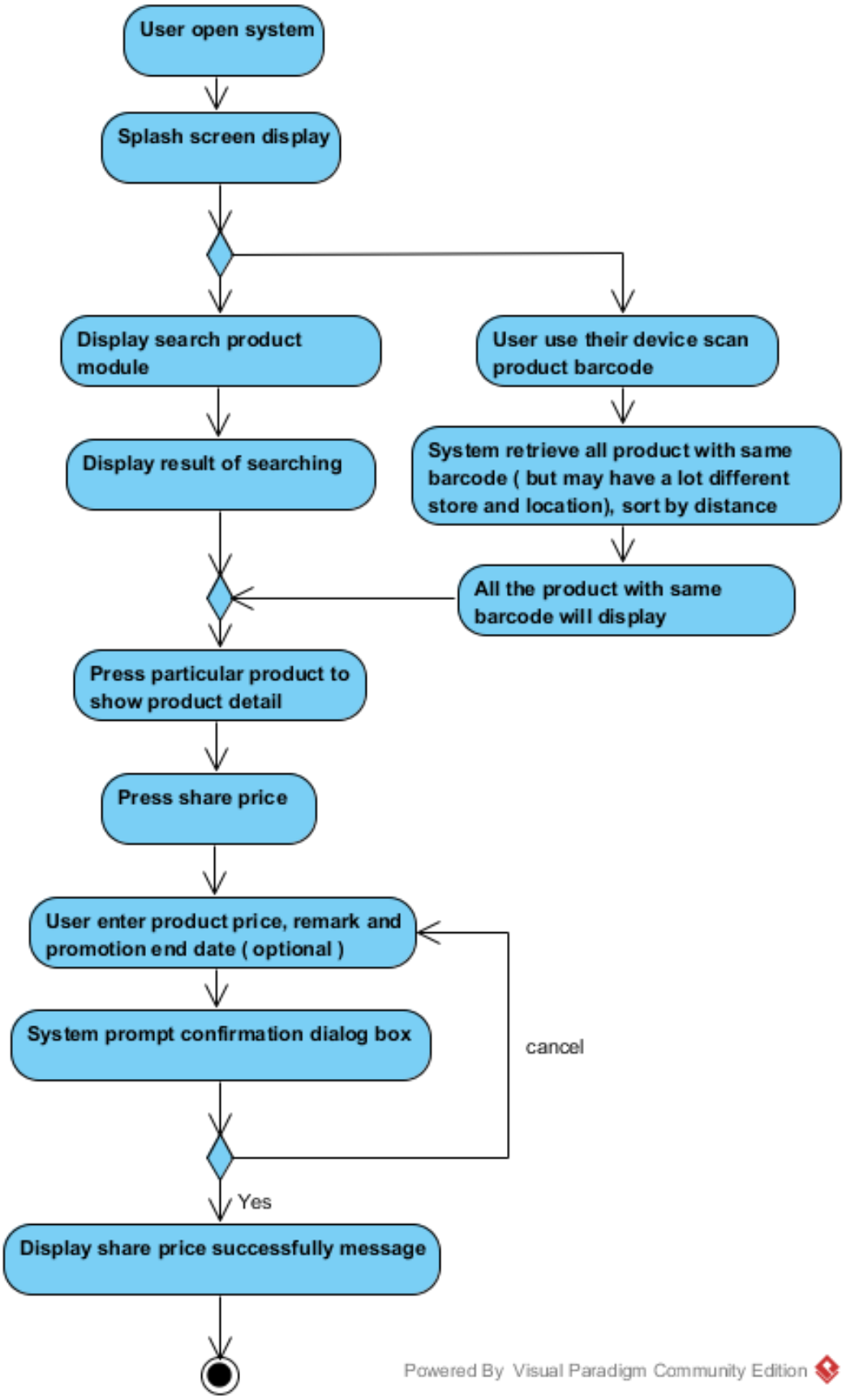



Figure 3-18 Activity Diagram of Share Price Module

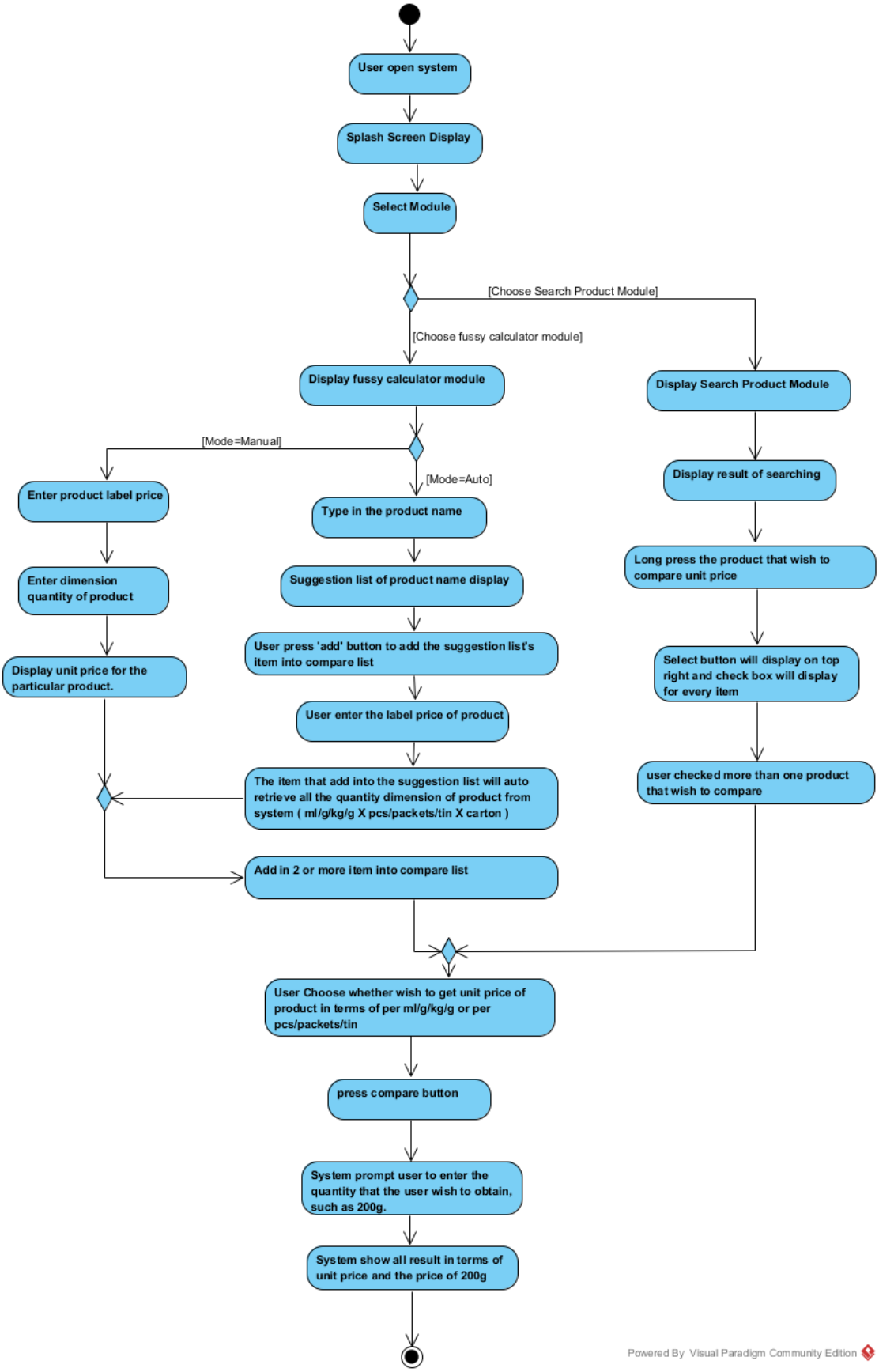



Figure 3-19 Activity Diagram of Fussy Calculator Module

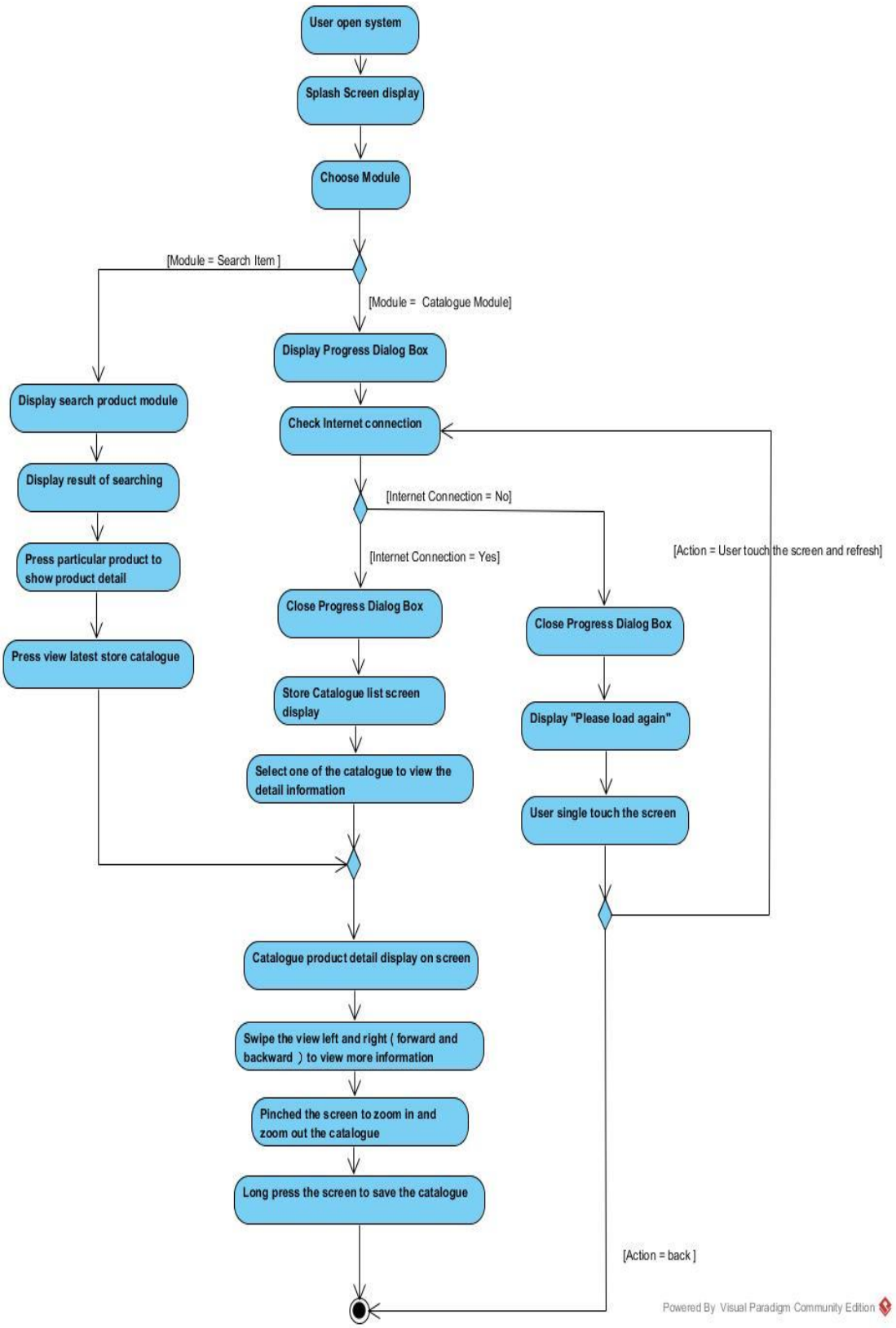



Figure 3-20 Activity Diagram of Catalogue Module

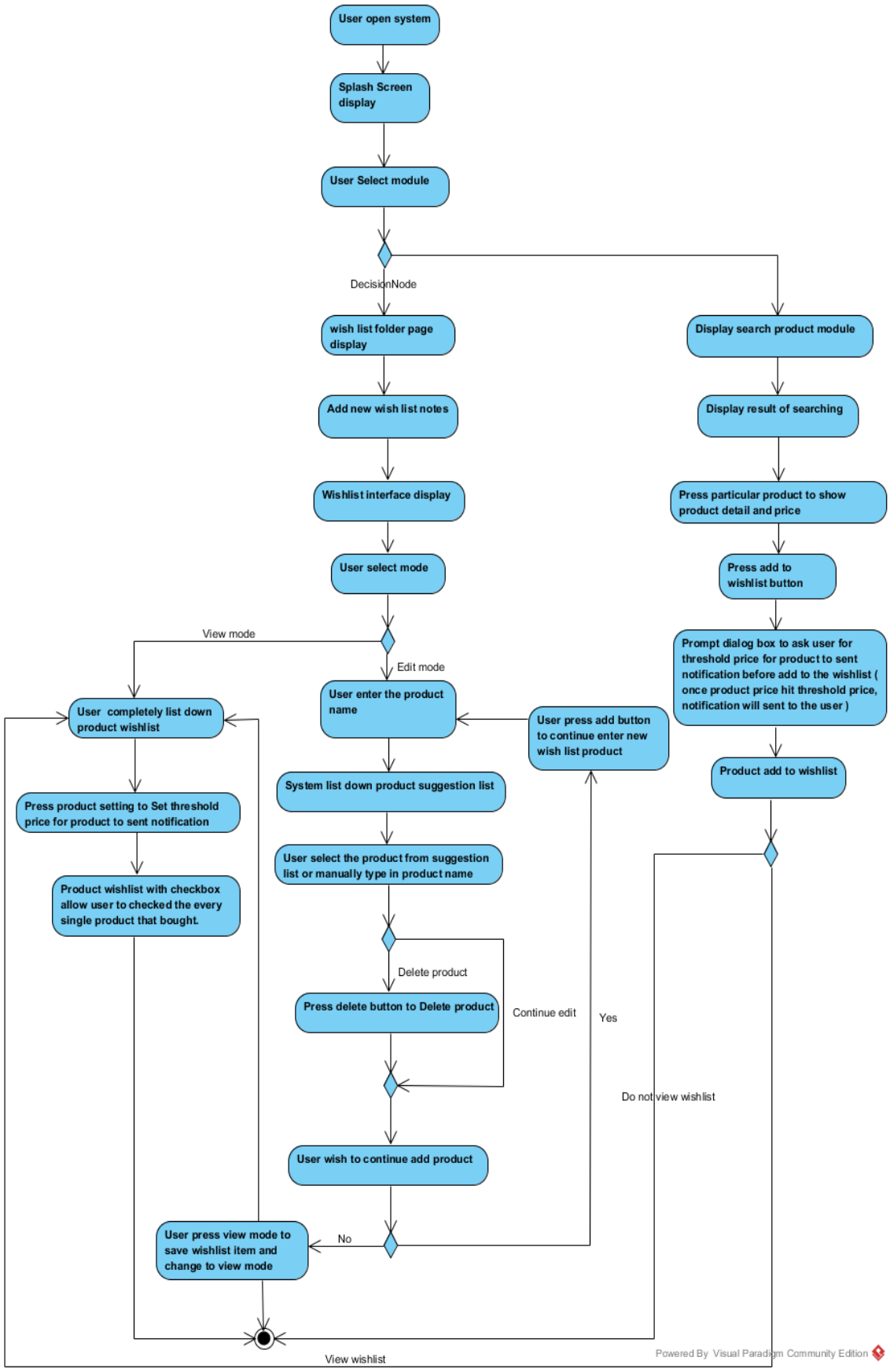



Figure 3-21 Activity Diagram of favorite list Module

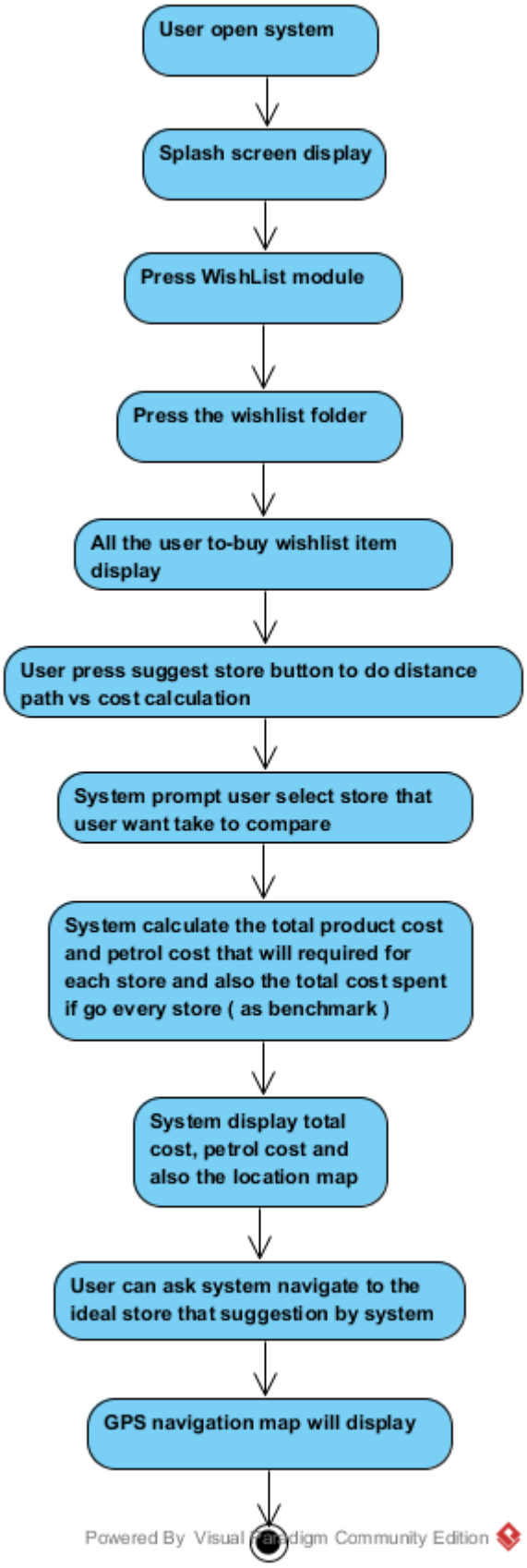



Figure 3-22 Activity Diagram of Ideal Store Suggestion

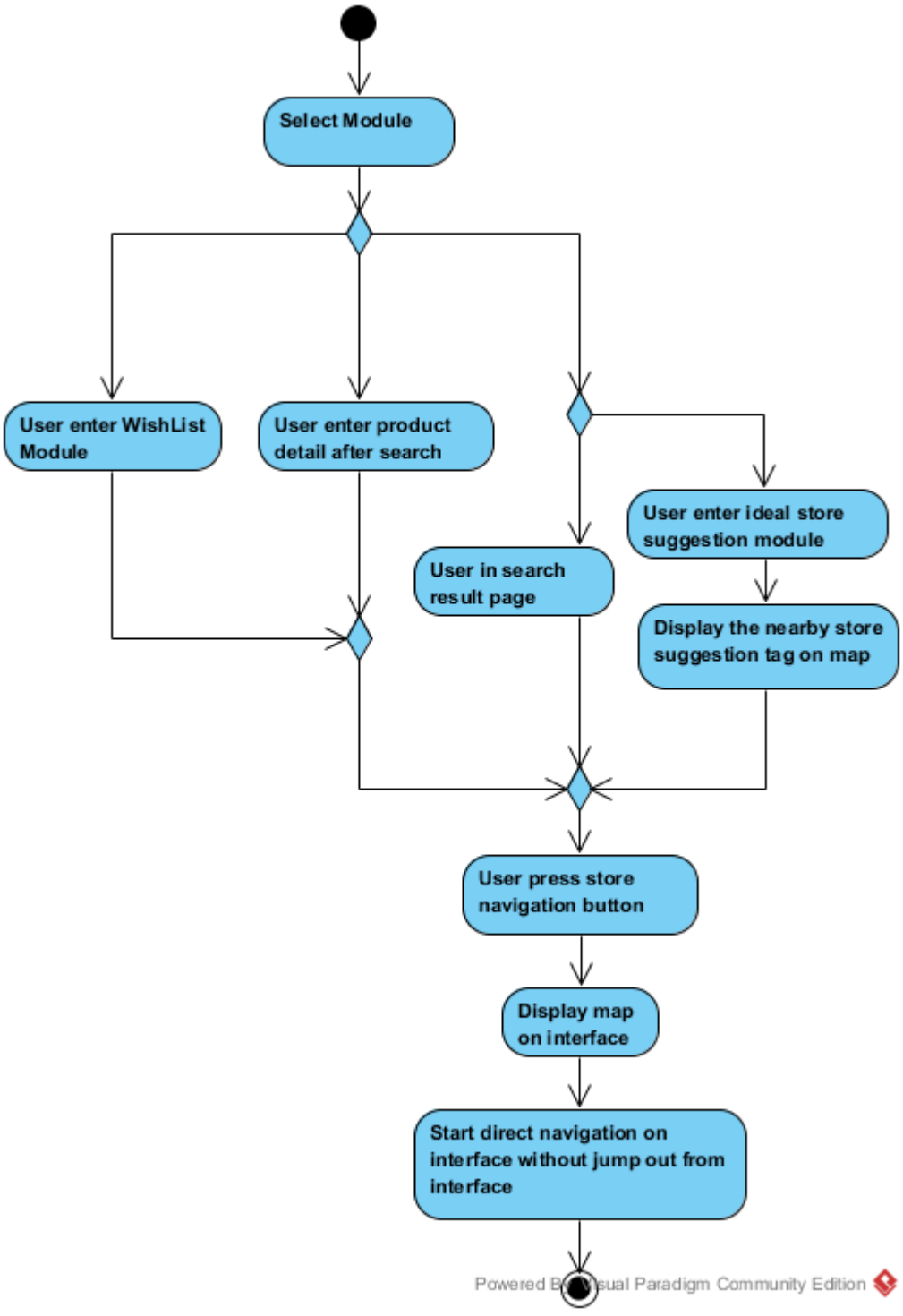



Figure 3-23 Activity Diagram of Location Navigation

### 3.7.3 Class Diagram

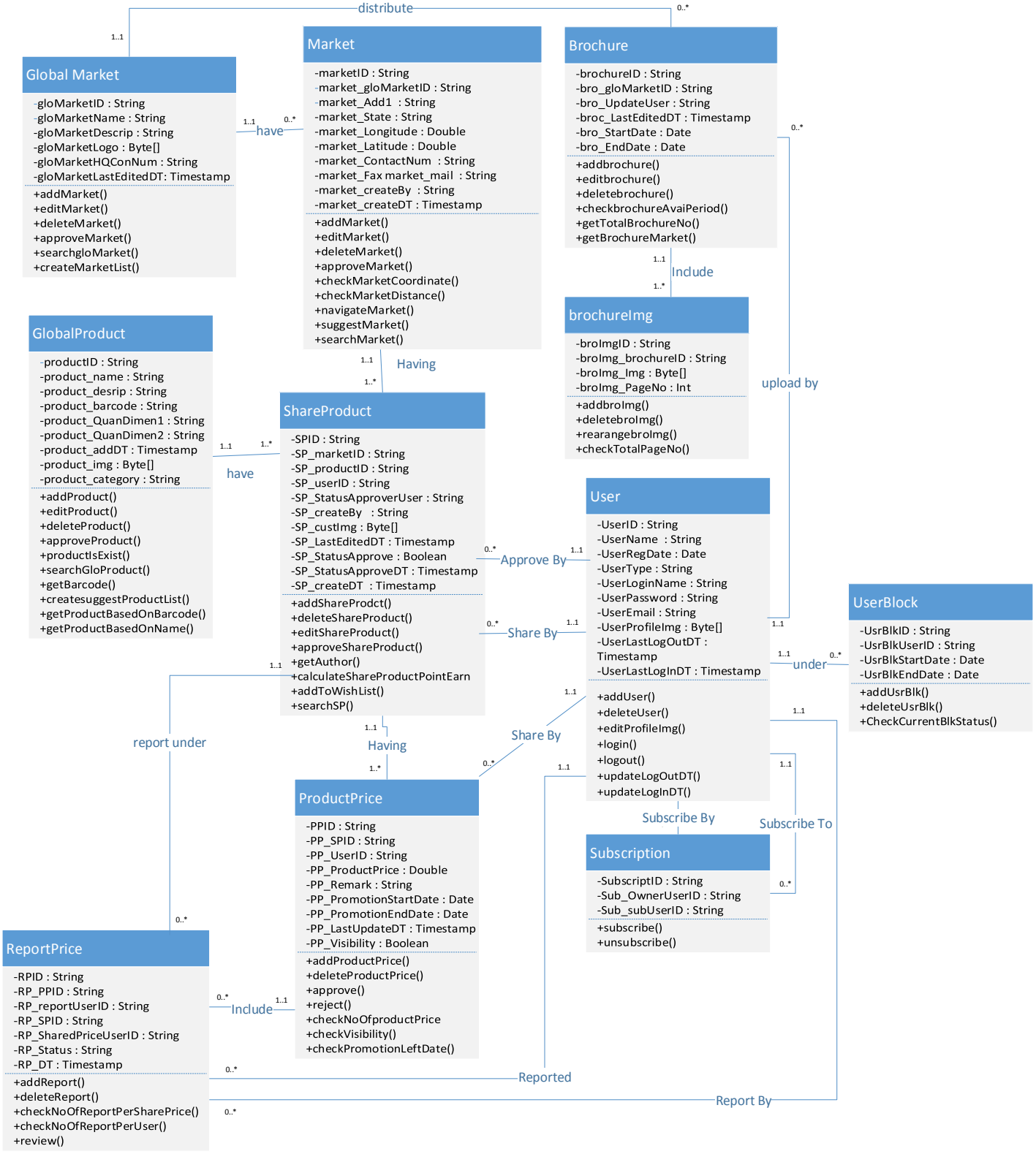



Figure 3-24 Class Diagram for Server Class

Firstly, global market having zero or a lot of market branch throughout whole country and it will distribute more than 1 brochure at any given time but some may not distribute any brochure. Then a brochure normally is distribute periodically (e.g. per week) and it contains many pages together with page number.

In another hand, any global product is uniquely identify by product barcode, can have more than one shared product, which mean user may share particular product that exist in specific market. With this, user can retrieve product information or detail by scanning product. A shared product must be shared under one and only one market but a market can have many shared product.

Besides that, every under shared product, user are allowed to share product price with some description so product price is child table of share product. It mean that every shared product must have at least one or more shared price and share by only one user. Through product detail page, user can subscribe user thus this will be add into subscription table.

If user think that the price shared by particular user is fake, they can press report button and report record will be inserted into report price table. One shared price can be reported by none people or many people but report price is child table or share product price thus any report price must contains foreign key from product price primary key. State and petrol price table is lookup table thus both them are standalone table.

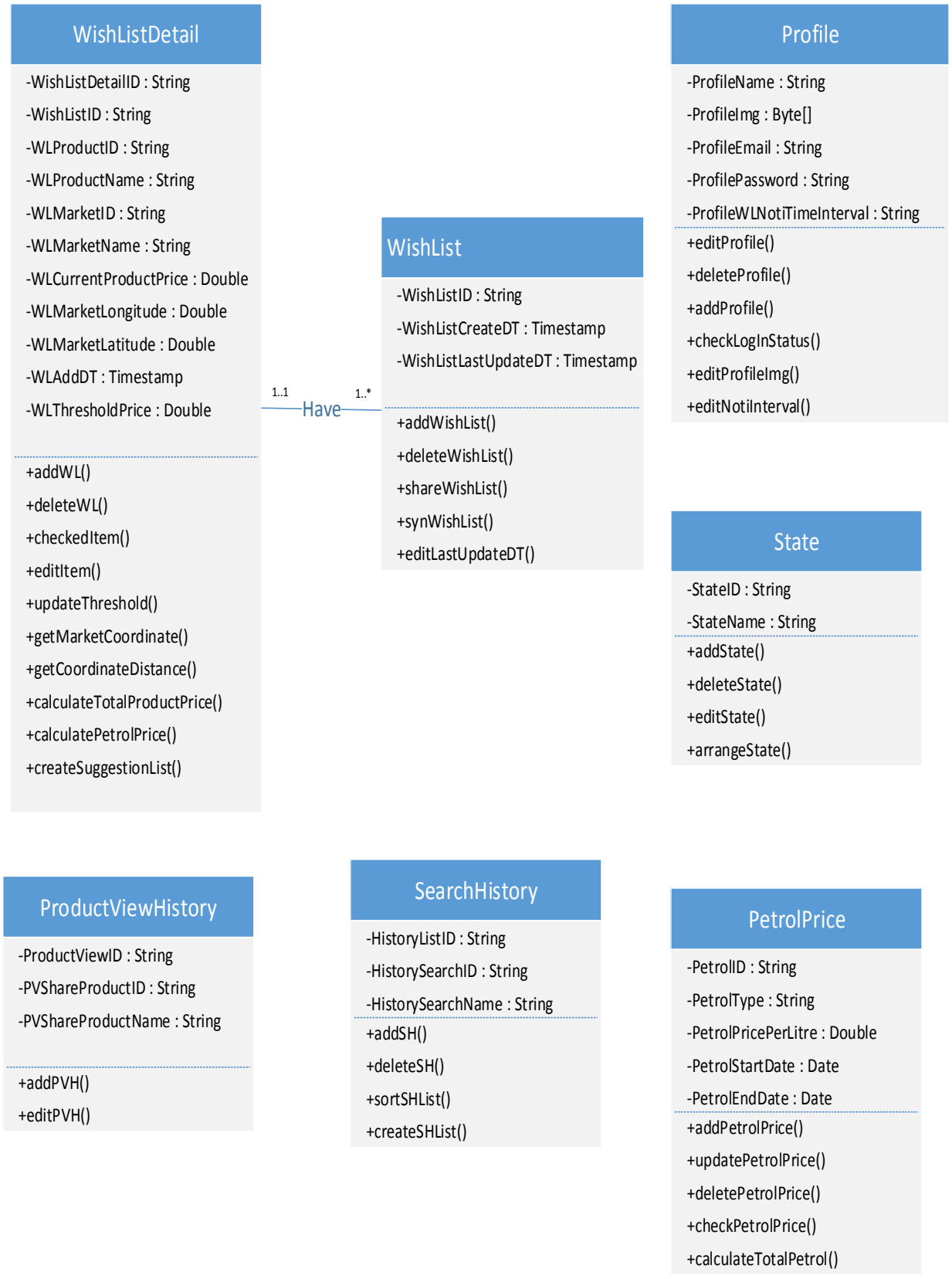



Figure 3-25 Class Diagram for Client Class

### 3.7.4 Object Diagram

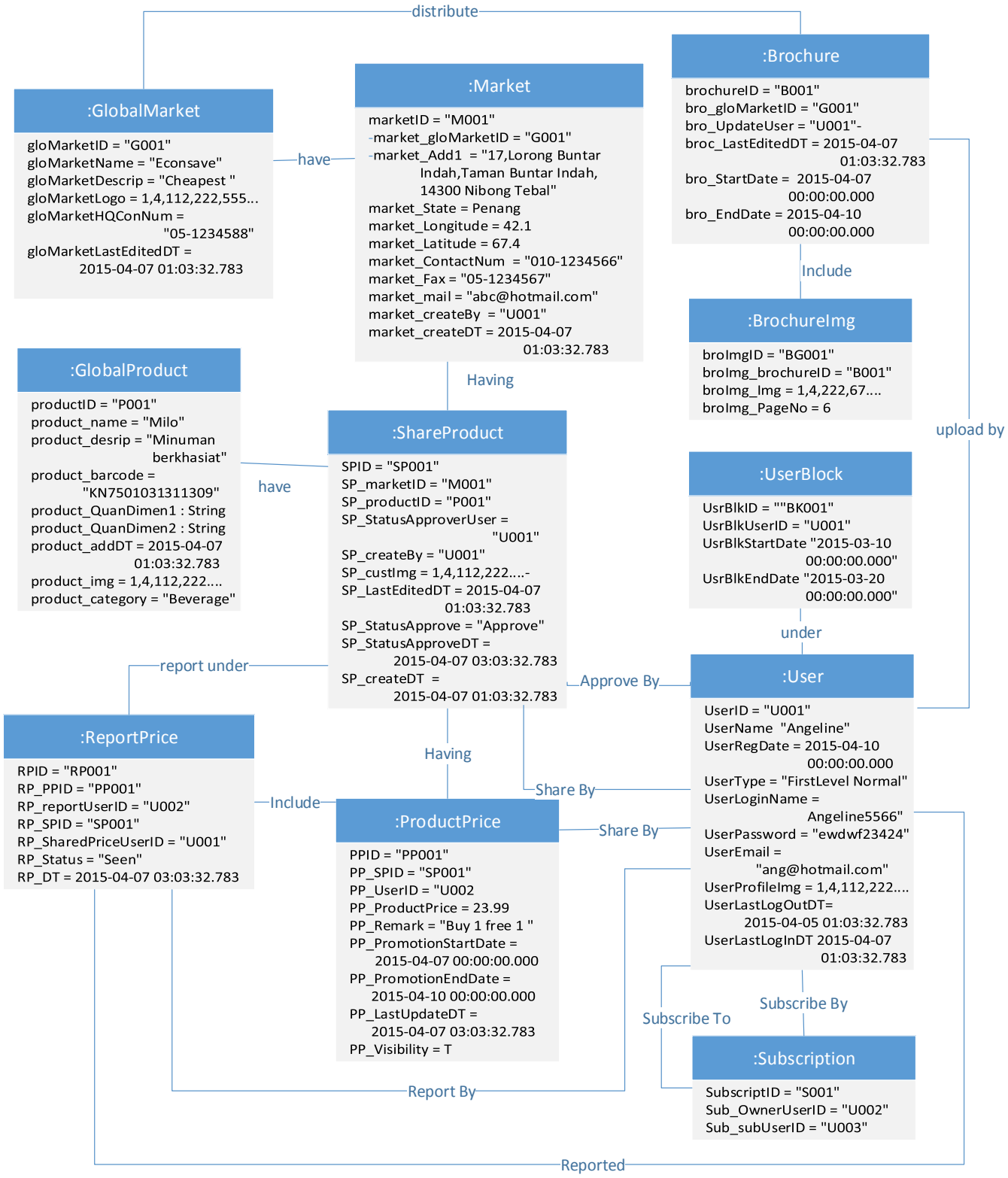



Figure 3-26 Object Diagram for Server Class

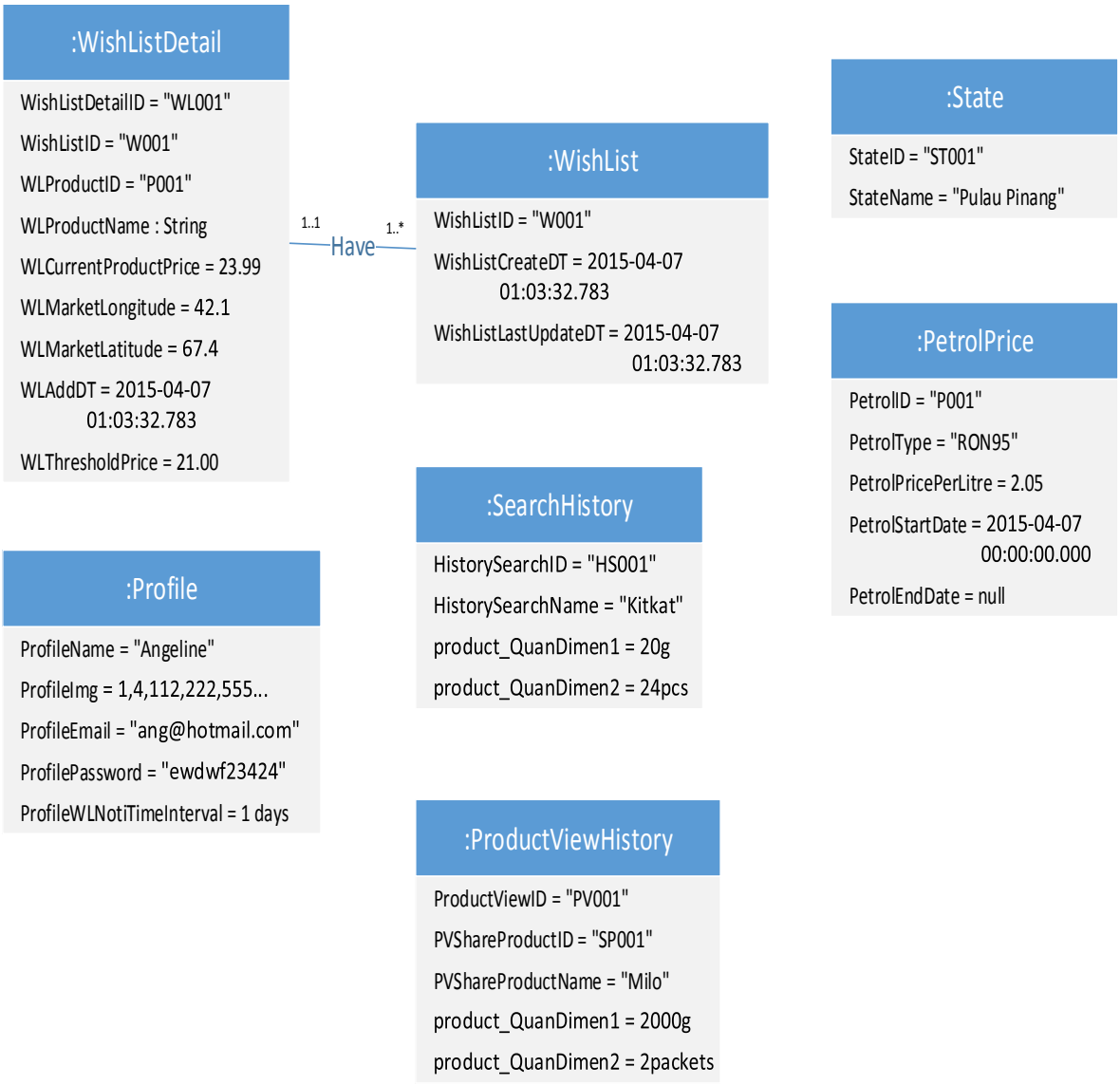



Figure 3-27 Object Diagram for Client Class

# CHAPTER 4 System Design

## 4.1 Chapter Overview

In this chapter, Conawa system design will be show at here. This include the business flow design which required in each module, graphical user interface design, icon and logo design and architectural design that will be use in this system that will incorporate in the system. The system design is carry out after all the system requirement had been gathered.

## 4.2 Graphic Design

### 4.2.1 Icon & Logo 1st draft

First logo design utilize pink color which depict the happiness of shopping are full of blossom.


Figure 4-1 Logo of Conawa 1st draft

### 4.2.2 Icon & Logo 2nd draft

Second logo design turn all functionality into the small icon to let user have better understand what the apps about but when the logo display on device desktop, that is very small relatively and the color too complicated.


Figure 4-2 Icon & Logo 2nd draft

### 4.2.3 Icon & Logo 3rd draft

Third logo emphasize on simple and clean. It keep the happiness pink color from first version and add in addition orange aesthetic color. The main logo still emphasize on shopping bag but the small image contains cart trolley which suit our theme well.


Figure 4-3 Icon & Logo 3rd draft

### 4.3 System Screen Flowchart Diagram

Generally there are 10 module integrate together and become 4 tab. In this subchapter show all the window screen flowchart to let user have overview concept about the system flow and in chapter 4.4 User Interface Design will show all the screen flow with application screenshot. The detail business flow and screen flow will be further discuss and elaborate in chapter 4.5 which is Application Screen Flow with Classes and Methods.

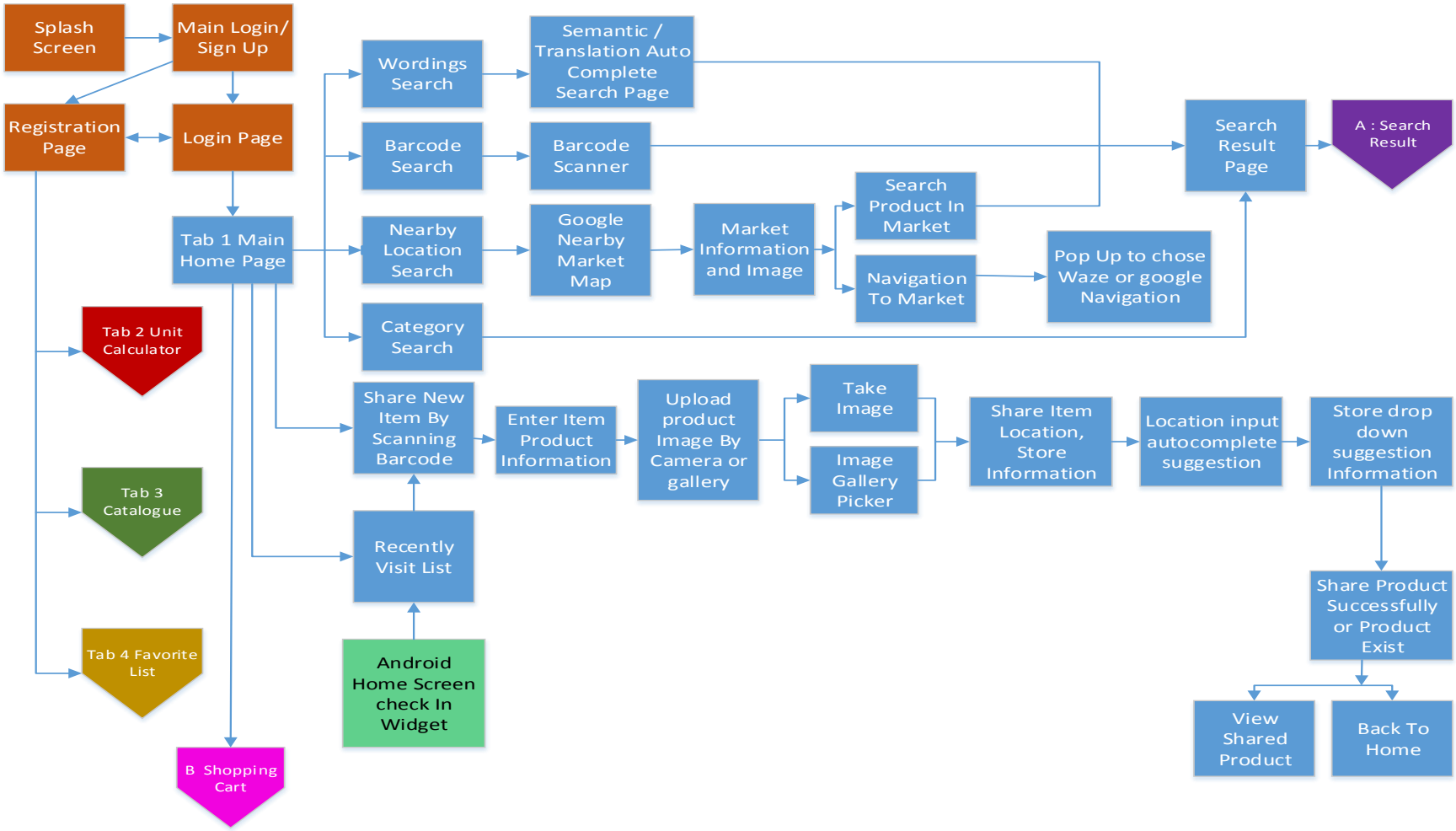



Figure 4-4 Flowchart 1 (Main Home Page and Search Module)

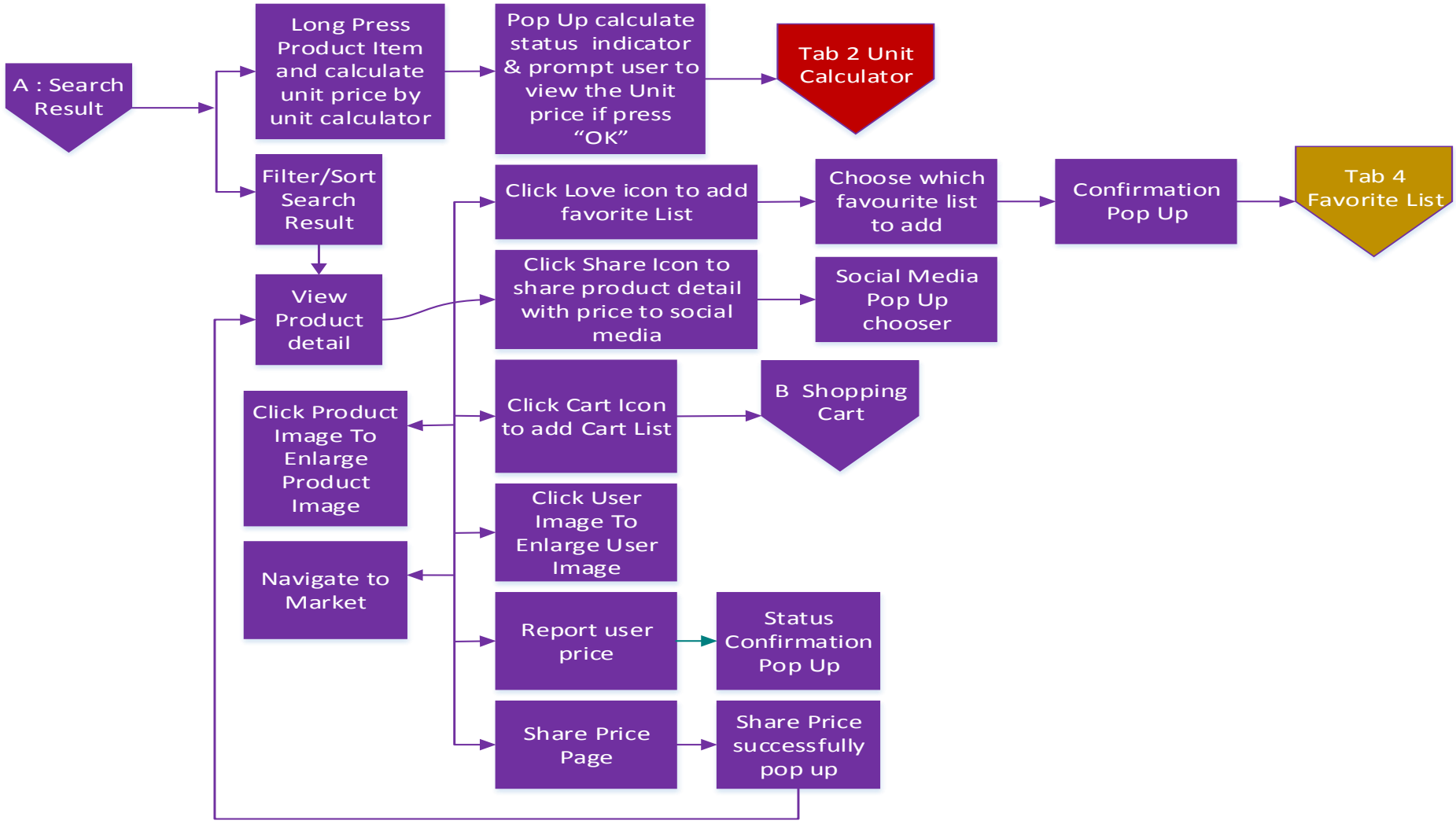



Figure 4-5 Flowchart 2(Search Result Module)

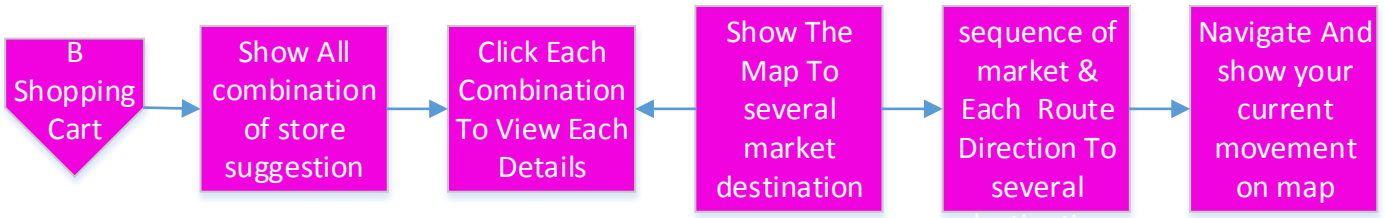



Figure 4-6 Flowchart 3(Shopping Cart Module)

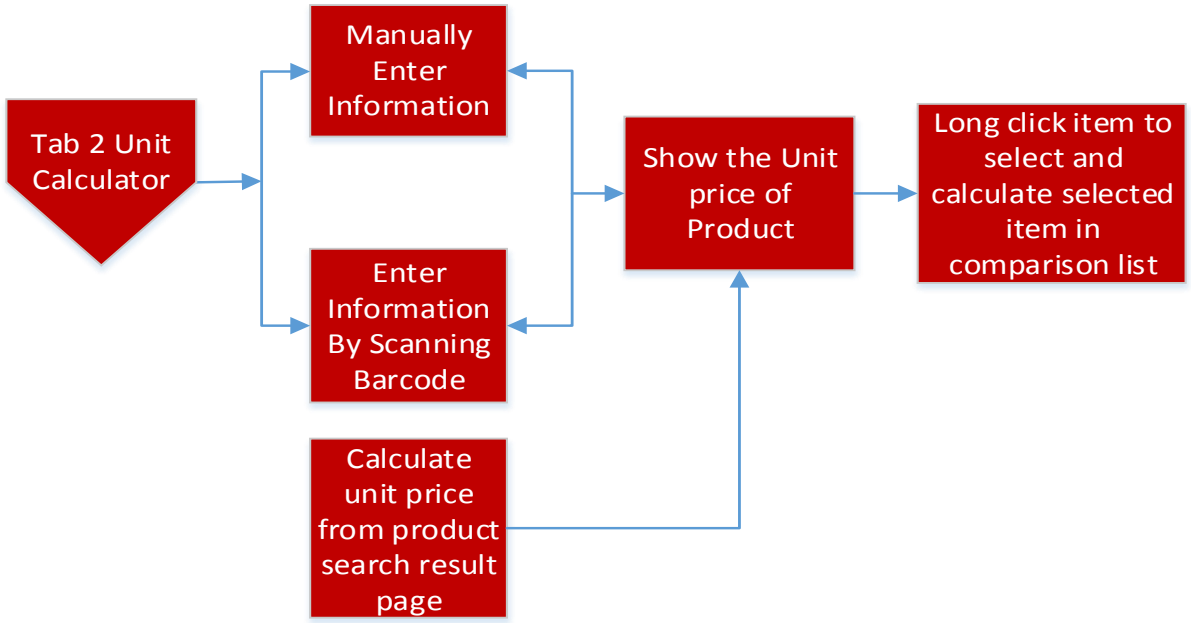



Figure 4-7 Flowchart 4(Unit Calculator Module)

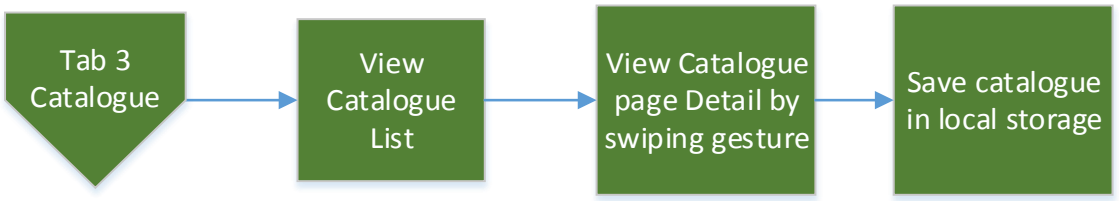



Figure 4-8 Flowchart 5(Catalogue Module)

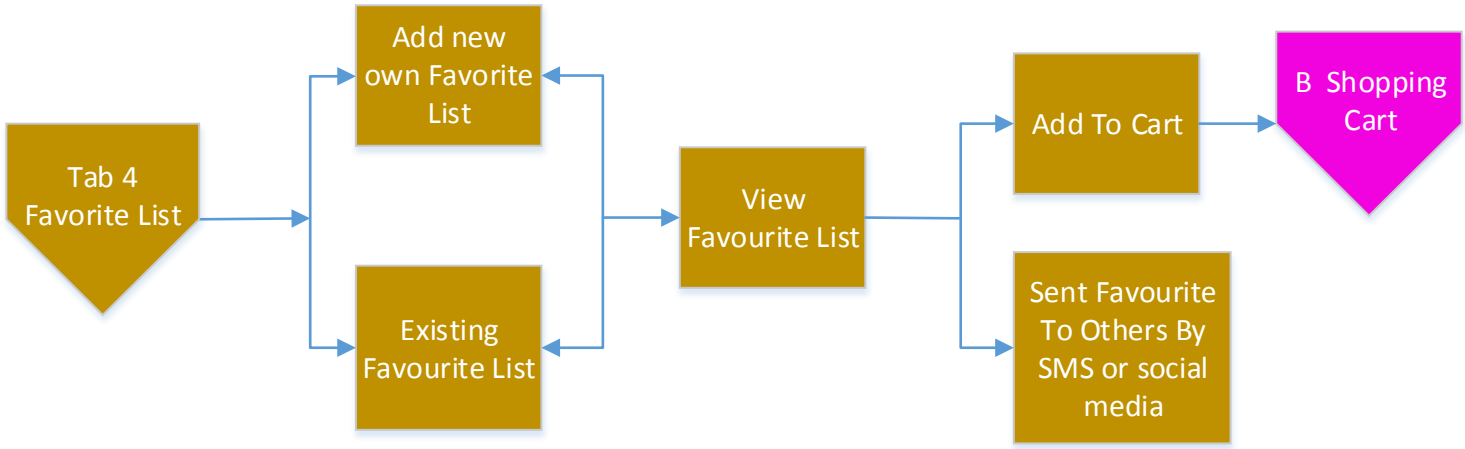



Figure 4-9 Flowchart 6(Favorite List Module)

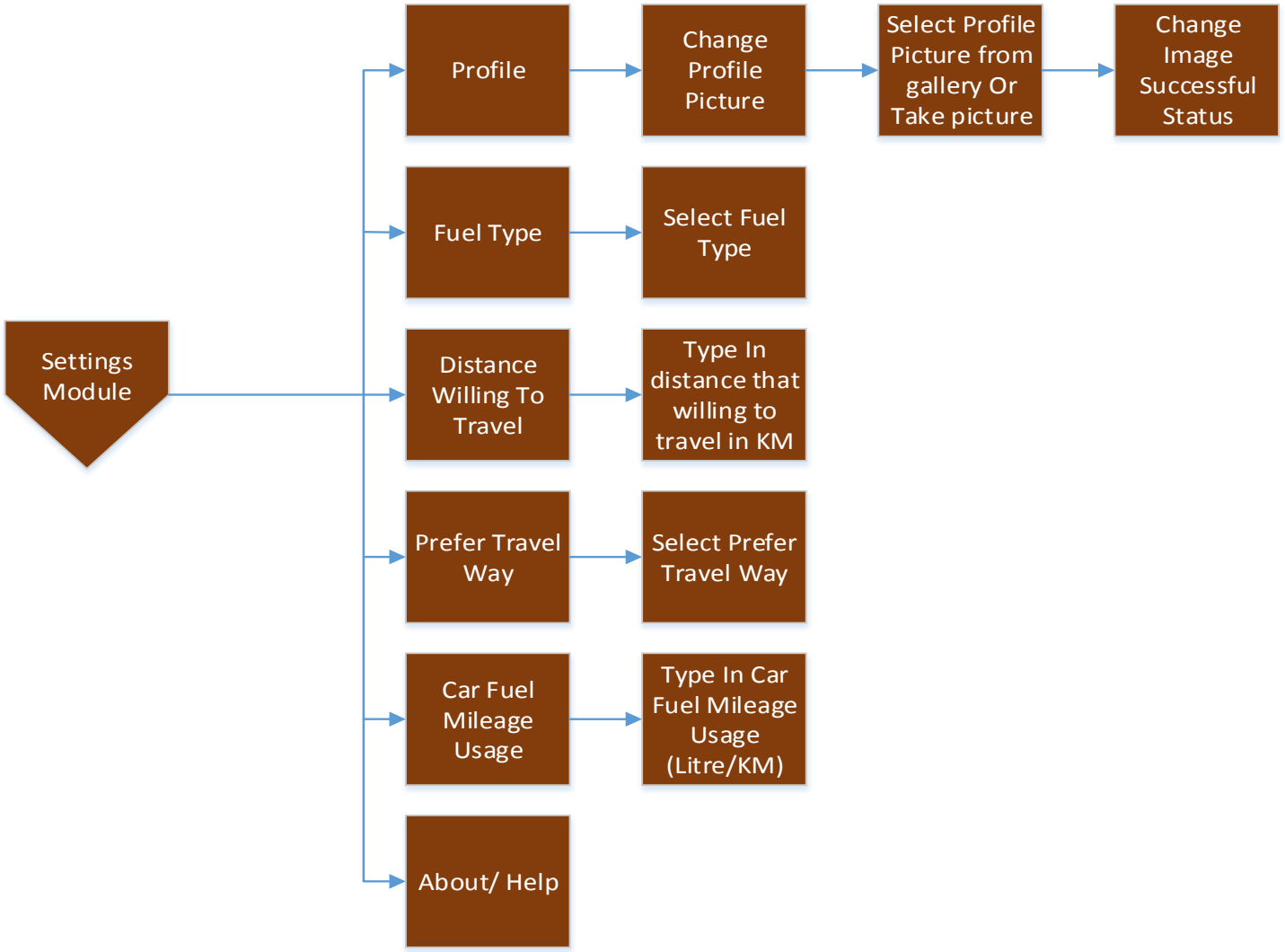



Figure 4-10 Flowchart 7(Settings Module)

### 4.4 User Interface Design

The picture below demonstrate the basic draft and mock of some module in system.

The screen below act as story board and arrange according to business flow.

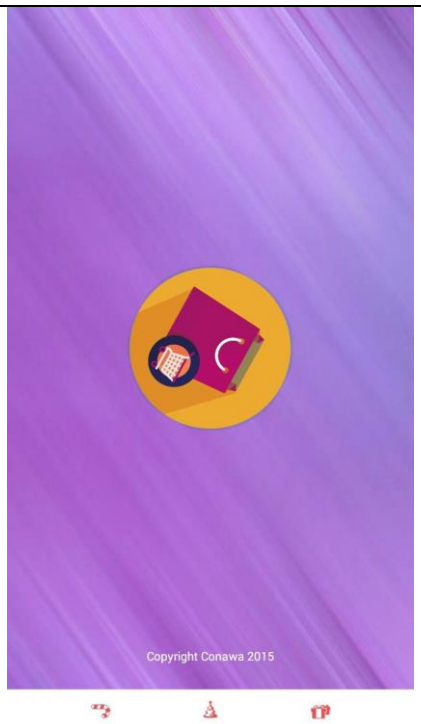
<table>
<tr>
<td> 

Figure 4-11 Splash Screen</td>
<td> 

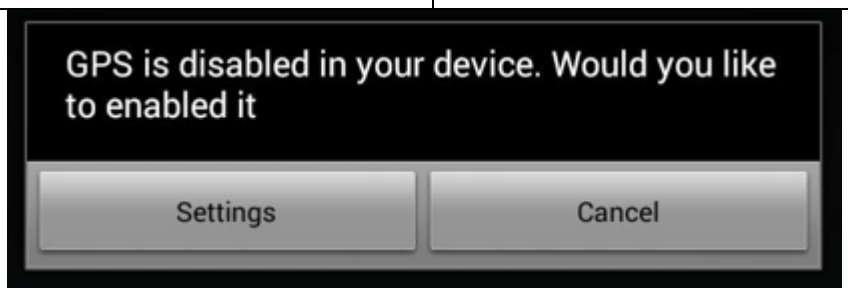
Figure 4-12 Main Sign In /Sign Up</td>
</tr>
<tr>
<td>Splash Screen</td>
<td>Main Sign In /Sign Up</td>
</tr>
<tr>
<td colspan="2"> 

Figure 4-13 GPS Enabler Prompt</td>
</tr>
<tr>
<td colspan="2">Prompt user to open GPS signal if system can’t access to location information</td>
</tr>
</table>

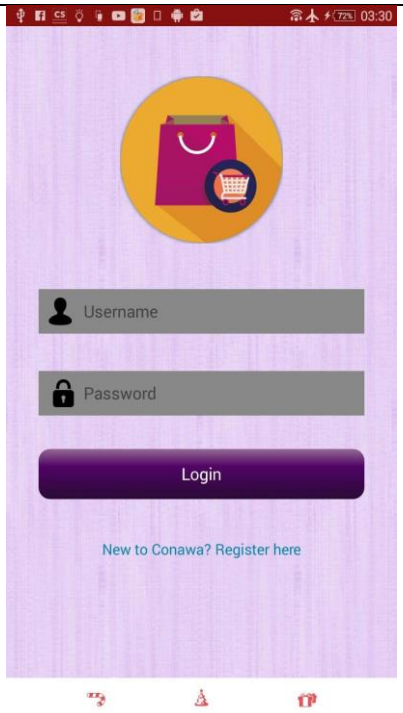



Figure 4-14 Log In Page

Log In Page

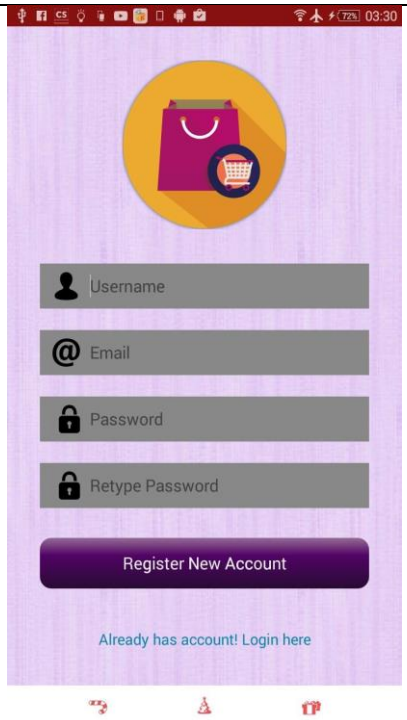



Figure 4-15 Register Page

Register Page


Figure 4-16 Main Home Tab

Main Home Tab


Figure 4-17 Main Home Tab 2

Main Home Tab 2

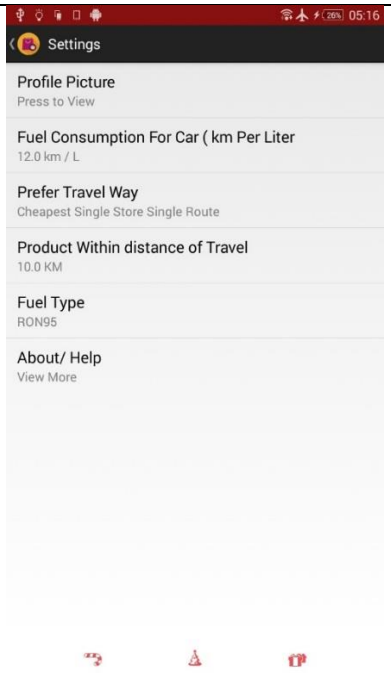
<table>
<tr>
<td> 

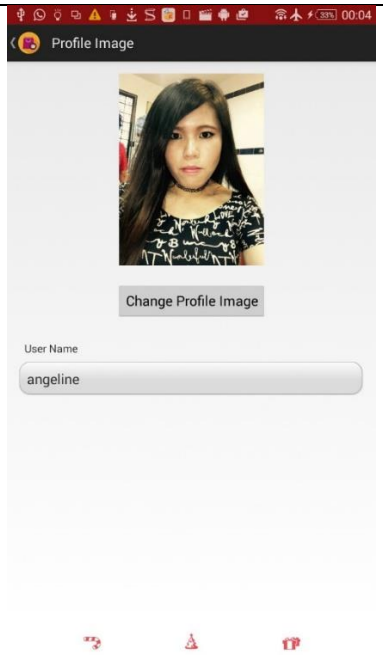
Figure 4-18 Settings Page</td>
<td> 

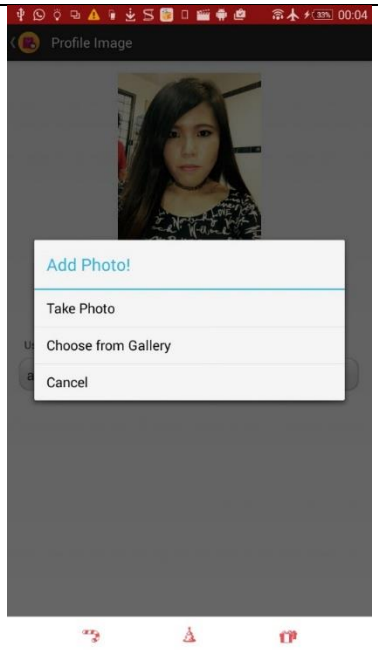
Figure 4-19 Profile Image and name settings Page</td>
</tr>
<tr>
<td>Settings Page</td>
<td>Profile Image and name settings Page, user can change profile image.</td>
</tr>
<tr>
<td> 

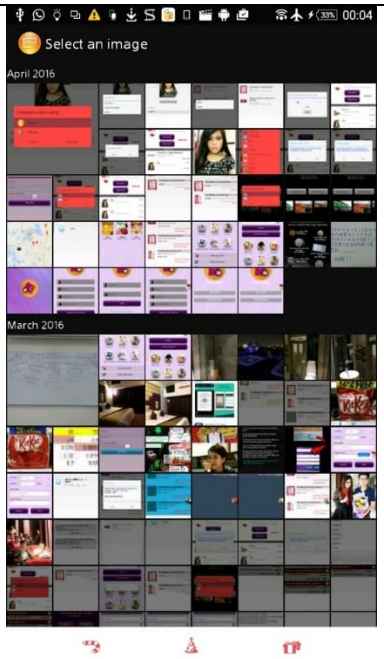
Figure 4-20 Change Profile Image pop up</td>
<td> 

Figure 4-21 Gallery Picker</td>
</tr>
<tr>
<td>User can Change profile image by take photo or from gallery.</td>
<td>If user choose gallery choice, gallery picker will display.</td>
</tr>
</table>

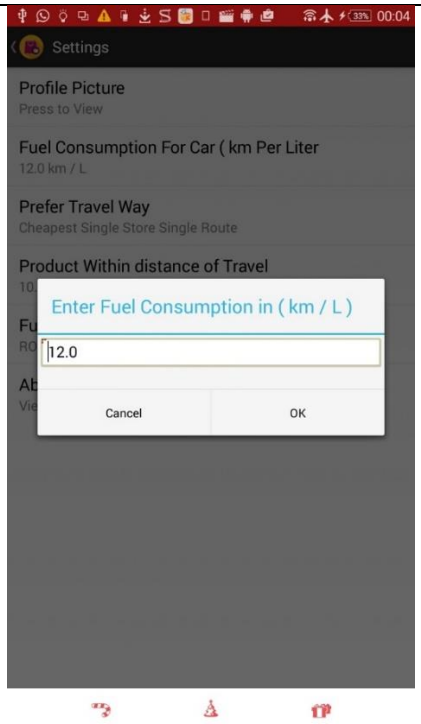



Figure 4-22 Fuel Consumption Settings

Fuel Consumption Settings

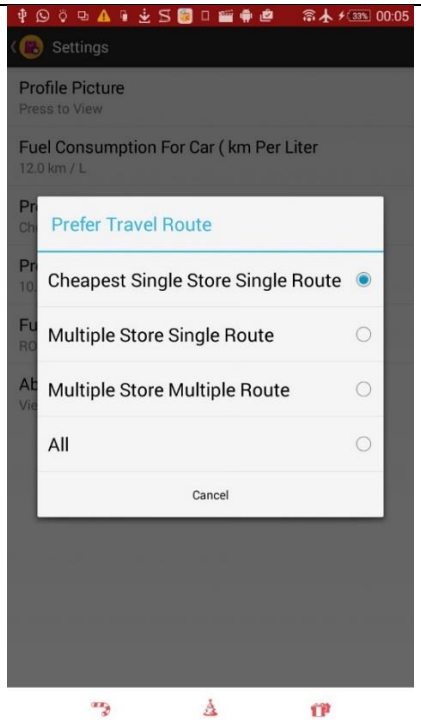



Figure 4-23 Prefer Travel Route Settings

Prefer Travel Route Settings

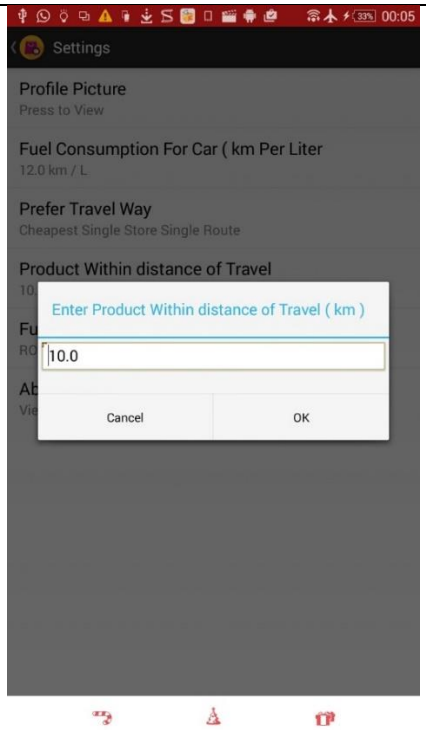



Figure 4-24 Distance Travel Settings

Settings for maximum distance user willing to travel

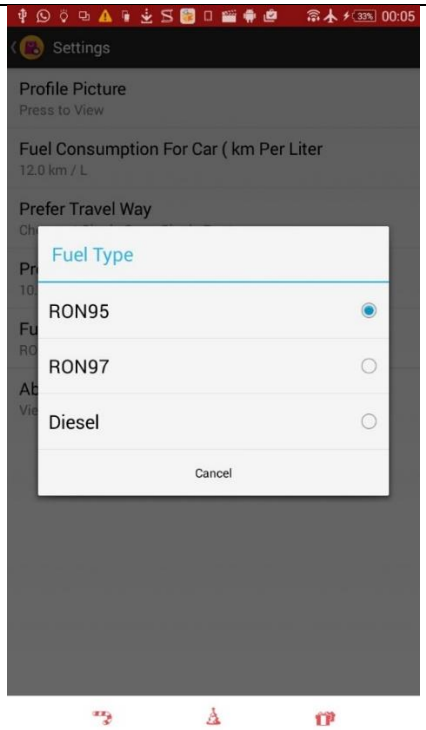



Figure 4-25 Fuel type Settings

Settings for fuel type

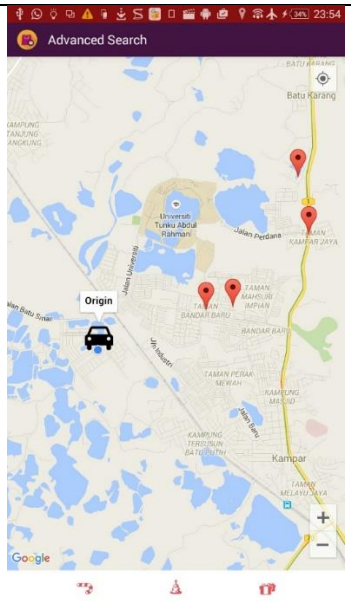
<table>
<tr>
<td> 

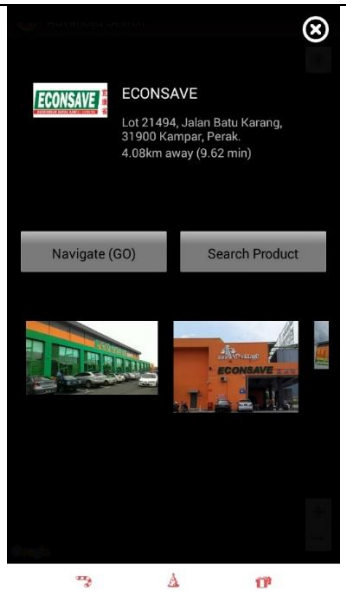
Figure 4-26 Advanced Search Maps</td>
<td> 

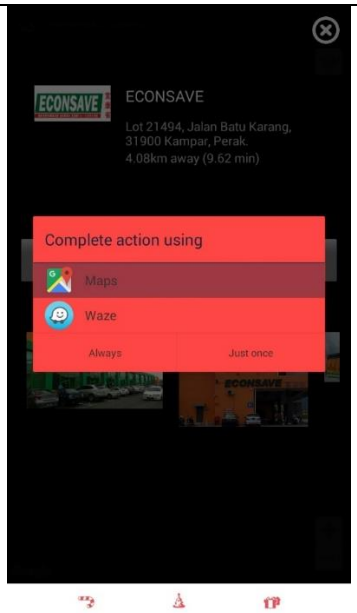
Figure 4-27 Market Information</td>
</tr>
<tr>
<td>User can click on marker in Advanced Search Maps and a partially black transparent activity will display, show in next figure.</td>
<td>Market Information when user click on marker</td>
</tr>
<tr>
<td> 

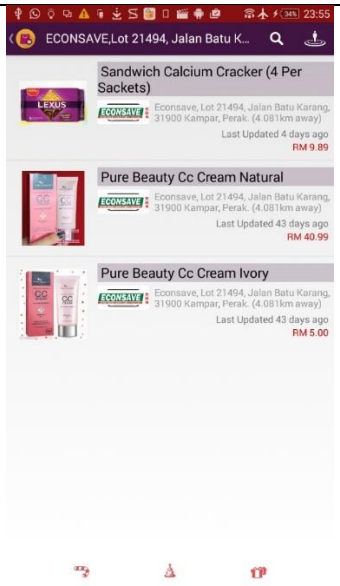
Figure 4-28 Market navigation from market information</td>
<td> 

Figure 4-29 Search Product Result in particular market</td>
</tr>
<tr>
<td>Market navigation from market information</td>
<td>Search Product Result in particular market by clicking search product button in market information page</td>
</tr>
</table>

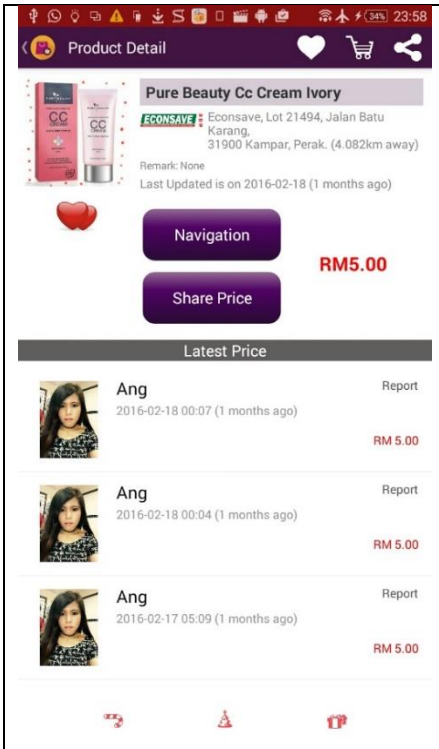



Figure 4-30 Product Detail Page

Product Detail Page

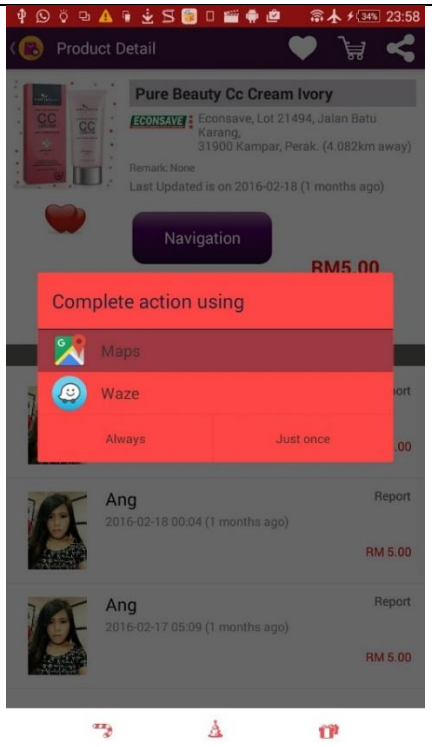



Figure 4-31 Navigation to market

Navigation to market

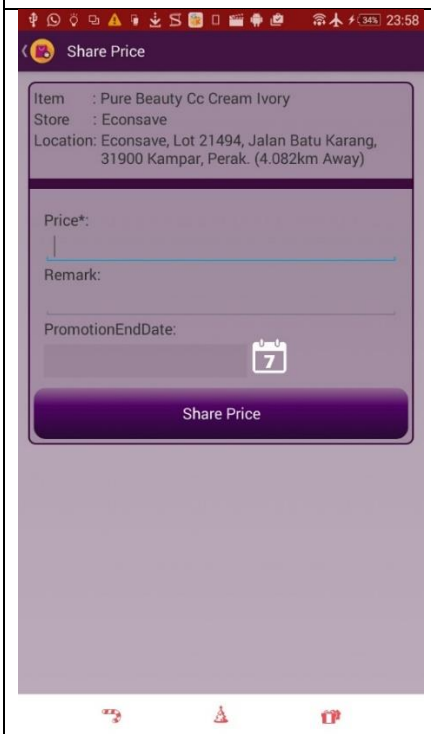



Figure 4-32 Share Price

Share Product Price

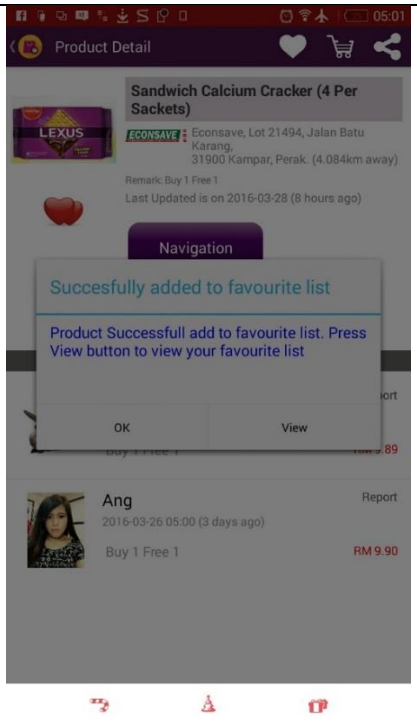
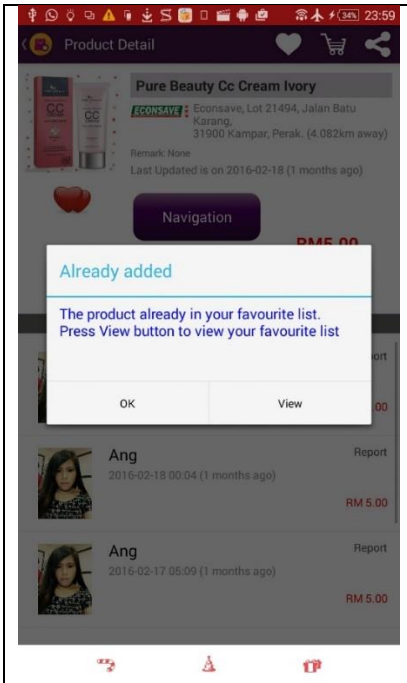
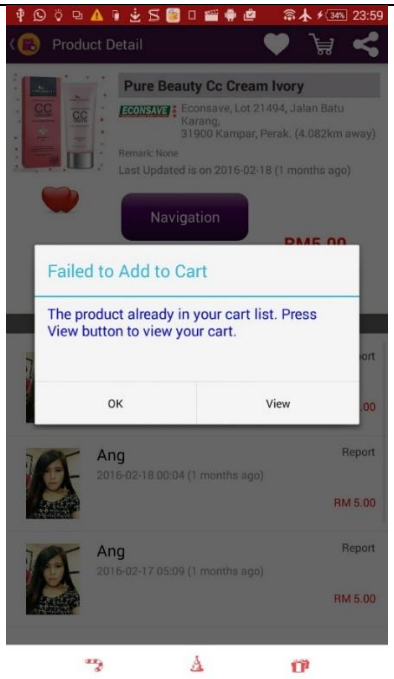
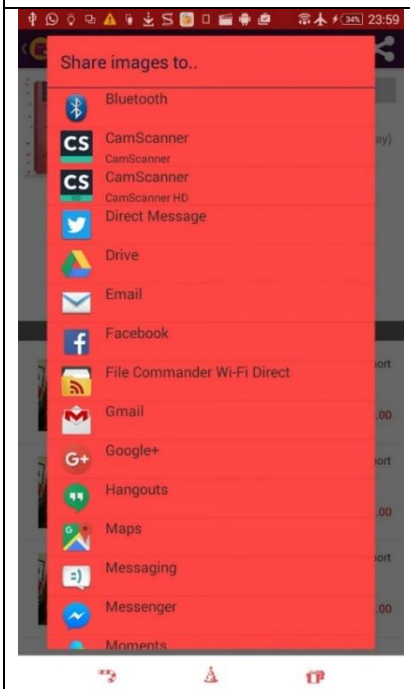
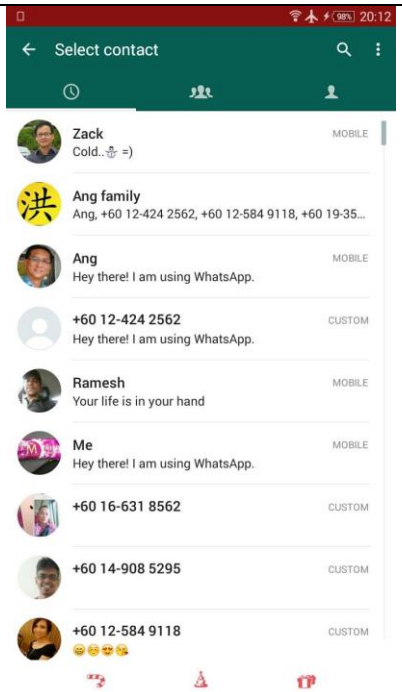



Figure 4-33 Add to favorite list

Add to favorite list

<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Figure 4-34 Product Already in Favorite List</td>
<td>Figure 4-35 Add To Cart</td>
</tr>
<tr>
<td>If product existed in favorite list, the pop up will display as show above</td>
<td>Add Product To Cart</td>
</tr>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Figure 4-36 Share social media</td>
<td>Figure 4-37 Whatsapp chooser</td>
</tr>
<tr>
<td>Share Product to social media</td>
<td>Whatsapp chooser</td>
</tr>
</table>

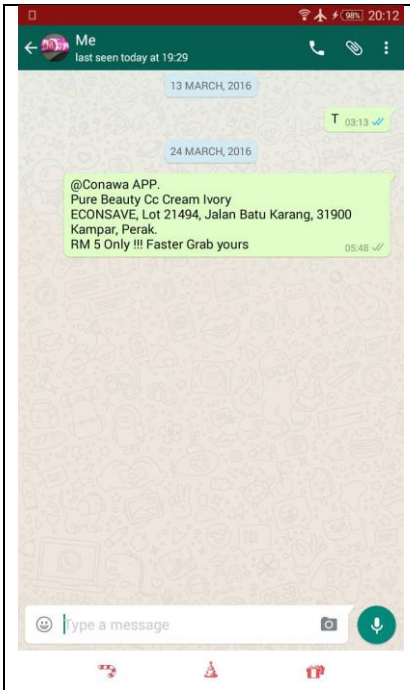



Figure 4-38 Successfully share to whatsapp

Successfully share to whatsapp

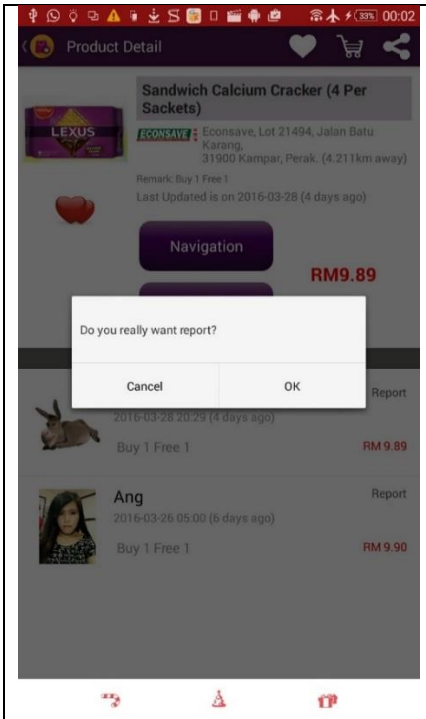



Figure 4-39 Report Price pop up

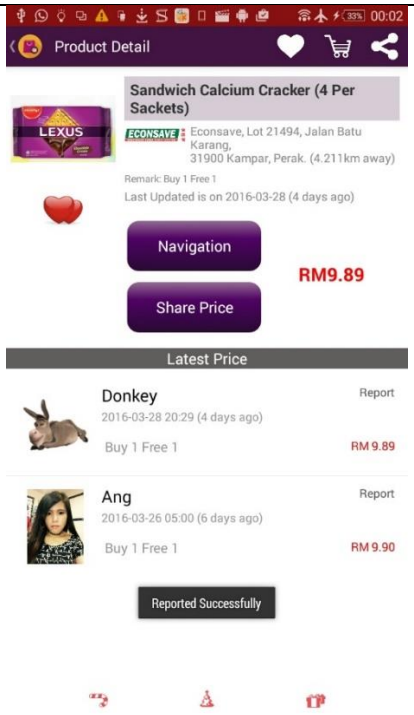



Figure 4-40 Report Price Successfully

| Report Price | Report Price Successfully |
|---|---|

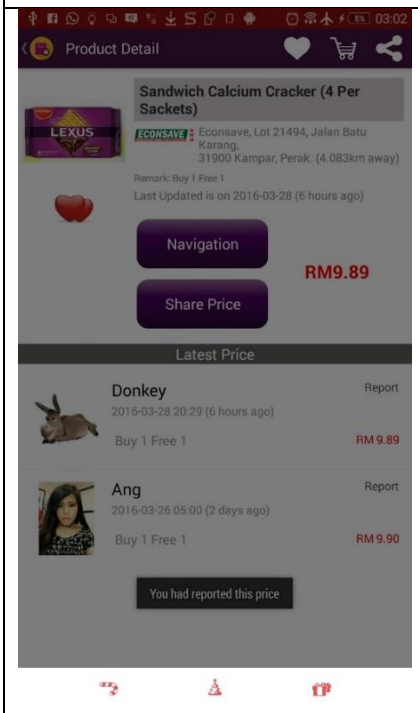



Figure 4-41 Reported Price, Fail

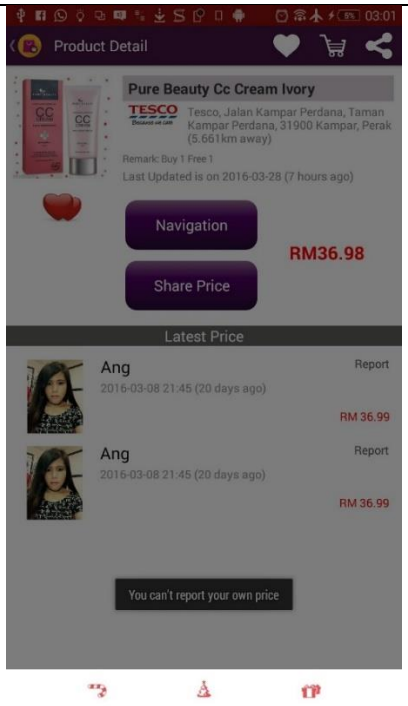
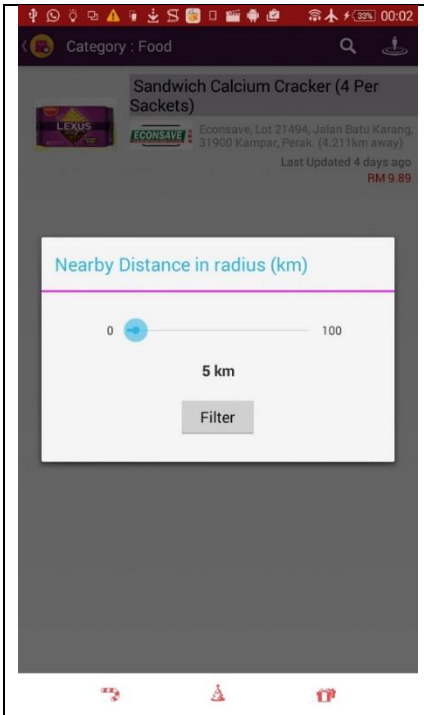
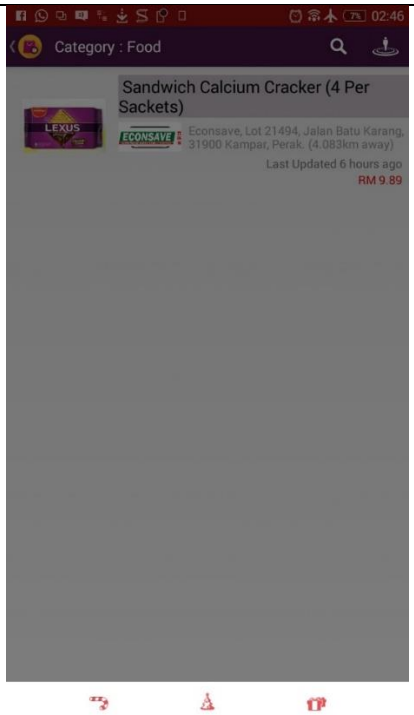
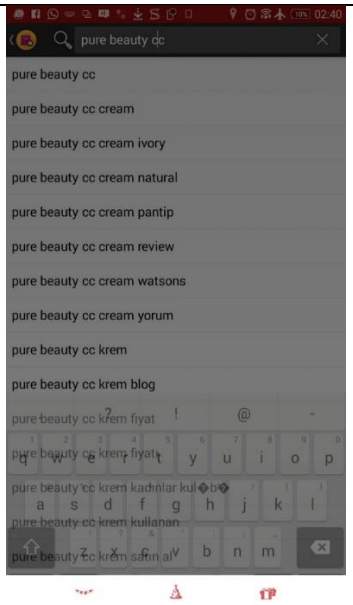
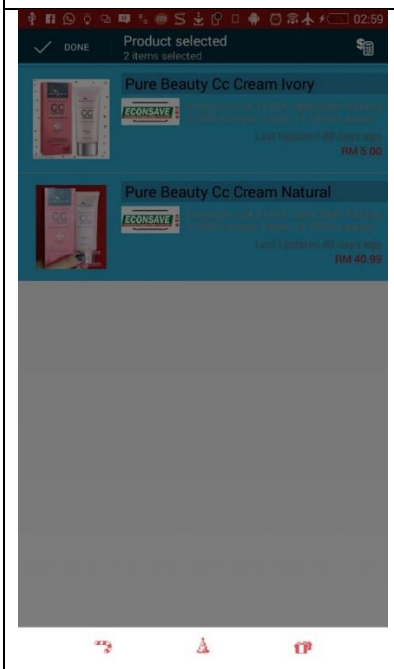
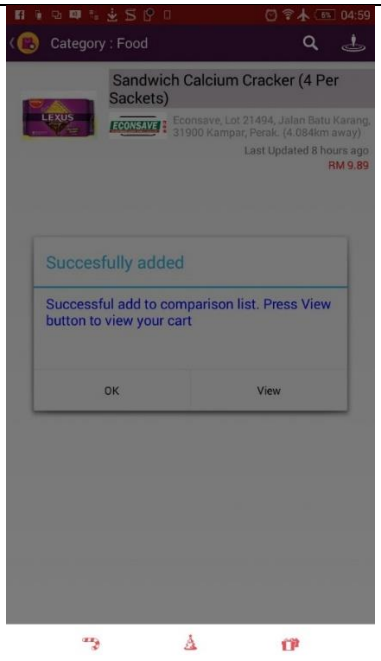



Figure 4-42 Report Own Price, Fail

| Reported Price, Fail | Report Own Price, Fail |
|---|---|

<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Figure 4-43 Nearby Market Filtering</td>
<td>Figure 4-44 Search By Category</td>
</tr>
<tr>
<td>Nearby Market Filtering</td>
<td>Search By Category, in this example is search by food ( action bar )</td>
</tr>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Figure 4-45 Search by product name</td>
<td>Figure 4-46 Search product name in mandarin</td>
</tr>
<tr>
<td>Search by product name</td>
<td>Search product name in mandarin</td>
</tr>
</table>

<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Figure 4-47 Search Product by mandarin translation</td>
<td>Figure 4-48 Product Semantic meaning suggestion</td>
</tr>
<tr>
<td>Search Product by mandarin translation</td>
<td>Product Semantic suggestion</td>
</tr>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Figure 4-49 Calculate product unit price</td>
<td>Figure 4-50 Successfully add product to unit calculator history</td>
</tr>
<tr>
<td>Calculate product unit price</td>
<td>Successfully add product to unit calculator history and press view to view unit price</td>
</tr>
</table>

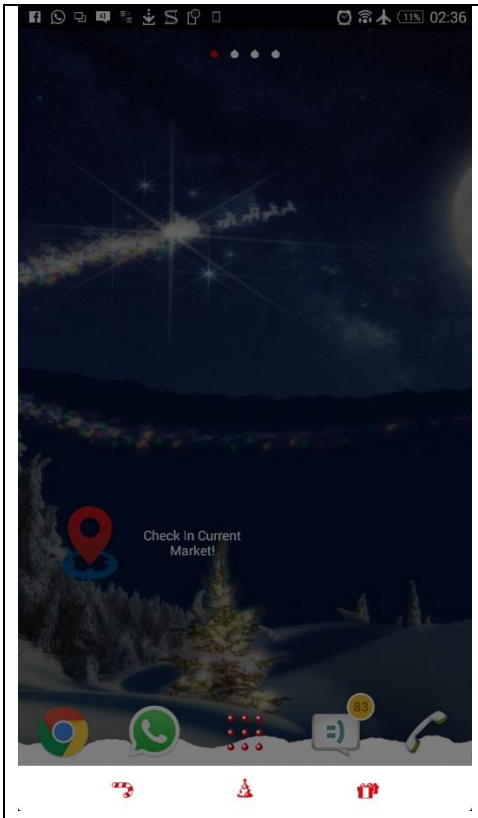



Figure 4-51 Check in current market

Check in current market

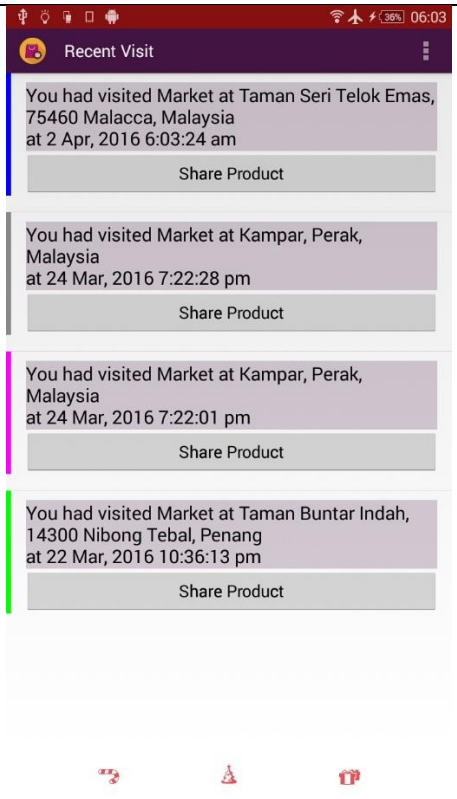



Figure 4-52 Market Visit History

Market Visit History


Figure 4-53 Scan Barcode

Scan Barcode

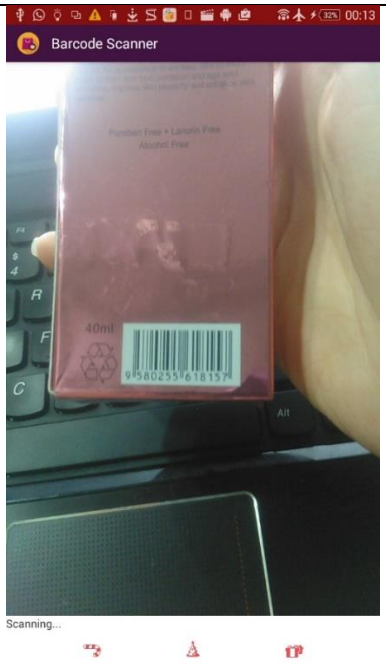



Figure 4-54 Start Detecting Barcode

Start Detecting Barcode

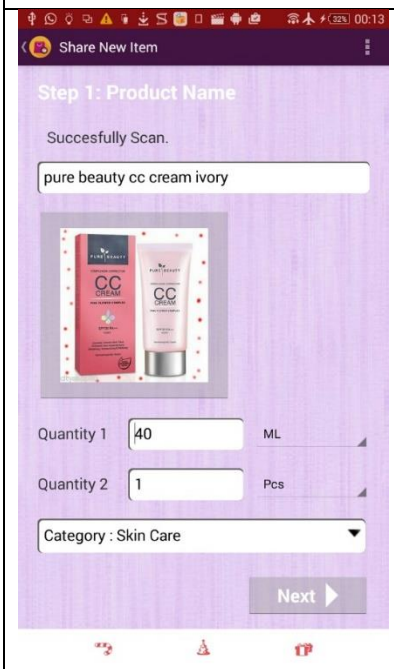



Figure 4-55 Share New Item Step 1

Share New Item Step 1

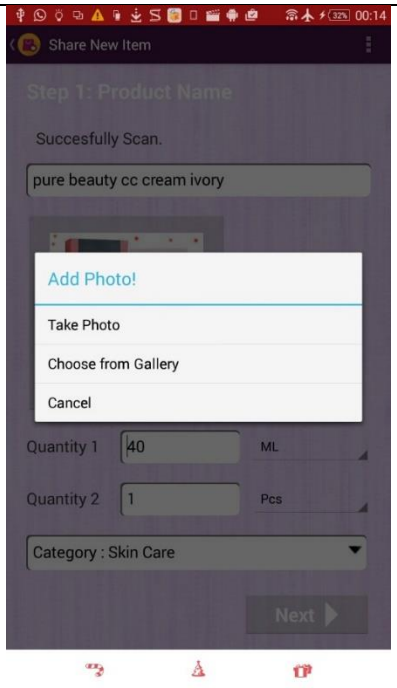



Figure 4-56 Share New Item add photo

Share New Item add photo

<table>
<tr>
<td> 

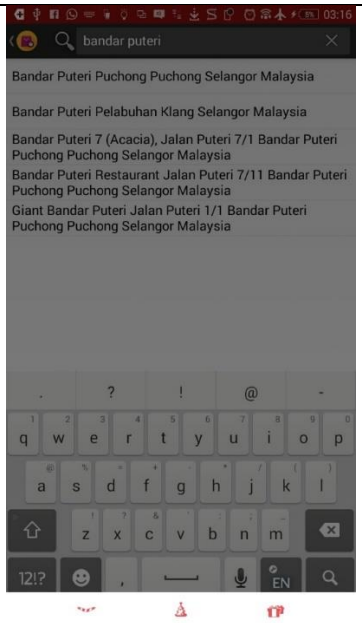
Figure 4-57 Share New Item Step 2</td>
<td> 

Figure 4-58 Place Address Semantic Suggestion</td>
</tr>
<tr>
<td>Share New Item Step 2: Location originally in current location which is kampar</td>
<td>Click editext and change address by Place Address Semantic Suggestion</td>
</tr>
<tr>
<td> 

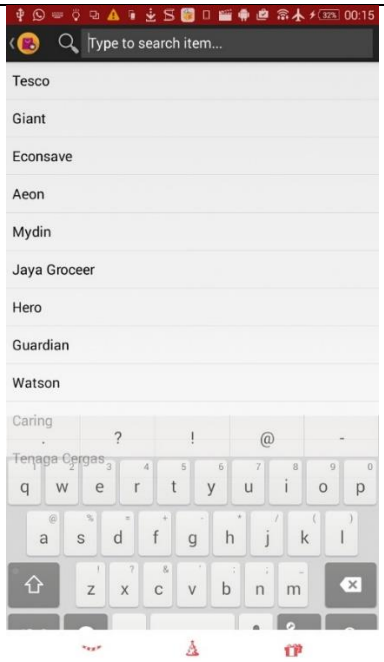
Figure 4-59 Successfully change location address</td>
<td> 

Figure 4-60 Store/market drop down</td>
</tr>
<tr>
<td>Successfully change location address</td>
<td>Store/market drop down</td>
</tr>
</table>

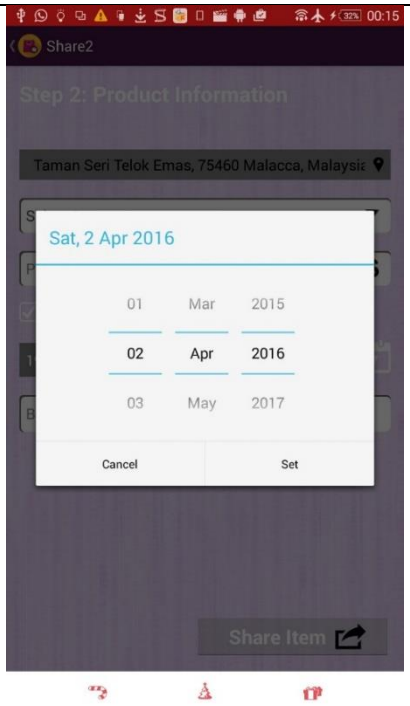



Figure 4-61 Change Promotion End Date

Change Promotion End Date

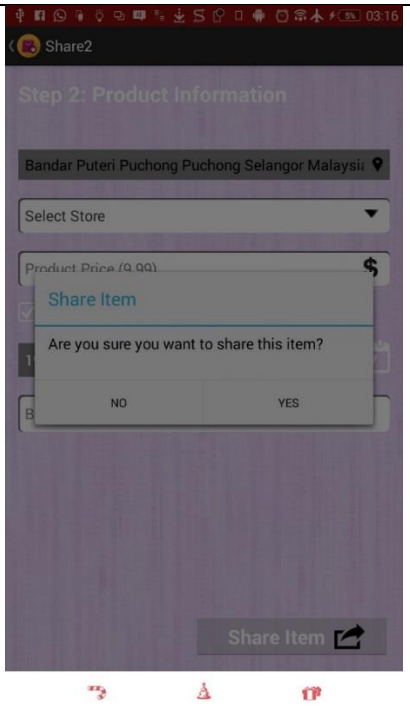



Figure 4-62 Share Product Confirmation Pop Up

Share Product Confirmation Pop Up

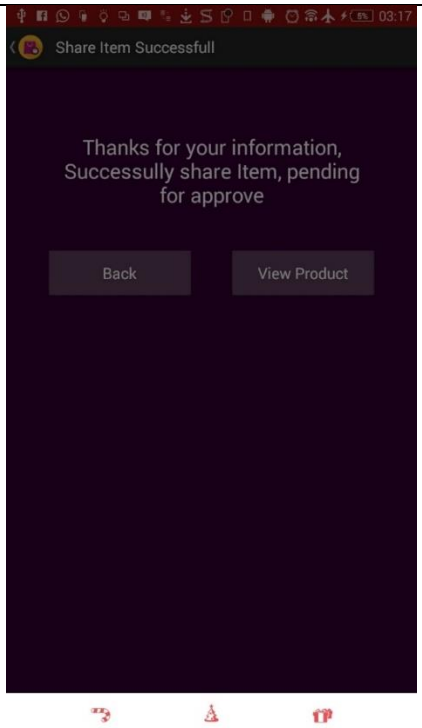



Figure 4-63 Share Product Successfully

Share Product Successfully


Figure 4-64 Catalogue list

Catalogue list


Figure 4-65 Catalogue detail swipe view 1

Catalogue detail swipe view 1


Figure 4-66 Catalogue detail swipe view 2

Catalogue detail swipe view 2


Figure 4-67 Unit Calculator Module Show Calculated History

Unit Calculator Module History

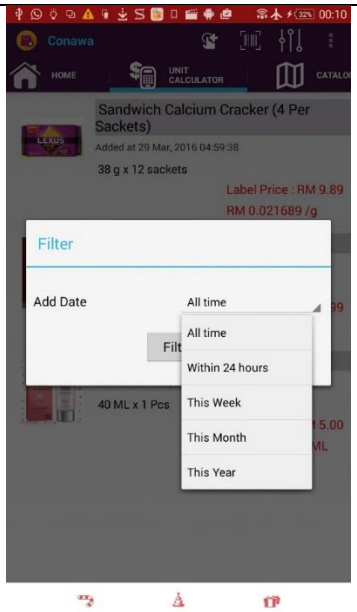



Figure 4-68 Filter Product Result by time

| Filter Product Result by time |
|---|

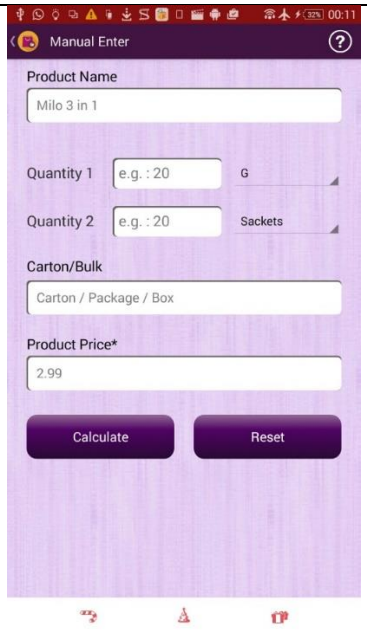



Figure 4-69 Manual Way

| Manual Way display when click manual button on action bar |
|---|

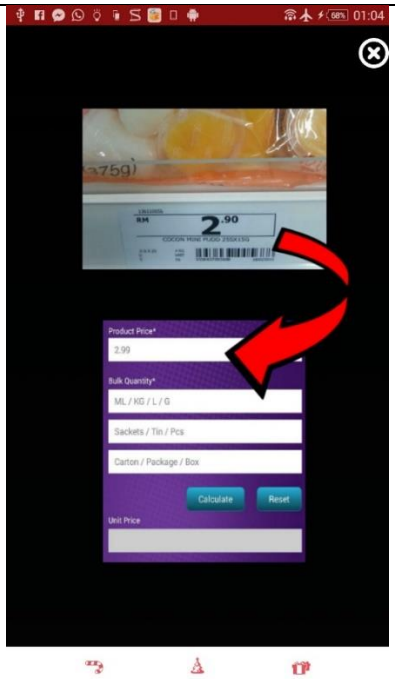



Figure 4-70 Help Page in unit calculator module

| Help Page in unit calculator module |
|---|


Figure 4-71 Scanning Way to get product unit price

| Scanning Way to get product unit price |
|---|

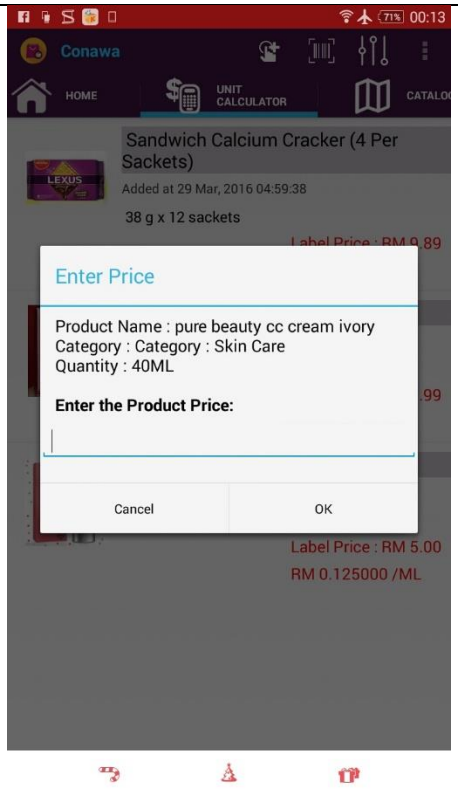



Figure 4-72 Enter price after barcode scan

Enter price after barcode scan


Figure 4-73 Compare selected product

Compare selected product

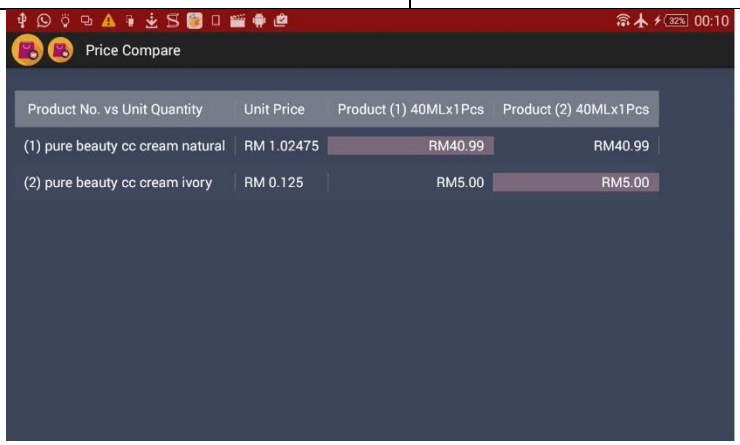



Figure 4-74 Price Comparison

Price Comparison

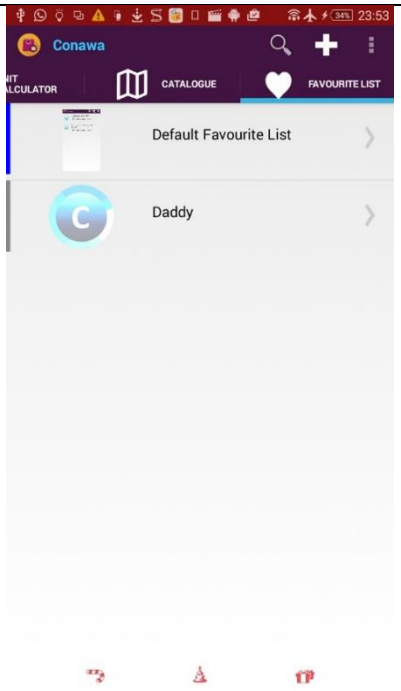
<table>
<tr>
<td> 

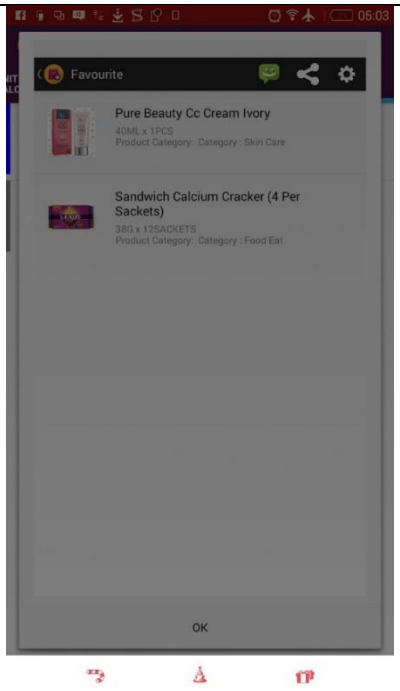
Figure 4-75 Custom Favorite List</td>
<td> 

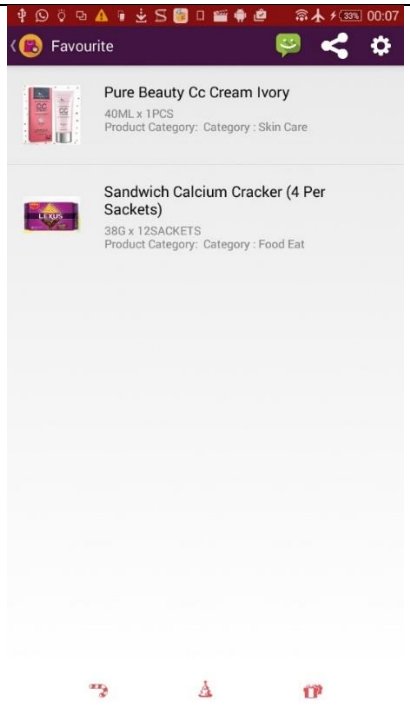
Figure 4-76 Preview Favorite List</td>
</tr>
<tr>
<td>Custom Favorite List</td>
<td>Preview Favorite List before click inside</td>
</tr>
<tr>
<td> 

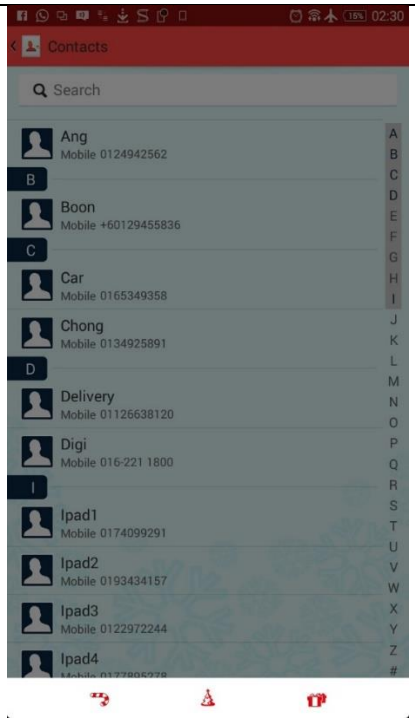
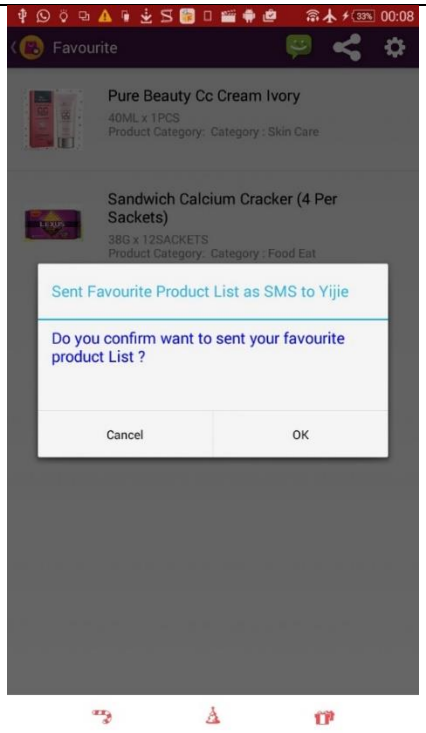
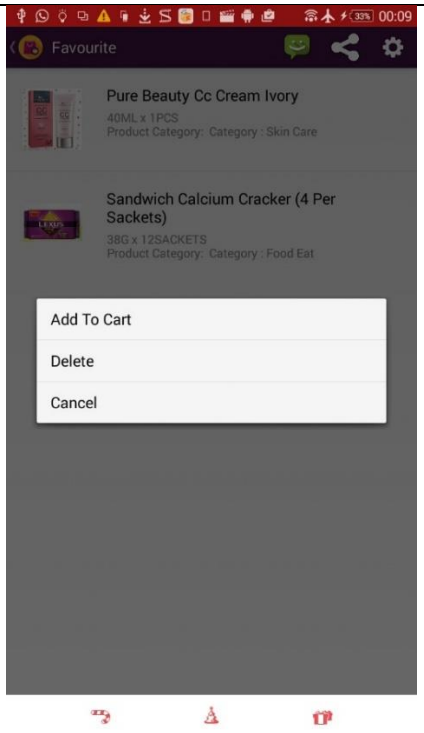
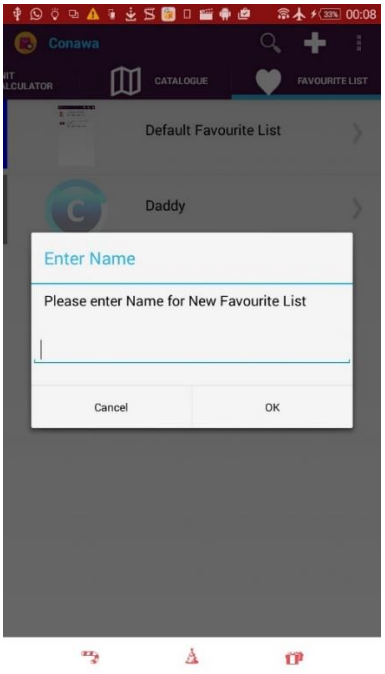
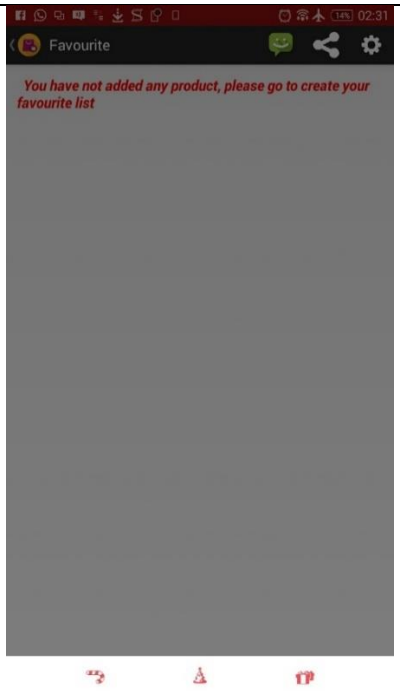
Figure 4-77 Favorite List Detail</td>
<td> 

Figure 4-78 Share Favorite to contact by SMS</td>
</tr>
<tr>
<td>Favorite List Detail</td>
<td>Share Favorite to contact by SMS</td>
</tr>
</table>

<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Figure 4-79 Send Favorite List by SMS Confirmation Pop Up</td>
<td>Figure 4-80 Add To cart from favorite list</td>
</tr>
<tr>
<td>Send Favorite List by SMS Confirmation Pop Up</td>
<td>Add To cart from favorite list</td>
</tr>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Figure 4-81 Add New Custom Favorite List</td>
<td>Figure 4-82 Empty Favorite List</td>
</tr>
<tr>
<td>Add New Custom Favorite List</td>
<td>Empty Favorite List</td>
</tr>
</table>

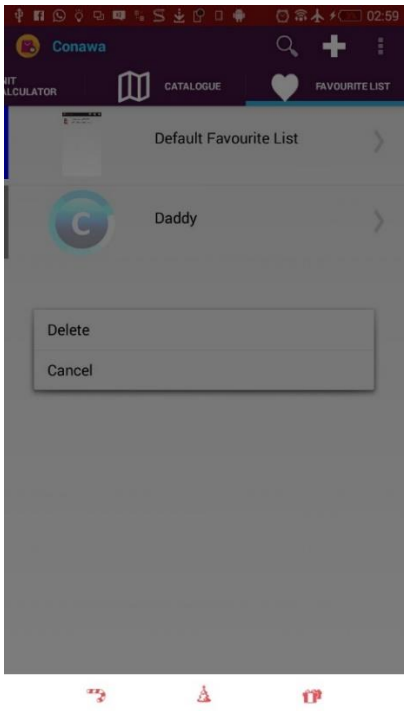



Figure 4-83 Delete Custom Favorite List

Delete Custom Favorite List

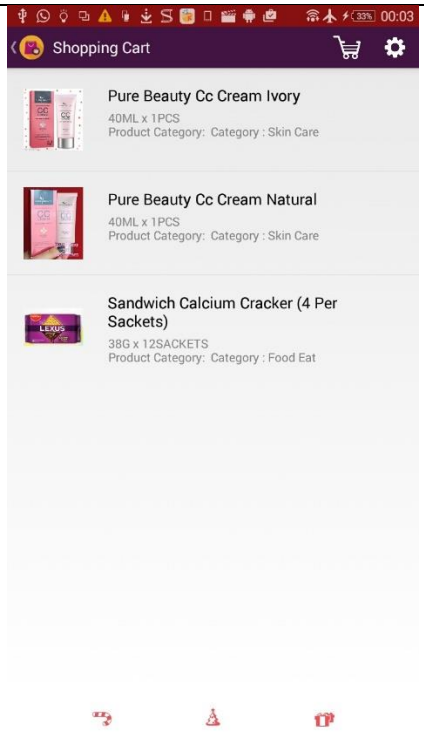



Figure 4-84 Shopping Cart

Shopping Cart

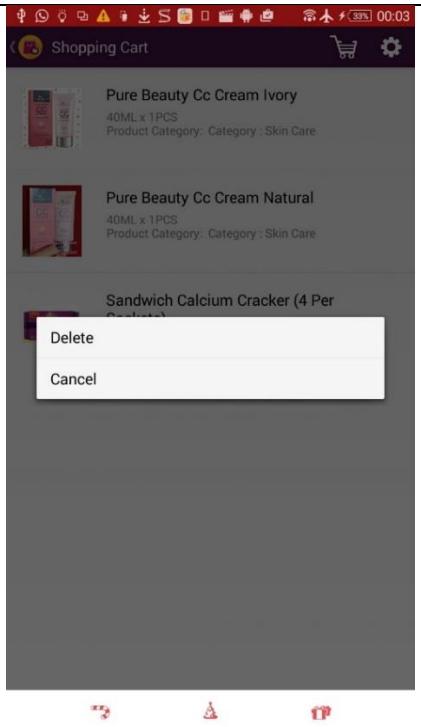



Figure 4-85 Delete Product in Cart

Delete Product in Cart


Figure 4-86 Ideal Store Suggestion 1

Ideal Store Suggestion 1

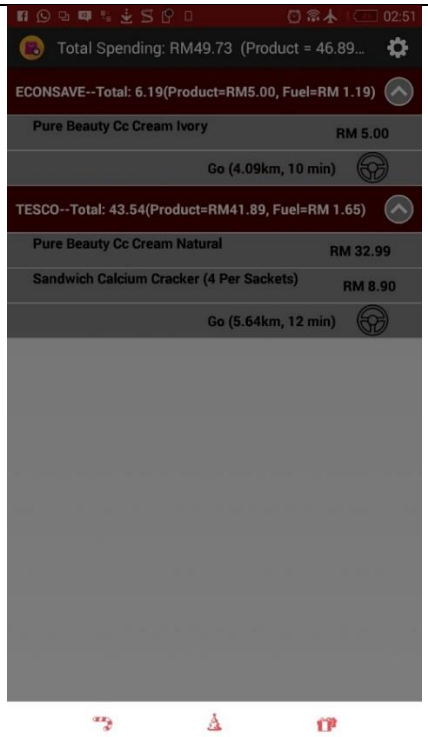



Figure 4-87 Ideal Store Suggestion 2

Ideal Store Suggestion 2

## 4.5 Application Screen Flow with Classes and Methods

### 4.5.1 Overview of Application Development

Android development basically encourage isolation between user interface and backend logic. In android development, xml file mostly use for user interface design while Java use for logic design. In Java, we need to inflate the ui xml file element (e.g. button, textview) and bind the processed data to ui in java. Table below show the folder exist in system development and the purpose of each folder.

Table 4-1 Program file Type

| **Program file Type** | **Purpose** |
|---|---|
| XML layout file | Design the user interface layout separately either by component or by page. (Design whole page individually or design single element and reuse the UI element in every android page as long as the page need it). |
| Java Client Side | All the backend logic and algorithm are done here. It bind processed data to UI element. |
| XML Menu file | Define Actionbar behavior. Design option menu of each activity/ page. |
| Drawable file( Image and XML drawable) | All the image that required in system and UI element selector (behavior of UI element when pressed, focus, disable mode). Besides that, the XML file in this folder all are related to UI element design such as drop shadow, gradient, stroke. |
| anim | Create animation behavior for UI element. |
| Libs | Libraries and APIs that incorporate in this application are place here. |
| Manifest.xml | Application Properties and permission to access mobile device features. |

In android, all the pages are called as activity. When the application first started, it will execute application class to initiate application. In manifest.xml, clearly stated which activity should run when application first started. Thus, after application class execute, launcher activity will execute.

#### 4.5.1.1 Design Pattern Implemented In Application

Android itself implemented MVC design Pattern. MVC is already implemented in android by default.

1) View = layout, resources, drawable, UI class such as Button derived from android.view.View class. This include XML, resources, language file.
2) Controller = Activity, Service, BroadcastReceiver that implement business logic. For instance, we extend classes like FragmentActivity, ListActivity and make use of inflators to inflate XML files.
3) Model = The classes controlling and format the data. For instance, in android there are a lot Utils have been ready such as Database Helper, Html. We implemented WebService serve as model to control the flow of data and format the data between application layer and web service.

Besides that, android also uses ViewHolder Design Pattern. Basically this used to improve performance of ListView when scrolling action by access each item in listview without the need of look up the layout with frequent call of findViewById() thus it saving processor memory.

#### 4.5.1.2 Application Life Cycle

All application runs on Android platform are the building blocks in system. The building blocks call activity and these activities can exist in different states in android platform namely Created, Started, Paused, Resumed, Stopped and Destroyed.

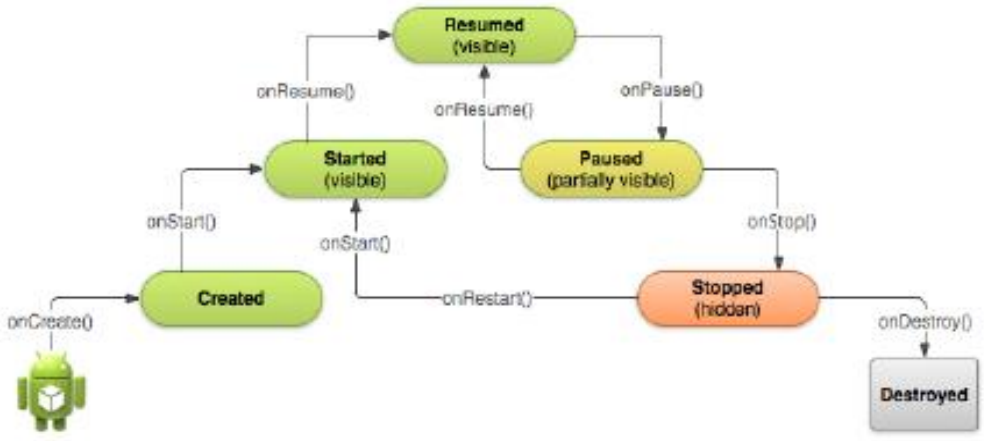



Figure 4-88 Application Life Cycle

Table 4-2 Application Life Cycle

| **States of Activities** | **Description** |
|---|---|
| **Active or Running** | Activities considered in this states when running in foreground. It having highest priority in application and can only be killed by the OS. |
| **Paused** | When activity visible to user but partially hidden by later or non-full sized activity. It still considered as alive and all the context information and data still maintain on UI. It is visible but does not receive any user input |
| **Stopped** | Activity is completely run in background and hidden from users view. In this case, there are other full page activity on top of it. This has the lowest priority among this three states. If the system required more memory for higher priority thread or activity, the activity in this state will be killed first. |
| **Restarted** | After the activity that run in background has been killed previously and user wishes to relaunch the activity, it will be restore easily from memory since the state and context of previous killed process will save in temporary memory. |

#### 4.5.1.3 Type of Class

Table 4-3 Type of Class

| | |
|---|---|
| Activity Class | Inflate, draw, place layout correctly, contains business logic and bind data from server to UI |
| Model Class | Class to populate data from database or external API. |
| Adapter Class | Populated data in and place in UI. Often use to converts ArrayList Object that pass from activity into view items. |
| AsyncTask Class | Perform background operations and publish result back to UI thread in activity. Normally called by Acitvity Class |

| | |
|---|---|
| Configuration Class | Define all the global static variable that will be used throughout whole application |

#### 4.5.1.4 Custom Adapter

We create custom adapter frequently in this application to populate listview, gridview, tablegrid data.

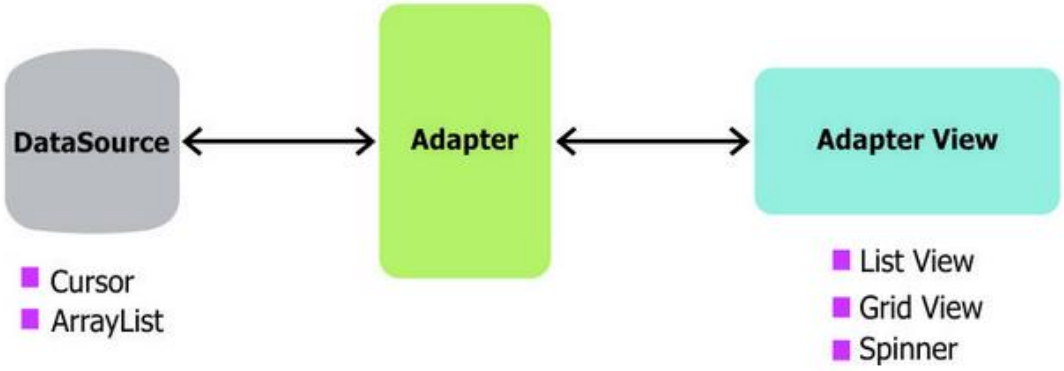



Figure 4-89 Custom Adapter

Data source imply the array or list to use in constructing the view. We connect listview or gridview to an custom adapter, the adapter will instantiate rows until the listview has been fully populated enough items to fill the full height of list. When user scroll through the list, items that hidden from user view are kept in memory and every new row display to user view reuses older row kept in memory previously. Normally, adapter will receive activity context, and arraylist that contain the object that user populated in model class.

```
ListView listView = (ListView) findViewById(R.id.lvItems);
listView.setAdapter(itemsAdapter);
```

Code Segment 4-1 Custom Adapter

#### 4.5.1.5 Global Configuration

This interface file was created at first in development and it serves as global configuration for whole application. All the values that will be use in development are define as static final variable here. Whole application under same package will be able to access this interface and no other objects are allowed to make any changes to these variables.

```
public interface Config {
      // server using java
```

```
        static final String APP_SERVER_URL =
        "http://192.168.43.78:8082/abcdWS/";
       static final String GOOGLE_PROJECT_ID = "275715266693";
       static final int colourbg = 0xe9330033;
}
```

Code Segment 4-2 Global Configuration

#### 4.5.1.6 Overview of API implemented

Below listed some API that will integrate in application.

Table 4-4 API List

| Google Maps Distance Matrix API |
|---|
| https://maps.googleapis.com/maps/api/distancematrix/json?origins=Vancouver+BC\|Seattle&destinations=San+Francisco\|Victoria+BC&key=APIKEY |
| Calculate estimate duration and distance of user current distance and destination |

| Google Maps Directions Waypoints API |
|---|
| https://maps.googleapis.com/maps/api/directions/json?origin=Boston,MA&destination=Concord,MA&waypoints=Charlestown,MA\|Lexington,MA&key=YOUR_API_KEY |
| Calculate optimize route by rearranging the waypoints in more efficient order, return multi-destination for a series of waypoints |

| Google Places Autocomplete API |
|---|
| https://maps.googleapis.com/maps/api/place |
| Provide location based autocomplete predictions like search on Google Maps |

| Language Detection API |
|---|
| http://ws.detectlanguage.com/0.2/detect?q=query&key=APIKEY |
| Language Detection API accepts text and produces result with detected language code and score. |

| Microsoft Translator API |
|---|
| Download a Java wrapper for the Microsoft Translator API from http://mvnrepository.com/artifact/com.memetix/microsoft-translator-java-api/0.3 |
| A Java wrapper for the Microsoft Translator API |

| Zbar API |
| --- |
| Download a Java wrapper for the Zbar API from https://sourceforge.net/projects/zbar/files/AndroidSDK/ |
| This library facilitate QR and barcode scanning. This is basically a barcode and QR code scanning API |

| Google Play Service API |
| --- |
| Download a Java wrapper for the Google Play Service API from SDK Manager |
| With this API, the app can use google-powered features such as Maps and location navigation. |

| Google Play Service API |
| --- |
| Download a Java wrapper for the Google Play Service API from SDK Manager |
| With this API, the app can use google-powered features such as Maps and location navigation. |

| AdMob for Android API |
| --- |
| https://developers.google.com/admob/android/download |
| Integrate the Google Mobile Ads SDK into app and use it to display an AdMob banner ad |

| Gson API |
| --- |
| http://www.java2s.com/Code/Jar/g/Downloadgson222jar.htm |
| A Java library that can convert Java Objects into JSON, sent to JSP and back. |

| Parse API |
| --- |
| http://www.java2s.com/Code/Jar/g/Downloadgson222jar.htm |
| Parse serve as cloud file server in our application. This API allow apps upload and download file or data that will be used in application. |

### 4.5.2 Initial Module

#### 4.5.2.1 Splash Screen

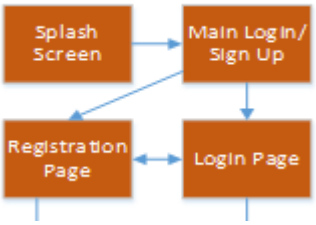



Figure 4-90 Main Screen Flow

Splash Activity would be the launcher activity of whole system, which mean when the application first started, this will be the first activity or page that display to user. After user click the icon from Android desktop home screen, user will be able to see animated splash screen with application logo and name. At the same time of splash screen displaying, the system load the settings. The double arrow between registration and login page indicate the dual direction navigation. The code below show the implementation of animation:

```
setContentView(R.layout.activity_splash);

            final ImageView myImage = (ImageView) findViewById(R.id.splash);
            AnimationSet as = new AnimationSet(true);
            final Animation myRotation = AnimationUtils.loadAnimation(
                        getApplicationContext(), R.anim.rotate);
            as.addAnimation(myRotation);
```

Code Segment 4-3 Animation

Firstly, rotate.xml was created to indicate the movement of animation. The activity_splash layout design the UI layout. The layout file was inflated and bind the splash logo to the animation behaviour.

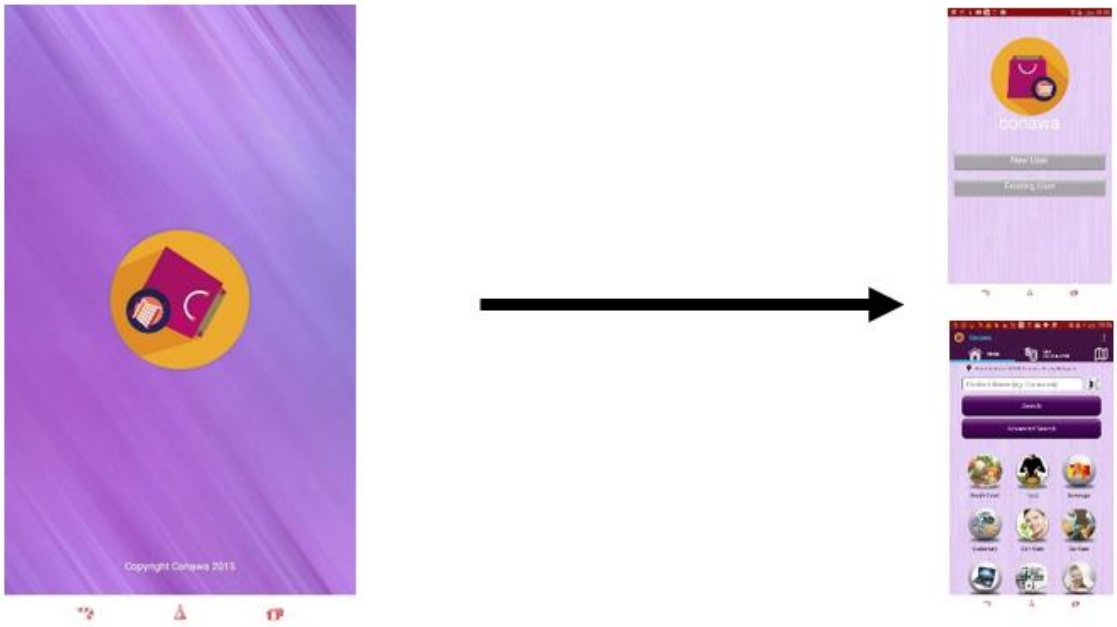



Figure 4-91 Home screen Demonstration

#### 4.5.2.2 User Authentication

After splash screen displayed, if user had sign in previously and does not sign out yet, the system will display main home page right after splash screen, otherwise Main Sign In / Sign Up page view be show, the image above depict the transition of two activity.

The above scenario is the case when device GPS was opened prior access to this application. However, if this application detect that user have not enabled GPS location, it will show a pop up to prompt user to enable GPS data.

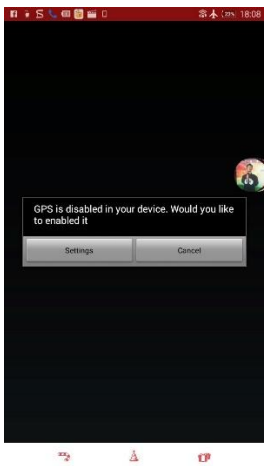



Figure 4-92 GPS Enable Checking

The code below show the implementation of this pop up:

```
lm = (LocationManager) this.getSystemService(this.LOCATION_SERVICE);
boolean enabled = lm.isProviderEnabled(LocationManager.GPS_PROVIDER);

if (!enabled) {
                            // notify user
AlertDialog.Builder dialog = new AlertDialog.Builder(this);
dialog.setMessage("GPS is disabled in your device. Would you like to enabled
it");
dialog.setPositiveButton("Settings",new DialogInterface.OnClickListener() {

      @Override
      public void onClick(DialogInterface dialog,int which) {

      Intent myIntent = new Intent(
      Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                                                  startActivity(myIntent);

      }
});
```

Code Segment 4-4 Location Enable Checking

The location manager will check whether provider enable (GPS signal on). If system detect that the device GPS location sensor was not turn on, it will create the pop up dialog box to prompt user to turn on GPS sensor. Right after the system able to getting the GPS sensor on, application will work together with internet connection and mobile network to get the reading of the user's current location coordinate. When the GPS signal is used and working along with internet and mobile connection, the application drain battery faster than normal mode but definitely it will be more accurate than if only either one to be used. The slight change of device movement can be detected up to +-1km bearing accuracy.

User are given two options which are New User for Sign Up or Existing User to Sign In to existing account. If user click new user, registration screen will be display. After the user fill in all the information and click register new Account, the system will do input type, length validation. If the input are acceptable, system will call worker thread to send information to server and then insert the information into database. Worker thread will use POST method to send information as shown in following:

```
HttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new HttpPost("http://192.168.43.78:8082/abcdWS/ SignUp");
                            // Post Data
List<NameValuePair> nameValuePair = new ArrayList<NameValuePair>(
                                        2);
```

```
nameValuePair.add(new BasicNameValuePair("name", userName));
nameValuePair.add(new BasicNameValuePair("password", password));
nameValuePair.add(new BasicNameValuePair("email", email));

httpPost.setEntity(new UrlEncodedFormEntity(nameValuePair));

HttpResponse response = httpClient.execute(httpPost);

InputStream is = response.getEntity().getContent();
```

Code Segment 4-5 Log In Call Post Method

Before insert information into database, the server are required to check whether there are existing user with same UserName in POST method.

```
String queryCheck = "SELECT [UserName] from [Conawa].[dbo].[UserAccount] where
[UserName]=?";
                    PreparedStatement psCheck =
conn.prepareStatement(queryCheck);
                    psCheck.setString(1, name);
                    ResultSet rsCheck = psCheck.executeQuery();
```

Code Segment 4-6 Log SQL Query

If there is no error happen, server will give feedback and return User ID to indicate successfully register. After that, all the UserID, UserName and default settings will write into shared Preferences. If there is an error, error message “Sign Up Failed, please try again” will be toast. Example of saving information into shared preference are as following:

```
SharedPreferences sharedPref = getSharedPreferences(
"USER_DETAIL", Context.MODE_PRIVATE);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("userName", userName);
editor.putString("password", password);
```

Code Segment 4-7 Log In Information save to Shared Preferences


Figure 4-93 Registration

If user choose second option which is login, login activity will be displayed. After application done validation of username and password input, the information will send to server and matching will be done in server. If username found in database and password is match, then the system will allow user to continue accessing other features in application. At the same time, the worker thread will save the information into shared preferences to indicate login status, if the system cannot found any username and userid in shared preferences, that mean the user already log out the system. This is because the username and userid in shared preferences will be delete if user choose to log out the system. After that, system will redirect user to Main Home Page.

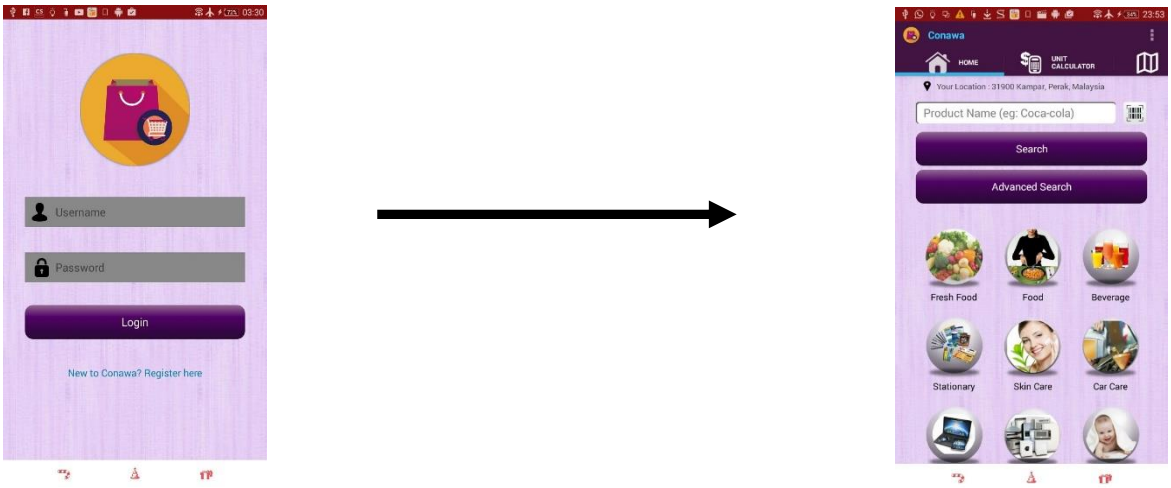



Figure 4-94 Log In Screen

Right after user login successfully to the application, Main Home page with four tab will displayed in user view namely Home Tab, Unit Calculator Tab, Store Catalogue Tab and Favorite List Tab. Before we start to discuss the content, we will like to discuss settings module.

### 4.5.3 Settings Module

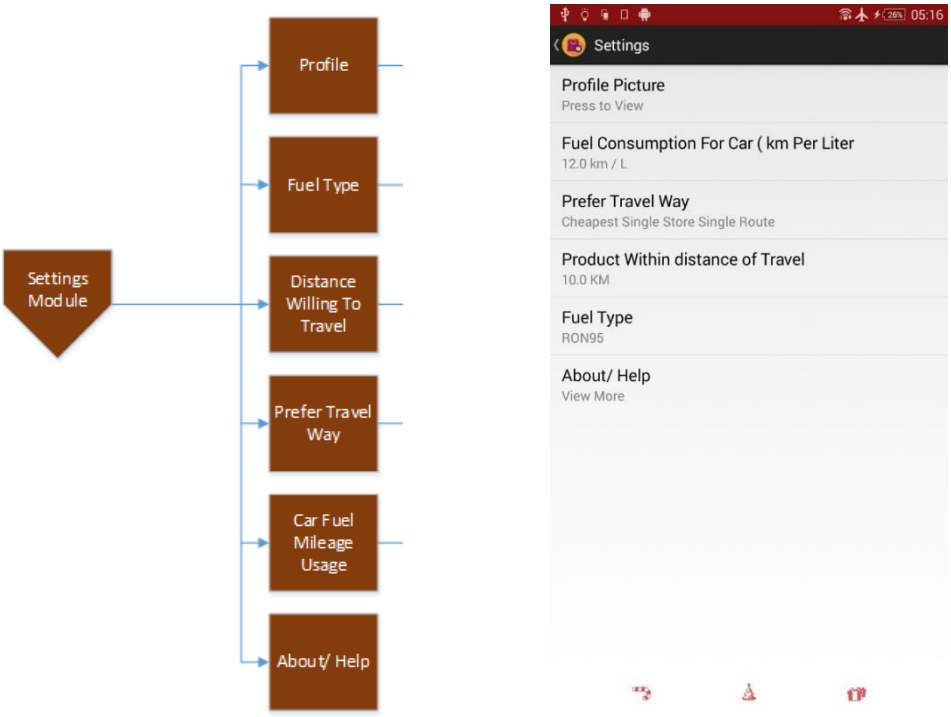



Figure 4-95 Left-Settings Screen Flow

Figure 4-96 Right-Settings Screen

There are 5 settings in application which is profile, fuel type, nearby distance willing to travel, prefer travel way, car fuel mileage usage.

#### 4.5.3.1 Settings List

The settings module using ListView to implement the scrollable List. All the value will first insert in arraylist and then the arraylist will be inserted in self-create Custom Adapter. The listview actually construct of more than one individual layout building block.

```
<ListView
    android:id="@+id/list22"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true" >
</ListView>
```

Item 1
Sub Item 1

Item 2
Sub Item 2

Item 3
Sub Item 3

Item 4
Sub Item 4

Item 5
Sub Item 5

Item 6
Sub Item 6

Item 7
Sub Item 7

Code Segment 4-8 ListView Layout Implementation

Custom adapter will first inflate the individual layout (e.g. linearlayout), get all the UI element from layout XML file and set the text in arraylist object into UI element. The image below show the single element that will inflate in custom adapter, and the data will be throw inside the single item. After adapter collect all the single item, listview will arrange the individual adapter item in list format.


Figure 4-97 Single Item Layout in Listview

Since the adapter are simply a connector to put data inside single item element. Thus after the data had been throw inside adapter, the adapter must be add into listview( inflate from another layout file ) in order to combine all the individual element together and become a listview.

### 4.5.3.2 Parse Application

Parse.com is an online cloud storage. We store image inside a table with primary key, Parse.com called table as “Class”. There are some step are required to configure an account and get API Key.

Step 1: Create an Account and create a new app in dashboard

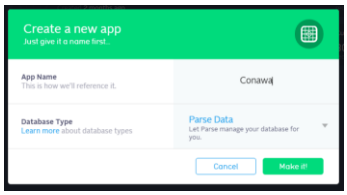



Figure 4-98 Create New App in Parse

Step 2: Go to Security & Keys under App Settings, you can obtain app ID and API Keys at here. These will be used in application later.

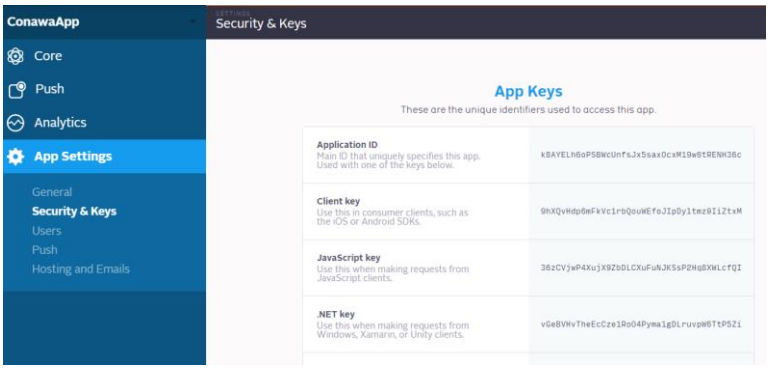



Figure 4-99 API Key

Step 3: Create a new Class as new table. Leave the type of class as custom class and add some data in it.

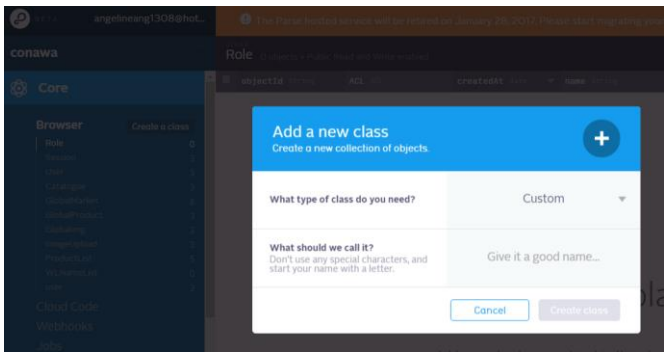



Figure 4-100 Create new class and table

Step 4: Create a custom application class in Project.

The application class is the first class to loaded when application launch. If the application do not have application class, android will use hidden generic application class in library, but here we create custom application to do the initial setup for parse application.

```
import android.app.Application;

import com.parse.Parse;
import com.parse.ParseACL;
import com.parse.ParseUser;

public class ParseApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();

        // Add your initialization code here
        Parse.initialize(this,
        "qqRkyQWJiKnRsY0FdcRhCXKQxpHCyVNbUkSG7A3S",
        "C3bdyqPX57Hngbd3LF5tEgMoJOLPRD75uUWRLKhc");

        ParseUser.enableAutomaticUser();
        ParseACL defaultACL = new ParseACL();

        // If you would like all objects to be private by default, remove this
        // line.
        defaultACL.setPublicReadAccess(true);

        ParseACL.setDefaultACL(defaultACL, true);

         //FacebookSdk.sdkInitialize(getApplicationContext());
    }
```

Code Segment 4-9 Parse Application Initialization

The initialization is in format below:

```
Parse.initialize(this, APPLICATION_ID, CLIENT_KEY);
```

Code Segment 4-10 Initialize by API KEY and ID

We create a class called ParseApplication and extends android.app.Application class. In the manifest.xml, set the name attribute in the application tag with the class name ParseApplication.

```
<application
    android:name="com.javapapers.android.swipetablayout.app.ParseApplication"
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
```

```
android:largeHeap="true"
android:theme="@style/AppBaseTheme" >
```

Code Segment 4-11 Parse Application in Manifest

Step 5: Download Parse.jar from official website and copy into libs folder. After that, set Java build path point to that library.

Step 6: Execute application

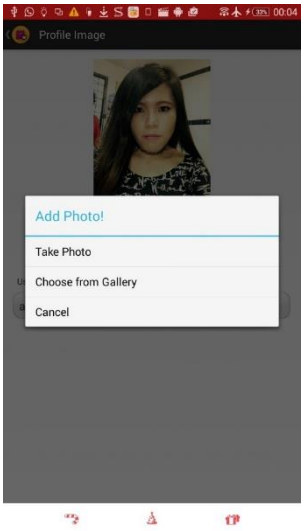



Figure 4-101 Left-Profile Screen

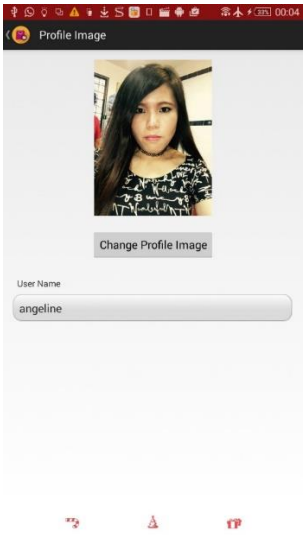



Figure 4-102 Right-Change Profile Picture Screen

#### 4.5.3.3 First setting: profile

In here, user can change their profile photo as they like. The profile photo can choose from gallery or directly take photo by device camera. After the photo successfully change, toast message will display to show status to indicate whether photo is failed or successfully change.

The implementation of image retrieval are rather simple.

```
ParseQuery<ParseObject> query = ParseQuery.getQuery("user");

query.whereContains("userid", userid);
List<ParseObject> arg0 = query.find();
ParseFile fileObject = (ParseFile) arg0.get(0).get("images");
byte[] tempImg;
tempImg = fileObject.getData();
bmp = BitmapFactory.decodeByteArray(tempImg, 0, tempImg.length);
profileImg.setImageBitmap(bmp);
```

Code Segment 4-12 Parse Application Image Retrieval

Parse.com is an online cloud storage. We store image inside a table with primary key, Parse.com called table as “Class”. The first line of code implementation trying to get the table named “user” which is the class / table that store our user image. After get the table class, the table will try to find userid column and compare to query input. The code above equivalent to the sql below:

```
Select images from user where userid = userid
```

Code Segment 4-13 Profile Picture Retrieval SQL query

After the images file retrieved, it store in ParseObject. ParseObject is an object type in Parse API use to store binary file. Later, the bytes data in file object will be retrieved and stored in byte array. The bytes array required to decode into bitmap type so that the image can set to the imageView.

```
ParseFile fileObject = (ParseFile) arg0.get(0).get("images");
tempImg = fileObject.getData();
bmp = BitmapFactory.decodeByteArray(tempImg, 0, tempImg.length);
profileImg.setImageBitmap(bmp);
```

Code Segment 4-14 Converting Parse Object to bitmap

After the image successfully updated, the image link will save to sql server database. There are two option of uploading image which are take picture or from gallery. Code below are useful for taking picture directly from camera:

```
Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
File f = new
File(android.os.Environment.getExternalStorageDirectory(),temp.jpg");
intent.putExtra(MediaStore.EXTRA_OUTPUT,Uri.fromFile(f));
```

Code Segment 4-15 Take Photo

The code above will open device camera and retrieved stored image in temporary storage whereas code below are used if wish to implement image selector from gallery:

```
Intent intent = new Intent(Intent.ACTION_PICK,
      android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
startActivityForResult(intent, 4);
```

Code Segment 4-16 Open Gallery Selector and Retrieve Image

After images are retrieved we use code below to process the image.

```
Uri selectedImage = data.getData();
// Set the Image in ImageView after decoding the String
profileImg.setImageBitmap(decodeUri(selectedImage));
ByteArrayOutputStream stream = new ByteArrayOutputStream();
// Compress image to lower quality scale 1 - 100
decodeUri(selectedImage).compress(Bitmap.CompressFormat.JPEG,100, stream);
final byte[] image = stream.toByteArray();
```

Code Segment 4-17 Decode and Convert Image from URI to byte

The code below show the Parse.com implementation of image saving action, the equivalent sql to explain the logic below are:

```
Select * from user where userid= userid input

If return row>0

Update user

Set image = new image

Where userid = userid input

Else

Insert user values( userid,images)
```

Code Segment 4-18 Equivalent SQL Query select and update image

```
ParseQuery<ParseObject> query = ParseQuery.getQuery("user");
query.whereEqualTo("userid", "userid");
ParseObject arg0 = query.getFirst();
if (arg0 != null) {
	ParseFile file1 = new ParseFile("Profile.jpg",image);
	file1.saveInBackground();
	arg0.put("images", file1);
	arg0.saveInBackground();
} else if (arg0 == null) {
	ParseFile file1 = new ParseFile("Profile.jpg",image);
	file1.saveInBackground();
	ParseObject imgupload = new ParseObject("user");
	imgupload.put("userid", userid);
	imgupload.put("images", file1);
	imgupload.saveInBackground();
```

Code Segment 4-19 Upload image to Parse storage

#### 4.5.3.4 Second Setting: Fuel Consumption for Car (km per Liter)

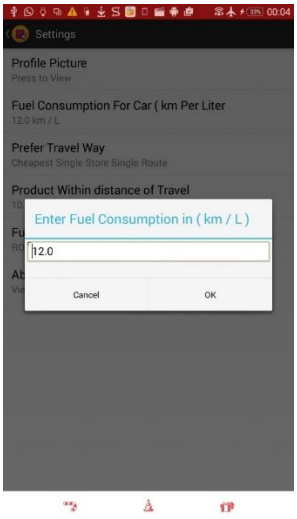



Figure 4-103 Fuel Consumption

This setting is a pop up dialog box and the editext only allow digit and decimal type of input, when click on the editext, only numeric softkeyboard will displayed but not alphanumeric.

Basically this setting will allow user to enter their car fuel consumption in km per Liter. If they did not do any changes to this setting, default will be 12km per Liter which is the average consumption according to research that had done by researcher from USM (Chapter 3.3).

User are free to change the setting according to their needs, since they may have different preference. The car nowadays provide car fuel usage along with car speed meter. The image below are Honda City 2.0cc car (Left) and Perodua Myvi 2012 1.5cc (Right) that was taken previously. It show the car fuel usage: 16.8km/L. Thus from the car meter, user may adjust the setting according to their needs.


#### 4.5.3.5 Third Setting: Travel Route Way

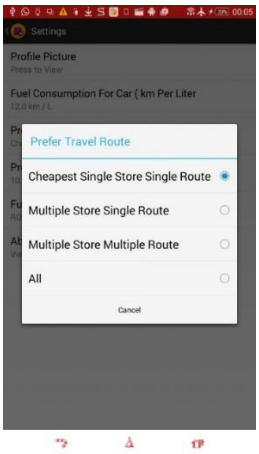



Figure 4-104 Travel Route Way

This setting will be used in ideal store suggestion module, further detail will be discussed in ideal store suggestion module.

There are four type of Travel Route Way in this setting:

Table 4-5 Four Type of Travel Route Way in Settings

| Setting | Description |
|---|---|
| Cheapest Single Store Single Route | Cheapest single store after sum up fuel consumption and product total price |
| Multiple Store Single Route | Cheapest store combination and travel all store together ( minimum distance travel to all store ) |
| Multiple Store Multiple Route | Display cheapest store combination and travel all store by star topology |
| All | Display all the combination of market and all kind of travel way |

#### 4.5.3.6 Fourth Setting: Nearby Distance Willing to Travel

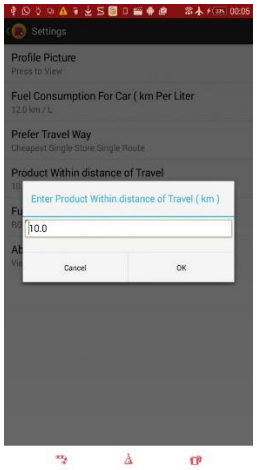



Figure 4-105 Nearby Distance Willing to Travel

This is the third setting which define the maximum distance user willing to travel and only numeric inputs are allowed in this field. This will be a user preferences for system retrieved all the data. This mean that, only all the distance of market and location fall into the range within the user define nearby distance will be filter out and display to user.

#### 4.5.3.7 Fifth Setting: Fuel Type

There are three types of fuel in international standard which is RON95, RON97 and Diesel. Since Fuel Price are fluctuate every month, so the system will calculate Total Fuel Required base on their mileage usage and car fuel type. The price of current fuel type will retrieve from database in this way.

```
SELECT fuelPrice from Fuel
WHERE fuelType='RON95'
AND EndDate IS NULL
AND StartDate<=CURRENTTIMESTAMP
```

Code Segment 4-20 SQL Query Get Current Fuel Price from SQL Server

### 4.5.4 Home Page Module



Figure 4-106 Home Page 1

Figure 4-107 Home Page 2

The image above show main home tab of whole application. It create by viewpager and customize actionbar together with swipe tab adapter. Every tab are called as fragment, the transition between fragment

```
viewPager.setOnPageChangeListener(new ViewPager.OnPageChangeListener() {})
```

Code Segment 4-21 Transition between Fragment

#### 4.5.4.1 Location Detection and Address Translation

First part of first in application show current location of user. The fragment need to extends LocationListener as follow:

```
public class Home extends Fragment implements LocationListener{}
```

Code Segment 4-22 Implements Location Listener

Location Change Listener will trigger itself every 5 seconds and sent request to satellite in order to get current location. The satellite will received request and return current coordinate back to application.

```
public void onLocationChanged(Location location) {
            // TODO Auto-generated method stub
      loc = new LatLng(location.getLatitude(),location.getLongitude());
      longitude = location.getLongitude();
      latitude = location.getLatitude();
}
```

Code Segment 4-23 Get New Location When Location Changed

When application received new coordinate it may update itself with new address. The address translation API which is "Geocoder" need to be done in `onLocationChanged` method right after getting new longitude and latitude. The code below show the process of translating coordinate of location to a full address.

```
Geocoder geocoder;
List<Address> addresses;
geocoder = new Geocoder(activity.context, Locale.getDefault());

addresses = geocoder.getFromLocation(latitude, longitude ,1);
String province = addresses.get(0).getAdminArea();
                    // get country
String country = addresses.get(0).getCountryName();
                    // get postal code
String postalCode = addresses.get(0).getPostalCode();
                    // get place Name
String knownName = addresses.get(0).getFeatureName(); // Only if
```

Code Segment 4-24 Address Translation

#### 4.5.4.2 Advertisement

The advertisement that display at the bottom of screen is actually provide by Google AdMob Advertisement API. This API exist in Google Play Services latest version unlike previously need to download AdMob API explicitly. In order to use AdMob in our application, there are several step need to follow.

Step 1: Sign up an account in Admob Publisher

Step 2: Log In to account and go to user dashboard.

Step 3: Create an ad unit for application.

There can be many ad unit in application and one ad unit represent one banner

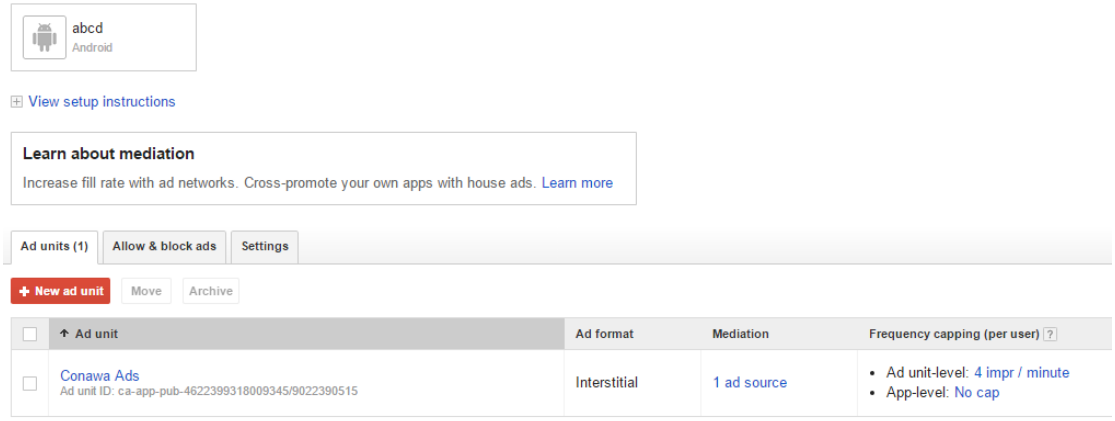



Figure 4-108 Create an ad unit for application

Step 4: Record down publisher ID

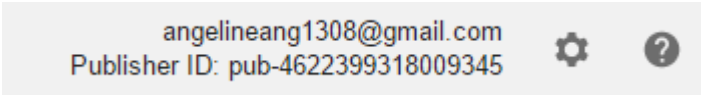



Figure 4-109 Record down publisher ID

Step 5: Implementation of AdMob

The code below show the implementation of AdMob in application

```
TelephonyManager tManager = (TelephonyManager) getActivity()
                          .getSystemService(Context.TELEPHONY_SERVICE);
String uid = tManager.getDeviceId();
AdView adView = (AdView) rowView3.findViewById(R.id.adView);
AdRequest adRequest = new AdRequest.Builder().addTestDevice(
                          "0F99AB86BB9EC655066CF457EFF4193C").build();
adView.loadAd(adRequest);
```

Code Segment 4-25 Back End Logic of Advertisement

Step 6: Layout of AdMob

```
    <com.google.android.gms.ads.AdView
        xmlns:ads="http://schemas.android.com/apk/res-auto"
        android:id="@+id/adView"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        ads:adSize="SMART_BANNER"
        ads:adUnitId="ca-app-pub-4622399318009345/9022390515" >
    </com.google.android.gms.ads.AdView>
```

Code Segment 4-26 Layout of Advertisement

In development and testing phase, sample ads are displayed but not real ads. We need to write the code below to indicate currently is in development phase or else we may get fine because we are not allowed to click the ads by own (same google account in device and developer).

```
AdRequest adRequest = new AdRequest.Builder().addTestDevice(
                                "0F99AB86BB9EC655066CF457EFF4193C").build();
```

Code Segment 4-27 Add Test Device

### 4.5.5 Searching Module

Basically there are four (4) ways to do searching


Figure 4-110 Searching Way

i. By Product Name and semantic autocomplete search (Support Mandarin and English dual way translation).
ii. By Barcode scanning, once user scan on product barcode, it will go database to find related product. If item found, particular product in different store will displayed. However if the product not in store it will toast a status message to indicate item not found.
iii. By advanced search, the application will display all the available market around user by

marker on google Map. Whenever user click on certain marker on google Map, transparent activity that show all the information of particular market include image, distance, contact number will be displayed and user can view all the product in particular market.

iv. There are 3 x 3 size table on home tab, user can choose to view only the product that fall in specific category (e.g. Skin Care : All the product related to skin care will be display ).

#### 4.5.5.1 Search by Dual Language Product Name and semantic autocomplete list

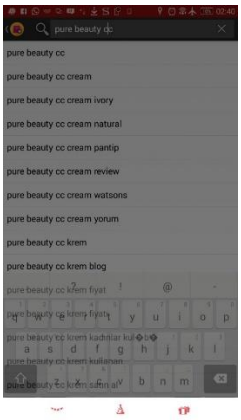



Figure 4-111 Search by Dual Language Product Name

When the Search Editext ( Hint : Product Name (e.g.Coca-cola) ) is clicked, a new listview activity will be open. The listView Activity show all the semantic related words to ease user in entering product name. Every time new character are typed, new set of related words will be update.

Everytime new character is typed in editext, the following listener will be trigger:

```
onQueryTextChange(String newText)
```

Code Segment 4-28 Text change listener

Once the editext received every new input character in onQueryTextChange method, Google Suggestion Queries API will be called.

```
http://suggestqueries.google.com/complete/search?client=chrome&q=input
```

Code Segment 4-29 Product Query Suggestion

Http Get request will then sent to this API and JSON response will be received after that.

```
url3 = new URL(urlStr3);
HttpURLConnection conn3 = (HttpURLConnection) url3.openConnection();
conn3.setRequestMethod("GET");
String line3;
BufferedReader reader3 = new BufferedReader(new
InputStreamReader(conn3.getInputStream()));
```

Code Segment 4-30 Http Get Request

The following code will be use to receive the response from API and parsing JSON data:

```
while ((line3 = reader3.readLine()) != null) {
            outputString3 += line3;
}

JsonParser parser2 = new JsonParser();
JsonArray nestedArray2 = parser2.parse(outputString3).getAsJsonArray().
      get(1).getAsJsonArray();
for (JsonElement e : nestedArray2) {
      String temp = e.toString().replaceAll("\"", "");
      sg.add(e.toString().replaceAll("\"", ""));
}
```

Code Segment 4-31 Parsing Json Data Return by suggestqueries

Basically the code above parse the Json data and get JsonArray. Later, second element of JsonArray will be retrieved out and all the possible words are stored in JsonArray. After that for loop was applied to retrieve all the suggestive words in JsonArray and add into ArrayList call “sg”.

Since the API are called during each character input, so the listview will then keep updating when every single character are entered.

#### 4.5.5.1.1 Translation Search Query

After user click on desired product name and back to home tab, translation of product will be carried out. The system will first check whether the input is Chinese or English. If Chinese, then system will translate to English or English to Chinese. Which mean that there are always two set of input will be sent to database to do searching. Below show the API was applied for detecting language.

```
http://ws.detectlanguage.com/0.2/detect?q=input&key=04abc411ef4e44ff64be7632386
2aa89
```

Code Segment 4-32 Language Detection

In here, Microsoft Translation API was applying and Microsoft Translator Client ID and Client secret must be obtained prior access to the API. Below show the way to register API.

First Step: Create a new account Windows Azure Marketplace


Figure 4-112 Create a new account Windows Azure Marketplace

Second Step: Go to "Data" Tab of Microsoft Azure Market Place and Sign Up to first option which is 2,000,000 characters/month

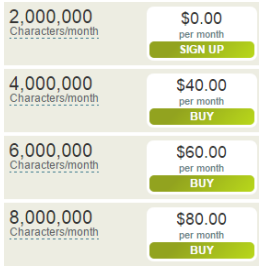



Figure 4-113 Microsoft Azure Market Place

Third Step: Register application and get Client ID and Client secret.

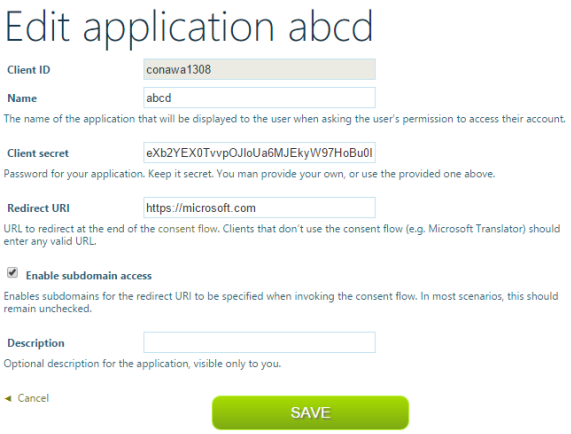



Figure 4-114 Register application and get Client ID and Client secret

Fourth Step: The client ID and Client Secret registered above will be used in translation as illustrate below.

```
Translate.setClientId("conawa1308");
Translate
       .setClientSecret("eXb2YEX0TvvpOJloUa6MJEkyW97HoBu0IzlpdygPWHI=");
if (lang.equals("en")) {
       Out = Translate.execute(ProName, Language.ENGLISH,
       Language.CHINESE_SIMPLIFIED);
} else {
       Out = Translate.execute(ProName,Language.CHINESE_SIMPLIFIED,
       Language.ENGLISH);
}
```

Code Segment 4-33 Language Translation

Later on, product name with both language will sent to database to do like search. Database will retrieved out all the product that related to product name. The data that will be retrieved out listed as below:

```
productList.SPID ,productList.marketID ,productList.ProductID,
productList.LatestUpdate, productList.latitude, productList.longitude,
productList.gloMarketID,productList.ProductName,productList.ProductCategor,
productList. product_QuanDimen1, productList.product_QuanDimen2,
productList.product_barcode,productList.ProductDescription,
productList.product_QuanDimen1Unit, productList.product_QuanDimen2Unit,
productList.MarketName, productList.LatestPrice, productList.MarketCity
```

Code Segment 4-34 Populated Product List Data

Right after all the data has been retrieving, another API will be called by Http Get Method to calculate distance of market with current location.

```
https://maps.googleapis.com/maps/api/distancematrix/json?origins=latitude,
longitude&destinations=productList.latitude,productList.longitude&mode=driving&
language=en-EN&sensor=true&key=AIzaSyD4SpLzS4x6KyKns1-o0Gh7C2oPNOBt28g
```

Code Segment 4-35 Distance Matrix API

Below show the sample response from API

```
{
   "destination_addresses" : [ "San Francisco, CA, USA" ],
   "origin_addresses" : [ "Vancouver, BC, Canada" ],
   "rows" : [
      {
         "elements" : [
            {
               "distance" : {
                  "text" : "1,528 km",
                  "value" : 1528359
               },
               "duration" : {
                  "text" : "15 hours 6 mins",
                  "value" : 54341
               },
```

```
               "status" : "OK"
            }
         ]
      }
   ],
   "status" : "OK"
}
```

Code Segment 4-36 Sample Response from API

The Json Data need to be parse in order to retrieve the data.

```
JSONObject jObj = new JSONObject(jsonn);
JSONArray array = jObj.getJSONArray("rows");
JSONObject rows = array.getJSONObject(0);
JSONArray elements = rows.getJSONArray("elements");
JSONObject steps = elements.getJSONObject(0);
JSONObject distance = steps.getJSONObject("distance");
double Distance = distance.getDouble("value") / 1000;
JSONObject duration = steps.getJSONObject("duration");
timetaken = Parse.Double(duration.getString("value"))/60;
```

Code Segment 4-37 Parsing Json Data return from Distance Matrix API

After we get the distance and time taken travel to the market, we will filter out those market which distance that are not fall within user preferences (maximum nearby distance to travel in settings).

#### 4.5.5.1.2 Sort Product List

```
Collections.sort(finalPdList,new Comparator<ProductList>() {
       @Override
       public int compare(ProductList lhs,ProductList rhs) {
       return Double.compare(lhs.LatestUpdate, rhs.LatestUpdate);
       }
});
```

Code Segment 4-38 Product List Date Time Sorting

Comparator can be used when we want to sort object based on user defined variable in the object. To do this, we need to provide Comparator interface. This method will do auto sorting and the output produce are already in correct order. This can do to any type of ArrayList or collection and it support all kind of object such as String, Double, Integer. For example: Double.compare(lhs.distance, rhs. distance);

Below show the result after sorting.


Figure 4-115 Search result after sorting

This has been implemented in Search Result Activity. Same as previous, this activity will load the image from parse.com by using the same way as discussed in loading profile picture in settings module.

#### 4.5.5.1.3 Nearby Filter

User can click search icon in actionbar to continue further search or click the nearby icon (rightmost in actionbar) to change the Product Result with different distance. After filter button is clicked, a new set of Product Result will be fetch from database.

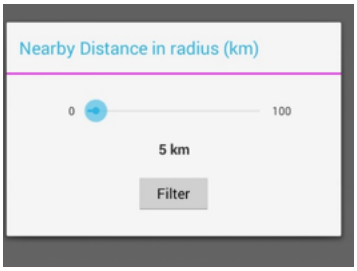



Figure 4-116 Nearby Filter

#### 4.5.5.1.4 Product Detail

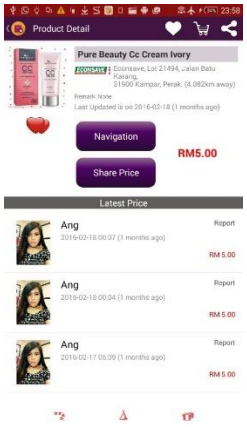



Figure 4-117 Product Detail Screen Flow

This page will be displayed whenever user click on the Product in search result page.

#### 4.5.5.1.5 Navigation

First purple color button is navigation button and it will open a list of navigation apps to let user choose their desire application. After user choose one of the application, the target application will open and the application immediately show the navigation route to market without any extra input from user anymore.

```
String uri = "geo:0,0?q=destlatitude, destlongitude"
startActivity(new Intent(android.content.Intent.ACTION_VIEW, Uri.parse(uri)));
```

Code Segment 4-39 Navigation through Waze or Maps

#### 4.5.5.1.6 Share Price


Figure 4-118 Share Price

User are allowed to share price in any product. The latest price will be on top of list and the latest price list are sort base on share price date in descending order. This is the case when the product do not have any price which update by official market account. However if particular having official updated price, the official price will always place on top before normal user updates.

#### 4.5.5.1.7 Report Price

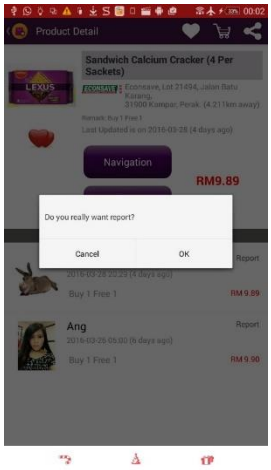



Figure 4-119 Report Price

User can report the price share by others (exclude official account price). In the above example, first price which shared by donkey are reported. If the user had reported particular price previously, the system would not allow and will display error message as below:

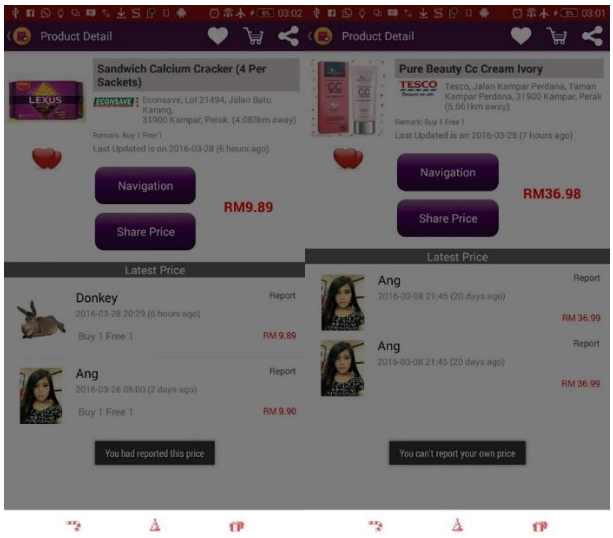



Figure 4-120 Reported Price

Furthermore, user are not allowed to report their own price. The reported price will add into database by insertion operation. The application will check in database whether particular user had reported specific price. If database does not return any row that mean the user are allowed to report particular price. Thus insertion will be execute with reportDateTime as currentTimestamp.

```
String queryCheck = "SELECT [ReportID] from [Conawa].[dbo].[Report] where
[ReportUserID]=? and [ReportUpdateID]=?";
                    PreparedStatement psCheck =
conn.prepareStatement(queryCheck);
                    psCheck.setString(1, ReportUserID);
                    psCheck.setString(2, ReportUpdateID);

                    ResultSet rsCheck = psCheck.executeQuery();

if (!rsCheck.next())
{

String query = "INSERT INTO [Conawa].[dbo].[Report]
VALUES(CURRENT_TIMESTAMP,?,?,?)";
PreparedStatement ps = conn.prepareStatement(query);
ps.setString(1, ReportUserID);
ps.setString(2, ReportUpdateID);
ps.setString(3, "N");
// Statement state = conn.createStatement();
int rs = ps.executeUpdate();
```

Code Segment 4-40 Report Price Web Service Scripting

#### 4.5.5.1.8 Action Bar Menu

Action Bar Menu also called as Option Menu. These are placed on top right of interface. Following example show the step of adding favorite icon on action bar:

```
public boolean onPrepareOptionsMenu(Menu menu) {

       MenuItem mi = menu.add("favourite");
       mi.setIcon(R.drawable.hearts);
       mi.setShowAsActionFlags(MenuItem.SHOW_AS_ACTION_IF_ROOM);
       mi.setOnMenuItemClickListener(new OnMenuItemClickListener() {
           @Override
           public boolean onMenuItemClick(MenuItem arg0) {
              //Do anything at here
           }
      });

       return super.onPrepareOptionsMenu(menu);
}
```

Code Segment 4-41 Add Option Menu Programmatically

Firstly, a menuItem named favorite is added to Menu, the icon of menu set as R.drawable.hearts.Secondly,mi.setShowAsActionFlags(MenuItem.SHOW_AS_ACTION _IF_ROOM) will allow all menuItem align on top horizontally instead of compress as drop down box.

#### 4.5.5.1.9 Database Handler

Sqlite is local database that stores data to a database file on a android, the database file (.db extension is hidden in android device unless the device is rooted). To access this database, one do not need to establish any connections (e.g. JDBC,ODBC) like SQL Server, Oracle database. In here we just need to create a class extends database open helper and all the insertion, updating, deletion, creation of data to sqlite local database in application later just required call the method inside this class:

```
public class DatabaseHandler extends SQLiteOpenHelper {}
```

Code Segment 4-42 Database Handler Helper Class

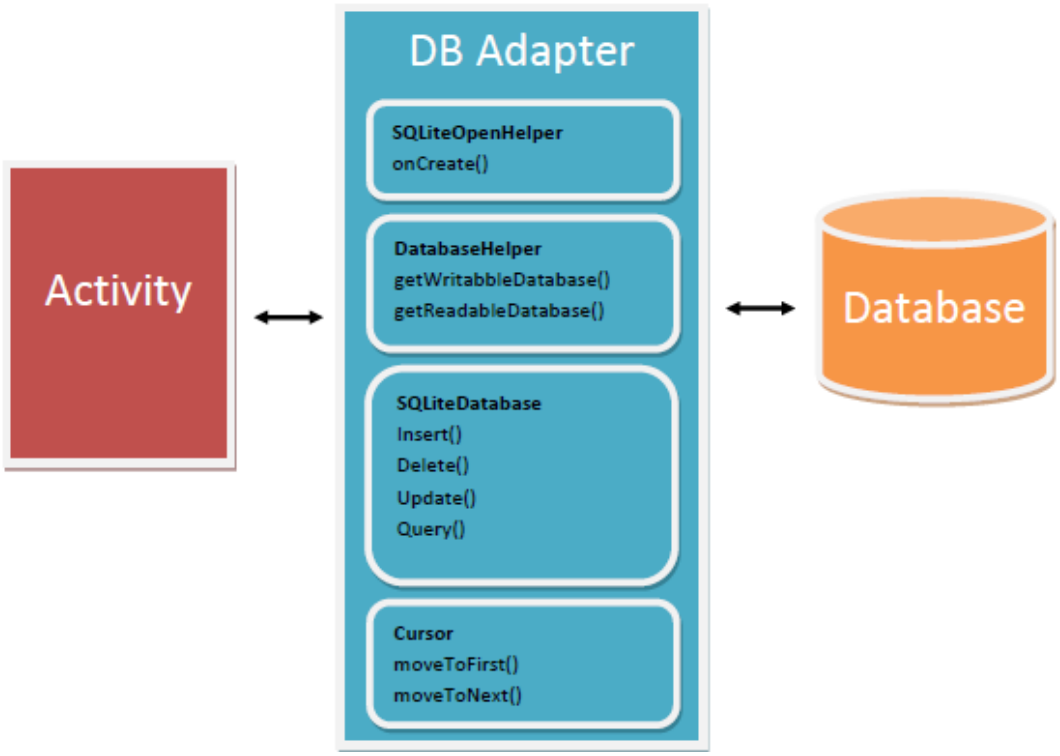



Figure 4-121 DB Adapter

There are several method that are required to implement in this class:

**Constructor**

```
private static final int DATABASE_VERSION = 1;
public DatabaseHandler(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
```

Code Segment 4-43 Database Handler Constructor

Table 4-6 SQLiteOpenHelper method

| **Method implement from SQLiteOpenHelper** | **Description** |
|---|---|
| onCreate (SQLiteDatabase db) | Will be called when database created, we write all create statements here. |
| onUpgrade (SQLiteDatabase database, int oldVersion, int newVersion) | When database is upgrade or modify from existing structure or column etc. |
| void addData(Data data) | Create/ add data operation, accept Data object as parameter. |
| Data getData(int id) | Read data operation, accept integer key as parameter and return retrieved data. |
| int updateData(Data data) | Update data operation, accept Data object as parameter and return number of row have been updated. |
| Void deleteData(Data data) | Delete data operation |
| ArrayList<Data> getAllData(int id) | Retrieved all data operation, accept integer key as parameter and return retrieved data as ArrayList. |

When there is a needs to upgrade database, we need to change database version to a higher value.

```
private static final int DATABASE_VERSION = 1;
to
private static final int DATABASE_VERSION = 2;
```

Code Segment 4-44 Database Version

#### 4.5.5.1.10 Add To favourite List

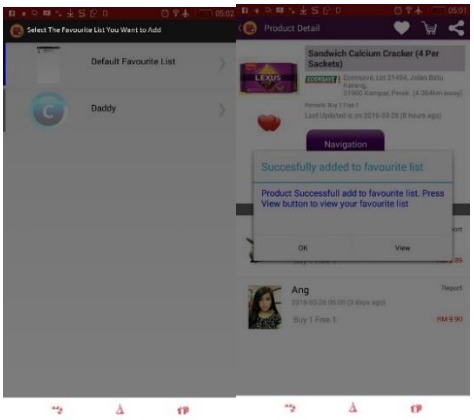



Figure 4-122 Add To Custom Favorite Selector

Figure 4-123 Add to Favorite list

The love button under product image or actionbar can let user save particular product to their favourite list. After clicked either button, another activity will display to let user to choose whether which favourite list user wish to add. For example, if user click “Daddy”, then the product will be add to daddy list and a confirmation pop up will display. User can click “ok” to close the dialog pop up box and continue other operation or click “view” to view “Daddy List”. The application will first check whether the product already exist in particular list from Sqlite Database.

```
DatabaseHandler db;
if (db.getFavouriteCount(PdItem.ProductID) == 0) {
	Log.d("Insert: ", "Inserting ..");
	Intent i = new Intent(context, FavouriteChoice.class);
	startActivityForResult(i, 5);
}
```

Code Segment 4-45 Check Favorite List exist same product

If the product not exist yet, system will start new intent activity to open new activity to let user choose the favorite list they wish to add. On the other hand, if the product already added previously, the application will directly display pop up and inform user “The product already in your favorite list”. User can Press “View” button to view daddy favorite list” or press “OK” to cancel the pop up dialog box and continue other operation.

```
DatabaseHandler db = new DatabaseHandler(context);
db.addFavourite(favouriteList);
```

Code Segment 4-46 Add Favorite List

#### 4.5.5.1.11 Add To Cart

Cart is different from favorite list. Basically, favorite list will store the products that are favorite by user. It act as bookmark to save the product in favorite list whereas cart store the product that the user wish to buy at that moment. For example, user may add Milo to their favorite list but not refrigerator since that may not be their favorite product. However if the refrigerator in their spoil suddenly, they may need buy it as soon as possible. In this case they can directly add this product to cart list but not favorite list. Besides that, when user scroll through their favorite list, they can add the product from favorite list to cart when they really decide want to buy the favorite products (e.g. Milo 3 in 1 2KG). After product added to the cart, application will suggest some ideal store base on cart list. This is under ideal store suggestion module and will be discussed in next part.

```
DatabaseHandler db = new DatabaseHandler(context);
if (db.getCartCount(PdItem.ProductID) == 0) {
       // Assignment data into cartData object
       String mydate = java.text.DateFormat.getDateTimeInstance()
                                   .format(Calendar.getInstance().getTime());
       cartData.cartAddDT = mydate;
       db.addCart(cartData);
}
```

Code Segment 4-47 Add Product into Cart

When the cart button is clicked, ProductID will used as primary key to check in sqlite Database whether the product exist in cart list. The product will be insert to cart if not exist else it will display status message to indicate the product already added previously and prompt user whether want to view the product. When the product insert to cart, we need to insert addDateTime as well, thus we use Calender object to generate current Date and Time.

#### 4.5.5.1.12 Share to Social Media

There is a standard share button at the right most of actionbar. This is create by menuItem as show below:

```
       MenuItem mi3 = menu.add("sharing");
              mi3.setIcon(R.drawable.sharable);
              mi3.setShowAsActionFlags(MenuItem.SHOW_AS_ACTION_IF_ROOM);
```

Code Segment 4-48 Share Social Media Menu Item

User can sent product viewed to social media by code below:

```
Intent sendIntent = new Intent();
sendIntent.setAction(Intent.ACTION_SEND);
sendIntent.putExtra(Intent.EXTRA_TEXT, "SomeText ")
sendIntent.setType("image/jpeg");
sendIntent.putExtra(Intent.EXTRA_STREAM, PdItem.proImg);
startActivity(Intent.createChooser(sendIntent,"Share product to.."));
```

Code Segment 4-49 Share Social Media Apps Chooser

The outcome show as below, the application will list out all application chooser to let user choose which application to send.

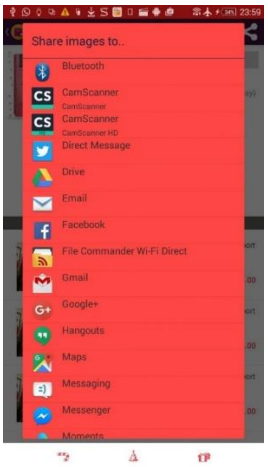



Figure 4-124 Application Chooser

### 4.5.5.2 Advanced search

In this search option, a full page google Map with market marker will be displayed. User will be able to know their current location and nearby store. This is useful when the user new to the environment and wish to know the store that exist surrounding. To place the map fragment on activity, there are some step that must follow closely in order to make the map to work.

#### 4.5.5.2.1 Google Play Services API

Prior importing the API, developer must register themselves in Google console developer as google developer to get a signing API key. The Google Play services will worked only if you get the Android Certificate and Google Map API Key since Google Map API required API Key to authenticate the device to Google Server.

**First Step:** Create New Project

Go to Google Developers Console https://console.developers.google.com/project and create a new project and enable all required API in API tab (only enable those API that will be using).

**Second Step:** In the sidebar, select credentials in API manager to obtain API key. You will first see screen below.

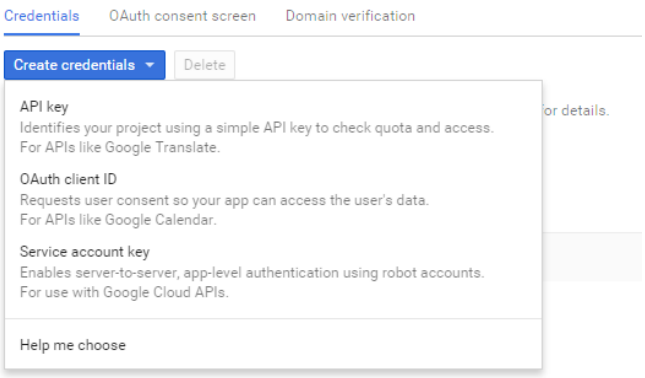



Figure 4-125 Credentials in API manager to obtain API key

Click on API key and next, a pop up will display

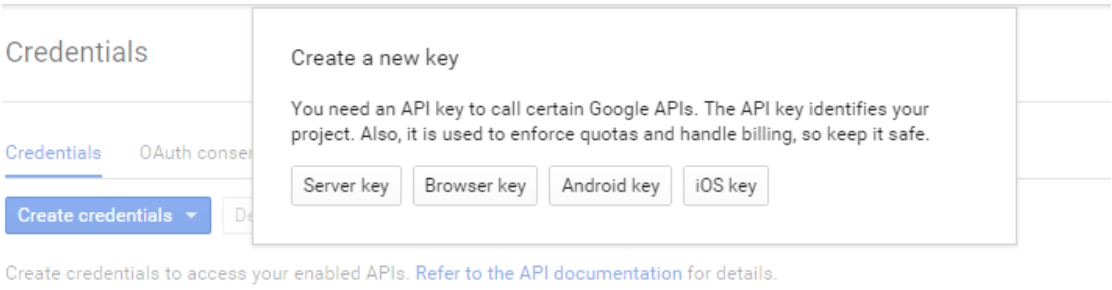



Figure 4-126 Create a Server Key

Next, you need to enter the application SHA-1 fingerprint and package name (package name must be same as the package name use in the project).

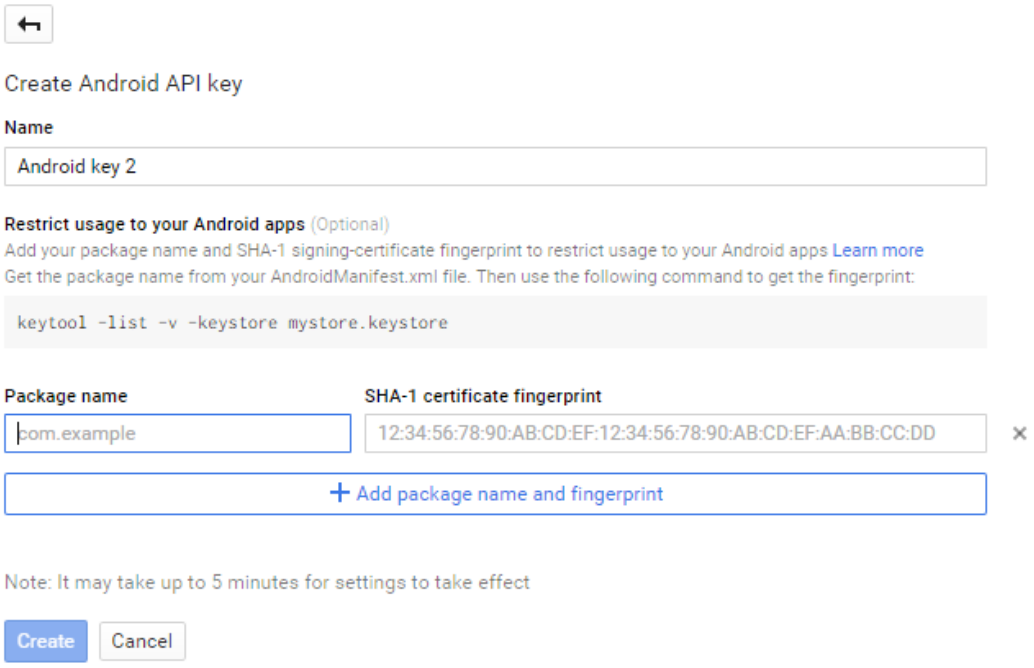



Figure 4-127 Insert Certification Fingerprint

which can be obtained by two method as demonstrate below:

**First method**: by keystrore

```
Keytool -list -v –keystore %USERPROFILE%\.android\debug.keystore" -alias
androiddebugkey
-storepass android -keypass android
```

Code Segment 4-50 Get SHA 1 fingerprint

**Second method**: by build number

Go to windows -> preferences -> Android -> build -> SHA1 fingerprint

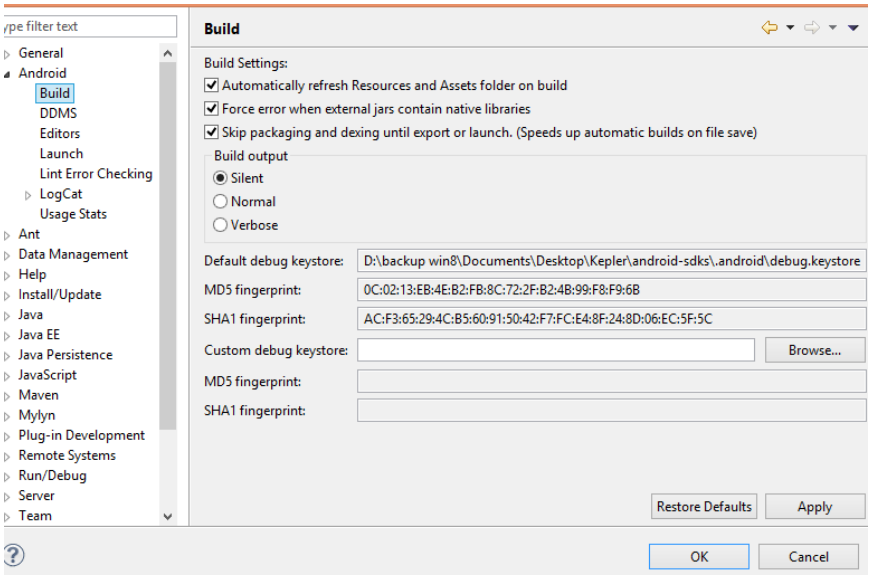



Figure 4-128 Build number to get SHA 1 Fingerprint

Third Step: Get the SHA1 fingerprint and paste in credentials. The API key will generate within 1 minute. Get the generated key and place it in the application manifest file.

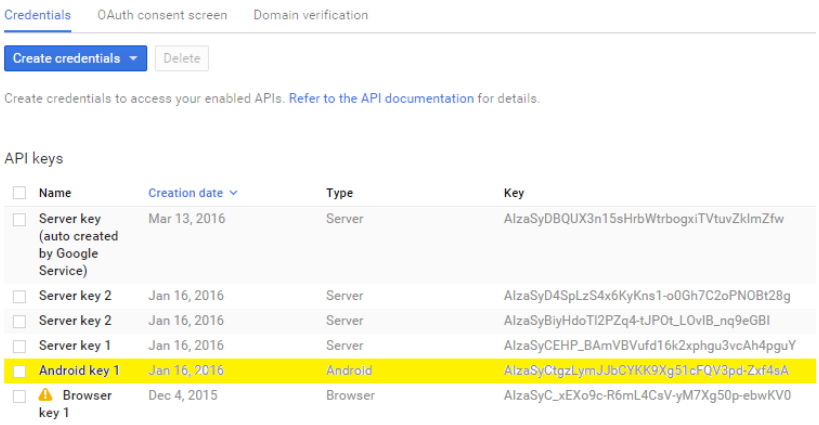



Figure 4-129 API Key to put in application manifest file

```
<meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="AIzaSyCtgzLymJJbCYKK9Xg51cFQV3pd-Zxf4sA" />
        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version" />
```

Code Segment 4-51 Google Play Servie in Manifest

#### 4.5.5.2.2 Permission

Besides that, another things that need to be noted is place related permission in manifest file. The permission required listed below:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

Code Segment 4-52 Permission for Accessing Google Map

The permission listed in manifest file are the permission that application will gain these permission from user before installing the application to allow application access to these hardware and features.

#### 4.5.5.2.3 OpenGL

Google use OpenGL ES Version 2 to render the map view of the Google Map thus the piece of code show below must be added to made the map show:

```
<uses-feature
   android:glEsVersion="0x00020000"
   android:required="true"/>
```

Code Segment 4-53 OpenGL accessing Google Map

OpenGL ES Version 2 let the device rendering 3D map by power of GPU.

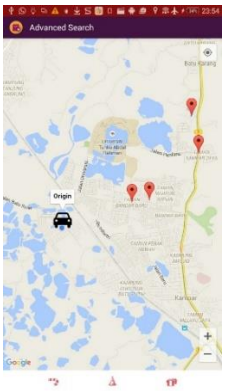



Figure 4-130 Google Map on Advanced Search

When first load this activity, application will detect current user position (coordinate) and place a marker onto the map indicate current position. By this, user can know their exact location and will have better idea whether which market are nearer.

```
marker = map.addMarker(new MarkerOptions()
       .position(loc)
       .icon(BitmapDescriptorFactory
       .fromResource(R.drawable.origin)).title("Origin"));
marker.showInfoWindow();
```

Code Segment 4-54 Add Marker of Current Location on Google Map

At the same time, UI thread will assign a worker thread to request server and retrieved all the nearby market list. The worker thread will sent user current location to server and based on user current location. The server will retrieve database market data (coordinate) and use distance matrix API to get the market distance. Later, system will sort the market based on distance accordingly. After all market retrieve successfully back to application, the application will loop through market list, get the market coordinate and place the marker on map.

```
for (int i = 0; i < marketmarkerFinal.size(); i++) {
       LatLng loctemp = new LatLng(marketmarkerFinal
              .get(i).latitude, marketmarkerFinal
              .get(i).longitude);
       marker2.add(map.addMarker(new MarkerOptions().position(loctemp).
       title(marketmarkerFinal.get(i).MarketName.toUpperCase()+","+
       marketmarkerFinal.get(i).MarketCity)));
}
```

Code Segment 4-55 Add Marker of Market on Google Map

Every marker are set with listener so they will be able to listener to click event. When user click on the marker, application will assign worker thread to fetch related market information such as market image, market contact number and so on from server and database.

```
map.setOnMarkerClickListener(new OnMarkerClickListener() {
       @Override
       public boolean onMarkerClick(Marker arg0) {
              for(int i=0; i<marker2.size();i++)
              {
                     if(marker2.get(i).equals(arg0))
                     {
                           MyThread connectThread = new
                           MyThread(marketmarkerFinal.get(i), mHandler);
                           connectThread.start();
                           break;
                     }
```

```
                }
                return true;
        }
});
```

Code Segment 4-56 Set Listener to Marker

When user click on the marker, a partially transparent activity will float on top of google Map as show below. If user wish to close this activity and continue other operation, they can always click on the "cross" button on the right top of screen.

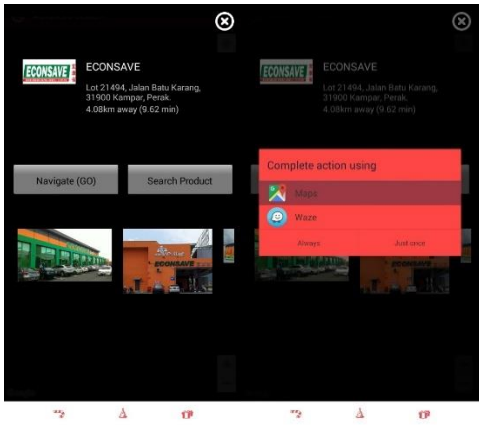



Figure 4-131 Partially Transparent Activity when click on marker on map

#### 4.5.5.2.4 Navigation

Navigate button on the left will redirect user to navigation apps. When the button is clicked, navigation apps chooser will display and let user to choose whether which application they wish to use to navigate them to the market. This has been discussed in search By Product Name and semantic autocomplete module.

#### 4.5.5.2.5 Search Product

If user wish to discover more product in particular market after they saw this pop up, user can click on search product button and search result screen ( same as search by Product Name and Semantic Autocomplete module) will be display (sort by latest update time by default). The screen display all the product in particular market only.

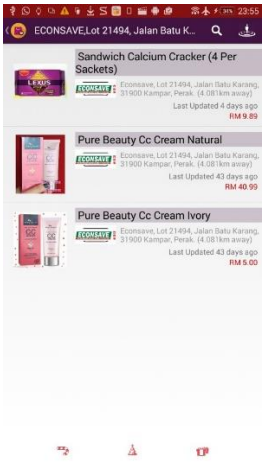



Figure 4-132 Search Product Result in Market

### 4.5.5.3 Search by Barcode Scanning

#### 4.5.5.3.1 Camera Permission

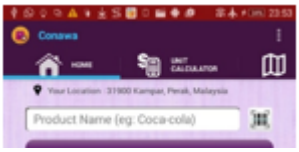



Figure 4-133 Scan Product Barcode

The barcode icon that place at right of Search Box is for barcode searching. User can scan the product and all the product of that barcode will return and sort distance in ascending order.


Figure 4-134 Detecting Product Barcode

This is the third option to search for product. User can use barcode scanner to scan the product on their hand. They must focus the camera to the product barcode and the scanning should be carry out in bright environment. After the camera successfully focus and detect on the barcode, the application will request worker thread to retrieved product list data at nearby market from server based on the barcode. The result will be sort by update date and time but user can further filter the result to smaller distance range by the nearby distance icon which had discussed in previous module. There are few permission need to add in manifest.xml in order access to device camera.

```
    <uses-permission android:name="android.permission.CAMERA">
    <uses-feature android:name="android.hardware.camera">
    <uses-feature android:name="android.hardware.camera.autofocus">
```

Code Segment 4-57 Permission to access Camera

#### 4.5.5.3.2 Zbar API

With Zbar library API, barcode/ QR code scanning becoming simple. This library able to differentiate multiple type of barcode/ QR code. Basically we are creating an intent and calling cameraScanning activity with StartActivityOnResult(intent) in the scanner icon button listener and returning the scanned result once successfully detected by:

Calling Scanner:

```
Intent intent = new Intent(getActivity().getApplication(),
                                  CameraActivity.class);
startActivityForResult(intent, Request_Barcode);
```

Code Segment 4-58 Call Zbar Scanner

Result Returned:

```
intent=new Intent();

intent.putExtra("barcode",codeNum);

setResult(1,intent);
```

Code Segment 4-59 Get Return Barcode from Zbar API

```
autoFocusHandler = new Handler();
```

```
mCamera = getCameraInstance();

/* Instance barcode scanner */

scanner = new ImageScanner();

scanner.setConfig(0, Config.X_DENSITY, 3);

scanner.setConfig(0, Config.Y_DENSITY, 3);

mPreview = new CameraPreview(this, mCamera, previewCb, autoFocusCB);

FrameLayout preview = (FrameLayout) findViewById(R.id.cameraPreview);

preview.addView(mPreview);
```

Code Segment 4-60 Implementation of Detecting Barcode

The codes above are the implementation of barcode scanner. Prior using ImageScanner class, we need to download zbar API and import it as library together with set build path to this library.

After the barcode successfully detected, we have to release camera the API in onStop() method.

```
if (mCamera != null) {
	previewing = false;
	mCamera.setPreviewCallback(null);
	mCamera.release();
	mCamera = null;
}
```

Code Segment 4-61 Release Camera After Succesfully Detected

SetResult method will return the result to home tab and camera Activity finish. When using StartActivityForResult to open new activity, the result return will trigger the method below instead of reload home tab fragment.

```
public void onActivityResult(int requestCode, int resultCode, Intent data) {
	if (requestCode == Request_Barcode) {
	    barcodeNum = data.getStringExtra("barcode");
	    MyThread connectThread = new MyThread(CurrentLoc,
	          disTra,barcodeNum, SuggestList, mHandler);
```

```
            connectThread.start();
        }
}
```

Code Segment 4-62 Search Product By Barcode

MyThread will sent request to server and retrieved related product that match with barcode number and filter based on user current location.

#### 4.5.5.4 Search by Category


Figure 4-135 Search By Category

All this icon represent different category of product. User can click on their desire category to view the product in those category.

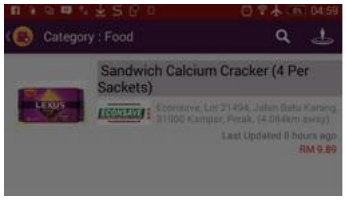



Figure 4-136 Product Search Result by Category

Similar as search by product name, but in this option, when the category icon is clicked, the category id instead of name string will sent to server:

```
MyThread5 connectThread5 = new MyThread5(disTra, catID, mHandler);
connectThread5.start();
```

Code Segment 4-63 Create Worker Thread and Passing Category ID

and the following servlet is called:

```
HttpClient httpClient = new DefaultHttpClient();
```

```
HttpPost httpPost = new HttpPost(
      "http://192.168.43.78:8082/abcdWS/getProductByCat");

nameValuePair.add(new BasicNameValuePair("categoryID", catID));

httpPost.setEntity(new UrlEncodedFormEntity(nameValuePair));

HttpResponse response = httpClient.execute(httpPost);
```

Code Segment 4-64 Call Http Post method to pass data to web service

After the server return result in Json format, we need to parse the JsonArray and get all the product in this category then sort according to latest date time or distance by distance matrix API. (Shown in produce name option module)

#### 4.5.5.4.1 Add To Unit Calculator

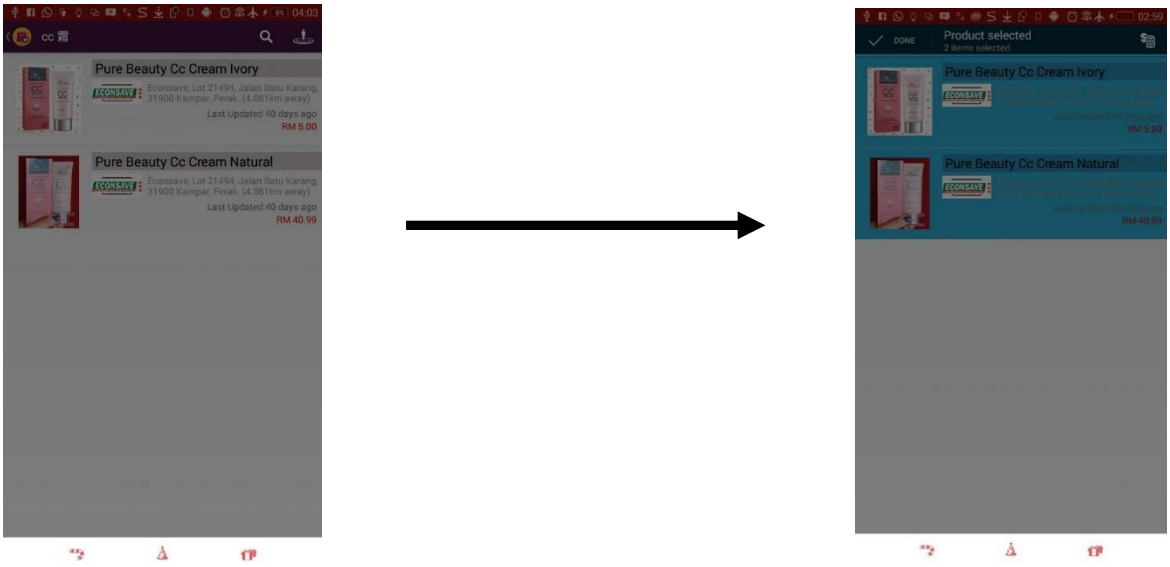



Figure 4-137 Search by Chinese and Add to Unit Calculator from Search Result

In Search Result page, the product item accept both type of event named click and long click event. Both these event handle by different listener.

On click listener will open product detail activity whereas On long click event will trigger on long click listener and Floating Context Menu will be display. When user long click the item, a contextual action bar appears on top of screen and overwrite the exist action to allowed user perform some action on currently selected item(s).

#### 4.5.5.4.2 Contextual Action Bar

User can select one or more items or deselect item after first long click event by just touching them. In here, either all the items deselected, back key are press or the check mark button (top left) from contextual action bar is click will dismiss

The contextual action bar and also removes all the selection on data. To make this feature work, the listview must set a listener on it and set choice mode as multiple modal.

```
list.setMultiChoiceModeListener(new MultiChoiceModeListener() {});
list.setChoiceMode(list.CHOICE_MODE_MULTIPLE_MODAL);
```

Code Segment 4-65 Contextual Action Bar

There are few method that must implement in this listener:

Table 4-7 Method Implemented from MultiChoiceModeListener

| | |
|---|---|
| public boolean onCreateActionMode(ActionMode mode, Menu menu){} | Action mode first created when long click on item, supply it a menu.xml layout and it will generates the button on top right of action bar |
| public boolean onPrepareActionMode(ActionMode mode, Menu menu) {} | Whenever it is invalidated, this will be called to refresh the action mode's action menu |
| public void onItemCheckedStateChanged(ActionMode mode, int position, long id, boolean checked) {} | Will be trigger when item is checked or unchecked |
| public boolean onActionItemClicked(ActionMode mode, MenuItem item) {} | Trigger when user clicks on any of action bar button |
| public void onDestroyActionMode(ActionMode mode) {} | When action mode = false or contextual action bar going to dismiss |

#### 4.5.5.4.3 Add Operation when item click

In application, all the Sqlite saving operation will be done in the method below:

```
public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
       SparseBooleanArray selected = list.getCheckedItemPositions();
       StringBuilder message = new StringBuilder();
       DatabaseHandler db = new DatabaseHandler(getApplication());
       for (int i = 0; i < selected.size(); i++) {
              if (selected.valueAt(i)) {
                  db.addFussy(PdList.get(selected.keyAt(i)));
               }
       }
       mode.finish();
}
```

Code Segment 4-66 Method Trigger When Item Long Click

If there is no error happened, a pop up will display the unit price and prompt user to view unit calculator history list. The last line in method which is mode.finish method is to dismiss the contextual action bar when saving action done.

### 4.5.6 Catalogue Module

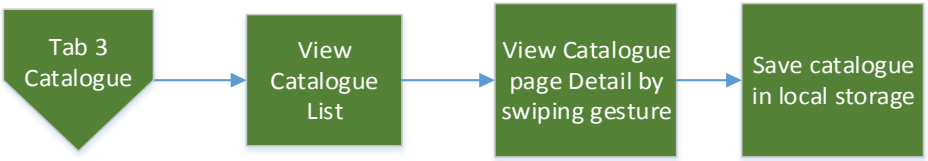



Figure 4-138 Catalogue Module Screen Flow

Figure 4-139 Catalogue List Screen

Catologue class as show below has been used to populate the image of catalogue.

```
public class CatalogClass  implements Serializable{
    public String brochureID;
    public String marketName;
    public byte[] prgmImages ;
    public String broStart,broEnd;
    public String broImgAdd;
    public Integer totalBroPage;
}
```

Code Segment 4-67 Model Class of Catalog

All the catalogue cover page will be display in the grid view and user are allowed to click on any of catalogue to view the catalogue detail.

Below show the grid view layout of catalogue list.

```
    <GridView
        android:id="@+id/gridView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView1"
        android:numColumns="3" >
    </GridView>
```

Code Segment 4-68 Layout of Catalogue

```
    <TextView
        android:id="@+id/tv_empty"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="58dp"
        android:text="Loading data..."
        android:textSize="18dp" />
```

Code Segment 4-69 Layout Display when no data populated yet

In java catalogue activity class we will inflate the grid.xml layout and find the gridview so that we can set all the image as adapter in gridview. When the adapter does not populated any data yet (still in retrieving process from server), gridview.setEmptyView method will display the textview that contains "Waiting For Retrieving".

```
rowView1 = inflater.inflate(R.layout.grid, null);
gridview = (GridView) rowView1.findViewById(R.id.gridView1);
gridview.setEmptyView(rowView1.findViewById(R.id.tv_empty));
gridview.setAdapter(new CustomAdapter(context, catalog));
```

Code Segment 4-70 Set Adapter to GridView for Catalogue list

Initially the catalog arraylist is an empty list, as soon as the images and data retrieved successfully from server and parse.com cloud storage then put in Catalog ArrayList object, the adapter will able to detect that there are changes in Catalog ArrayList if we put the 2 line of code below after the catalog arraylist populate all the data. It will update the gridview automatically with new loaded catalogue.

```
catalog.add(catalogSingle);
adapter.notifyDataSetChanged();
list.invalidateViews();
```

Code Segment 4-71 Method call After all data populated succesfully

Second line in code above `adapter.notifyDataSetChanged` use to notify the adapter that there are changes in Catalog ArrayList and third line is called list to re draw the overall list layout.

The query below has been used to retrieving the catalogue and information:

```
Select [brochureID],[broStart],[broEnd],[marketName],[broImgAdd],[broPage]
from [Conawa].[dbo]. [Brochure] "
where [broEnd] >= CAST(CURRENT_TIMESTAMP AS DATE) or
       [broEnd] is null
ORDER BY marketName ";
```

Code Segment 4-72 SQL Query for retrieving Catalogue

#### 4.5.6.1 Catalogue Detail


Figure 4-140 Catalogue Detail Swipe View Tab

After all the catalogue display on screen, user are allowed click on any catalogue to view the detail by swipe view. The data were populate in class below, it implements Comparable interface in order to sort the catalogue according to page number so CatalogDetail must implement compareTo(CatalogDetail o) method.

```
public class CatalogDetail  implements Serializable,Comparable<CatalogDetail>{
    public String brochureID;
    public String brochureIMGID;
    public byte[] prgmImages ;
    public int broPageNum;
    public int compareTo(CatalogDetail o) {
                return (broPageNum - o.broPageNum);
        }
}
```

Code Segment 4-73 Catalogue Detail Model Class

### 4.5.7 Share Item Module

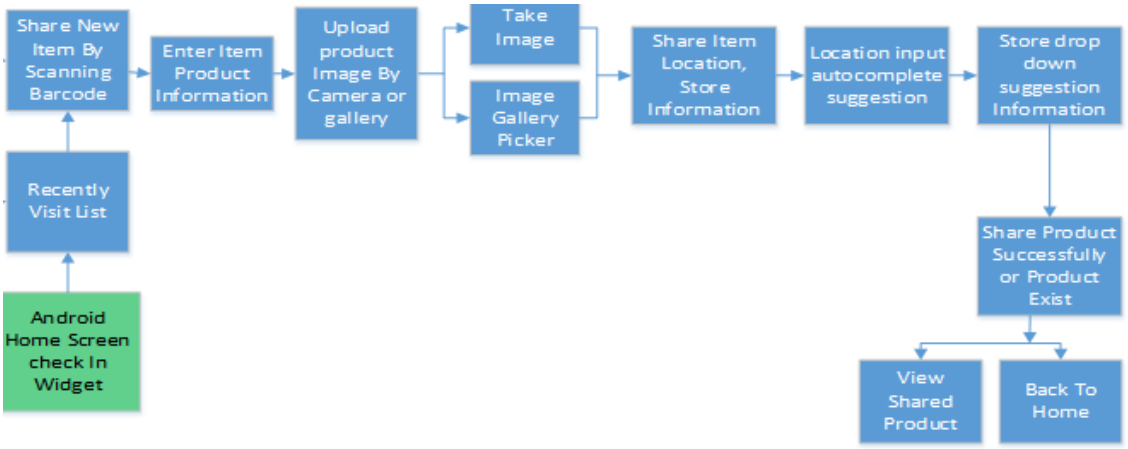



Figure 4-141 Share Item Module Screen Flow

#### 4.5.7.1 Location Check In Widget

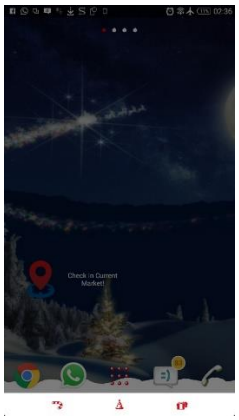


Figure 4-142 Location Check in Widget

There are few step that we must follow to create a widget in this application:

Step 1. Create a new widget layout file in layout folder.

```
    <Button
        android:id="@+id/buttonwid"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@android:color/transparent"
        android:drawableLeft="@drawable/checkin"
        android:text="Check In Current Market!" />
```

Code Segment 4-74 Android Desktop and lock screen widget layout

Step 2. Create a new folder call XML and create a file name widgetAppnfo in this which define the properties of widget such as size, frequency of update and so on.

The widget size is flexible (e.g. make it larger or smaller) by adding resizeMode="horizontal|vertical" whereas widgetCategory="home_screen|keyguard can make the widget possible too add as home screen or lock screen widget

```
<appwidget-provider xmlns:android="http://schemas.android.com/apk/res/android"
       android:minWidth="180dp"
       android:minHeight="142dp"
       android:updatePeriodMillis="1000"
       android:initialLayout="@layout/widgetlayout"
       android:widgetCategory="home_screen|keyguard"
       android:resizeMode="horizontal|vertical">
</appwidget-provider>
```

Step 3: Create a receiver name AppWidgetProvider to build user interface and listener

```
public class AppWidgetProvider extends AppWidgetProvider   {}
```

Code Segment 4-75 Service Provider for Widget

Similar with application life cycle, widget has its own cycle. When the class extends AppWidgetProvider, it must implement onUpdate method. This method will be called for every update of the widget.

Step 4: Define broadcast receiver with intent filter for android.appwidget.action. APPWIDGET_UPDATE in manifest. Meta-data must be specify as well via a attribute call android:name="android.appwidget.provider. The configuration file referred to widgetAppnfo which created in step 2.

```
<receiver
       android:name="com.javapapers.android.swipetablayout.app.AppWidgetProvider" >
          <intent-filter>
       <action     android:name="android.appwidget.action.APPWIDGET_UPDATE" />
            </intent-filter>
            <meta-data
                android:name="android.appwidget.provider"
                android:resource="@xml/widgetAppnfo " />
</receiver>
```

Code Segment 4-76 Receiver Define for Widget

#### 4.5.7.2 Visit History

When check in widget is clicked on home screen or lock screen, user current location will be translate to address by geocoder. Both coordinate and full location address will be recorded into Sqlite local database.

```
DatabaseHandler db = new DatabaseHandler(context);
CheckInlist ckL = new CheckInlist();
ckL.latitude = latitude;
ckL.longitude = longitude;
Geocoder geocoder = new Geocoder(getApplicationContext(),
Locale.getDefault());
List<Address> addresses = geocoder.getFromLocation(ckL. latitude,
                          ckL. longitude, 1);
ckL.address= addresses;
db.addCheck(ckL);
```

Code Segment 4-77 Inserting User Current location


Figure 4-143 Visit History

With this, user can refer back the market they went before and they can click on share product button to share the product that they had bought.

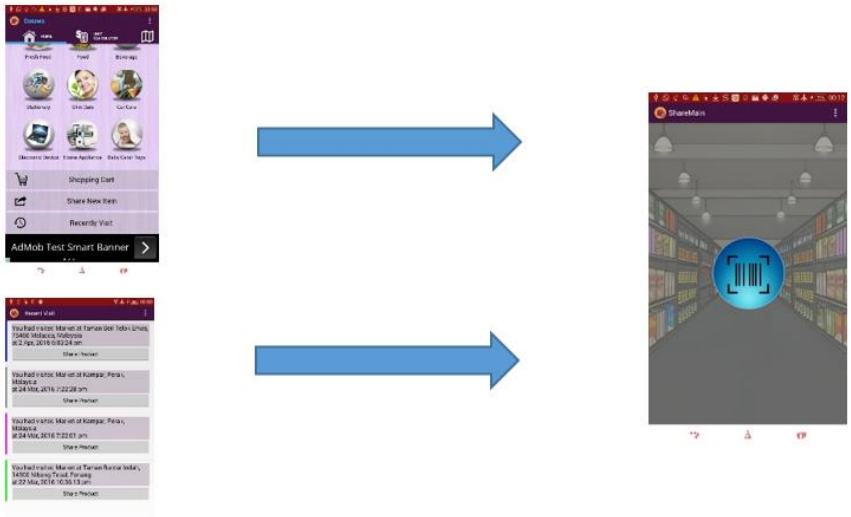



Figure 4-144 Share Product Demonstration

Other from clicking share product button in visit history, there is another option to share product which is click share item in main home tab. If user access share item page from main page, the location of shared product will be user's current location by default whereas if user access from visit history, the visited location will be the location of shared product. Thus the visited history page can let user recall the place they went and reduce the needs of enter market location data whenever they wish to share any new data.

#### 4.5.7.3 Share Item

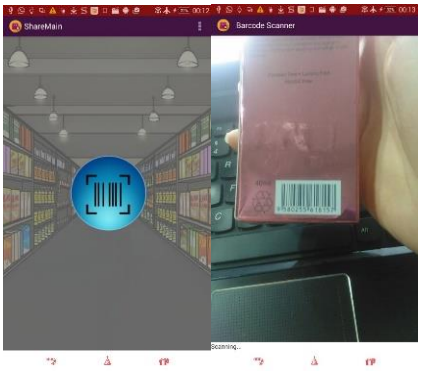



Figure 4-145 Scan Product Barcode Page

Figure 4-146 Camera Scanning Page

When user wish to share any item, the first action that user must do is scan barcode of product. User can click on the scan icon button in middle of screen, then system will access camera and start detecting product barcode. This is the portion of code that implement in the scan button listener.

```
IMGbarcodeshare.setOnClickListener(new OnClickListener() {
	public void onClick(View v) {
		Intent i = new Intent(context, CameraActivity.class);
		startActivityForResult(i, Request_Barcode);
	}
});
```

Code Segment 4-78 Scan barcode to share product

After system detect product barcode successfully, the application sent request to server and retrieve the product data that match with that barcode string. Subsequently, application code will use the data return from server and put in the product information field as show below. Image below demonstrate that application successfully detect the product barcode and the

product is exist in database, so application retrieve all available data from database and display to user. User can choose to accept the data and continue next step or change to other data or images.


Figure 4-147 Share Product Step 1 Information

The UI above demo that all the data had been fetching successfully. The product name which is the first editext, category which is the fourth editext field is clickable and it will open another activity and allow user to enter their desired product with assist of semantic autocomplete features. Basically this is the same page that mentioned in home tab search bar. That mean both the share item, search box will navigate to same page, class and same activity.

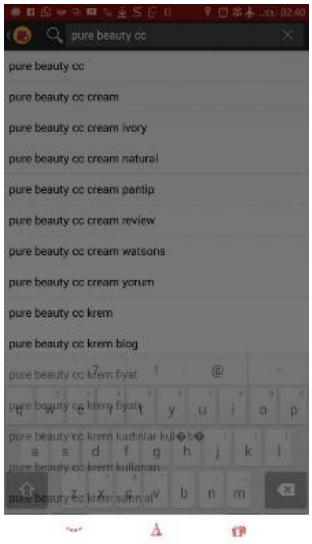



Figure 4-148 Product Name Suggestion in Share Item Module

After user fill in all the field, user can click next button and step 2 activity will display. Step 1 actually is for basic information of product whereas step 2 is the GPS location and market information.


Figure 4-149 Share Product Step 2 Information

#### 4.5.7.4 Location Semantic Suggestion

The first editext is the location of market. Here will have the difference if user click share item button from visit history. The location address will be the history location instead of current location. However if user does not satisfy to the location, he or she can always change manually by clicking the gray color editext and the page below will be displayed. If the location search bar in action bar is empty, nearby market with address will list in listview but if the user start to type any character to the search bar in action bar, location suggestion will display. In short, every single character that user type will trigger onQueryTextChange method and retrieve new set of suggestion address.

```
public boolean onQueryTextChange(String newText) {
connectThread = new MyThread(newText.toString(), mListView);
connectThread.start();
}
```

Code Segment 4-79 Text Change Listener implementation for Place Suggestion

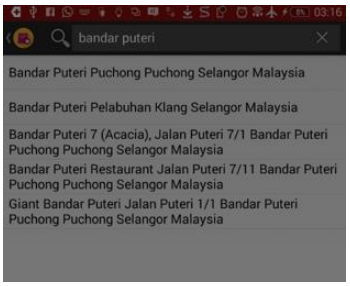



Figure 4-150 Place Address Suggestion in Share Item Module

#### 4.5.7.4.1 Place API

The API that use to retrieve nearby place suggestion is

```
https://maps.googleapis.com/maps/api/place/autocomplete/json?key=
APIKEY&components=country:my &input=query
```

Code Segment 4-80 API use for place suggestion

Sample response:

```
"predictions" : [
      {
         "description" : "Tesco Kampar Jalan Perdana Taman Kampar Perdana Kampar Perak
Malaysia",
         "id" : "bc265216fcf838e3281f8a178041a7ed8edb75cd",
         "matched_substrings" : [
            {
               "length" : 12,
               "offset" : 0
            }
         ],
         "place_id" : "ChIJkZ6K98niyjERE8efnKu2GPw",
         "reference" : "CmRUAAAAUT-SVxiZGpqPsyYInEDn1f3eBrYU0I ",
         "terms" : [
            {
               "offset" : 0,
               "value" : "Tesco Kampar"
            },
            {
               "offset" : 13,
               "value" : "Jalan Perdana"
            },
            {
               "offset" : 27,
               "value" : "Taman Kampar Perdana"
            },
            {
               "offset" : 48,
               "value" : "Kampar"
            },
            {
               "offset" : 55,
               "value" : "Perak"
            },
            {
               "offset" : 61,
               "value" : "Malaysia"
            }
         ],
         "types" : [ "establishment" ]
      }
```

Code Segment 4-81 Sample Response for place suggestion

#### 4.5.7.4.2 Json Parsing

That data return from this API is in Json format, we need to parse it and put in addrSugest arrayList.

```
int read;
char[] buff = new char[1024];
while ((read = in.read(buff)) != -1) {
jsonResults.append(buff, 0, read);
```

```
}
JSONObject jsonObj = new JSONObject(jsonResults.toString());
JSONArray predsJsonArray = jsonObj.getJSONArray("predictions");
for (int i = 0; i < predsJsonArray.length(); i++) {
       addrSugest.add(predsJsonArray.getJSONObject(i).getString(
       "description"));
       placeid.add(predsJsonArray.getJSONObject(i).getString("place_id"));
}
```

Code Segment 4-82 Parsing Json Data and retrieving place suggestion

The second editext is market name and it is clickable. It will open up a page which list down all market that fetch from database in listview.

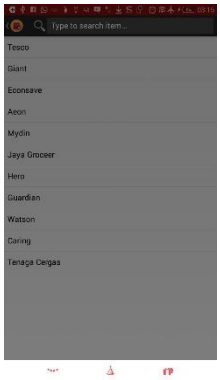



Figure 4-151 Store Suggestion in Share Item Module

After user fill in third editext which is price of product, they can click share item to submit the data. All the data will be validating, process and sent to server and database. When the saving operation done, a confirmation page will be display and user can choose to view shared product or back to home tab.

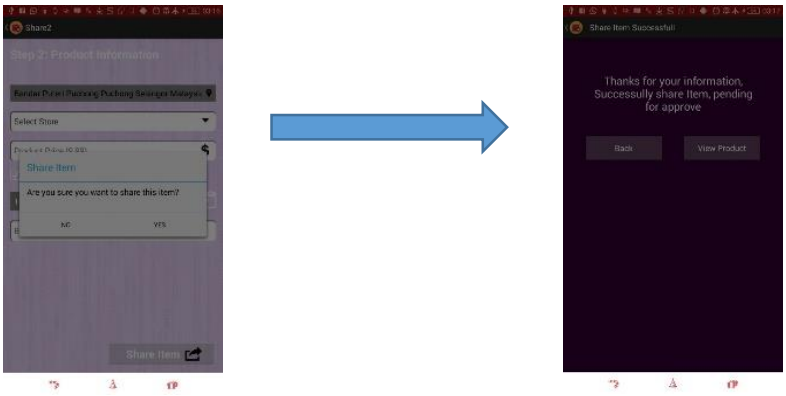



Figure 4-152 Share Product Successfully Demonstration

### 4.5.8 Favorite List

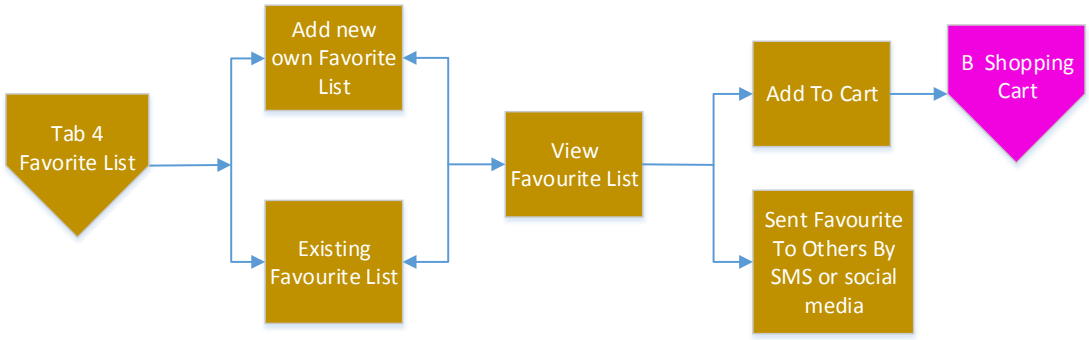



Figure 4-153 Favorite List Screen Flow

his is third tab in home swipe tab and it allow user save their favorite item or the item they wish to purchase.

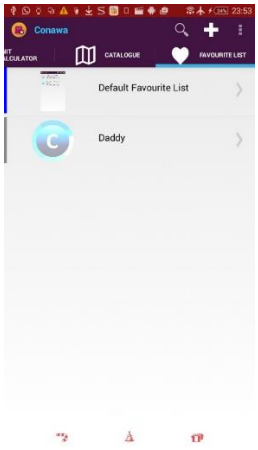



Figure 4-154 Custom Favorite List

The preview images are load from sd storage by directory and image file name:

```
File imageFile = new File(Environment.getExternalStorageDirectory().toString(),
DistinctName.get(i).WLlistNameID + ".png");
if (imageFile.exists()) {
	FileInputStream fis = new FileInputStream(imageFile);
	ByteArrayOutputStream bos = new ByteArrayOutputStream();
	byte[] buf = new byte[1024];
	for (int readNum; (readNum = fis.read(buf)) != -1;) {
	// Writes to this byte array output stream
		bos.write(buf, 0, readNum);
	}
	byte[] bytes = bos.toByteArray();
	fis.close();
	DistinctName.get(i).WLlistNameImg = bytes;

}
```

Code Segment 4-83 Print Screen Programmatically

All these preview image are capture programmatically when new data added in favorite list from product detail, the implementation show below:

Step 1: Write permission in manifest.xml

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Code Segment 4-84 Permission Required to print screen and save image to SD card

Step 2: Capture screen right after data added to sqlite database.

```
private void captureScreen() {

	if (favouriteList.size() != 0) {
		View v = getWindow().getDecorView().getRootView();
		v.setDrawingCacheEnabled(true);
		Bitmap bmp = Bitmap.createBitmap(v.getDrawingCache());
		v.setDrawingCacheEnabled(false);
		FileOutputStream fos = new FileOutputStream(new File(
		Environment.getExternalStorageDirectory().toString(),
					suName + ".png"));
		bmp.compress(CompressFormat.PNG, 100, fos);
		fos.flush();
		fos.close();

	}
}
```

Code Segment 4-85 Capture Screen and save to Sd card

Basically this approach getting a bitmap cache from layout in page or a UI view element so first need to v.setDrawingCacheEnabled(**true**) to a layout or view and then v.getDrawingCache() is the method call to get the screenshot programmatically and then store Bitmap bmp = Bitmap.createBitmap(v.getDrawingCache()) by FileOutputStream into sd card storage. This method allow hiding the view and taking screenshot in the background even the UI not displaying.

#### 4.5.8.1 Add New Custom Favorite List

User can create at most 12 favorite list by clicking the "+" sign on the action bar. Besides that, user can also delete created favorite list but not default favorite list.

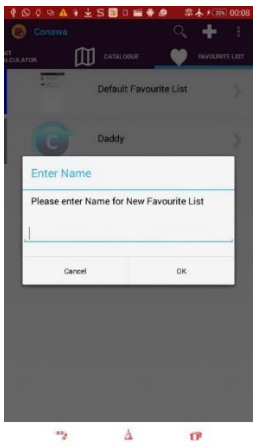



Figure 4-155 Add New Custom Favorite List

As soon as user press "OK" button, a new favorite list record will be insert in sqlite database.

#### 4.5.8.2 Preview Favorite List

Before click inside the favorite list, user can click the image on left of each favorite list and a preview pop up will be show to let user have overview about what is inside in case they had forgotten.

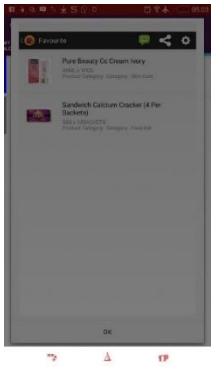



Figure 4-156 Preview Favorite List

This can help them to decide whether continue click inside to continue subsequent operation. After they have an idea what is inside the list, they may like to view detail and continue other operation. Image below show the default favorite list and empty daddy list after they clicked inside.

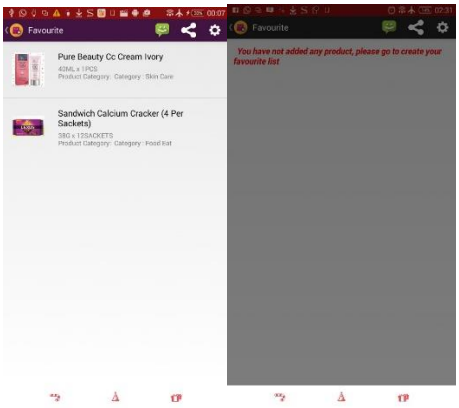



Figure 4-157 Left- Favorite List Contains Item/ Product

Figure 4-158 Right- Favorite List Does not Contains Item/ Product

### 4.5.8.3 Sent SMS

The green color icon on the action bar is self-explanatory, it allow user to send particular favorite list to relatives, friends or family member through contact list. When the green button is clicked, a contacts picker will display as show in following, user can choose the person they want to send their favorite to through SMS.

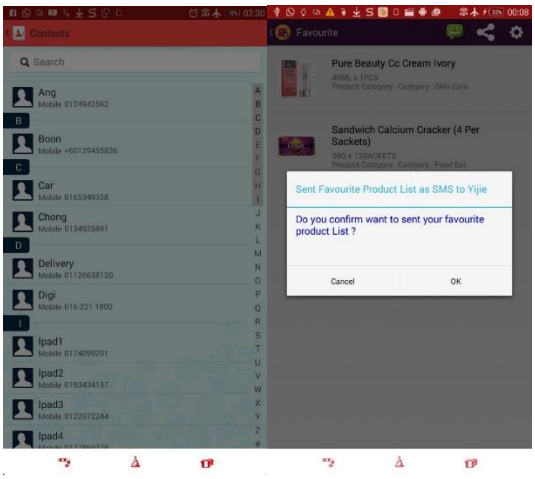



Figure 4-159 Left- Contact List Chooser to Send SMS

Figure 4-160 SMS Sending

As demonstrated above user choose a user called YiJie from contact list and send the favorite list to him. A confirmation pop up will displayto let user confirm their operation and prevent miss press. Below show the way to open the contact picker.

Write these permission in manifest file:

```
  <uses-permission
android:name="com.javapapers.android.swipetablayout.app.permission.C2D_MESSAGE"
/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.WRITE_SMS" />
    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    <uses-permission android:name="android.permission.RECEIVE_MMS" />
    <uses-permission android:name="android.permission.SEND_SMS" />
    <uses-permission android:name="android.permission.READ_CONTACTS" />
```

Code Segment 4-86 Permission for Sending SMS

Open and reading contact information:

```
Intent intent = new Intent(Intent.ACTION_PICK,
       ContactsContract.Contacts.CONTENT_URI);
int PICK_CONTACT = 0;
intent.setType(ContactsContract.CommonDataKinds.Phone.CONTENT_TYPE);
startActivityForResult(intent, PICK_CONTACT);
```

Code Segment 4-87 Contacts Selector

and process the return data in onActivityResult method

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

if(requestCode==0)

{

     String name = null;
     Uri uri = data.getData();
     Cursor cursor = getContentResolver().query(uri, null, null,null,null);
     cursor.moveToFirst();

     int phoneIndex = cursor
       .getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBER);

     int nameIndex = cursor
       .getColumnIndex(ContactsContract.CommonDataKinds.Phone.DISPLAY_NAME)

     phoneNo = cursor.getString(phoneIndex);
     name = cursor.getString(nameIndex);
}
```

Code Segment 4-88 Retrieving Phone Number and Contact Name

The code above showcase the way to retrieve phone number and name form contact picker (content provider). After application get the contact number, it will start the SMS sending operation.

To be able to send out messages, SMS manager API must be apply. Messages sending is simple by filling the message and phone number in method below.

```
SmsManager sms = SmsManager.getDefault();
ArrayList<String> parts = sms.divideMessage(message);
sms.sendMultipartTextMessage(phoneNumber, null, parts, null, null);
```

Code Segment 4-89 Message Sending

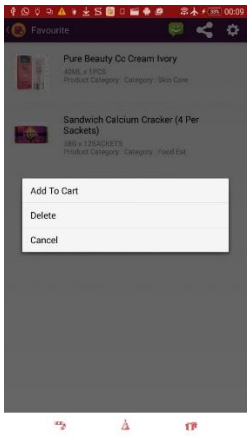



Figure 4-161 Add Favorite List to Cart

Besides that, when user long click the product item, they can delete the item that they no longer needed or no need for time being.

#### 4.5.8.4 Add To Cart

Same as add cart feature in product details but at here is add product from favorite list to cart list. For instance, user may want to buy the product that added inside favorite list, so they can press add to cart choice in the pop out list. The data will be add into cart table of sqlite database by utilize DatabaseHandler class.

### 4.5.9 Cart & Ideal Store Suggestion

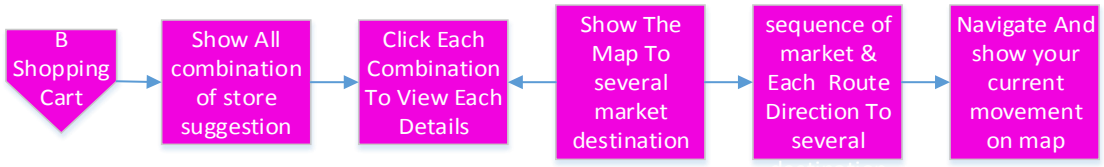



Figure 4-162 Cart and Ideal Store Suggestion Screen Flow

#### 4.5.9.1 Cart List

User can access this module by clicking shopping cart button in main home tab or view cart button in pop up dialog box when execute add to cart operation.

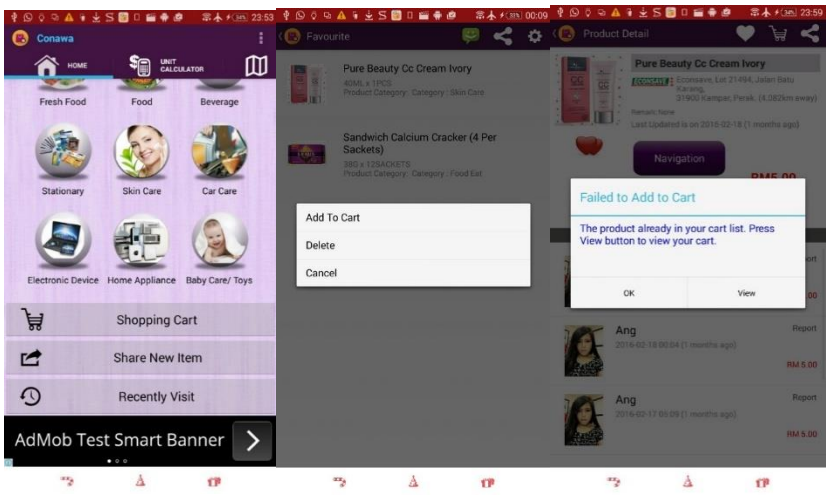



Figure 4-163 Left- Enter Cart Module from Main Home Page Tab

Figure 4-164 Middle- Enter Cart Module from add favorite list to cart Page

Figure 4-165 Add to Cart Page from Product Detail

Once the shopping cart button clicked, the first page that fall into user view is the list of product that added from either favorite list or item detail page. Shopping cart list down all the product in listview UI element through custom adapter. The custom adapter collect all the data and arrange it in listview accordingly. Furthermore, if user wish to delete added product, they can always long click the product in shopping cart and a pop up listview selector will display. When user click delete button, another confirmation pop up will display in user eye to prevent user carry out the operation mistakenly and since delete operation is persistent and unrecoverable thus we provide extra step to let user confirm again their choice properly.

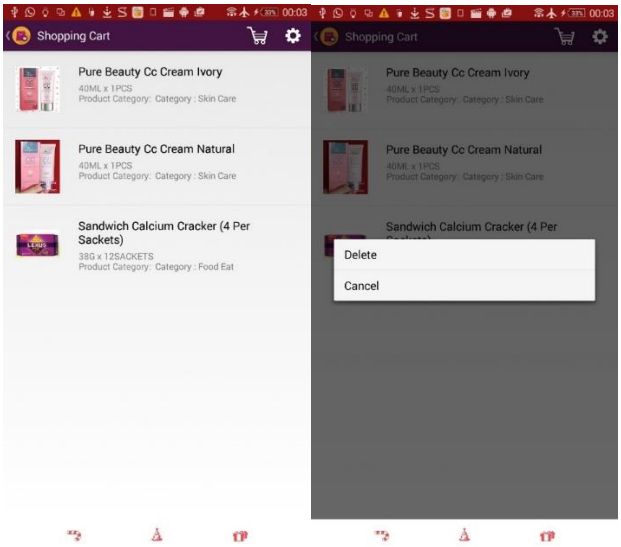



Figure 4-166 Left- Shopping Cart Page

Figure 4-167 Right- Delete Shopping Cart Item

### 4.5.9.2 Ideal Store Suggestion

When user click on the cart button on top right side of action bar, application will list down suggest store based on user settings. This module heavily rely on settings. All the store suggestion output are based on user preferences that preset in settings. The settings and description list down below.

Table 4-8 Ideal Store Suggestion Settings

| Setting | Description |
|---|---|
| Cheapest Single Store Single Route | Cheapest single store after sum up fuel consumption and product total price |
| Multiple Store Single Route | Cheapest store combination and travel all store together ( minimum distance travel to all store ) |
| Multiple Store Multiple Route | Display cheapest store combination and travel all store by star topology |
| All | Display all the combination of market and all kind of travel way |

#### 4.5.9.2.1 Settings = Cheapest Single Store Single Route

| Store | Total Product Price(RM) | Total Distance ( km) | Total Petrol Price(RM) | Total Price Required(RM) |
|---|---|---|---|---|
| Giant, Kampar | RM 104.99 | d4 | p4 | RM 104.99+ p4 |
| Tesco, Kampar | RM 107.99 | d1 | p1 | RM 107.99+ p1 |
| Econsave, Kampar | RM 102.99 | d2 | p2 | RM 102.99+ p2 |
| Tenaga Cergas, Kampar | RM 110.99 | d3 | p3 | RM 110.99+ p3 |

If user choose this setting, the system will display the store that having lowest total price after sum up fuel required and total product price from cart list among all market/store. Image below show the store suggested when user prefer travel way is Cheapest Single Store Single Route.

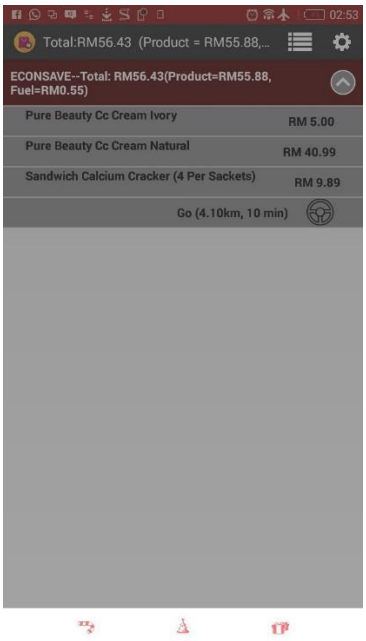



Figure 4-168 Cheapest Single Store Single Route Settings Store Suggestion

As show above, there are three (3) product had been added to cart list. This is only the single cheapest store so there is only one expendable list was displaying. There will be more expendable list if user choose prefer travel way other than single cheapest store. After computation, application suggest that Econsave Kampar is the most ideal store for single

cheapest store only. At the bottom of page, the distance and estimate required travel time between user current location and market are shown. The steering icon on the right of last row allow user navigate themselves to the market by device existing GPS APPS.

Furthermore, user can click on the first icon on action bar if they wish to view more information. A table with total product price, fuel spent and total required spent will sort and rank in ascending order. This ease user if they wish to refer the price offer by other store so they can have more flexible choice. Besides that, user can click on any row to view product price and detail breakdown of fuel and product price in other market.

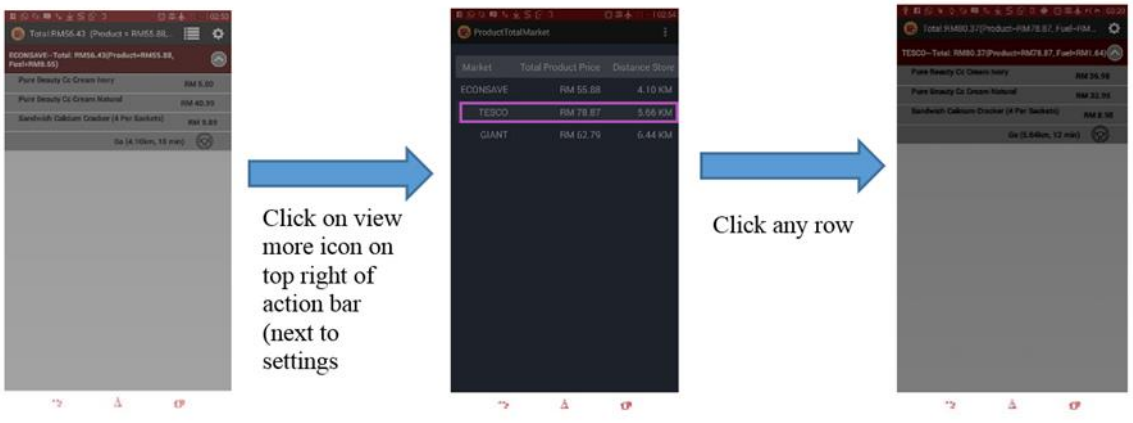



Figure 4-169 View More Information in Cheapest Single Store Single Route Settings

#### 4.5.9.2.1.1 Logic Behind of Cheapest Single Store Single Route Settings

Step 1: Get the maximum distance willing to travel from shared preferences.

```
SharedPreferences sharedPref = this.getSharedPreferences("USER_DETAIL",
                            Context.MODE_PRIVATE);
fuel = Double.parseDouble(String.valueOf((sharedPref
                            .getFloat("fuel", 0))));
fueltype = sharedPref.getString("fueltype", "");
distanceTra = sharedPref.getFloat("distanceTra", 0);
```

Code Segment 4-90 Get Shared Preferences value

Step 2: Calculate out market distance from user current location for every store by SQL statement. Use the maximum distance value to filter the market from database. Those market which locate beyond this distance will not be included.

```
https://maps.googleapis.com/maps/api/distancematrix/json?origins=latitude,
longitude&destinations=productList.latitude,productList.longitude&mode=driving&
language=en-EN&sensor=true&key=AIzaSyD4SpLzS4x6KyKns1-o0Gh7C2oPNOBt28g
```

Code Segment 4-91 Distance Matrix API

Step 3: Return market list that distance fall within range back to application layer from server.

```
if (marketList.dist <= distr) {
      for (int f = 0; f < cartList.size(); f++) {}
}
```

Code Segment 4-92 Filter Market List

Step 4: Loop through cart list, and search the cart list product in every market.

```
for(int i=0 ; i<marketList.size();i++)
      for (int f = 0; f < cartList.size(); f++) {
          //Sent cart Product ID and market ID to server. Get Product
            Information and price for each product in each market
      }
}
```

Code Segment 4-93 Get Cart Product in each market

After server received market ID and product ID, sql below will be execute.

```
SELECT TOP 50
ShareProductID,SP_marketID,SP_productID,SP_LastEditedDT,
market_Latitude,market_Longitude,market_gloMarketID,product_name,
product_desrip,product_barcode,product_QuanDimen1,product_QuanDimen2,
product_category,product_QuanDimen1Unit,product_QuanDimen2Unit,
gloMarketName,gloMarketLogo,market_Add1
FROM [Conawa].[dbo].[ProductList] as PL
inner join [Conawa].[dbo].[Market] as MK
on (PL.SP_marketID=MK.marketID)
inner join [Conawa].[dbo].[GlobalProduct] as GP
on(GP.productID= PL.SP_productID)
inner join Conawa.dbo.GlobalMarket as GM
on (GM.gloMarketID=MK.market_gloMarketID)
where GP.[productID] =cartProductID
and pl.[SP_marketID]= marketID
```

Code Segment 4-94 SQL Query Selecting Product in market

and sql below to get latest price and time updated of product.

```
SELECT TOP 1 UpdatePrice,UpdateDateTime FROM
[Conawa].[dbo].[UpdateProductPrice]
where [UpadateProductID]= shareproductID
order by UpdateDateTime desc
```

Code Segment 4-95 SQL Query Selecting Latest Update Time for each shared product

In each market model class, there is market cart product ArrayList list object to store the information and price of cart product that available in market, every single product return from sql above will add into market product list of each market respectively.

| | | | | |
|---|---|---|---|---|
| Market 1 → | Cart Product 1 | C.P 2 | C.P 3 | C.P 4 |
| Market 2 → | Cart Product 1 | C.P 2 | C.P 3 | C.P 4 |
| Market 3 → | Cart Product 1 | C.P 2 | C.P 3 | C.P 4 |
| Market 4 → | Cart Product 1 | C.P 2 | C.P 3 | C.P 4 |

Figure 4-170 Assignment of Cart Product into Market

Step 5: Remove market that does not have any product listed in cart.

```
ArrayList<Integer> tempArr = new ArrayList<Integer>();
for (int i = 0; i < MarketSingle.size(); i++) {
	Boolean flag = false;
	 for (int i1 = 0; i1 < MarketSingle.get(i1).MarketProduct.size();
	 i1++) {
	if (MarketSingle.get(i).MarketProduct.get(i1).LatestPrice != null) {
			flag = true;
			break;
		}
	}

	if (flag == false) {
		MarketSingle.remove(i);
		i--;
	}
}
```

Code Segment 4-96 Remove market that does not have any product listed

Step 6: Calculate total product price per store

```
// ============================================calculate
// total per store
for (int i = 0; i < MarketSingle.size(); i++) {
    for (int i1 = 0; i1 < MarketSingle.get(i).MarketProduct
            .size(); i1++) {

        MarketSingle.get(i).totalPerStore += ((MarketSingle
                .get(i).MarketProduct.get(i1).LatestPrice) == null) ? MarketSingle
                .get(i).MarketProduct.get(i1).avgprice
                : Double.parseDouble(MarketSingle.get(i).MarketProduct
                        .get(i1).LatestPrice);
    }
}
```

Code Segment 4-97 Calculate total product price per store

Step 7: Sort market based on distance

```
Collections.sort(MarketSingle,
        new Comparator<ProductList>() {
            public int compare(ProductList arg0,
                    ProductList arg1) {
                return (arg1.dist)
                        .compareTo((arg0.dist));
            }
        });
```

Code Segment 4-98 Sort market based on distance

Step 8: Add only top 4 nearest market to final market list

```
for (int h = 0; h < 4; h++) {
    MarketSingleTemp.add(MarketSingle.get(h));
}

MarketSingle.clear();
MarketSingle.addAll(MarketSingleTemp);
```

Code Segment 4-99 Add only top 4 nearest market to final market list

Step 9: Get price of fuel per Litre from database

```
Select price from FuelRate
Where StartDate<=CurrentTimeStamp
And EndDate is null
And FuelType='RON95'
```

Code Segment 4-100 SQL Query Get Latest Fuel Price

Step 10: Calculate Fuel Spent to travel each single store and total required spent

```
for (int i = 0; i < MarketSingleTemp.size(); i++) {
    litreUsage = MarketSingleTemp.get(i).dist * 2/ fuel;
    MarketSingleTemp.get(i).litreUsage = litreUsage ;//x2 round trip
            //rm per L value get from database
    totalfuelrm = litreUsage * rmperL;

    MarketSingleTemp.get(i).totalfuelrm = totalfuelrm;

    MarketSingleTemp.get(i).TotalSpentSingRouteSingStore = totalfuelrm
                + MarketSingleTemp.get(i).totalProductPerStore;
}
```

Code Segment 4-101 Calculate Fuel Spent to travel each single store

Step 11: Sort the market list and display accordingly.

#### 4.5.9.2.2 Settings = Multiple Store Single Route

| Store | Total Product Price(RM) | Total Distance ( km) | Total Petrol Price(RM) | Total Price Required(RM) |
|---|---|---|---|---|
| Giant, Kampar | RM 104.99 | d4 | p4 | RM 104.99+ p4 |
| Tesco, Kampar | RM 107.99 | d1 | p1 | RM 107.99+ p1 |
| Econsave, Kampar | RM 102.99 | d2 | p2 | RM 102.99+ p2 |
| Tenaga Cergas, Kampar | RM 110.99 | d3 | p3 | RM 110.99+ p3 |
| **Giant, Kampar +Tesco, Kampar + Econsave, Kampar + Tenaga Cergas, Kampar** | **RM 97.99** | **d5 ( shortest path )** | **p5** | **RM 97.99+ p5** |

When system detect the user preferred travel route in setting is Multiple Store Single Route, the system will display the cheapest store combination (RM97.99 in the example above) base on product cart list and distance required to travel all store together (minimum travel distance to all store in single route). Basically, if user choose this settings, the total product price will be the lowest among all other choice since it take the market which having cheapest price for each product into total price calculation. The sample output of this setting show below. In this example, the total product price is RM44.89 whereas fuel price is RM1.89 to travel all the 3 market together.

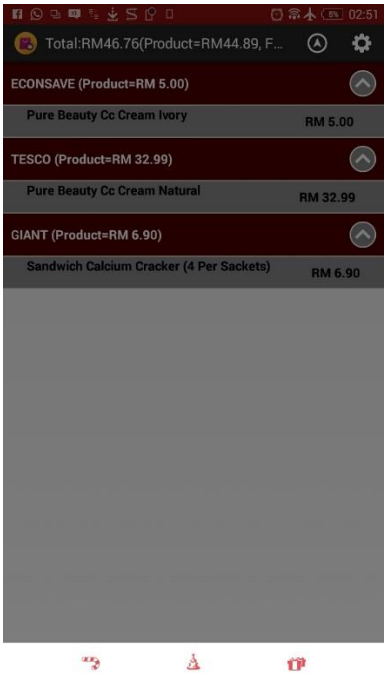



Figure 4-171 Multiple Store Single Route

At here demonstrate the cheapest product in each store together with its price. In order to travel all the three market in single route, we use the minimum spanning tree (salesman travelling problem) to get the sequence of market travel so the distance is optimal and shortest (sequence and route to travel each market so that total distance travel are shortest).

User can click on the navigation button on action bar to view overall travel route and sequence. When user start moving, map will zoom in from 14f to 16.5f and map will keep track the route of user movement, blue polyline is used to track down the path that user went through.

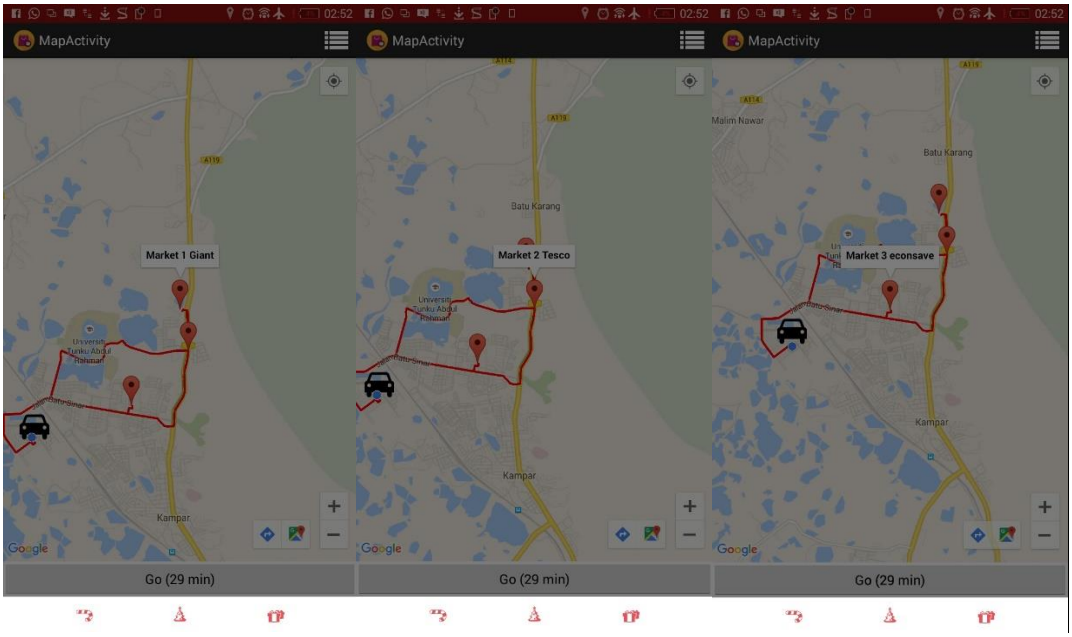



Figure 4-172 Overall travel route and sequence for Multiple Store Single Route

User can click on the view more button on top right of action bar to view overall route and path.

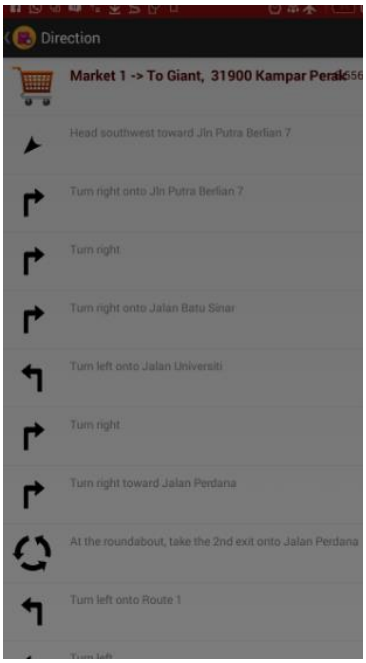



Figure 4-173 Path of Multiple Store Single Route

#### 4.5.9.2.2.1 Logic Behind of Multiple Store Single Route Settings

Step 1: Get the maximum distance willing to travel from shared preferences.

```
SharedPreferences sharedPref = this.getSharedPreferences("USER_DETAIL",
                            Context.MODE_PRIVATE);
fuel = Double.parseDouble(String.valueOf((sharedPref.getFloat("fuel",  0))));
fueltype = sharedPref.getString("fueltype", "");
distanceTra = sharedPref.getFloat("distanceTra", 0);
```

Code Segment 4-102 Get Shared Preferences value

Step 2: Calculate out market distance from user current location for every store by SQL statement. Use the maximum distance value to filter the market from database. Those market which locate beyond this distance will not be included.

```
https://maps.googleapis.com/maps/api/distancematrix/json?origins=latitude,
longitude&destinations=productList.latitude,productList.longitude&mode=driving&
language=en-EN&sensor=true&key=AIzaSyD4SpLzS4x6KyKns1-o0Gh7C2oPNOBt28g
```

Code Segment 4-103 Get estimate distance from distance matrix API

Step 3: Return market list that distance fall within range back to application layer from server.

```
if (marketList.dist <= distr) {
      for (int f = 0; f < cartList.size(); f++) {
      }
}
```

Code Segment 4-104 Filter out market list which beyond range

Step 4: Loop through cart list, and search the cart list product in every market.

```
for (int i=0 ; i<marketList.size();i++)
{
      for (int f = 0; f < cartList.size(); f++) {
         //Sent cart Product ID and market ID to server. Get Product
           Information and price for each product in each market
      }
}
```

Code Segment 4-105 Get Cart Product in each market

After server received market ID and product ID, sql below will be execute.

```
SELECT TOP 50
ShareProductID,SP_marketID,SP_productID,SP_LastEditedDT,
market_Latitude,market_Longitude,market_gloMarketID,product_name,
product_desrip,product_barcode,product_QuanDimen1,product_QuanDimen2,
product_category,product_QuanDimen1Unit,product_QuanDimen2Unit,
gloMarketName,gloMarketLogo,market_Add1
FROM [Conawa].[dbo].[ProductList] as PL
inner join [Conawa].[dbo].[Market] as MK
on (PL.SP_marketID=MK.marketID)
inner join [Conawa].[dbo].[GlobalProduct] as GP
on(GP.productID= PL.SP_productID)
inner join Conawa.dbo.GlobalMarket as GM
on (GM.gloMarketID=MK.market_gloMarketID)
where GP.[productID] =cartProductID
and pl.[SP_marketID]= marketID
```

Code Segment 4-106 SQL Query Selecting Product in market

and sql below to get latest price and time updated of product

```
SELECT TOP 1 UpdatePrice,UpdateDateTime FROM
[Conawa].[dbo].[UpdateProductPrice]
where [UpadateProductID]= shareproductID
order by UpdateDateTime desc
```

Code Segment 4-107 SQL Query Selecting Latest Update Time for each shared product

In each market model class, there is market cart product ArrayList list object to store the information and price of cart product that available in market, every single product return from sql above will add into market product list of each market respectively.

| | | | | |
|---|---|---|---|---|
| Market 1 → | Cart Product 1 | C.P 2 | C.P 3 | C.P 4 |
| Market 2 → | Cart Product 1 | C.P 2 | C.P 3 | C.P 4 |
| Market 3 → | Cart Product 1 | C.P 2 | C.P 3 | C.P 4 |
| Market 4 → | Cart Product 1 | C.P 2 | C.P 3 | C.P 4 |

Figure 4-174 Assignment of Product that match Cart Product to Market

Step 5: Sort market based on distance

```
Collections.sort(MarketSingle,
        new Comparator<ProductList>() {
            public int compare(ProductList arg0,
                    ProductList arg1) {
                return (arg1.dist)
                        .compareTo((arg0.dist));
            }
        });
```

Code Segment 4-108 Sort market based on distance

Step 6: Add only top 4 nearest market to final market list

```
for (int h = 0; h < 4; h++) {
    MarketSingleTemp.add(MarketSingle.get(h));
}

MarketSingle.clear();
MarketSingle.addAll(MarketSingleTemp);
```

Code Segment 4-109 Add only top 4 nearest market to final market list

Step 7: Get Different combination of market List

Combination define as the number of ways to choose r objects from a group of n object without considering order factor.

Given a market of size 4, generate and get all possible combinations of r elements in new market arraylist. For example, if input market is {A, B, C, D} and r is 2, then output should be:

Table 4-9 Market Possible Combination Outcome

| **r** | **Possible outcome** |
|---|---|
| r=2 | {A, B}, {A, C}, {A, D}, {B, C}, {B, D} and {C, D} |
| r=3 | {A,B,C}, {A,B,D}, {B,C,D} |
| r=4 | {A,B,C,D} |

Step 8: Scan through all market for every single product in cart list

to get the market with lowest price for particular product in cart. The code below implement the logic of this step. There are 2 for loop here, outer for loop control the element of cart

product whereas inner for loop control the market list, this mean it scan through all available market for every single cart list product and get the lowest market id and price.

```
if (MarketSingleTemp1.size() > 0) {
    for (int i = 0; i < MarketSingleTemp1
            .get(0).MarketProduct.size(); i++) {
        Double minObj = Double.parseDouble(MarketSingleTemp1.get(0).MarketProduct
                        .get(i).LatestPrice);
        cartList.get(i).lowestMarket = 0; // Element

        for (int i1 = 0; i1 < MarketSingleTemp1.size(); i1++) {
            if (MarketSingleTemp1.get(i1).MarketProduct.get(i).LatestPrice != null) {
                Double item = Double.parseDouble(MarketSingleTemp1.get(i1).MarketProduct
                                .get(i).LatestPrice);

                if (item.compareTo(minObj) < 0) {

                    cartList.get(i).lowestMarket = i1; // Element
                    minObj = item;
                }
            }
        }
        cartList.get(i).lowestpri = minObj;
    }
}
```

Code Segment 4-110 Scan through all market to get the market with lowest price

Step 9: Assign lowest price product to respective market

Create hash table for each single market and assign lowest price product to respective market. Those market which does not have any cart product mean that there do not have any cheapest product in particular market, it having higher price for every cart product compare to other market. The conceptual model show below. The first array list depict there are 10 product in cart list.

| C.P 1 | C.P 2 | C.P 3 | C.P 4 | C.P 5 | C.P 6 | C.P 7 | C.P 8 | C.P 9 | C.P10 |
|---|---|---|---|---|---|---|---|---|---|

Figure 4-175 Cart List Illustration

Second arraylist show that cart product 2, 3 having cheapest price in market 1, product 1, 4, 6, 10 having cheapest price in market 2 and so on. In this example, market 4 do not have any cheapest price product and perhaps it having relatively expensive price for all product compare to market 1, 2, 3.

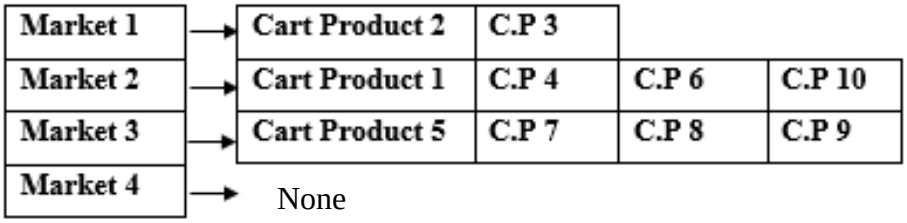



Figure 4-176 Assignment of Cheapest Product in Market

Step 10: Remove the market that having none cheapest product (cheapest product arraylist size = 0)

```
for (int i = 0; i < MarketSingleTemp1.size(); i++) {
    if (MarketSingleTemp1.get(i).CheapestProduct.size() == 0) {
        MarketSingleTemp1.remove(i);
        i--;
    }
}
```

Code Segment 4-111 : Remove the market that having none cheapest product

Step 11: Calculate total cheapest product price in each store

```
for (int k = 0; k < MarketSingleTemp1.size(); k++)
{
       for (int k1 = 0; k1 < MarketSingleTemp
                           .get(k).CheapestProduct.size(); k1++)
       {
       MarketSingleTemp1.get(k).cheapestTotal += MarketSingleTemp1
       .get(k).CheapestProduct.get(k1).lowestpri;

       }
}
```

Code Segment 4-112 Calculate total cheapest product price in each store

Step 12: Calculate optimize route and shortest distance required to travel all store that having cheapest product. The optimization route below is a Travelling Salesman Problem algorithm. You can pass waypoints attribute in URL to optimize route by rearranging the waypoints in more efficient order, otherwise the response will just follow the sequence of waypoints without optimize sequence

The URL following demonstrate a Directions request for a route between Boston, MA and Concord, MA with stopovers in Charlestown and Lexington:

```
https://maps.googleapis.com/maps/api/directions/json?origin=Boston,MA
&destination=Concord,MA&waypoints=Charlestown,MA|Lexington,MA&key=YO
UR_API_KEY
```

Code Segment 4-113 Direction Waypoints API

For each waypoint in the request above (Charlestown, MA| Lexington, MA ), the response includes legs array to indicate detail information to reach every single waypoint :

legs 1 destination= Charlestown, MA or Lexington, MA,

legs 2 destination = Lexington, MA or Charlestown, MA,

legs 3 destination = Concord,MA

Sample Output:


```
"legs": [ {
  "steps": [ {
    "travel_mode": "DRIVING",
    "start_location": {
      "lat": 41.8507300,
      "lng": -87.6512600
    },
    "end_location": {
      "lat": 41.8525800,
      "lng": -87.6514100
    },
    "polyline": {
      "points": "a~l~Fjk~uOwHJy@P"
    },
    "duration": {
      "value": 19,
      "text": "1 min"
    },
    "html_instructions": "Head \u003cb\u003enorth\u003c/b\u003e on \u003cb\u003eS Morgan St\u00
    "distance": {
      "value": 207,
      "text": "0.1 mi"
    }
  },
  ...
  ... additional steps of this leg
...
... additional legs of this route
  "duration": {
    "value": 74384,
    "text": "20 hours 40 mins"
  },
  "distance": {
    "value": 2137146,
    "text": "1,328 mi"
  },
  "start_location": {
    "lat": 35.4675602,
    "lng": -97.5164276
  },
  "end_location": {
    "lat": 34.0522342,
    "lng": -118.2436849
  },
```


Code Segment 4-114 Sample Response from direction waypoints API

Table 4-10 Direction Waypoints Json Output Description

| Json Output field | Description and example |
|---|---|
| Html_instructions | Contains formatted instructions for this step, in HTML encoded string. |
| Distance | distance from this step until next step |
| Duration | time required to travel from this step to next step (each legs) |
| Start Location | coordinate of start point of step |
| End Location | coordinate of end point of step |
| Polyline | Single points object holds encoded polyline representation of the step.The polyline can be decoded and draw directly on google map. |

Step 13: Get shortest distance and time taken to travel all market (single route multiple store)

```
JSONObject distance = ((JSONObject)jLegs.get(j)).getJSONObject("distance");

KMSingRouteMulStore += distance.getDouble("value") / 1000;

TimetotalTravelSingRouteMulStore += ((JSONObject) jLegs.get(j)).
                                    getJSONObject("duration").getInt("value");
```

Code Segment 4-115 Get shortest distance and time taken to travel all market

Step 14: Calculate summation of total cheapest product price of all store

```
tempData1.KMSingRouteMulStore = KMSingRouteMulStore;
tempData1.TimeTravelSingRouteMulStore = TimeTravelSingRouteMulStore;
tempData1.RMfuelSingRouteMulStore = (KMSingRouteMulStore / fuel) * rmperL;

for (int k = 0; k < MarketSingleTemp1.size(); k++) {
       RMProSingRouteMulStore += MarketSingleTemp1.get(k).cheapestTotal;
}
tempData1.RMProSingRouteMulStore = RMProSingRouteMulStore;
RMTotalSpentSingRouteMulStore = RMProSingRouteMulStore+
                                RMfuelSingRouteMulStore;

tempData1.RMTotalSpentSingRouteMulStore = RMTotalSpentSingRouteMulStore;

Combination.add(tempData1)
```

Code Segment 4-116 Calculate summation of total cheapest product price

Step 15: Repeat step 7 – 14 for different combination of market.

Step 16: Sort Market Combination based on total money spent in ascending order.

#### 4.5.9.2.3 Settings = Multiple Store Multiple Route

| Store | Total Product Price(RM) | Total Distance ( km) | Total Petrol Price(RM) | Total Price Required(RM) |
|---|---|---|---|---|
| Giant, Kampar | RM 104.99 | d4 | p4 | RM 104.99+ p4 |
| Tesco, Kampar | RM 107.99 | d1 | p1 | RM 107.99+ p1 |
| Econsave, Kampar | RM 102.99 | d2 | p2 | RM 102.99+ p2 |
| Tenaga Cergas, Kampar | RM 110.99 | d3 | p3 | RM 110.99+ p3 |
| Giant, Kampar +Tesco, Kampar + Econsave, Kampar + Tenaga Cergas, Kampar | RM 97.99 | d5 ( shortest path ) | p5 | RM 97.99+ p5 |
| **Giant, Kampar +Tesco, Kampar + Econsave, Kampar + Tenaga Cergas, Kampar** | **RM 97.99** | **d1 + d2 + d3 + d4** | **p1 + p2 + p3 + p4** | **RM 97.99+ p1 + p2 + p3 + p4** |

The store suggest will be similar with settings in single store single route therefore the total product price will be same. However, by this settings user may wish to visit different store at different occasion, instead of travel all store in single route. Thus, the route will be in star topology. Therefore the distance travel and fuel spent will be the summation of single route to single store.

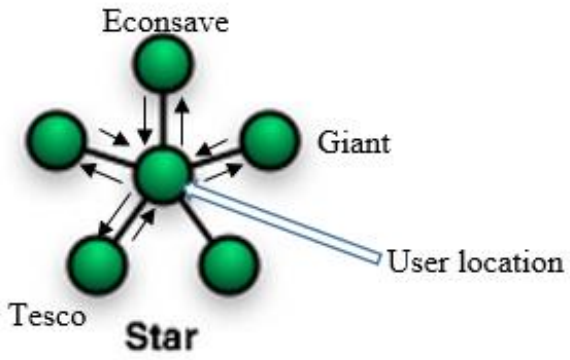



Figure 4-177 Multiple Store Multiple Route Travel in Start Topology

The result display below:

Figure 4-178 Multiple Store Multiple Route Page

This screen depicted the store that offering cheapest price for every given product. In the example above, Pure beauty CC cream Ivory having cheapest price in Econsave, Kampar but not Tesco and Giant whereas Sandwich Cracker having cheapest price in Giant, Kampar but not in other two since the another two may sell the same product with higher price or not selling the given product. At bottom of every sub expendable list, are the distance of particular store with user current location and time required to travel to the particular store. With the estimated time and distance for each store, user can choose to navigate to the market by click on the steering button at the bottom of each market (bottom of sub expendable list) and store. User can choose to change the settings anytime by clicking the setting button on actionbar. For instance, the setting button will be seen in every single pages since this is the central configuration of overall application.

#### 4.5.9.2.3.1 Logic Behind of Multiple Store Multiple Route Settings

Step 1: Get the maximum distance willing to travel from shared preferences.

```
SharedPreferences sharedPref = this.getSharedPreferences("USER_DETAIL",
                            Context.MODE_PRIVATE);
fuel = Double.parseDouble(String.valueOf((sharedPref
                            .getFloat("fuel", 0))));
fueltype = sharedPref.getString("fueltype", "");
distanceTra = sharedPref.getFloat("distanceTra", 0);
```

Code Segment 4-117 Get Shared Preferences value

Step 2: Calculate out market distance from user current location for every store by SQL statement. Use the maximum distance value to filter the market from database. Those market which locate beyond this distance will not be included.

```
https://maps.googleapis.com/maps/api/distancematrix/json?origins=latitude,
longitude&destinations=productList.latitude,productList.longitude&mode=driving&
language=en-EN&sensor=true&key=AIzaSyD4SpLzS4x6KyKns1-o0Gh7C2oPNOBt28g
```

Code Segment 4-118 Get estimate distance from distance matrix API

Step 3: Return and filter market list that distance fall within range by if statement and for statement below back to application layer from server.

```
if (marketList.dist <= distr) {
       for (int f = 0; f < cartList.size(); f++) {

       }
}
```

Code Segment 4-119 Filter Market List

Step 4: Loop through cart list, and search the cart list product in every market.

```
for (int i=0 ; i<marketList.size();i++)
{
       for (int f = 0; f < cartList.size(); f++) {
          //Sent cart Product ID and market ID to server. Get Product
            Information and price for each product in each market
       }
}
```

Code Segment 4-120 Get Product that match with Cart Product in each market

After server received market ID and product ID, sql below will be execute.

```
SELECT TOP 50
ShareProductID,SP_marketID,SP_productID,SP_LastEditedDT,
market_Latitude,market_Longitude,market_gloMarketID,product_name,
product_desrip,product_barcode,product_QuanDimen1,product_QuanDimen2,
product_category,product_QuanDimen1Unit,product_QuanDimen2Unit,
gloMarketName,gloMarketLogo,market_Add1
FROM [Conawa].[dbo].[ProductList] as PL
inner join [Conawa].[dbo].[Market] as MK
on (PL.SP_marketID=MK.marketID)
inner join [Conawa].[dbo].[GlobalProduct] as GP
on(GP.productID= PL.SP_productID)
inner join Conawa.dbo.GlobalMarket as GM
on (GM.gloMarketID=MK.market_gloMarketID)
where GP.[productID] =cartProductID
and pl.[SP_marketID]= marketID
```

Code Segment 4-121 SQL Query Selecting Product in market

and sql below to get latest price and time updated of product

```
SELECT TOP 1 UpdatePrice,UpdateDateTime FROM
[Conawa].[dbo].[UpdateProductPrice]
where [UpadateProductID]= shareproductID
order by UpdateDateTime desc
```

Code Segment 4-122 SQL Query Selecting Latest Update Time for each shared product

In each market model class, there is market cart product ArrayList list object to store the information and price of cart product that available in market, every single product return from sql above will add into market product list of each market respectively.

| | | | | |
|---|---|---|---|---|
| **Market 1** → | **Cart Product 1** | **C.P 2** | **C.P 3** | **C.P 4** |
| **Market 2** → | **Cart Product 1** | **C.P 2** | **C.P 3** | **C.P 4** |
| **Market 3** → | **Cart Product 1** | **C.P 2** | **C.P 3** | **C.P 4** |
| **Market 4** → | **Cart Product 1** | **C.P 2** | **C.P 3** | **C.P 4** |

Figure 4-179 Assignment of Cart Product that available to every Market

Step 5: Sort market based on distance

```
Collections.sort(MarketSingle,
        new Comparator<ProductList>() {
            public int compare(ProductList arg0,
                    ProductList arg1) {
                return (arg1.dist)
                        .compareTo((arg0.dist));
            }
        });
```

Code Segment 4-123 Sort market based on distance

Step 6: Add only top 4 nearest market to final market list

```
for (int h = 0; h < 4; h++) {
    MarketSingleTemp.add(MarketSingle.get(h));
}

MarketSingle.clear();
MarketSingle.addAll(MarketSingleTemp);
```

Code Segment 4-124 Add only top 4 nearest market to final market list

Step 7: Get Different combination of market List

Combination define as the number of ways to choose r objects from a group of n object without considering order factor.

Given a market of size 4, generate and get all possible combinations of r elements in new market arraylist. For example, if input market is {A, B, C, D} and r is 2, then output should be:

Table 4-11 Market Possible Combination Outcome

| r | Possible outcome |
|---|---|
| r=2 | {A, B}, {A, C}, {A, D}, {B, C}, {B, D} and {C, D} |
| r=3 | {A,B,C}, {A,B,D}, {B,C,D} |
| r=4 | {A,B,C,D} |

Step 8: Scan through all market for every single product in cart list

to get the market with lowest price for particular product in cart. The code below implement the logic of this step. There are 2 for loop here, outer for loop control the element of cart product whereas inner for loop control the market list, this mean it scan through all available market for every single cart list product and get the lowest market id and price.

```
if (MarketSingleTemp1.size() > 0) {
    for (int i = 0; i < MarketSingleTemp1
            .get(0).MarketProduct.size(); i++) {
        Double minObj = Double.parseDouble(MarketSingleTemp1.get(0).MarketProduct
                        .get(i).LatestPrice);
        cartList.get(i).lowestMarket = 0; // Element

        for (int i1 = 0; i1 < MarketSingleTemp1.size(); i1++) {
            if (MarketSingleTemp1.get(i1).MarketProduct.get(i).LatestPrice != null) {
                Double item = Double.parseDouble(MarketSingleTemp1.get(i1).MarketProduct
                                .get(i).LatestPrice);

                if (item.compareTo(minObj) < 0) {

                    cartList.get(i).lowestMarket = i1; // Element
                    minObj = item;
                }
            }
        }
        cartList.get(i).lowestpri = minObj;
    }
}
```

Code Segment 4-125 Scan through all market to get the market with lowest price

Step 9: Assign lowest price product to respective market

Create hash table for each single market and assign lowest price product to respective market. Those market which does not have any cart product mean that there do not have any cheapest product in particular market, it having higher price for every cart product compare to other market. The conceptual model show below. The first array list depict there are 10 product in cart list.

| C.P 1 | C.P 2 | C.P 3 | C.P 4 | C.P 5 | C.P 6 | C.P 7 | C.P 8 | C.P 9 | C.P10 |
|---|---|---|---|---|---|---|---|---|---|

Figure 4-180 Cart List Illustration

Second arraylist show that cart product 2, 3 having cheapest price in market 1, product 1, 4, 6, 10 having cheapest price in market 2 and so on. In this example, market 4 do not have any cheapest price product and perhaps it having relatively expensive price for all product compare to market 1, 2, 3.

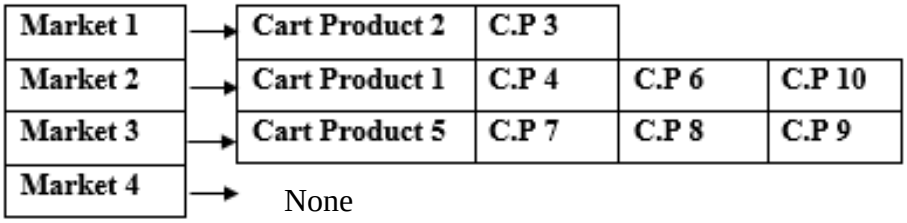



Figure 4-181 Assignment of Cheapest Product in Market

Step 10: Remove the market that having none cheapest product (cheapest product arraylist size = 0)

```
for (int i = 0; i < MarketSingleTemp1.size(); i++) {
    if (MarketSingleTemp1.get(i).CheapestProduct.size() == 0) {
        MarketSingleTemp1.remove(i);
        i--;
    }
}
```

Code Segment 4-126 Remove the market that having none cheapest product

Step 11: Calculate total cheapest product price in each store

```
for (int k = 0; k < MarketSingleTemp1.size(); k++)
{
       for (int k1 = 0; k1 < MarketSingleTemp
                           .get(k).CheapestProduct.size(); k1++)
       {
       MarketSingleTemp1.get(k).cheapestTotal += MarketSingleTemp1
       .get(k).CheapestProduct.get(k1).lowestpri;

       }
}
```

Code Segment 4-127 Calculate total cheapest product price in each store

Step 12: Calculate Fuel Spent to travel each single store and total required spent in each single store. In this case, it is in multiple store multiple route setting thus fuel spent will be separately calculated for every store like cheapest single store did. However, the product price must get cheapest one among all store thus it is similar with Multiple Store Single Route setting. After total of cheapest product calculated for each store, it will sum up together with totalfuelrm to get the overall price that may need to pay for each store, store in variable rmMulRouteMulStore. RMfuelMulRouteMulStore basically use to store the overall money that will spent travel to every store.

```
for (int i = 0; i < MarketSingleTemp.size(); i++) {
    litreUsage = MarketSingleTemp.get(i).dist * 2/ fuel;
    MarketSingleTemp.get(i).litreUsage = litreUsage ;//x2 round trip
            //rm per L value get from database
    totalfuelrm = litreUsage * rmperL;

    MarketSingleTemp.get(i).totalfuelrm = totalfuelrm;
    RMfuelMulRouteMulStore += totalfuelrm;
    MarketSingleTemp.get(i).rmMulRouteMulStore = totalfuelrm
                + MarketSingleTemp.get(i).cheapestTotal;
}
```

Code Segment 4-128 Calculate Fuel Spent and total required spent travel to each store

Step 13: Calculate overall fuel spent, total product price and money required spent. First line sum up the total cheapest product price in each store and assign into another variable call RMProMulRouteMulStore whereas second line in for loop sum up all the money that will spent in each store (product+fuel) and assign into a variable call RMTotalAllMulRouteMulStore.

```
for (int k = 0; k < MarketSingleTemp1.size(); k++) {
    RMProMulRouteMulStore += MarketSingleTemp1.get(k).cheapestTotal;
    RMTotalAllMulRouteMulStore += MarketSingleTemp1.get(k).rmMulRouteMulStore;

}
```

Code Segment 4-129 fuel spent, total product price and money required spent

#### 4.5.9.2.4 Settings = All

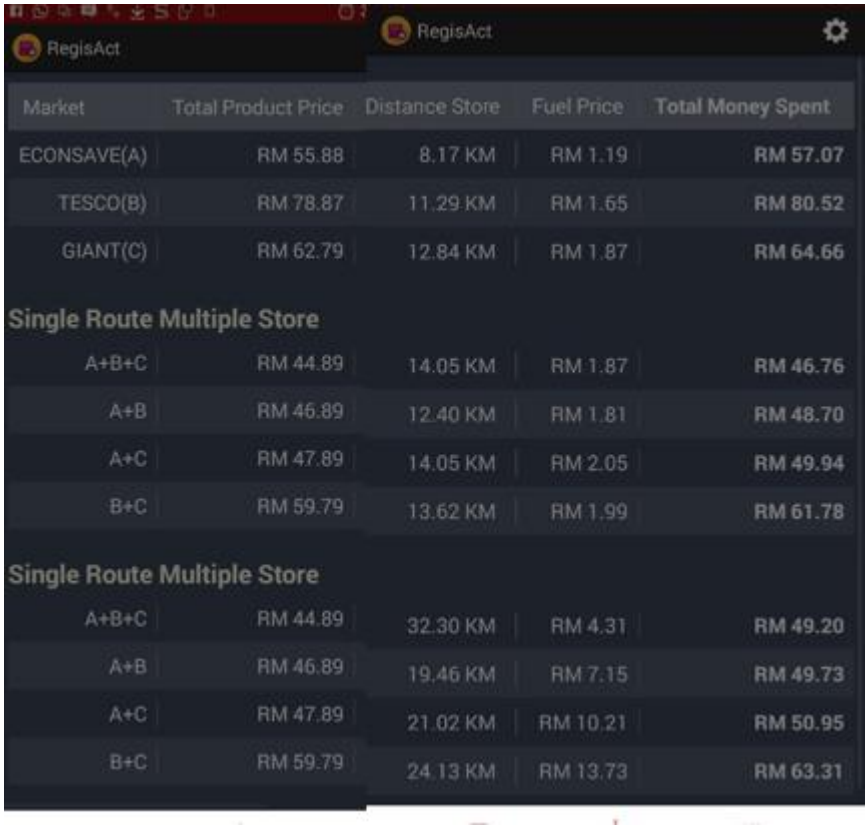



Figure 4-182 Store Suggest Display when Setting = ALL

With this setting, user will be able to view all the combination of store (maximum four store) and its ranking (total money spent sort in ascending order) so this can assist user to make a better decision by looking at the total product spent and fuel spent since everyone having different preferences in market comparison. For instance, some people may prefer nearest store even the total product price is higher than other choice whereas some people will like to compare price of single product in different store whereby lesser fuel and shorter distance may not be their priority to take into consideration. The result sort single market by distance in ascending order and sort both Single Route Multiple Store and Multiple Route Multiple Store by Total Money Spent in ascending order.

Users are allowed to click on each row and the detail information for each store and product price will be show.

For instance, if user click on Econsave (A), screen below will display. The UI is quite similar with cheapest single store settings but this screen does not provide "view more" functionality.

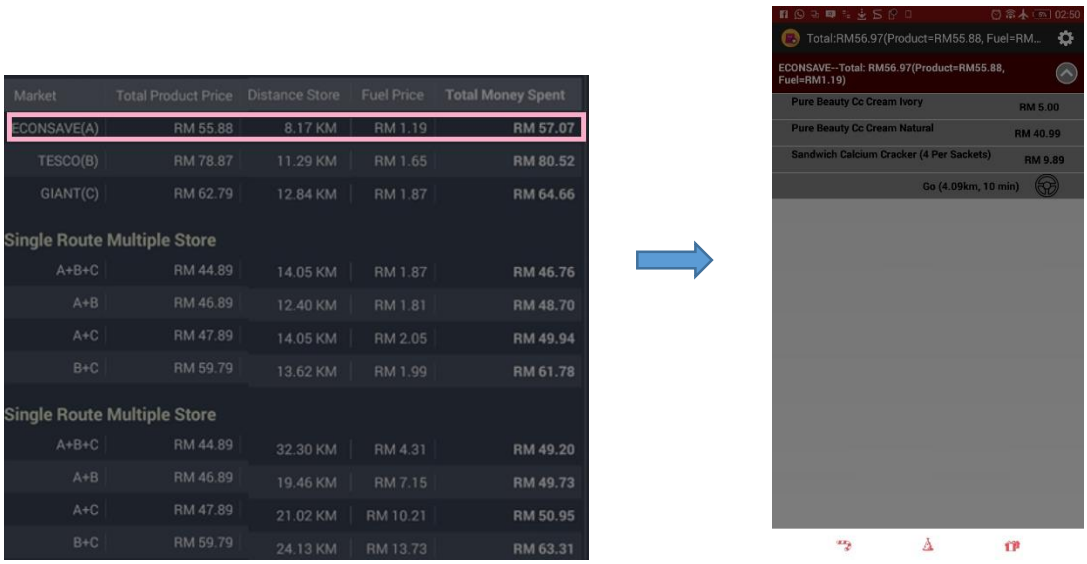



Figure 4-183 View Detail for each Ideal Store Suggestion 1 (Setting=ALL)

If user click on combination of A+B+C under Single Route Multiple Store, the screen below will display. This is the same screen and page that use in Multiple Store Single Route Setting.

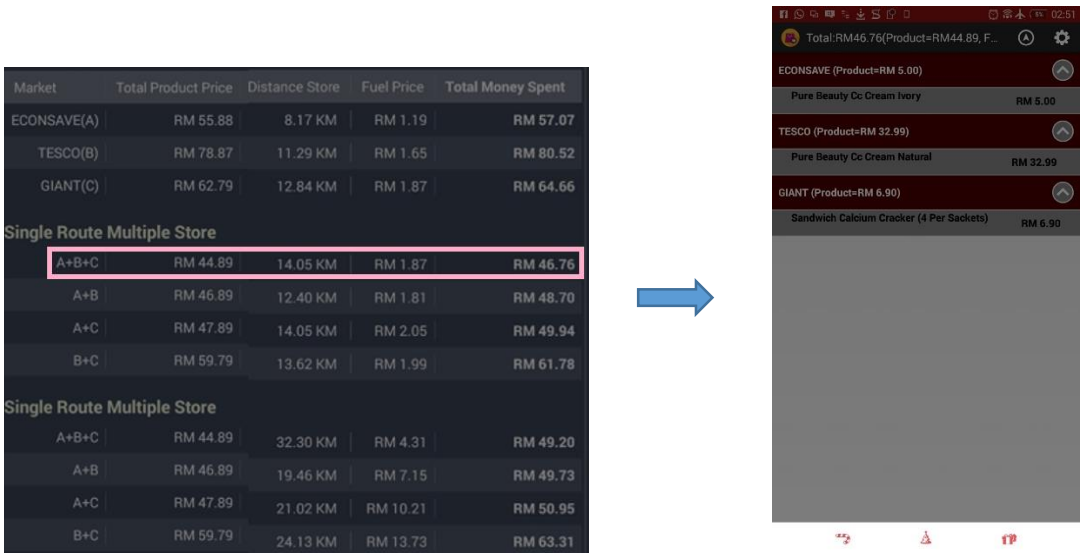



Figure 4-184 View Detail for each Ideal Store Suggestion 2 (Setting=ALL)

When user choice is A+B or B+C combination under single route multiple store, system will display only the 2 store, and the product that should buy in each store.

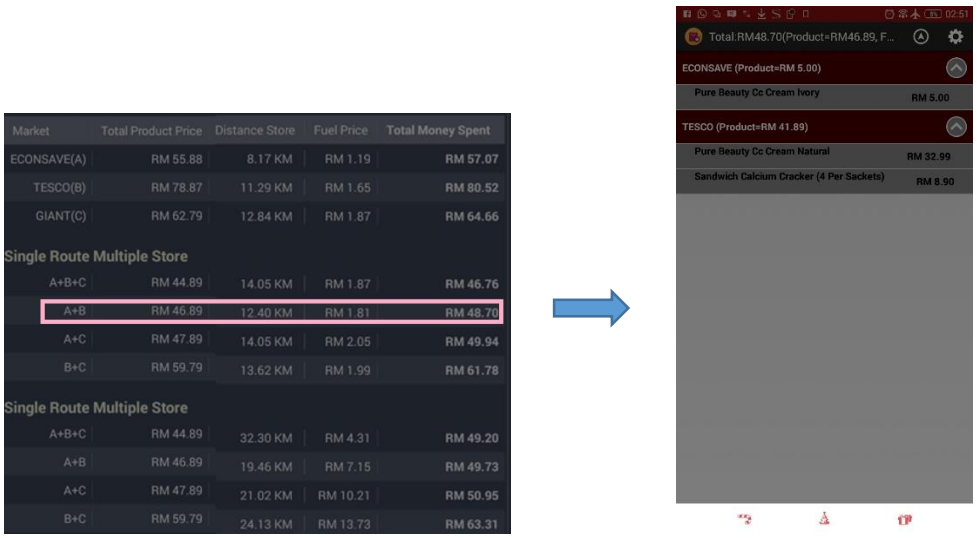



Figure 4-185 View Detail for each Ideal Store Suggestion 3 (Setting=ALL)

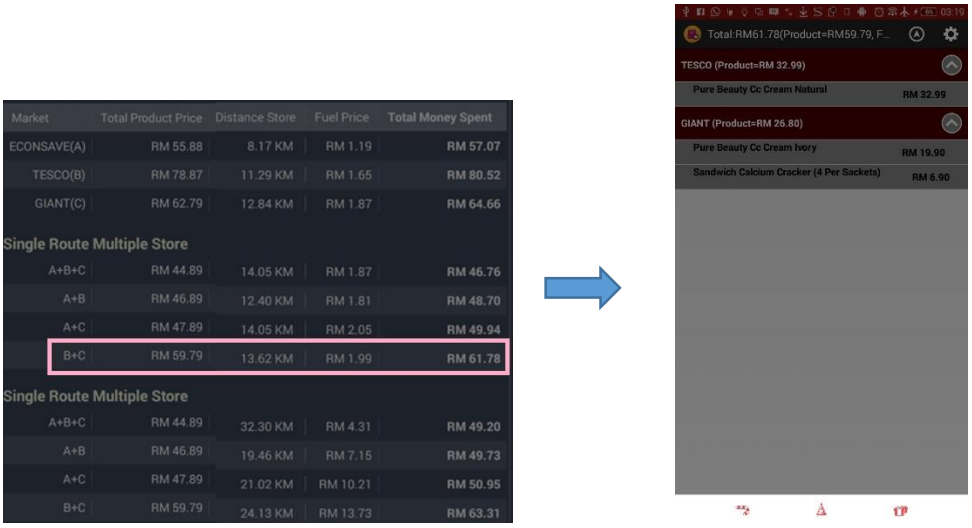



Figure 4-186 View Detail for each Ideal Store Suggestion 4 (Setting=ALL)

Besides that, when user click on combination of A+B+C under Multiple Route Multiple Store, product price detail in each store and estimate travel distance with time taken of each store.

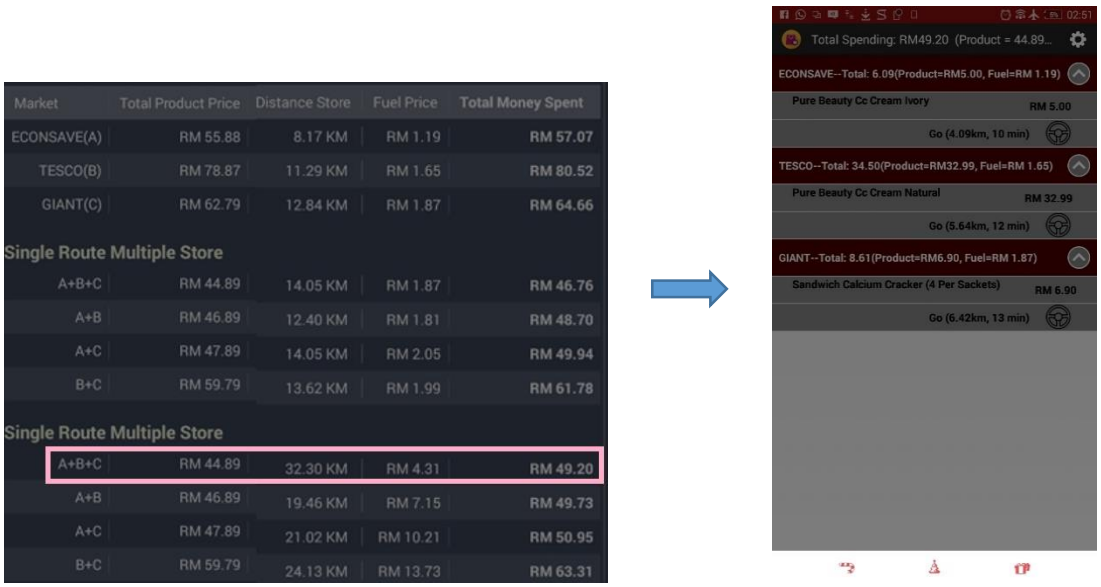



Figure 4-187 View Detail for each Ideal Store Suggestion 5 (Setting=ALL)

Lastly, below demonstrate the store suggestion for choice of A+B market combination under Single Route Multiple Store.

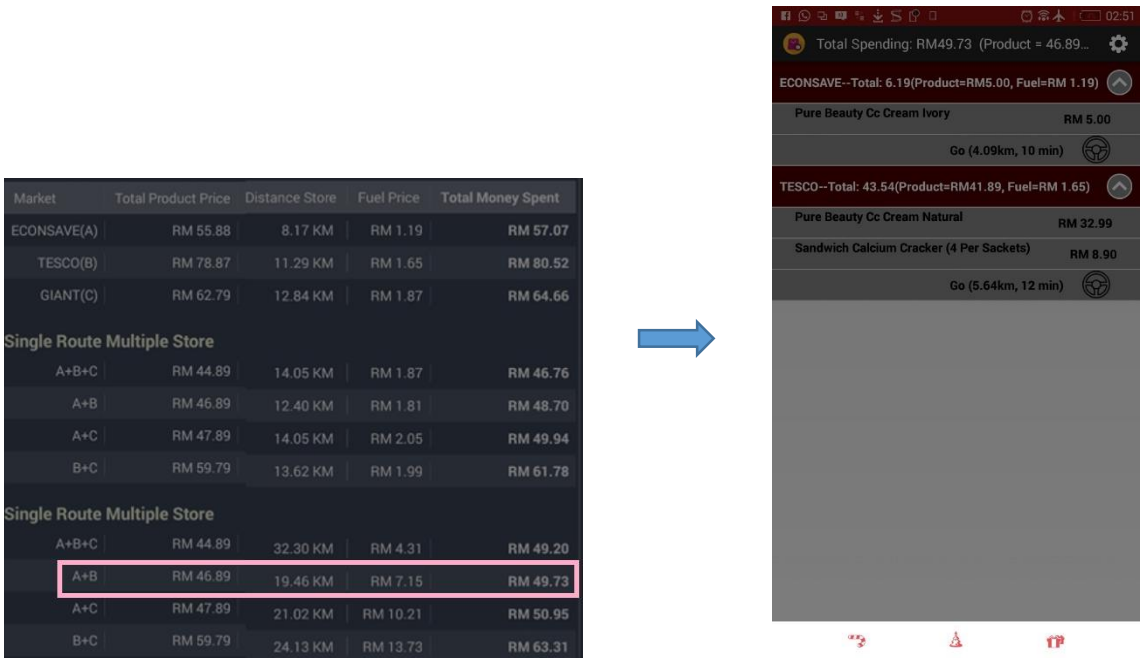



Figure 4-188 View Detail for each Ideal Store Suggestion 6 (Setting=ALL)

### 4.5.10 Unit Calculator Module

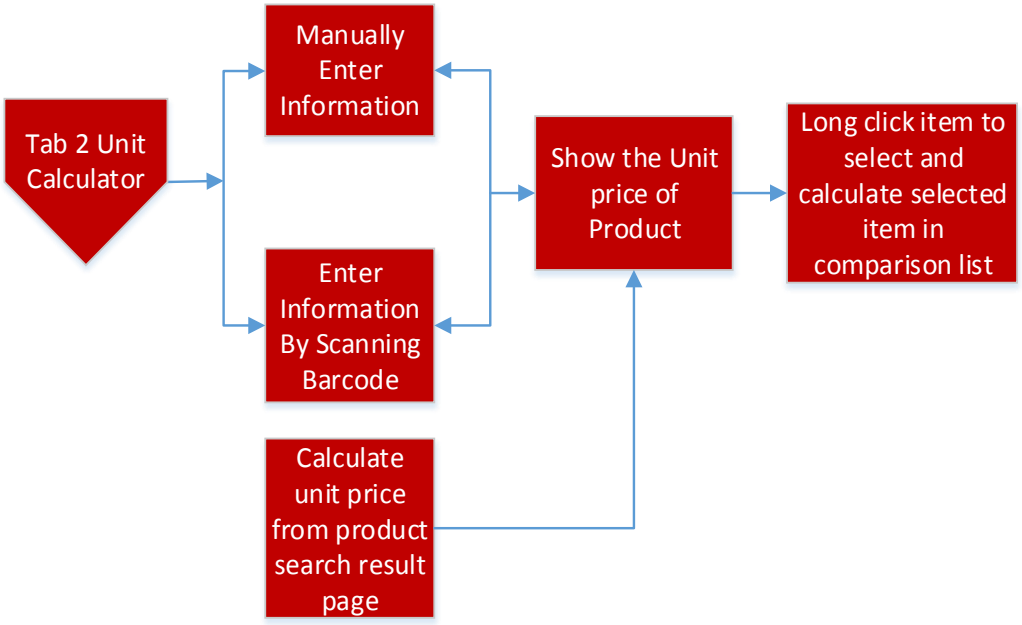



Figure 4-189 Unit Calculator Module Screen Flow


Figure 4-190 Unit Calculator Tab Screen

Unit calculator is the second tab in application and it list down all the calculated history through manual way or scanning way. It allow user to delete the item by long click the item they wish to delete. The system will display the calculated unit price and append each record to the history section. Based on the unit price and price comparison calculated by the system, user can make better decision on which item to purchase.

Besides that, the result can be filter by time for example, within 24 hours, this week, this month, and this year. As user click on filter button (second from right), the filtering will start to execute.

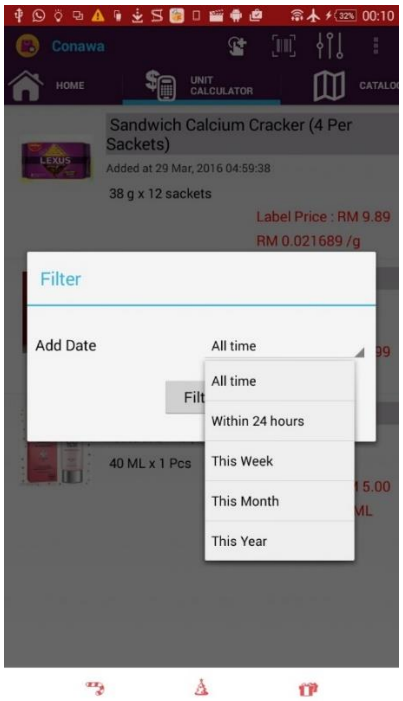



Figure 4-191 Filter Unit Calculator History by Time

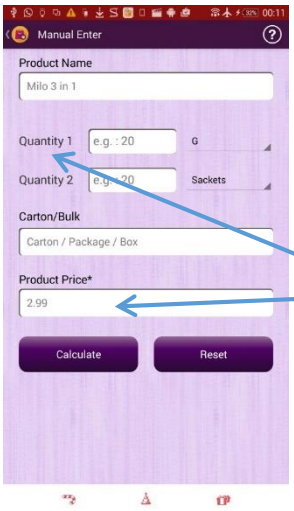
### 4.5.10.1 Calculate from product search result page

There are three (3) way to calculate unit price namely, manual, scanning and add from product search result. First way allow user to calculate unit price of product directly from product search result page by long click any product they wish to know unit price.


Figure 4-192 Unit Calculator from Search Product Result

### 4.5.10.2 Manual Way

User can manually enter the product quantity 1 (G/ML/L/KG), quantity 2 (Sackets/ Cartons/ Pcs) and product price to calculate unit price.







| Chesdale Cheese Plain | | | |
|---|---|---|---|
| *Quantity | *Price | Unit Price | Price Comparison (12pcs) |
| 6pcs = 125g | Rm6.15 | 6.15/6= RM1.025/pcs | RM12.30 |
| 12pcs = 250g | RM10.60 | 10.6/12= RM0.883/pcs | RM10.60 |
| 24pcs=500g | RM20.25 | 20.25/24=RM0.844/pcs | RM10.13 |

Note: * represents input from user

Figure 4-193 Unit Calculator Manual Way Demonstration

This screen will be displaying when the question mark on action bar is clicked.

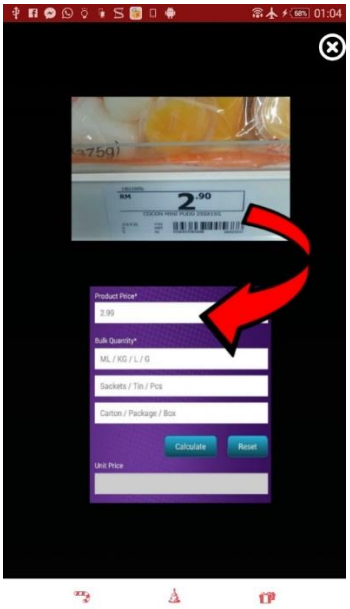



Figure 4-194 Unit Calculator Manual Way Help Screen

### 4.5.10.3 Scanning Way

User can use device camera to scan the product barcode. If the product exist in database, a pop up dialog box with product information will be display and users are allowed to enter product price, then user can view the unit price immediately.

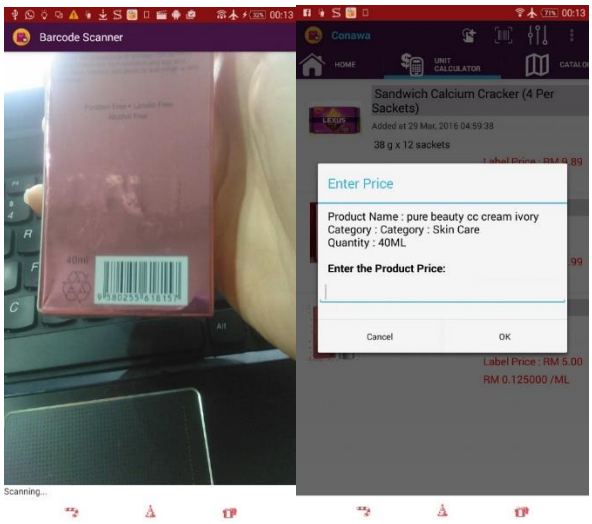



Figure 4-195 Left- Unit Calculator Barcode Scanning Detecting Process

Figure 4-196 Right- Unit Calculator Detecting successfully and enter price

### 4.5.10.4 Comparison List


Figure 4-197 Comparison List from Unit Calculator

Referring back to the previous example, the system will display the calculated unit price and append each record to the history section.

After user calculates the unit price of the 2 different yet similar product, they can long click on the item of each history record, and proceed to press compare price on contextual action bar (another actionbar floating on top of original action bar) as show in image below.

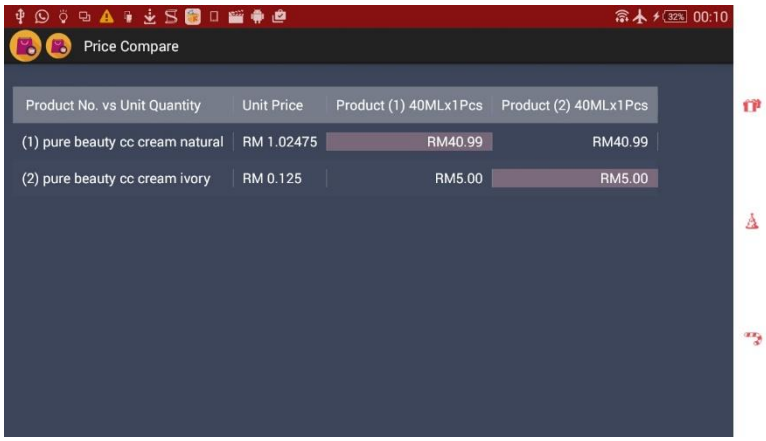



Figure 4-198 Comparison list Demonstration

By this comparison, user can know what will be the price for each item's unit price. The result will be useful for user analysis to make better decision on which product to buy.

Table 4-12 Illustration of decision making for same product but different quantity

| | | 50 | 75 | 100 | 250 |
|---|---|---|---|---|---|
| P1 | 0.052 | 2.6 | 3.9 | 5.2 | 13 |
| P2 | 0.0507 | 2.535 | 3.8025 | 5.07 | 12.675 |
| P3 | 0.045 | 2.25 | 3.375 | 4.5 | 11.25 |
| P4 | 0.034 | 1.7 | 2.55 | 3.4 | 8.5 |

## 4.6 Entity Relationship Diagram Model

### 4.6.1 Entity Relationship Diagram Model in Sql Server

We convert the model class to ERD class. Firstly, global market having zero or a lot of market branch throughout whole country and it will distribute more than 1 brochure at any given time but some may not distribute any brochure. Then a brochure normally is distribute periodically (e.g. per week) and it contains many pages together with page number.

In another hand, any global product is uniquely identify by product barcode, can have more than one shared product, which mean user may share particular product that exist in specific market. With this, user can retrieve product information or detail by scanning product. A shared product must be shared under one and only one market but a market can have many shared product.

Besides that, every under shared product, user are allowed to share product price with some description so product price is child table of share product. It mean that every shared product must have at least one or more shared price and share by only one user. Through product detail page, user can subscribe user thus this will be add into subscription table.

If user think that the price shared by particular user is fake, they can press report button and report record will be inserted into report price table. One shared price can be reported by none people or many people but report price is child table or share product price thus any report price must contains foreign key from product price primary key. State and petrol price table is lookup table thus both them are standalone table.

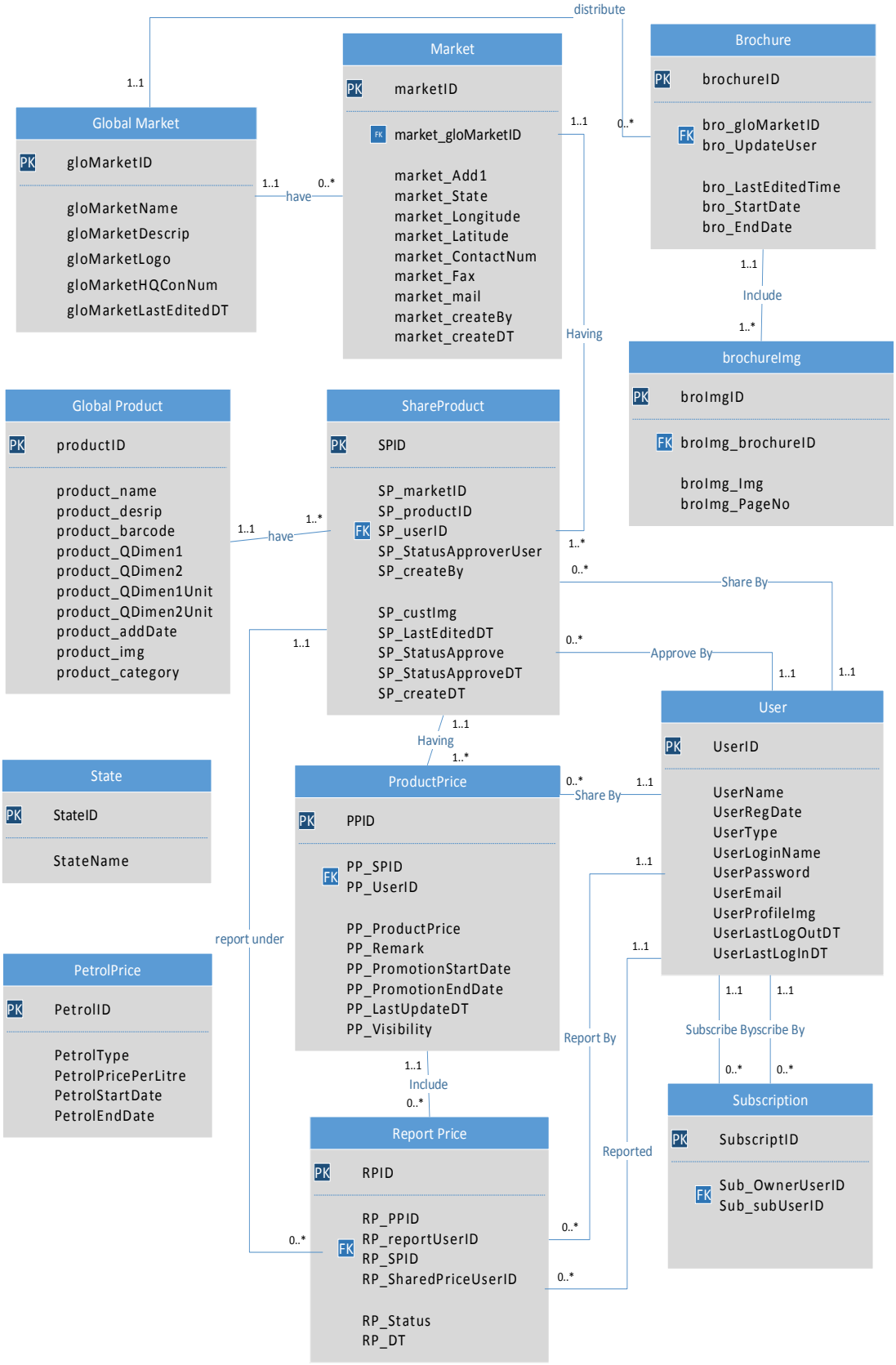



Figure 4-199 Entity Relationship Diagram Model in Sql Server

### 4.6.2 Entity Relationship Diagram Model in SqLite

When the information fetch from remote database server, user can choose to add product or information into their favorite list or cart list. Below show the six (6) table that create in local database Sqlite. Firstly, users are allowed to create any custom favourite list and sent the favorite to friends or families.

Wishlist table at here store favorite list, it is main table and wishListDetail is child detail that store detail and item contain in each wishList. We have cart table here to store cart list so that user can check out the cart product they want to buy and let system suggest them which store to go.

Furthermore, fussy table store the unit calculator history, it contains all calculated history whether in manual, scanning, or from product search result. Check In table is required in order to keep track user visited location so that the application can help them to recall back where they went and they can directly share product from particular check in history record. This reduce the troublesome for user to re-entering all location and store information and this can definitely help out those people who having alzheimers disease.

Lastly, search history table store all user previous search history, all the query that type by user will insert into this table.

In short, all table in Sqlite is not link with sql server database, but it still require to store the primary key of sql server for every record inserted in sqlite for refer purpose or data retrieving purpose.

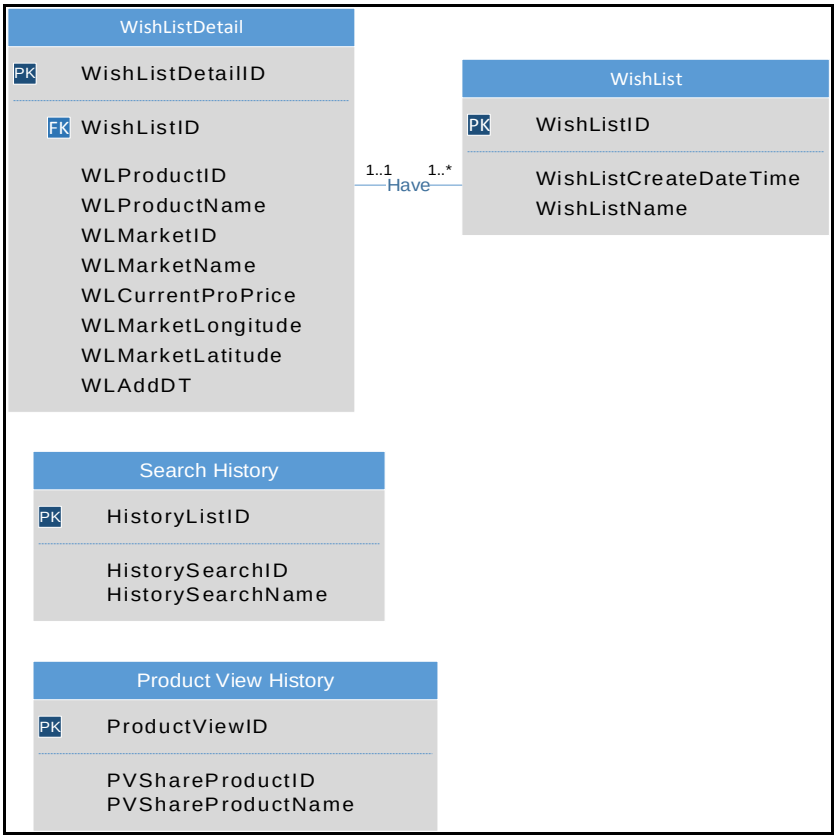



Figure 4-200 Entity Relationship Diagram Model in SqLite

### 4.6.3 Shared Preferences Storage

All the data store in shared preference are key value pairs. It store all user profile and settings together with authentication session information.

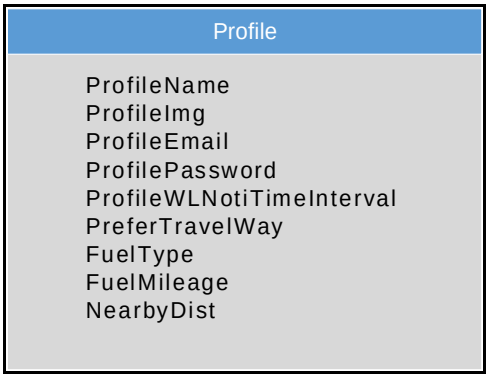



Figure 4-201 Shared Preferences

### 4.7 Data Dictionary

There are 3 different storage use in this application, so data type is slightly different for each type of storage.

SQL server data type that use in this application listed in table below:

| | |
|---|---|
| int | numeric |
| bigint | float |
| money | datetime |
| date | time |
| varchar | char |

Sqlite data type that use in this application listed in table below:

| | |
|---|---|
| Text | datetime |
| Integer | date |
| Blob | boolean |

Shared Preferences data type that use in this application listed in table below:

| | |
|---|---|
| String | Float |
| int | Boolean |

Table Name: User

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| UserID | User's unique identification number that use in this system | int | 8 | 0000 | PK | - |
| UserName | Staff's last name and Staff's first name together that use in system and display to let other user see | varchar | 2 | 00 | - | - |
| UserRegDate | User's start registration date to this system | date | 2 | 00 | - | - |
| UserType | User type which is consumer/ seller/ admin | varchar | 20 | - | - | - |
| UserLoginName | User's login name use when login | varchar | 20 | - | - | - |
| UserPassword | User's personal password in this system | varchar | 20 | XXXXXXXX | - | - |
| UserEmail | User's personal electronic mail | varchar | 50 | xxx@xxx.com | - | - |
| UserProfileImgAdd | User Profile image cloud storage link that upload from their application | varchar | 300 | - | - | - |
| UserLastLogOutDT | User last log out time | datetime | - | "M" or "F" | - | - |
| UserLastLogInDT | User last log in time | datetime | - | "P" or "T" | - | - |

Table Name: Brochure

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| brochureID | Brochure's unique identification number | int | 8 | 00 | PK | - |
| bro_gloMarketID | The market that the brochure belongs to | int | 8 | 0000 | FK | GobalMarket |
| broUpdateUser | The user that upload the brochure(admin / store ) | int | 8 | - | - | - |
| broc_LastEditedTime | The date and time that the brochure latest update | datetime | - | 'dd-mm-yyyy' | - | - |
| bro_StartDate | The start date for the brochure start ( period that the brochure effective ) | date | - | - | - | - |
| bro_EndDate | The end date for the brochure end ( period that the brochure effective ) | date | - | - | - | - |

Remarks: Brochure is abbreviated into "bro" to reduce the wordings of the attribute name.

Table Name: brochureImg

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| broImgID | Brochure image's unique identification number | int | 8 | 0 | PK | - |
| broImg_brochureID | Indicate the particular brochure image is from which brochure | int | 8 | 0 | FK | - |
| broImg_Img | The file server brochure image link | varchar | 300 | 'ab' | - | - |
| broImg_PageNo | The sequence of image in particular brochure | numeric | 2 | 0 | - | - |

Remarks: Brochure image is abbreviated into "broImg" to reduce the wordings of the attribute name.

Table Name: GlobalMarket

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| gloMarketID | Global Market lookup list unique identification number | int | 8 | 00 | PK | - |
| gloMarketName | Global Market lookup list name | varchar | 300 | 'ab' | - | - |
| gloMarketDescrip | Global Market lookup list description about the market | varchar | 300 | 'ab' | - | - |
| gloMarketLogo | Global Market lookup list logo image file server link | varchar | 300 | 'ab' | - | - |
| gloMarketHQConNum | Global Market lookup list head branch contact number | varchar | 50 | 05-1234567 | - | - |
| gloMarketLastEditedDT | Global market lookup list last add or edited date time | datetime | - | 2016-02-17 04:29:48.533 | - | - |

Remarks: Brochure is abbreviated into "bro" to reduce the wordings of the attribute name.

Table Name: Market

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| marketID | Local market unique identification number | int | 8 | 00 | PK | - |
| market_gloMarketID | Local market belong to which market lookup market | int | 8 | 00 | FK | GlobalMarket |
| market_Add1 | Local market address | varchar | 300 | 'abc' | - | - |
| market_State | Local market state | varchar | 300 | 'abc' | - | - |
| market_Longitude | Local market longitude coordinate | float | 3 | 0.00 | - | - |
| market_Latitude | Local market latitude coordinate | float | 3 | 0.00 | - | - |
| market_ContactNum | Local market contact number | varchar | 20 | 05-1234567 | - | - |
| market_Fax | Local market fax number | varchar | 20 | 05-1234567 | - | - |
| market_mail | Local market email | varchar | 100 | 'abc' | - | - |
| market_createBy | The market information create by which user | int | 8 | 00 | User | - |
| market_createDT | When the market information create | datetime | - | 2016-02-17 04:29:48.533 | - | - |

Table Name: State lookup list

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| StateID | State ID unique identification number | int | 2 | 00 | PK | - |
| StateName | State name in lookup list | varchar | 50 | 'ab' | - | - |

Table Name: GlobalProduct

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| productID | Product lookup list unique identification number that exist in system so far | int | 2 | 00 | PK | - |
| product_name | Product name In lookup list that exist in system so far | varchar | 200 | 'ab' | - | - |
| product_desrip | Product description In lookup list that exist in system so far | varchar | 500 | 'ab' | - | - |

| | | | | | | |
|---|---|---|---|---|---|---|
| product_barcode | Product barcode number In lookup list that exist in system so far | varchar | 15 | ‘ab’ | - | - |
| product_QuanDimen1 | Product quantity dimension 1 In lookup list that exist in system so far | numeric | 8 | - | - | - |
| product_QuanDimen1Unit | Product quantity unit( ml/g/ L/kg ) | varchar | 5 | ‘ml’ | - | - |
| product_QuanDimen2Unit | Product quantity unit( sackets/ tin/ pcs ) | varchar | 5 | ‘tin’ | - | - |
| product_QuanDimen2 | Product quantity dimension 2 In lookup list that exist in system so far | numeric | 8 | - | - | - |
| product_addDate | Date that the Product add in the system | datetime | - | 2016-02-17 04:29:48.533 | - | - |
| product_img | Product default image file server link ( as long as all same product in all store, will use same picture, but user can put customize image if they want ) | varchar | ‘ab’ | ‘ab’ | - | - |
| product_category | Product category | varchar | ‘ab’ | ‘ab’ | - | - |

Table Name: ShareProduct

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| SPID | The product unique identification number in particular store | int | 8 | 00 | PK | - |
| SP_marketID | The Product belongs to which market | int | 8 | 00 | FK | Market |
| SP_productID | The product in that market is which lookup list's item | int | 8 | 00 | FK | ShareProduct |
| SP_userID | Who are the one who share this product at first | int | 8 | 00 | FK | User |
| SP_StatusApproverUser | Before display , after the user add new product, will required system admin to approved. Here indicate the system admin ID | int | 8 | 00 | FK | User |
| SP_custImg | User can put customize image of particular product if they want. This is the link of file server | varchar | 300 | 'ab' | - | - |
| SP_LastEditedDT | Last add or edited date and time | datetime | - | 2016-02-17 04:29:48.533 | - | - |
| SP_StatusApprove | Whether the product approve by system or not (will be update by admin after admin review ). All the product that | varchar | 2 | 'Y' | - | - |

| | | | | | | |
|---|---|---|---|---|---|---|
| | share initially is unapproved status | | | | | |
| SP_StatusApproveDT | Product share approved date and time by admin | datetime | - | 2016-02-17 04:29:48.533 | - | - |

Table Name: Share ProductPrice

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| PPID | The price share by user unique identification number | int | 8 | 00 | PK | - |
| PP_SPID | The price share belongs to which product | int | 8 | 00 | FK | - |
| PP_UserID | The user who share the price | int | 8 | 00 | FK | - |
| PP_ProductPrice | The product price | money | 10,2 | 0.00 | - | - |
| PP_Remark | Extra remark for product share price | varchar | 300 | 'ab' | - | - |
| PP_PromotionStartDate | Mandatory. The share price available promotion period. Generate by system date | date | - | 2016-01-29 | - | - |

| | | | | | | |
|---|---|---|---|---|---|---|
| PP_PromotionEndDate | Optional. The share price available promotion period. | date | - | 2016-01-29 | - | - |
| PP_LastUpdateDT | The date and time when user update the price | datetime | - | 2016-02-17 04:29:48.533 | - | - |
| PP_Visibility | The product price visibility. If report by many people, after review by admin, will set visibility to disable | varchar | 1 | 'Y' | - | - |

Table Name: Report Price

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| RPID | The price reported by user unique identification number | int | 8 | 00 | PK | - |
| RP_PPID | The reported price product ID | int | 8 | 00 | FK | - |
| RP_reportUserID | The user who reported the price | int | 8 | 00 | FK | - |
| RP_SPID | The product price | int | 8 | 00 | FK | - |
| RP_SharedPriceUserID | The user who share the price | int | 8 | 00 | FK | - |
| RP_Status | The status of admin approval | varchar | 1 | 'Y' | - | - |
| RP_DT | Date and time when user report the price | datetime | - | 2016-02-17 04:29:48.533 | - | - |

Table Name: Subscription

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| SubscriptID | Subscription unique identification number by user A to user B ( user A subscribe user B) | int | 8 | 00 | PK | - |
| Sub_OwnerUserID | Subscription User A | int | 8 | 00 | FK | User |
| Sub_subUserID | Subscription User B | int | 8 | 00 | FK | User |

Table Name: PetrolPrice

| Attribute | Description | Data Type | Size | Format | Primary key(PK) / Foreign Key(FK) | Foreign key referenced table |
|---|---|---|---|---|---|---|
| PetrolID | Petrol History Price unique identification number | int | 2 | 00 | PK | - |
| PetrolType | Petrol Type whether RON95 or RON 97 | varchar | 5 | 'ab' | - | - |
| PetrolPricePerLitre | Petrol Price for that particular time | money | 10,2 | 0.00 | - | - |
| PetrolStartDate | The new petrol price enforced start date | date | - | 2016-01-29 | - | - |
| PetrolEndDate | The new petrol price enforced end date | date | - | 2016-01-29 | - | - |

## 4.8 System Architecture

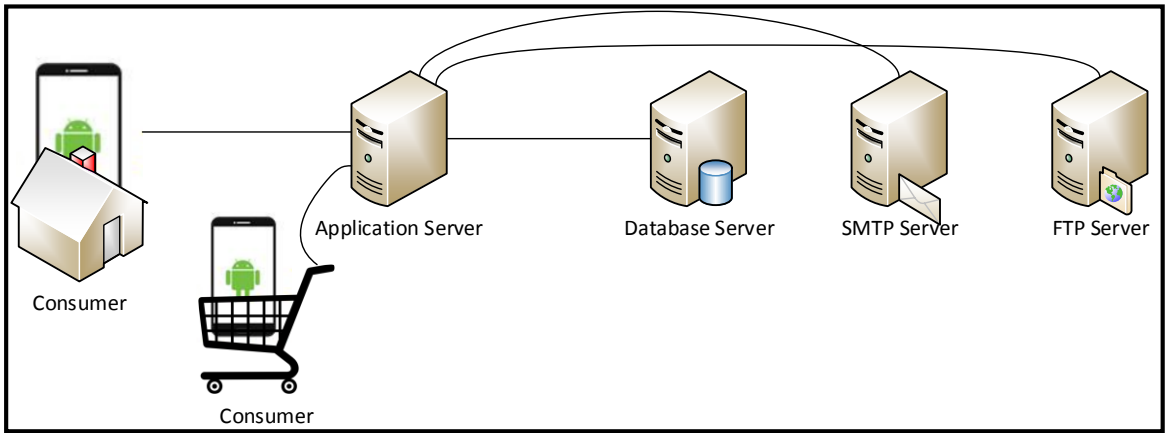



Figure 4-4-197 System Architecture

The proposed system will be build based on three tier architecture. They are namely the application layer, application server and database or better known as the file server. The mobile phone or personal computer serve as the presentation layer which act as user interface to accept input data. The store representative, seller or consumer will upload the information to application server. All input or hiding background data will store to database. Before storing to the database, the data received will need to go through application server.

The application server will responsible for all the communication with database server or other external server to ensure isolation and security protection. Basically the application server will act as middleware, it will coordinate the coordination with front end application and backend database file server. All the data passing or calling other external web server are required to go through application server. The application server will settle all the data formatting, coordination, communication in this case. This is to reduce the load of application layer which is having a limited resources such as CPU, RAM and so on.

The data will be processed before it further sends to server or database. For example, user name and password from the application layer will be send to server and uniquely validate the user account and retrieve all relevant information. The result of retrieving

will be processed to a user understandable format and arrange properly. Once done, it will then send back to presentation layer.

The figure below shows the overall development scripting or language use in each tier of architecture.

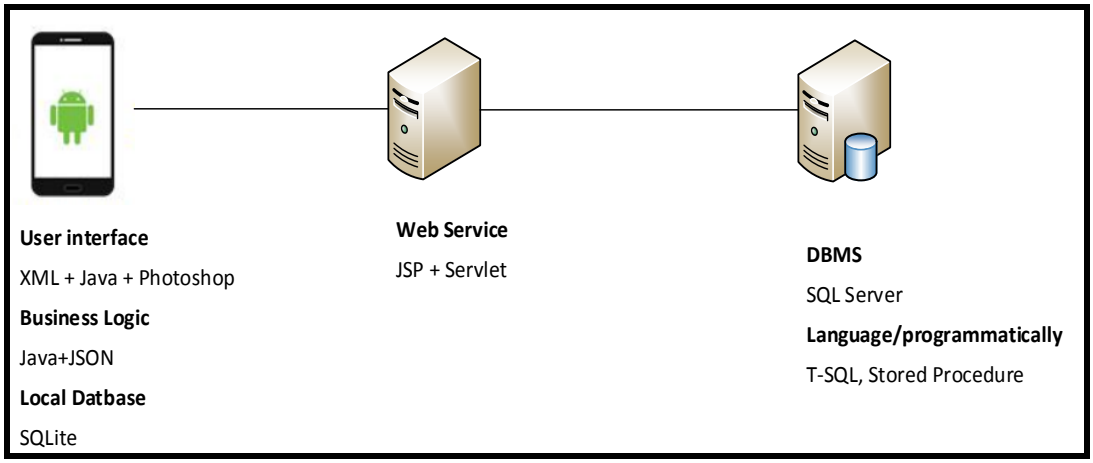



Figure 4-198 3 Tier Architecture

The application front end user interface will be coded by using java, xml with Photoshop together and SQLite as local database, whereas JSP+ servlet will be use to code the web service and SQL Server will serve as remote database. The system allow to install on android device only.

# CHAPTER 5 SYSTEM TESTING

## 5.1 Project Implementation & Testing

After the system design have been done, prototype will be throwaway and actual coding for system will be carry out. The architecture of system, user interface, network setup, algorithm, backend logic, internal and external server communication will start to carry out in development phase. Database will be built in SQL Server for data storing, and apache tomcat server will be built and execute for system communication between data storage tier and client tier. Later on, in the testing phase, system will be test by test cases based on test plan. The reliability will be test in user acceptance test and system test will be carry out to test the behavior of system. As long as there is no new severe incident raised, it is expected that the product has conformed to the objectives.

## 5.2 Implementation Issues and Challenges

After application design have been done, the design will turn into actual coding which start to implement all design features and architecture/ pattern. The actual implementation may have some slight discrepancy with what design had. This is due to some of the logic and problem that may arise in future cannot be forecast before actual implementation.

### (I) Image Transferring

**Challenge**

When the time I wish to pass image from application to web service, I had encountered some difficulties. That is because the way I sent text is using Json, and images is using byteArrayOutputStream. Initially I sent the image to network by using httpMultipartBuilder, it does not work, the bits or picture information lost in middle of sending to webservice. After that, I try several approach such as convert the image to hexademical, binary, byte Array and also try several different library method and sent to the webservice, it still fail to send through network or fail to receive in web service. All the error happens between the sending of image or not able to accept in web service.

**Solution**

Change strategy. Directly implement existing cloud stage server which is call parse.com instead of creating own server. The image link are store in database after every insertion of image to cloud. Whenever the image are required in application, the application will sent request to server and server will fetch the image cloud location from database back to client application. After that, the system will sent REST request to parse.com and retrieve image by the image link that fetched from database.

**(II) Database and server speed**

**Challenge**

The server performance very slow due to the server hardware have limitation. Personal computer with six (6) years ages act as server slow down the overall application performance. All the request sent to server required some time to perform and process, the arithmetic or algorithm operation required some time to be done. After the server process the data, it required another period of time to send through network to application side since low speed 0.5M WIFI internet are used in development. Besides that, the motherboard of server had been changed previously due to motherboard burned up. This issue cause the server processing may be slower than normal computer. In fact server process all client side request by execute different new thread, every user request, a new thread will be opened for every single new request. If a lot of thread execute concurrently in server, the server may crash and restart.

**Solution**

Development and testing in different workstation. Testing will be done in more powerful workstation and high speed internet.

**(III) Integration Of Project**

**Challenge**

This project consist of many different module. There exist many problem when integration of module. This is the first time we deal with this kind of big project. As the time goes on, experience are obtained and problem solving skill had

improved. As more module integrate together, there will have a lot of different issue such as:

1) Database design issue,
2) Version control issue,
3) Data passing issue

When A and B module successfully integrated, that does not mean B and C would not have problem when integrate and so on. The design of system are high coupling and all class are highly dependent on each other, thus more testing are required. Module B are start after module A finish. In this case, module A required data from table A join table B, there would not be any problem. However, when module B required table B and table C join data, problem then arise, this may due to table design problem, relationship problem or inheritance problem. This is because when module A are ongoing, we cannot forecast what will be happen in future and what data will be used in module B.

**Solution**

More experience are obtained after some period, and able to expect future needs and problem. This reduce the time of debugging error. Furthermore, each class was changed to less coupling after several error was occurred.

**(IV) Separation Testing**

**Challenge**

Since server and application are implemented separately and very often server codes were implement prior to application code. In this case it is hard to debug front end and backend separately. Backend end always will pending for frontend to complete so that it can pump data from front end to database. If we wait all the front end and backend to complete and do the test, it is harder to debug and may not know which part the error come from.

**Solution**

In order to do unit test on server code, we need to spent some time to create some stub (dummy client module) to test whether whole process flow are successful, or data passing between front end and server code are flawless or successful.

**(V) Screen Size and resolution of Different mobile device**

**Challenge**

The development phase only do testing on certain device which is Sony T2 Ultra. However there are many different device with different screen size for example 4.5, 5.0, 5.5, and 5.7 inch and resolution nowadays such as 780 x 1080p or 1028x2048p. This increase the difficulties to make the application fully operate able in different device (smart phone and tablet) and standardize UI in different device since this may cause content and UI of system not consistent in different device. Besides that, there is one minor problem which is network configuration. The network configuration in devices of different OEM (brand) are different. Same application is workable in Sony and Samsung devices but not workable in Xiao MI and HTC devices. This may due to network configuration or code problem. This issue had been raised officially by many developer in android website forum many times, yet this problem can't be solved.

**Solution**

Removed the module or change structure or module that trigger error if wish to continue used in Xiao MI of HTC devices

**(VI) Ideal Store Suggestion implementation**

This idea does not implement by any existing program yet, thus this required a lot of time to debug and create the algorithm. In fact the algorithm itself are not that hard, but need run many test to make sure the algorithm are applicable because in the process of implementing the algorithm, did not realized different test case or scenario. In other way, we not able to estimate different combination of data may break the algorithm flow and throw exception. In whole development, this module spent the most time and run through many test since many different combination of data are required.

## 5.3 Development Tools and Technology

1. **Android Debug Bridge ( ADB driver )**

   It's a "bridge" for developers to debug in their Android applications. This is done by connecting an android device by USB port that runs the ADB software through a PC.

2. **Google Location Services API**

   By using this API, developer do not need to detect location and do all calculation manually by own, the API can help to do all the calculation thus it simplify the development process.

3. **Gson API**

   A Java library that can convert Java Objects into JSON, sent to JSP and back. Thus this API are required import in both client application and server side.

4. **Microsoft Translator Java API**

   It build on top of Bing Translator. This API is a machine translation API. It can make the apps a translation enabled. This API replace the Google Translate API after deprecation of Google Translation API on December 1, 2011.

5. **Parse API**

   This is a REST API that can let the apps interact with Parse by HTTP request. Parse serve as cloud file server in our application. This API allow apps upload and download file or data that will be used in application.

6. **Google Play Services**

   With this API, the app can use google-powered features such as Maps and location navigation.

7. **Zbar API**

   This library facilitate QR and barcode scanning. This is basically a barcode and QR code scanning API. It allow apps to do image scanning and it able to recognize many types of Barcode and QR code standard such as EAN-13, UPC-A, UPC-E,

EAN-8.

8. **Commons Lang API**

   It provide extra utilities for string manipulation for classes in java.lang such as capitalize first character of every words or Date translation and calculation.

9. **V4 Android Support API**

   It provide a lot basic User interface design component and increase performance of Swipe Tab UI.

10. **v7 AppCompat Android Support API**

    This library build on top of existing android support v4 library. It adds support for actionbar and toolbar material design user interface. Besides that, it provide a lot of new User interface widget with better performance.

## 5.4 Testing

The application objective requirement validation test has been done by black box testing. Basically, black box testing is known as functional testing mainly focus on application behavior and determining application fulfills its functional requirements. Black box testing carry out by throwing input to system and monitor and output then comparing actual output with expected output without taking care of internal code structure.

### 5.4.1 Features to be tested

The following is the table that contains the function ID and its corresponding functions and estimated risk level.

### 5.4.2 Test Traceability Matrix

Table 5-1 Features to be tested

| Function ID | Function | Function Description |
|---|---|---|
| F001 | Provide Ideal Store Suggestion based on user location | Provide ideal store list based on user settings and user location. |
| F002 | Get Product Unit Price so can assist user in decision making | Calculate unit price of product |
| F003 | Search and get accurate and timely product information | Search Product by 4 different way which is product name, category, advanced google map search and barcode scanning search |
| F004 | Able create favorite list and cart list | Able to create custom product list, add and view product in favorite list , add and view product in cart list |
| F005 | Nearby store and location navigation | Navigate user to desired market and show all optimize route and path. |
| F006 | Sent product detail or user custom favorite list through social media or SMS | Sent information to social media including facebook, whatsapp, wechat, line, email and SMS as well |

### 5.4.3 Test Report

#### 5.4.3.1 F001 Provide Ideal Store Suggestion based on user location

| Feature | Input | Actual Output | Expected Output | Anomaly? (Yes or No) |
|---|---|---|---|---|
| Display ideal store combination in Setting=All mode | Cart List | Display all combination of market in all mode(single store, multiple store combination) | Display all combination of market in all mode(single store, multiple store combination) | No |
| Display ideal store combination in Setting=Cheapest Single Store Mode | Cart List | Display cheapest single store and single route information together with fuel and product price | Display cheapest single store and single route information together with fuel and product price | No |
| Display ideal store combination in Setting=Multiple Store Multiple Route | Cart List | Display Multiple Store Multiple Route information with single market navigation together with fuel and product price | Display Multiple Store Multiple Route information with single market navigation together with fuel and product price | No |
| Show Route and path sequence in Setting=Multiple Store Single Route and able to track user movement | Cart List | Display Multiple Store Single Route information with shortest distance travel to all store and show the route and path sequence travel to multiple store and able to track user movement / record down user path | Show the route and path sequence travel to multiple store and able to track user movement / record down user path | No |

### 5.4.3.2 F002 Get Product Unit Price so can assist user in decision making

| Feature | Input | Actual Output | Expected Output | Anomaly? (Yes or No) |
|---|---|---|---|---|
| Calculate Unit Price from product search result | Product information from SQL Server | Display unit price | Display unit price | No |
| Add Calculated History | Product information from SQLite | Display calculated unit price history | Display calculated unit price history | No |
| Calculate Unit Price by manual way | Product price and quantity | Display unit price | Display unit price | No |
| Calculate Unit Price by barcode detecting way | Detecting Barcode Number | Display unit price | Display unit price | No |
| Comparison Table | Unit Price Item | Display a table show the price for each product and each quantity ( how much will be price of product A if the quantity same as B and C) | Display a table show the price for each product and each quantity ( how much will be price of product A if the quantity same as B and C) | No |

#### 5.4.3.3 F003 Search and get accurate and timely product information

| Feature | Input | Actual Output | Expected Output | Anomaly? (Yes or No) |
|---|---|---|---|---|
| Search product by product category | Category Button | Display all search result product inside correct category | Display all search result product inside correct category | No |
| Search Product by product name in english | English Product name | Display all product search result that match the English and Chinese product name query | Display all product search result that match the English and Chinese product name query | No |
| Search Product by product name in mandarin | Mandarin Product name | Display all product search result that match the Chinese and English product name query | Display all product search result that match the Chinese and English product name query | No |
| Search Product by Advanced Search | Marker location | Display all product in particular market or store | Display all product in particular market or store | No |
| Search by Barcode | Barcode number | Display all product result in nearby market that match the barcode | Display all product result in nearby market that match the barcode | No |
| Share Product | Enter all product information | Share product successfully page display | Share product successfully page display | No |

#### 5.4.3.4 F004 Able create favorite list and cart list

| Feature | Input | Actual Output | Expected Output | Anomaly? (Yes or No) |
|---|---|---|---|---|
| Add product into favorite list from product detail | Product detail information | Successfully add product into favorite list | Successfully add product into favorite list | No |
| Add product into cart list from product detail | Product detail information | Successfully add product into cart list | Successfully add product into cart list together with fuel and product price | No |
| Add product from favorite list into cart list | Favorite list | Successfully add from favorite list to cart list | Successfully add from favorite list to cart list | No |
| Display custom favorite list | Favorite list from sqlite database | Display all custom favorite list correctly and after click inside, all the product added display correctly too | Display all custom favorite list correctly and after click inside, all the product added display correctly too | No |
| Display cart list | Cart List from sqlite database | Display all cart item that added previously correctly | Display all cart item that added previously correctly | No |
| Create new custom favorite list | User define custom favorite list | Successfully add a new favorite list | Successfully add a new favorite list | No |

#### 5.4.3.5 F005 Nearby store and location navigation

| Feature | Input | Actual Output | Expected Output | Anomaly? (Yes or No) |
|---|---|---|---|---|
| Navigate to market through maps | Market information and coordinate | Navigate user to correct market with optimize path and route | Navigate user to correct market with optimize path and route | No |
| Navigate to market through waze | Market information and coordinate | Navigate user to correct market with optimize path and route | Navigate user to correct market with optimize path and route | No |

#### 5.5.3.6 F006 Sent product detail or custom favorite list to social media or SMS

| Feature | Input | Actual Output | Expected Output | Anomaly? (Yes or No) |
|---|---|---|---|---|
| Sent Product detail to whatsapp | Product Detail from SQL Server database | Successfully sent Product detail to whatsapp | Successfully sent Product detail to whatsapp | No |
| Sent Product detail to facebook | Product Detail from SQL Server database | Successfully sent Product detail to facebook | Successfully sent Product detail to facebook | No |
| Sent favorite list to whatsapp | Favorite List from SQLite database | Successfully sent Favorite list to whatsapp | Successfully sent Favorite list to whatsapp | No |
| Sent favorite list to facebook | Favorite list from SQLite database | Successfully sent Favorite list to faceook | Successfully sent Favorite list to faceook | No |
| Sent favorite list through SMS | Favorite list from SQLite database | Sent to phone contact number successfully and successfully received by recipient device | Sent to phone contact number successfully and successfully received by recipient device | No |

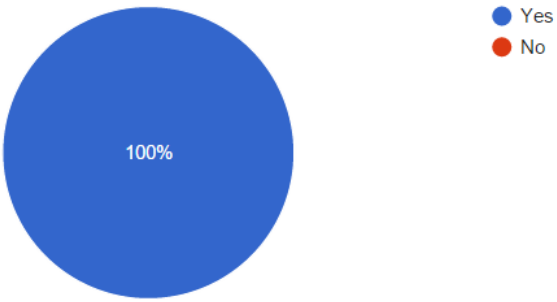
### 5.4.4 User Acceptance Test

The app were first giving to user and test it out. There are totally 6 feedback receiving regarding the APP. Some respondent did not provide valuable information but in overall, their feedback still give some positive feedback.

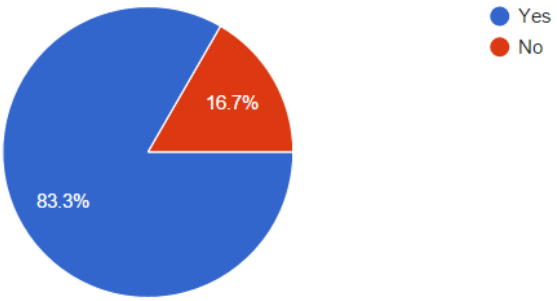
1. Do you think this apps able to help you save your effort and time to gather information?
(6 responses)

| |
|---|
| Yes |
| Yes |
| e |
| Yes. It save me a lot of time when making decision |
| Yes. |
| No. I cannot use it when I didn't have internet access. |

2. Will you think of using it when you buying on product (6 responses)

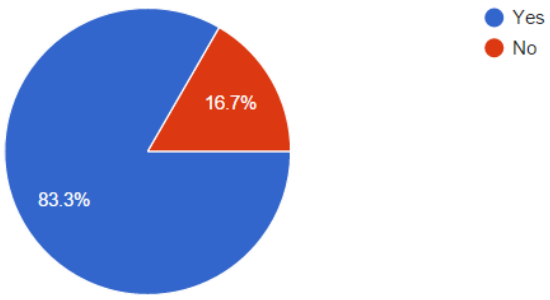



3. Do you think the ideal store suggestion features useful and will it able to solve your problem in decide which item to buy and which store to go?
(6 responses)



4. The unit calculator features, will you use this to decide which item worth to buy

(6 responses)




5. Other feedback (6 responses)

| |
|---|
| s |
| the apps able to help me to record best deals in my smartphone and share them easily and immediately to my friends. |
| It really useful when promotion of hypermarket is around the corner. I can buy all my goods in cheapest price and cost. |
| It assists me when shopping. It already become a helpful tools everytime I am shopping. |
| It save me a lot of cost of petrol. |
| My camera is not good enough for using the module of scanning barcode. |

6. Is it ease to use and able to solve your problem in short time and quick manner?(user friendly)

(6 responses)

| |
|---|
| s |
| Yes, it is user friendly. |
| yes. |
| Yes. It is fast and reliable. |
| Yes. It is an good apps. |
| Yes. It is easy to learn. |

# CHAPTER 6 SYSTEM EVALUATE AND DISCUSSION

## 6.1 Evaluation and Strength

In order to achieve the objectives of this project and solve the problem stated in problem statement, an Android application integrated with semantic search API, GPS technology, Parse cloud storage and barcode scanning technology together with translation dictionary API is developed. GPS technology use to detect user location and calculate the distance between market and user, then get the direction waypoints between multiple markets.

Besides that, semantic autocomplete, language detection and translation dictionary were used to complement each other and achieve higher efficient product searching. This two technology is the backbone of system to make the application workable. However, the most important one is Parse cloud storage, all of the binary file and image are stored in there.

In the final delivery system, the application brings enhancement to consumer shopping experience. This application able to assist them in decision making and purchasing planning. At the same time it works as your personal assistant, you can let the application record your current location or let the system suggestion you the product that worth to buy and also the store having cheapest product or perhaps the combination of store can let you spent the least and get the most product. Furthermore, the application allow you to get latest product information from brochure catalogue or product searching. Other from these, the application allow user share their favorite list to other friends through both social media or SMS will bring a lot of convenience whenever the user wish to ask for other help to do product purchasing.

## 6.2 Limitation

### I) Power draining

The application required a lot power to get the exact location of the user in every few second. The location Change Listener will trigger itself every 5 seconds and sent request to satellite in order to get current location. The satellite will received request and return current coordinate back to application. This operation execute very frequent and it turn out to drain battery faster than normal standby mode.

### II) Accuracy of Location

GPS device location tracking may not be very sensitive to location change and still this depends on device GPS signal. Typically, the sensitivity is around +-1km thus this is a big challenge for device to get market accurate location.

### III) Performance of application heavily depends on internet speed

The application required internet to perform several operation such as communication with API, retrieving data from database and retrieving large amount of image from cloud storage. The performance of application depends on user's device speed itself and the internet speed available. If the internet speed is low, user can expect to have lower performance when data loading and algorithm calculation.

### IV) Ideal Store Suggestion only allow maximum combination of 4 store

The store suggest in ideal store suggestion module allow maximum four (4) store combination. This is because processing speed will be heavier as combination increase. Besides that, through the survey done show that majority of user tends to do price market comparison for 3-4 store only. Therefore, 3-4 store is the ideal range of combination for store suggest. The application will only take top 4 nearest store into consideration on algorithm calculation.

### V) Multiple destination navigation

For instance, the application not able to carry out multiple destination voice navigation due to lack of current existing technology. However, it still can show the route and path to user together with path tracking to guide user to destination. We may need more R & D on this thus this will be put in future enhancement and release.

## 6.3 Future Enhancement

- **User Customize Ideal Store Suggestion Choice**

  For time being, all the store suggest are based on nearest location. However, user may have some preferred store and some disliked store. Therefore, in future enhancement, user can choose whether to get store suggestion based on nearest store and auto generated the nearest store list or system will display a list of nearby store and let user tick and decide the store they want to take into consideration are.

- **Notification Features**

  User will be allowed to customize own price alerts notification. The apps allow you to set the alert notification, so whenever if an item falls into your desired price range, it will send push notification when you device in online mode. The price are allow to send to social media such as twitter, facebook and whatsapp, and they can see where to get the best deal on those items.

- **Admin Module**

  Admin module will be added in future on product approval, location verification, price report action. Besides that, admin should be able to edit, remove and add product or catalogue information but not normal user. Furthermore, a user dashboard or statistic should be provided to detect user unusual behavior or spamming action.

- **User Loyalty Module**

  The user reputation (Star Rank at right hand side of each user name) will change based on the amount of like or report. This can encourage user to share more new item and correct information. After period of time and reach certain membership level (2 star), user can use these credits to redeem some gift or voucher through application.

- **Product Expiration Tracking**

This module will be added in future plan. The product added can be categorize by different category and application will display the freshness level of product based on product category by displaying different color of background or bulb indicator to indicate whether particular product is fresh or spoil.

## CHAPTER 7 CONCLUSION

In this project, the deliverable which is an application for the mobile device that works as a consumer awareness application that can assist consumer in decision making when shopping. The main purpose of this Conawa application is to serve as an information sharing platform mainly for both online and offline consumers and seller. This application will be able to assist consumer in decision making when shop for a product. This could be able to save the cost and time to be a smart consumer especially price inflation happened frequently nowadays. Conawa is a consumer awareness application that provide functionality for the consumer to check or calculate price of items, in different stores that are located near the user. It harness the power of consumer and retailer outlets to share the item information and the application will suggest the ideal store choice for consumer.

**REFERENCE**


A,E, Md, Saad, Z, Saad, A,K, Ahmad, S, Abdullah, F, Ab, Razak 2011 January, 'Mileage comparison between RON 95 and RON 97 in constant speed test', *2011 IEEE 7th International Colloquium on Signal Processing and its Applications.*

Annie,a 2015, *App Annie Index Market Q3 2014,* San Francisco: App Annie Designated Agent, pp. 3-7, viewed 14 August 2015, <https://s3.amazonaws.com/files.appannie.com/reports/App-Annie-Index-Market-Q1-2015.pdf >

Bäurle, F 2011, *A User Interface for Semantic Full Text Search*, pp.1-12, viewed 2 September 2015. <https://ad.informatik.uni-freiburg.de/files/ui_semantic_fulltext_search>

Buchhold, B 2010, *SUSI: Wikipedia Search Using Semantic Index Annotations* pp.15-44, viewed 31 August 2015, <https://ad.informatik.uni-freiburg.de/files/susi>

F, Suchanek, G, Kasneci, and G, Weikum 2007 *YAGO: A core of semantic knowledge*

Google.com 2005, *Mobile Analytics for Applications | Google Mobile Analytics – Google.* Available from: <http://www.google.com/analytics/mobile>. [2 September 2015].

Hannah, B, Florian, B, Bjorn, B, Elmar, H 2012, *Broccoli: Semantic Full-Text Search at your Fingertips,* CoRR, pp.1-10 viewed 25 August 2015 <http://arxiv.org/pdf/1207.2615.pdf >.

Hansen,T and Cumberland,F and Hans, S, Solgaard 2004, *How the Measurement of Store Choice Behaviour Moderates the Relationship between Distance and Store Choice Behaviour,* A Study of the Interface Between Retailers and Consumers, Kluwer Academic Publishers, pp12-17, viewed 2 October 2015. <http://www.marketing-trends-congress.com/archives/2013/pages/PDF/665.pdf>.

Holger, B, Alexandru, C, Fabian, S, Ingmar, W 2007, 'ESTER: efficient search on text, entities, and relations', *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '07*, New York, pp. 671–678. Available from: ACM Portal: ACM Digital Library. [2 September 2015].

Inazaruk (22 May 2011). *Android Application, Android libraries and Jar libraries*. Available from https://devmaze.wordpress.com/2011/05/22/android-applicationandroid-ibraries-and-jar-libraries/ [6 April 2015]

Intelligence on TAP 2009, November, *Advantages & Disadvantages of prototyping process model*. Available from <http://www.iotap.com/Blog/tabid/673/entryid/124/Advantages-Disadvantage-of-Prototyping-process-model.aspx> [10 April 2015]

McCarty B. (2011, December). *The History of the Smartphone*. Available from http://thenextweb.com/mobile/2011/12/06/the-history-of-the-smartphone/ [2 April 2015]

Mitre 2009, System *Design and Development*. Available from <http://www.mitre.org/publications/systems-engineering-guide/se-lifecycle-building-blocks/system-design-and-development>. [3 April 2015].

Perez, S 2013, *In-App Purchase Revenue Hits Record High: Accounts For 76% Of U.S. iPhone App Revenue, 90% In Asian Markets*, TechCrunch. Available from: <http://techcrunch.com/2013/03/28/in-app-purchase-revenue-hits-record-high-accounts-for-76-of-u-s-iphone-app-revenue-90-in-asian-markets/> [2 September 2015].

Radack, S n.d., *The system development life cycle*. Available from: <http://csrc.nist.gov/publications/nistbul/april2009_system-development-life-cycle.pdf> [27 October 2014].

Thinkgaming.com n.d., Top Grossing iPhone Games, Installs, ARPU & revenue estimates - United States. Available from: <https://thinkgaming.com/app-sales-data/> [2 September 2015].

Wright, R 2013, April, *The Evolution of the Smartphone In 7 Releases*. Available from: <http://www.crn.com/slide-shows/mobility/240152197/the-evolution-of-the-smartphone-in-7-releases.htm> [2 April 2015].

**APPENDIX A**

# FINAL YEAR PROJECT WEEKLY REPORT

*(Project I)*

WEEKLY REPORT 1

| | |
|---|---|
| Trimester, Year: Year 3 Trimester 3 | Study week no.:2 |
| Student Name & ID: Ang Sinyee & 1104648 | |
| Supervisor: Mr. Ku Chin Soon | |
| Project Title: Consumer shopping apps decision support tool | |

| |
|---|
| 1. WORK DONE<br><br>Start correction previous error in IIPSPW. |
| 2. WORK TO BE DONE<br><br>Do more fact finding, research of the journal paper |
| 3. PROBLEMS ENCOUNTERED<br><br>Lack of time. |
| 4. SELF EVALUATION OF THE PROGRESS<br><br>Normal. |

_____ _____

Supervisor's signature Student's signature

WEEKLY REPORT 2

| | |
|---|---|
| Trimester, Year: Year 3 Trimester 3 | Study week no.: 4 |
| Student Name & ID: Ang Sinyee & 1104648 | |
| Supervisor: Mr. Ku Chin Soon | |
| Project Title: Consumer shopping apps decision support tool | |

1. WORK DONE

Continue on correct and start system analysis part, rewrite system development methodology, use case diagram and so on

2. WORK TO BE DONE

Class diagram, activity diagram, object diagram.

3. PROBLEMS ENCOUNTERED

Lack of time.

4. SELF EVALUATION OF THE PROGRESS

Normal.

_____ _____

Supervisor's signature Student's signature

WEEKLY REPORT 3

| | |
|---|---|
| Trimester, Year: Year 3 Trimester 3 | Study week no.: 6 |
| Student Name & ID: Ang Sinyee & 1104648 | |
| Supervisor: Mr. Ku Chin Soon | |
| Project Title: Consumer shopping apps decision support tool | |

1. WORK DONE

Research on Java and JSP programming.
Finish design the use case diagram, activity diagram and so on

2. WORK TO BE DONE

Object diagram, research on more idea and design the business conceptual model.

3. PROBLEMS ENCOUNTERED

Lack of time.

4. SELF EVALUATION OF THE PROGRESS

Normal.

_____ _____

Supervisor's signature Student's signature

WEEKLY REPORT 4

| | |
|---|---|
| Trimester, Year: Year 3 Trimester 3 | Study week no.:8 |
| Student Name & ID: Ang Sinyee & 1104648 | |
| Supervisor: Mr. Ku Chin Soon | |
| Project Title: Consumer shopping apps decision support tool | |

1. WORK DONE

Finish the documentation.

2. WORK TO BE DONE

To design the system interface
To create a database that able to communicate with the system interface

3. PROBLEMS ENCOUNTERED

Lack of time.

4. SELF EVALUATION OF THE PROGRESS

Normal.

_____ _____

Supervisor's signature Student's signature

WEEKLY REPORT 5

| | |
|---|---|
| Trimester, Year: Year 3 Trimester 3 | Study week no.:10 |
| Student Name & ID: Ang Sinyee & 1104648 | |
| Supervisor: Mr. Ku Chin Soon | |
| Project Title: Consumer shopping apps decision support tool | |

| |
|---|
| 1. WORK DONE<br><br>Finish the documentation. |
| 2. WORK TO BE DONE<br><br>To design the system interface<br>To create a database that able to communicate with the system interface |
| 3. PROBLEMS ENCOUNTERED<br><br>Lack of time. |
| 4. SELF EVALUATION OF THE PROGRESS<br><br>Normal. |

_____ _____

Supervisor's signature Student's signature

WEEKLY REPORT 6

| | |
|---|---|
| Trimester, Year: Year 3 Trimester 3 | Study week no.:12 |
| Student Name & ID: Ang Sinyee & 1104648 | |
| Supervisor: Mr. Ku Chin Soon | |
| Project Title: Consumer shopping apps decision support tool | |

1. WORK DONE

Finish the documentation.

2. WORK TO BE DONE

To design the system interface
To create a database that able to communicate with the system interface

3. PROBLEMS ENCOUNTERED

Lack of time.

4. SELF EVALUATION OF THE PROGRESS

Normal.

_____ _____

Supervisor's signature Student's signature

## APPENDIX B

## ACADEMIC POSTER CONAWA

conawa

**CONSUMER SHOPPING APPS DECISION SUPPORT TOOL**

**By Ang Sinyee**

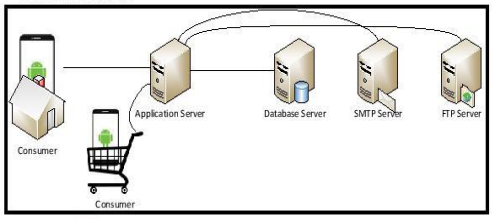

**Abstract**

This application brings enhancement to consumer shopping experience, it able assist them in decision making and purchasing planning. At the same time it works as your personal assistant, you can let the application record your current location or let the system suggestion you the product that worth to buy and also the store having cheapest product or perhaps the combination of store can let you spent the least and get the most product. Furthermore, the application allow you to get latest product information from brochure catalogue or product searching. They can share this information with the people around them as well.

After GST implementation and price inflation, when you stand in front of a shelf, would you hesitate before buying a product and curious whether the product sold with far lower price in other store?

**Conawa App**

**Architecture**





**Commercialization Value**

- Include Advertisement and google analytic that can get good revenue.
- International market or shopping complex that interested in this technology :

**Decision Making Support**

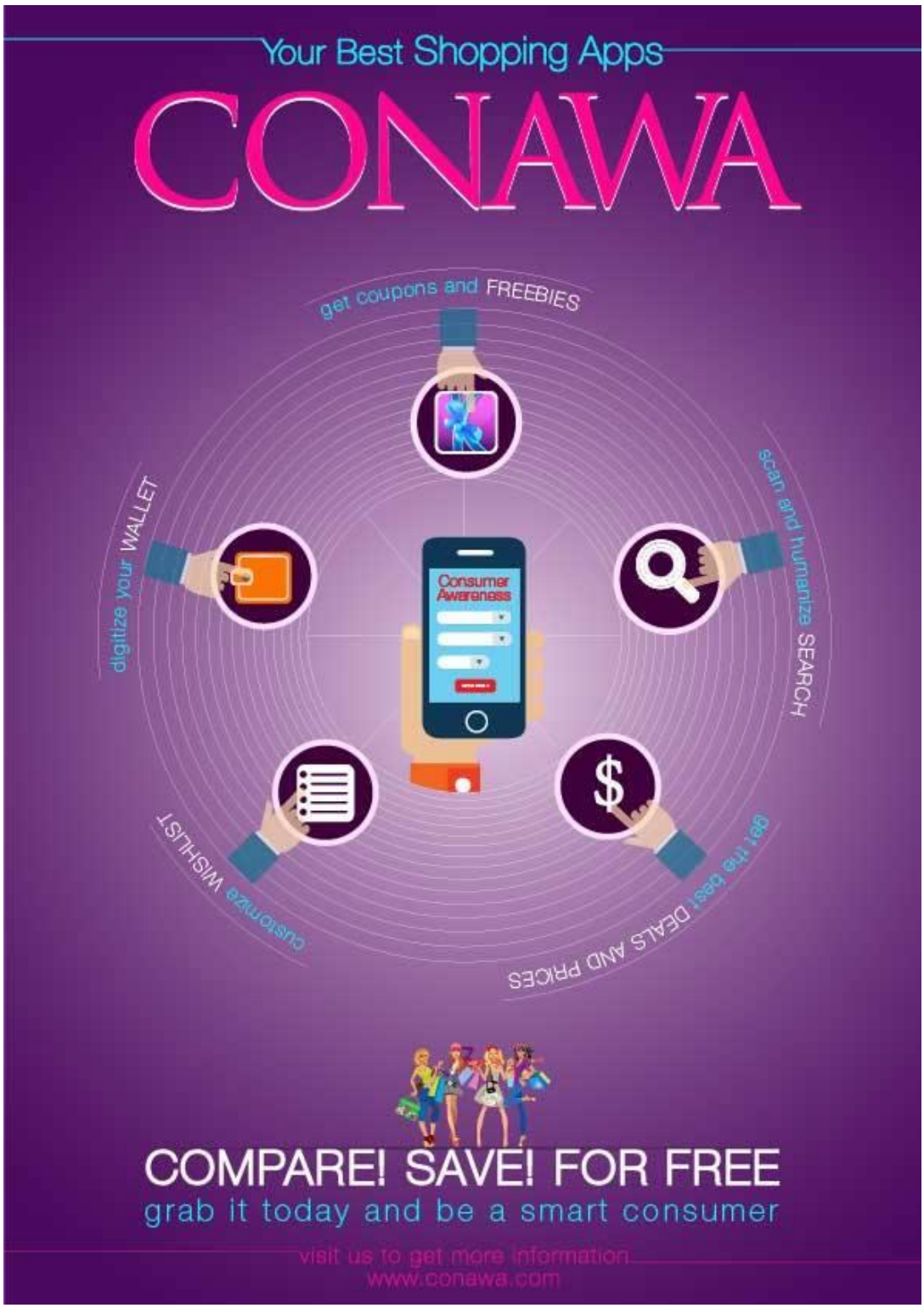
Application cannot force user to do decision since every user may have different preferences on store choice such as price or distance. Therefore, this idea produce some possible ideal store and let user to decide themselves whether which store is ideal to them.

**Key Features**

- Get Latest Product Promotion and store promotion information
- Suggest User which store to go and where to purchase based on user favorite and cart list
- Create your Own shopping list and share with friends
- Get Product Unit Price so can assist user in decision making in terms on packaging quantity and price

**APPENDIX C**

**MARKETING POSTER CONAWA**



## APPENDIX D

## ORIGINALITY REPORT