LOCATION BASED SOCIAL MOBILE APPLICATION FOR FOOD

KOH SOO CHENG

A proposal submitted
in partial fulfilment of the requirements for the award of
Bachelor of Science (Hons.) Software Engineering

Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

September 2013

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature	:	
Name	:	Koh Soo Cheng
ID No.	:	911111-06-5577
Date	:	26 August 2013

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"LOCATION BASED SOCIAL MOBILE APPLICATION FOR FOOD"** was prepared by **KOH SOO CHENG** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Hons.) Software Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature	:	
Supervisor	:	Dr. Simon Lau Boung Yew
Date	:	

LOCATION BASED SOCIAL MOBILE APPLICATION FOR FOOD

ABSTRACT

This project aims to produce a social mobile application for collaborative sharing of food reviews through Mobile application. The food review application includes features such as the location of the eating outlets via GPS location or network location to determine the users' location. With the location of the eating outlets and users' current location, the application can show the nearby foods or eating outlets. The application also enables users to interact with their friends or surrounding people by following their activities via a following-follower system to comment and rate food and food outlets. The application provides a user information about the food and eating outlets around him and help the user to decide where and what to eat. The system is developed based on a client-server architecture. Android native application is implemented as the client side while Apache Web server, MySQL database and PHP are implemented on the server side. User acceptance testing results show that objectives of this project have been achieved. Users are able to find and access information about the most popular, cheapest, most recent or nearest food and eating outlet around him and interact with other users by sharing food reviews.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS / ABBREVIATIONS

API	application programming interface
GPS	global positioning system
HTTP	Hypertext Transfer Protocol
JSON	JavaScript object notation
ORM	Object Relational Mapping
R	radius of the Earth

CHAPTER 1

INTRODUCTION

1.1 Background

In recent time, people are interested to document their lives, taking the photo of special occasion, share good food with others, etc. People like to take photos of food they ate and post on social platforms such as Facebook, Twitter, Instagram, Foursquare, etc. Statistics show that 52% of people take photos with their mobile phones and another 19% upload those photos to the web at least once a month (Todd Wasserman, 2011). Eating outlets will be promoted indirectly if all the food photos have reviews, ratings and locations attached to it and accessible via a mobile application.

1.2 Problem Definition

Nowadays, most people have their meals outside. People are interested to know where all the delicious food around them is without having to visit every restaurant. However, deciding on what and where to eat is difficult because there are many eating outlets where food quality is not known. Besides, good eating places may not be easily spotted. A location based social mobile application for food enables us to easily search for good quality food in our surrounding without the need to physically be present at all food premises. Users in a community may upload photos and reviews of food and the eating outlets via a mobile application for sharing with other users.

1.3 Problem Statement

This project is motivated by the following gap: There is a need for a social mobile application to assist users to make decision on the eating outlet and food considering the preferred locations, taste and price of the food as reviewed by other users.

1.4 Objectives

The objectives of this project are to design and develop a mobile application which:

- Allows users to search for nearby food or eating outlets in users' surrounding by category. Nearby food or eating outlets maybe displayed based on user preference without user intervention on the application.

- Creates a mobile online eating outlet food directory with information such as food list, food prices, address, phone number, illustrations of the food and reviews without having to go physically to the restaurant. Users can read reviews of an outlet to decide whether they want to eat in the outlet and make phone reservation; and

- Creates a social mobile application for users to share food information, pictures and reviews and interact with each other. Users can follow, like/dislike or comment other users' posts.

1.5 Project Scope

The system is developed based on a client-server architecture. The client side which displays the mobile user interface, handling user interaction and request services of the server is implemented as an Android native application. Apache Web server, MySQL database and PHP are implemented on the server side.

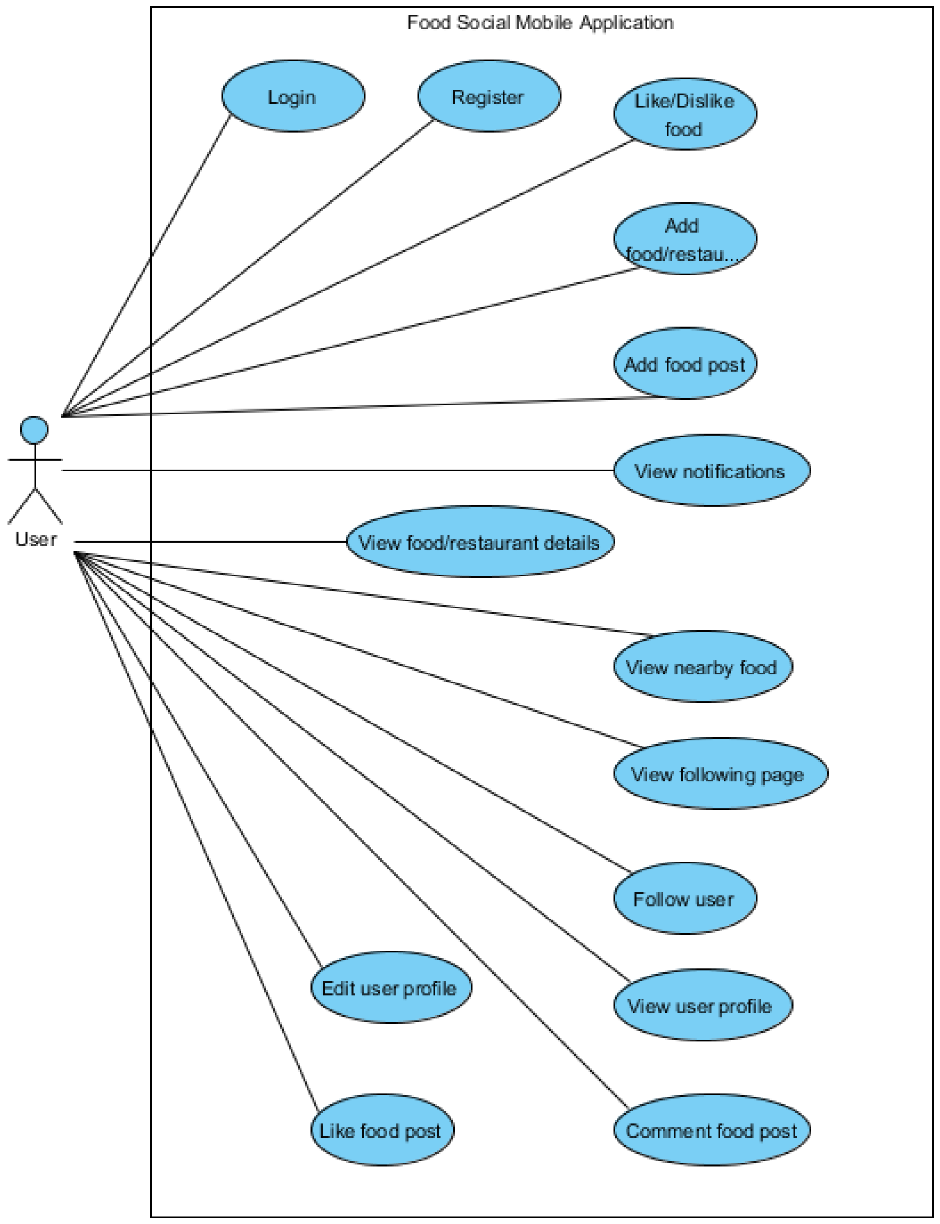


Figure 1.1: Use Case Diagram

Figure 1.1 shows the use cases of the application which include:

- Register and login
  - Login screen will be shown if users are not logged in.
  - A button "To Register" at the login screen for users to register.
- View nearby food
  - Users can choose to view nearby food.

- A list of nearby foods will be shown when users start up the application.

- View details of food/eating outlet
    - Details of food are shown when a food review is clicked. It includes the food picture, distance, location, price, creator name, created date, number of persons who like and dislike.
    - Details of eating outlets are shown when outlet’s name is clicked. It includes the location and the foods in the outlet.
- Comment food post
    - Users can comment on food reviews.
    - Comments on food reviews are shown in the food reviews.
- Like or dislike food
    - Users can like or dislike food.
    - The number of users’ likes and dislikes are shown in the food.
- Like food review
    - Users can like food review.
    - The number of users’ like are shown.
- Follow user
    - A user can follow other users.
    - After the follow relationship is established, any new food reviews posted by the followed user will be shown in a user’s following page.
- View following page
    - Following page shows the food reviews of the followed users.
- View user profile
    - Shows a user’s information such as username, profile picture, followers and own food reviews.
- Add food post
    - Users can add food post by taking picture using mobile devices’ camera or choose picture from gallery and then fill in the information such as restaurant name, food name, comment and price.
- View notification
    - Notification screen shows the activities which are performed on a user.

  - When other users comment, like a user's food reviews or follow the user, it will be shown on his notification screen.
- Edit user profile
  - Users can edit their details such as users' profile picture, biography and email.
- Add new food/outlet
  - Users can add new food in an outlet.
  - Users can add new outlet to the database.

The system is deployed and implemented according to the architecture as shown in Figure 1.2.Android native application is used as the client side of the system. HTTP request will be sent by the Android Native Application to PHP Server to request for services. At the PHP server, HTTP requests sent from the Android native application are processed by the Routing module first, it route the HTTP request to the Controller module according to the requests' types and names. Migration module is used to create or update database table. A table named migration is created to store the changes of the database so that the database can be rollback to previous state if anytime goes wrong. The PHP server and the database server is on the same local area network so that data from database can be accessed faster. Every database table has a Model class that is used to map the database table. The Model class is then used to access or update the database in the database server.

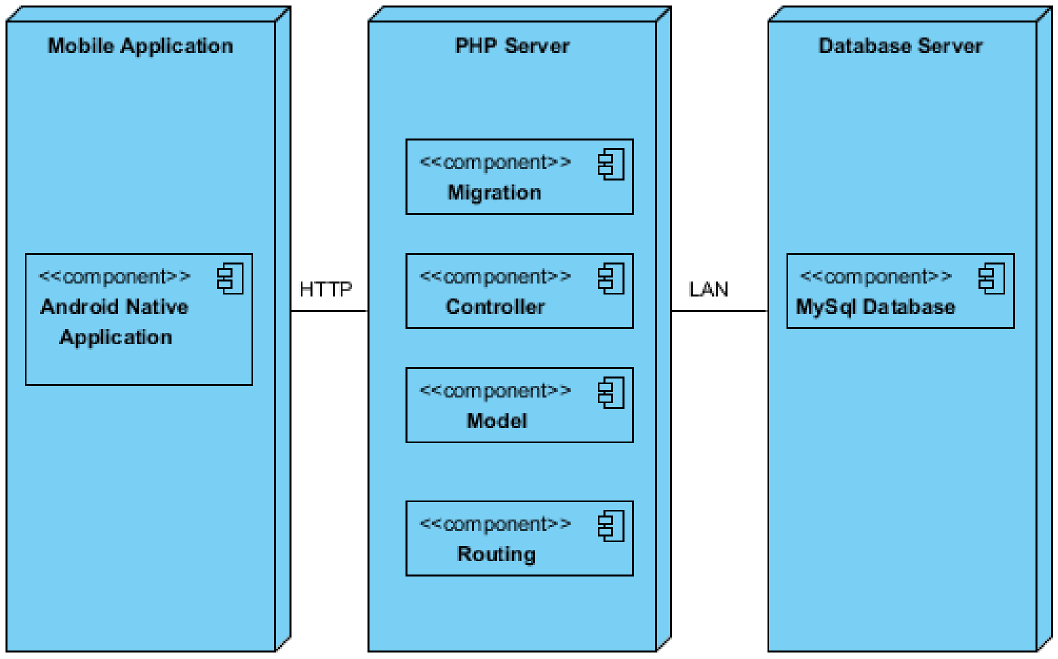


Figure 1.2: Deployment Diagram

1.6 Thesis Outline

The thesis is organized as follows:

In Chapter 2, key concepts related to this thesis together with existing related work on social mobile application and their respective features and limitations will be discussed.

In Chapter 3, software development methodology, technologies and tools used in this project will be discussed.

In Chapter 4, the design models of the system in the form of structural model such as entity relationship and interaction models such as sequence diagrams and activity diagrams will be showcased and explained.

In Chapter 5, outcomes of the project will be presented and discussed. The resulting runtime features of the mobile application will be snapshot. Furthermore, test results will also be discussed.

Chapter 6 will conclude and summarize the contributions of this thesis and recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter first introduces the key concepts related to this thesis, i.e. “crowdsourcing” and “location based services” and the Android Location API followed by existing related work on social mobile application and their respective features and limitations will be discussed.

2.2 Crowdsourcing

Large number of food reviews is required to recommend eating outlets to users. To get that number of food reviews, crowdsourcing is the way to gathers food reviews. Brabham (2008) states that “Crowdsourcing is an online, distributed problem-solving and production model”. Food reviews are written by users rather than hiring people to write it. Wikipedia is an example of successful crowdsourcing platform, its large database of information is written by many different people.

It is very difficult to hire people to write reviews for every eating outlets due to there are so many eating outlets out there and new eating outlets can be opened anytime. By using the crowdsourcing model,

i) Reviews can be contributed by users from anywhere.
ii) New or unknown eating outlets can be spotted by users.
iii) No money is spent on hiring people to write reviews for eating outlets.

However, there are some disadvantages of using crowdsourcing. Due to any user can write reviews, the reviews may not be accurate compared to reviews written by professional food reviewers. Therefore, there is a need to check the reviews written by users are appropriate or not. Location based services will be talked in next section.

2.3 Location Based Services

Location based services are services that are used to provide the location information to the users. With over 770 million GPS-enabled smartphones (Business Insider, 2013), location based services can be accessed from mobile devices. It allows application to be aware of users' location. Thus, it allows users to find the nearest eating outlets. By integrating location based services into mobile applications, some interesting features can be added to help recommend eating outlets such as:

- *Location based advertising*. Both real time and historical data mobile location data can be used by location based advertising to deliver to shoppers where mobile users actually shop and eat (Olenski, 2013). The advertisement will only be shown when users are near to the eating outlets so that the probability of users going to the eating outlets is higher because the eating outlet is nearby. Coupons or promotions can also be shown to attract more customers.
- *Awards and badges*. Some users may not like to review foods but only interested in reading others' reviews. Awards or badges can be awarded to users who meet certain criteria such as review more than 10 foods or review more than 3 foods in the same eating outlets.
- *Track families and friends*. Users can know where their families or friends eat at. Thus, users can join their families or friends to eat together.
- *Location of the food*. With the location on the food, the existence of the eating outlets can be alerted to users.

2.4 Android Location APIs

Android has updated their Location API in Google Play services for Android developer to obtain user's location easily on May 2013. It is easier to build location-aware applications without the need to focus on how the underlying location technology work.

By using the latest Location API, the mobile devices will intelligently get the best and most recent location from GPS or network location by specifying the needs like "high accuracy" or "low power". Thus, the power usage is minimized. Besides, the location can be accessed immediately because Google Play service will constantly store the best and most recent location. Therefore, nearby food and eating outlets of the user can be accessed immediately.

2.5 Related Work

Every recommendation application for eating outlets employs certain unique approach to recommend eating outlets such as showing trending place, leaving tips on a place or explore nearby eating outlets. Some of the existing food recommendation applications are:

2.5.1 Foursquare

Foursquare is a location based social networking application that allows users to check in and comment on their check in. Users are able to search for trending places and leave tips on a place. However, Foursquare is not specialized for food but for more general check-in type of application at various locations. Hence, users may be flooded with posts in their following page which are not related to food as shown in Figure 2.1.

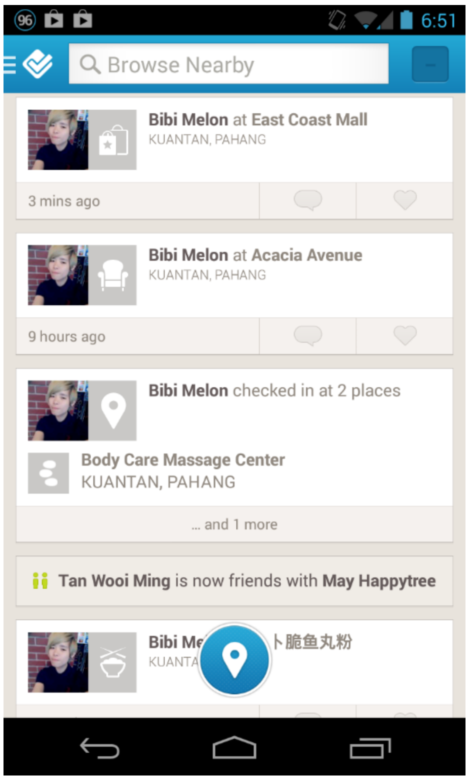

6:51
Browse Nearby
Bibi Melon at East Coast Mall
KUANTAN, PAHANG
3 mins ago
Bibi Melon at Acacia Avenue
KUANTAN, PAHANG
9 hours ago
Bibi Melon checked in at 2 places
Body Care Massage Center
KUANTAN, PAHANG
... and 1 more
Tan Wooi Ming is now friends with May Happytree
卜脆鱼丸粉

Figure 2.1: Foursquare

2.5.2 Foodspotting

Foodspotting is another location based social networking application specialized for food. It allows user to upload food photo and review the food. Besides, it also allows user to search for nearby food and follow friends' activities. Figure 2.2 shows the screenshot of Foodspotting, nearby food is shown in this page.

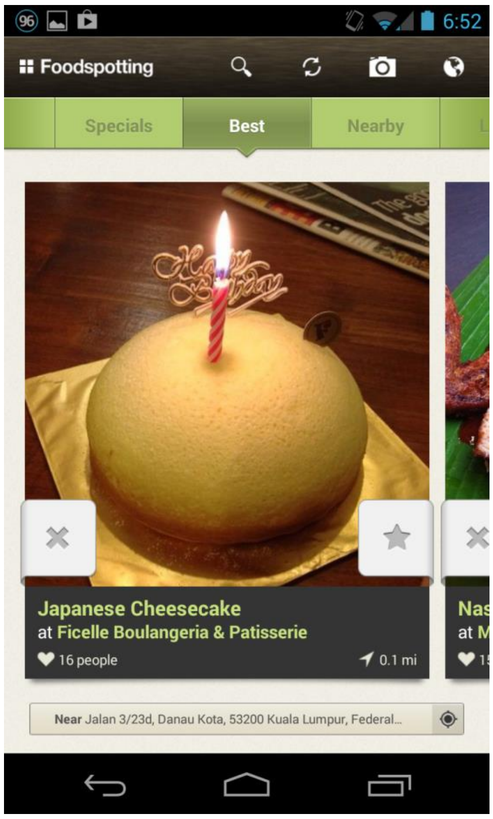

6:52
Foodspotting
Specials
Best
Nearby
Japanese Cheesecake
at Ficelle Boulangeria & Patisserie
16 people
0.1 mi
Near Jalan 3/23d, Danau Kota, 53200 Kuala Lumpur, Federal...

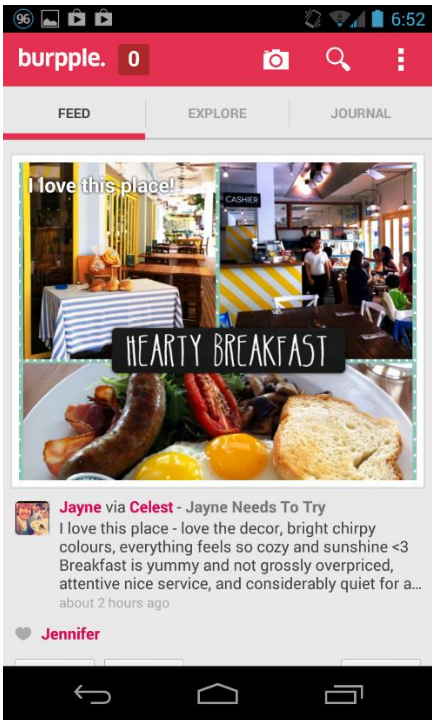
Figure 2.2: Foodspotting

2.5.3 MisoTrendy

MisoTrendy helps to find trending places from Foursquare even users are not near to the location because in Foursquare, users are only allowed to look for nearby places and friends' check-ins. Thus, users can search for eating outlets in a new city before going there.

2.5.4 Burpple

Burpple is a location based food recommendation application that has following-follower system like Foodspotting and Foursquare. Its uniqueness is that it supports uploading post to both Burpple and Instagram at one time and a special feature named "Explore". There are five options in the explore page. Users can explore nearby good food, famous food reviewer, popular food, coffee and brunch. Figure 2.3 shows the "Feed" page of Burpple, followed users' food posts are shown in this page.



Jayne via Celest - Jayne Needs To Try
I love this place - love the decor, bright chirpy colours, everything feels so cozy and sunshine <3 Breakfast is yummy and not grossly overpriced, attentive nice service, and considerably quiet for a...
about 2 hours ago

Figure 2.3: Burpple

From the study, key limitations and gaps of the applications reviewed are identified as:

i) *Showing the price of food*. Price is an important factor for deciding what to eat. Some people may want to save money and have a budget for their meal.

ii) *Categorization of food*. Users should be able to filter the food review by category, such as vegetable, fish, pork, etc.

iii) *Control of the search radius*. Users should be able to control the search radius so that they can decide how far they want to eat at.

Due to the abovementioned limitations and gaps, it is the vision of this research to design and develop a novel mobile application which allows users to search for food of interest to them based on location and feedbacks from the other users. Users are able to read food reviews without much effort like asking friends, searching the Internet or tasting out them physically. Besides, users can also have an online eating outlet menu that allows them to check for nearby food and its price so that user's budget can be met.

2.6 Conclusion

Related concepts such as “crowdsourcing” and “location based services” have been discussed in this chapter. Besides, there have been a comparative study and analysis of related works. From the study, limitations and research gaps are identified which include the non-existence of the price for food, categorization of food and search radius of the eating outlets. In the following chapter, the design and development methodology of the project will be discussed.

CHAPTER 3

METHODOLOGY

3.1 Introduction

The design and development methodology of the project will be described in this chapter. This chapter also includes the software development methodology, and technology and tools used in this project.

3.2 Project Description

This project aims to provide a social mobile application for sharing food reviews. The food reviews is associated with the location of the eating outlets by using the features on the mobile devices such as GPS location or network location to determine users' location. At the time a user starts up the mobile application, users' location will be determined and used to query the nearby eating outlets from the database. List of food reviews that are near to users will then be shown on the main screen of the mobile application as shown in Figure 3.1.

Features such as "following-follower", commenting and rating systems are also implemented in this project to allow users to interact with others. These features can increase user's engagement. Sharing, commenting and rating friends or families' food reviews cause users to spend more time on the application. By following other users, users can see the latest food reviews posted by their followees in the

"Following" screen of mobile application which as shown in Figure 3.1 but the list of food reviews are posted by the followees rather than the nearby users.

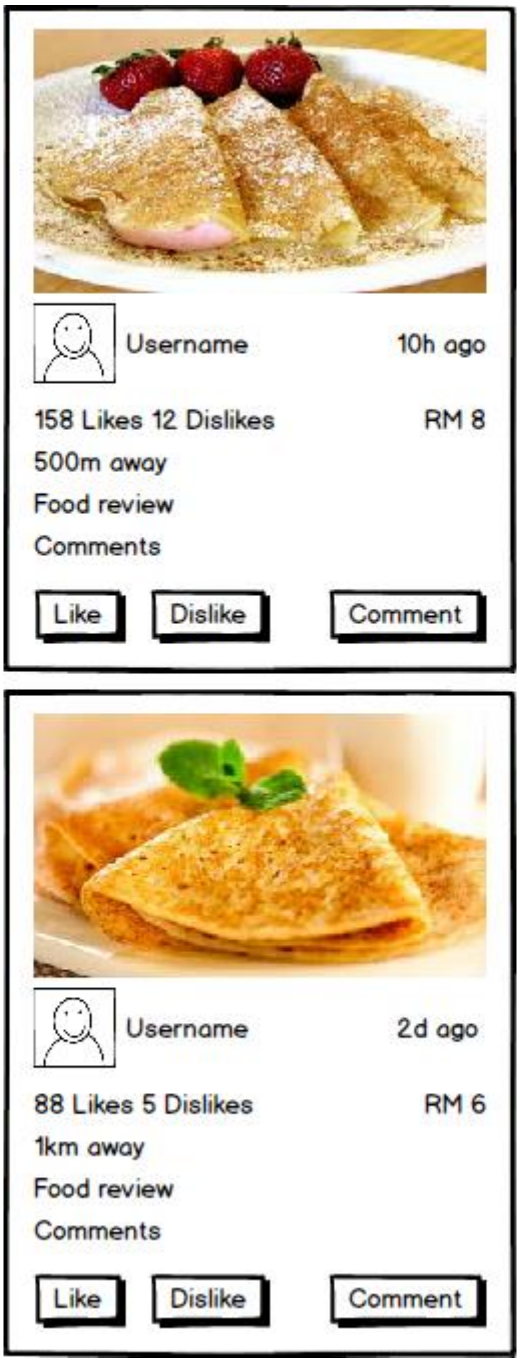


Figure 3.1: List of Food Reviews on Mobile application

For food reviews upload, food pictures can be taken using the camera on mobile devices or chosen from the gallery. The location of the food is determined using the user's location. A user's location (longitude and latitude) is sent to

Foursquare API to query the nearby eating outlets. Then, a list of eating outlets is shown on the screen of the Mobile application for users to choose.

3.3 System Architecture

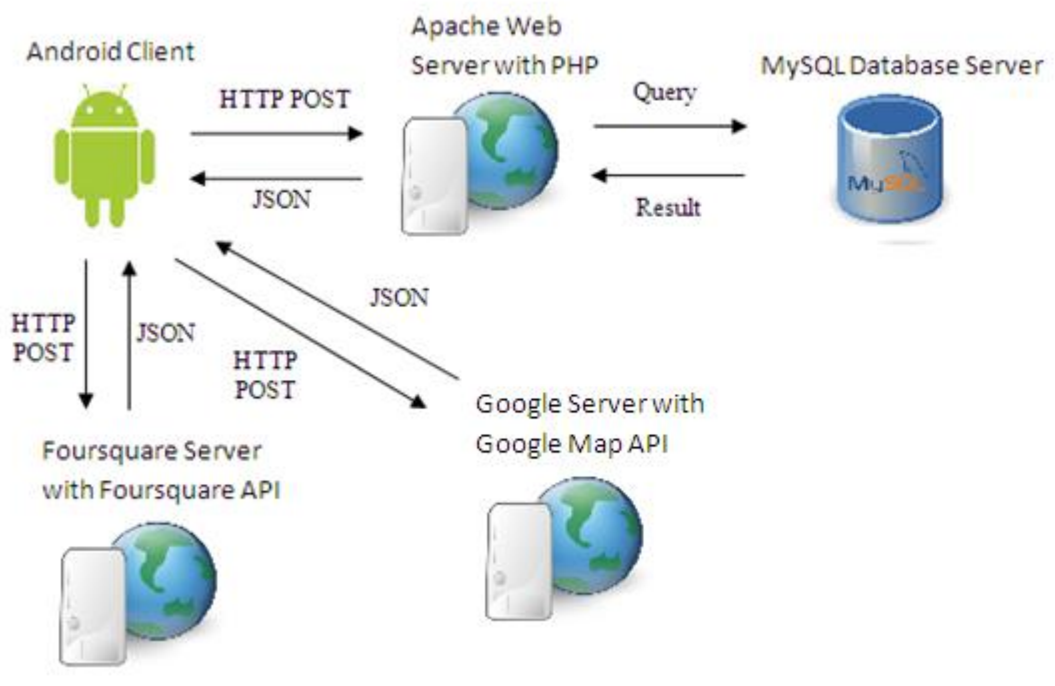

Android Client
Apache Web Server with PHP
MySQL Database Server
HTTP POST
JSON
Query
Result
HTTP POST
JSON
JSON
HTTP POST
Google Server with Google Map API
Foursquare Server with Foursquare API

Figure 3.2: System Architecture

As described in Figure 3.2, the mobile application acts as the client, users only interact with the mobile application. When a user performs a request for data in the database, the mobile application will request services from the Apache Web Server using HTTP POST. Then, the web services written using PHP in the web server will connect to the MySQL database and perform SQL operations. Results sent back from the MySQL database to the web server will be encoded in JSON format and then sent to the mobile application. Finally, the JSON format data will be decoded and displayed in the mobile application. Foursquare API is used to access Foursquare locations' data and Google Map API is used to display interactive and feature rich map in the mobile application.

3.4 Software Development Methodology

The software development methodology employed for this project is the agile development. Beck et al. (2001) states that agile development is a software development methodology that values:

i) Individuals and interactions over processes and tools
ii) Working software over comprehensive documentation
iii) Customer collaboration over contract negotiation
iv) Responding to change over following a plan

Agile development is selected because the mobile technology is changing fast. Agile development reduces development time. Changes to the requirements are accommodated easily because the software is delivered in small, incremental releases.

In this project, the functions of the application are divided into smaller part such as get nearby food subsystem, follower subsystem, commenting subsystem, like subsystem and add new food post subsystem. Each subsystem is developed and tested first before the next part start to be developed.

Therefore, the progress can be measured because of faster delivery of smaller working releases of the software. Moreover, testing is continuously throughout the development due to software is delivered in small, incremental releases. Therefore, bugs and problems can be easier to find.

3.5 Development Technologies and Tools

The development technologies and tools used for this project are:

3.5.1 Android SDK

The Android SDK is required to build applications for Android. API libraries and developer tools are provided by Android SDK to build, test and debugs applications for Android.

3.5.2 PHP

PHP is a widely used open source server-side scripting language. PHP is used for the server-side scripting language of this project because it is simple to use and provides many advanced features such as database support. Besides, it is compatible with most servers and platforms such as Apache, IIS, Windows, Linux, and Mac OS X. Other than that, PHP has large community support due to its open source. Tutorials, examples and questions for PHP can be easily found on the Internet. It is used in the server-side to communicate with the database.

3.5.3 Laravel PHP Framework

Laravel is a PHP framework that eases the writing of PHP code. Common task such as routing, authentication, and caching are made easier to write.

Laravel framework is used in this project for the server side API. Laravel helps to implement the Model View Controller easily. Routing using the API provided by Laravel framework is also easy. Besides, Laravel supports the Eloquent ORM (Object Relational Mapping) and a migration system. The migration system is used to create and update all database tables.

Another useful feature supported by Laravel is restful routing. Different type of HTTP request such as GET, POST, DELETE and PUT can be routed to the particular function by prepend[1] the name of the function with the request type such as getFunctionname or postFunctionname. For instance, A HTTP GET request to

[1] add something to the beginning of something else

www.foocious.com/api/comments will be routed to a function in the server controller named getComments.

The Eloquent ORM supported by Laravel is used a lot in this project. Every database table has a Model class that is used to map the database table. A lot of API is provided to easily access or update the database. Relationship between models can be defined in the models easily.

3.5.4 MySQL

DB-Engines Ranking (2013) states that MySQL is the world most widely used open source relational database management system. It is used for this project because it is free to use, provides a full-featured database management system and comply with SQL standards. It is used store the mobile application data such as user, food, eating outlets, reviews, comments, “likes” and “dislikes” so that the data can be accessed online by users in the mobile application via the server API.

3.5.5 Foursquare Venues API

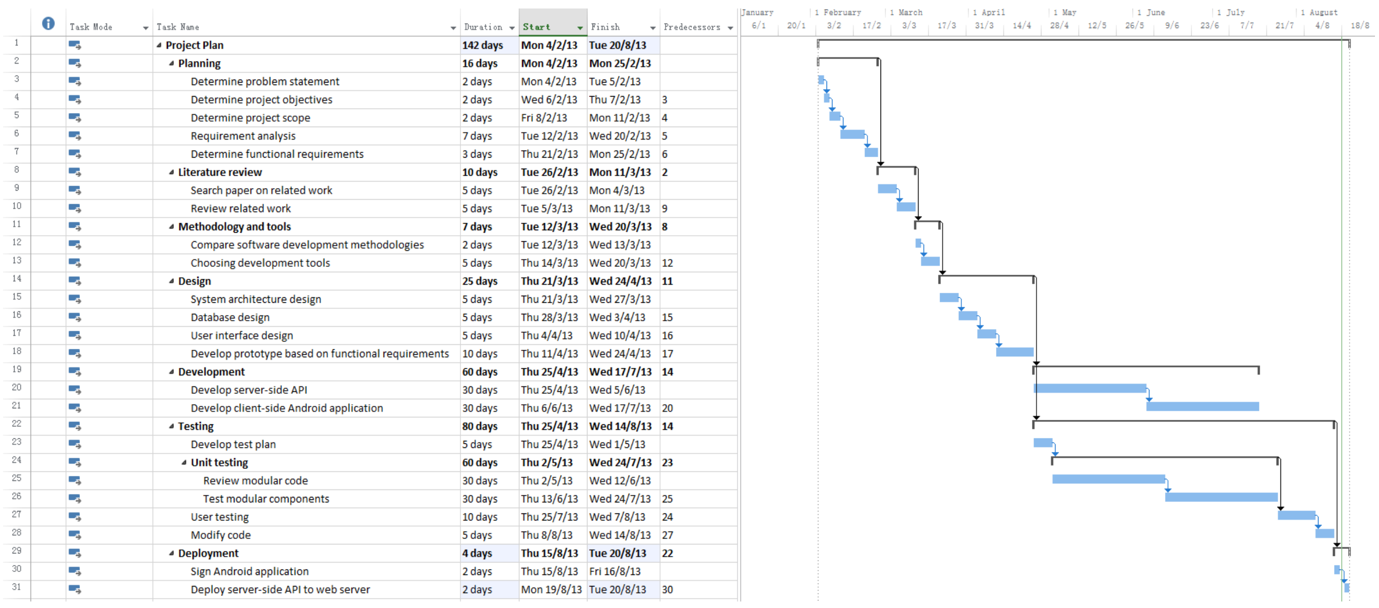
Foursquare API is a places data API that allows people to search for places and access a wealth of information about them, including addresses, popularity, tips, and photos (The foursquare Platform, n.d.). Not all users may like to add a place data for the application. But with Foursquare API, existing eating outlets data can be accessed so that the number of eating outlets information is big at early launch of the application. In this project, Foursquare API is used to obtain the existing eating outlets so that user can add food and food reviews to the eating outlets without needing to add new eating outlets to the mobile application on their own.

3.5.6 Google Maps Android API v2

Google Maps Android API v2 provides the API libraries to create interactive and feature rich maps in Mobile application. All the eating outlets within an area can be anchored to specific positions on the map using the API libraries provided. Google Maps Android API v2 is used to show the map of the eating outlets in the mobile application. Besides showing, user also can zoom and pan the map.

3.6 Project Planning

This project is planned and carried out according to schedule in Figure 3.3.



		Task Mode	Task Name	Duration	Start	Finish	Predecessors
1			**Project Plan**	**142 days**	**Mon 4/2/13**	**Tue 20/8/13**	
2			**Planning**	**16 days**	**Mon 4/2/13**	**Mon 25/2/13**	
3			Determine problem statement	2 days	Mon 4/2/13	Tue 5/2/13	
4			Determine project objectives	2 days	Wed 6/2/13	Thu 7/2/13	3
5			Determine project scope	2 days	Fri 8/2/13	Mon 11/2/13	4
6			Requirement analysis	7 days	Tue 12/2/13	Wed 20/2/13	5
7			Determine functional requirements	3 days	Thu 21/2/13	Mon 25/2/13	6
8			**Literature review**	**10 days**	**Tue 26/2/13**	**Mon 11/3/13**	**2**
9			Search paper on related work	5 days	Tue 26/2/13	Mon 4/3/13	
10			Review related work	5 days	Tue 5/3/13	Mon 11/3/13	9
11			**Methodology and tools**	**7 days**	**Tue 12/3/13**	**Wed 20/3/13**	**8**
12			Compare software development methodologies	2 days	Tue 12/3/13	Wed 13/3/13	
13			Choosing development tools	5 days	Thu 14/3/13	Wed 20/3/13	12
14			**Design**	**25 days**	**Thu 21/3/13**	**Wed 24/4/13**	**11**
15			System architecture design	5 days	Thu 21/3/13	Wed 27/3/13	
16			Database design	5 days	Thu 28/3/13	Wed 3/4/13	15
17			User interface design	5 days	Thu 4/4/13	Wed 10/4/13	16
18			Develop prototype based on functional requirements	10 days	Thu 11/4/13	Wed 24/4/13	17
19			**Development**	**60 days**	**Thu 25/4/13**	**Wed 17/7/13**	**14**
20			Develop server-side API	30 days	Thu 25/4/13	Wed 5/6/13	
21			Develop client-side Android application	30 days	Thu 6/6/13	Wed 17/7/13	20
22			**Testing**	**80 days**	**Thu 25/4/13**	**Wed 14/8/13**	**14**
23			Develop test plan	5 days	Thu 25/4/13	Wed 1/5/13	
24			**Unit testing**	**60 days**	**Thu 2/5/13**	**Wed 24/7/13**	23
25			Review modular code	30 days	Thu 2/5/13	Wed 12/6/13	
26			Test modular components	30 days	Thu 13/6/13	Wed 24/7/13	25
27			User testing	10 days	Thu 25/7/13	Wed 7/8/13	24
28			Modify code	5 days	Thu 8/8/13	Wed 14/8/13	27
29			**Deployment**	**4 days**	**Thu 15/8/13**	**Tue 20/8/13**	**22**
30			Sign Android application	2 days	Thu 15/8/13	Fri 16/8/13	
31			Deploy server-side API to web server	2 days	Mon 19/8/13	Tue 20/8/13	30

Figure 3.3: Gantt Chart

As shown in Figure 3.3, this project is categorized into the following phases:

i) Planning

In this phase, the problem statement, objectives and scope of this project were determined. Other than that, the functional requirements of the application were also determined in this phase. Use case diagram and description were created using the functional requirements determined earlier.

ii) Literature review

Journals, websites and book about related work were collected and put into a literature matrix in this phase. The findings of the literatures were then listed and critiqued.

iii) Methodologies and tools

Several software development methodologies were compared and the most suitable methodology for this project was chosen. The development tools such as the platform, database management system, and server-side programming language were also determined during this phase.

iv) Design, development and testing

System architecture design is carried out in the beginning of this phase. Then, the design, development and testing of each module is carried out iteratively. Database table design and user interface design were carried out for every module. Server-side API and client-side mobile applications are also developed module by module during this phase. Test plan is created for every module. After one module has been designed, developed and tested, the another module is designed, developed and tested. The test plan will include the unit testing and user testing. The modular code is reviewed and tested. After all the modules have been developed, the application is distributed to users for testing.

v) Deployment

After completion of the testing phase, the application is signed and uploaded on Google Play. The server-side API is deployed to the Apache web server.

3.7 Conclusion

Agile development is used for the software development methodology for faster delivery, easier testing and easier accommodation of changes to requirement. In this chapter, the development methodology, technology and tools used have been presented. In the following chapter, the design modelling process will be elaborated.

CHAPTER 4

DESIGN

4.1 Introduction

The design of database and interaction between entities in system will be discussed in this chapter. The interaction between entities in system will be illustrated with sequence diagrams and activity diagrams and the database design will be illustrated with entity relationship diagrams.

4.2 Database Design

Figure 4.1 shows the relationship between entities of this project. There are 12 entities represented as tables in the database. The entities are:

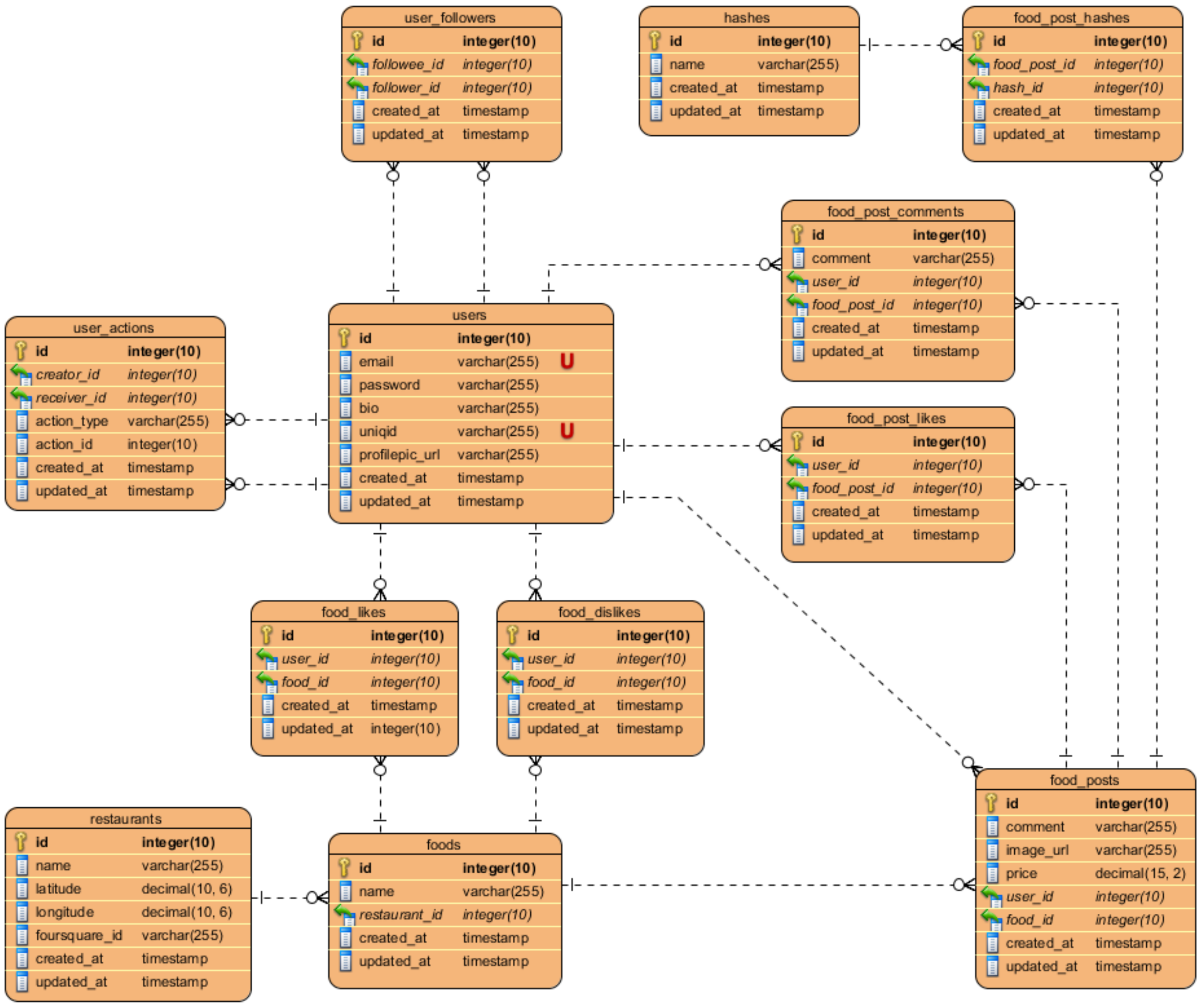


Figure 4.1: Entity Relationship Diagram

i) users

Store the information of the user such as email, password, bio, uniqid, profile picture URL and timestamp. The uniqid is used to authenticate the user when user try to edit his profile, add new food post, “like/dislike” food, “like” food post, follow or unfollow other users.

ii) user_actions

Store the actions of the user such as “like” food post, comment food post or follow other users. The receiver_id that is same with the user’s id means that the action is done on the user. For example, user A follows user B will create a row in user_actions table with action_type as followuser, creator_id as user A’s id and receiver_id as user B’s id. Then, this table is used to show the notification for user.

iii) user_followers

Store the following between users.

iv) hashes

Store the hashtag

v) food_post_hashes

Store the food posts with hashtag. hash_id is the id of the hashtag.

vi) food_post_comments

Store the comments of the food posts and the creator of the comments

vii) food_post_likes

Store the “likes” of the food posts and who “likes” the food posts.

viii) food_posts

Store the details of the food posts such as comment, food picture URL, price, creator id and the food id. food_id is used to know the food post belongs to which food.

ix) foods

Store the details of the food such as name and restaurant_id.

x) restaurants

Store the details of the restaurant such as name, latitude, longitude and the foursquare id. The latitude and longitude is used to know the location of the restaurant. The foursquare id is used to check whether the foursquare location has been added to this database.

xi) food_dislikes

Store the “dislikes” of the food and who “dislikes” the food.

xii) food_likes

Store the “likes” of the food and who “likes” the food.

4.3 Interaction Design

Interaction design involves the design of how entities in the system interact with each other. The interaction between the entities in the mobile application, server API, Foursquare venue API and Google Maps API were designed carefully to improve the performance, security and user satisfaction of the mobile application. Figure 4.2 shows the overview of the modules of the system and how the system interacts with each other. The system is divided into smaller modules as shown in Figure 4.2. Integration of Android native application, web server API and database server are required to make the system functional. Module Add Food Post use the Foursquare API to get the nearby eating outlets by sending HTTP request. Details of the interaction will be discussed later.

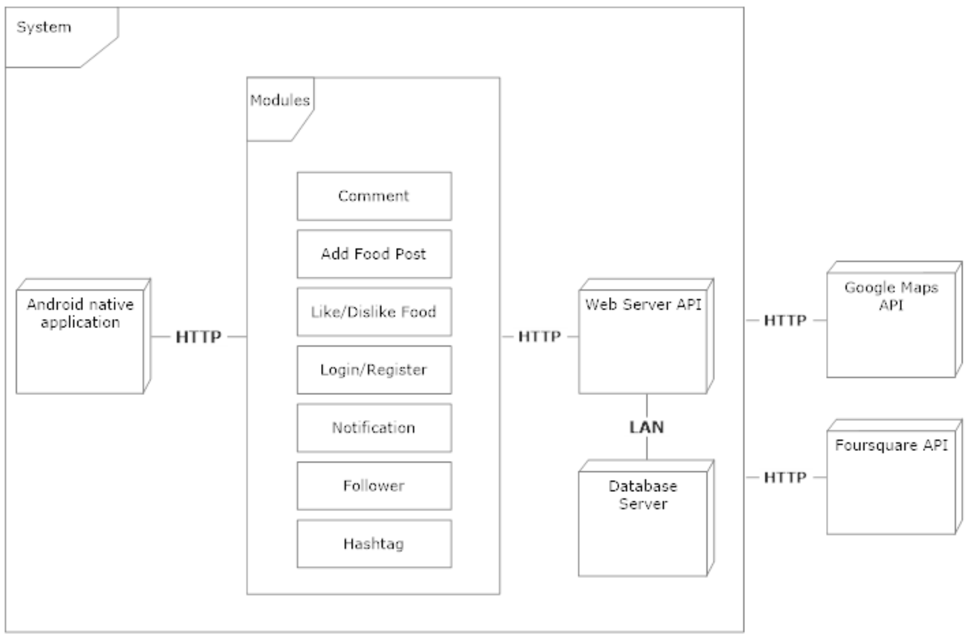


Figure 4.2: Physical Architecture of the System

4.3.1 Add New Food Post

Users can add food post by taking picture using mobile devices' camera or choose picture from gallery and then fill in the information such as restaurant name, food name, comment and price. Figure 4.3 shows the activity diagram of adding food post.

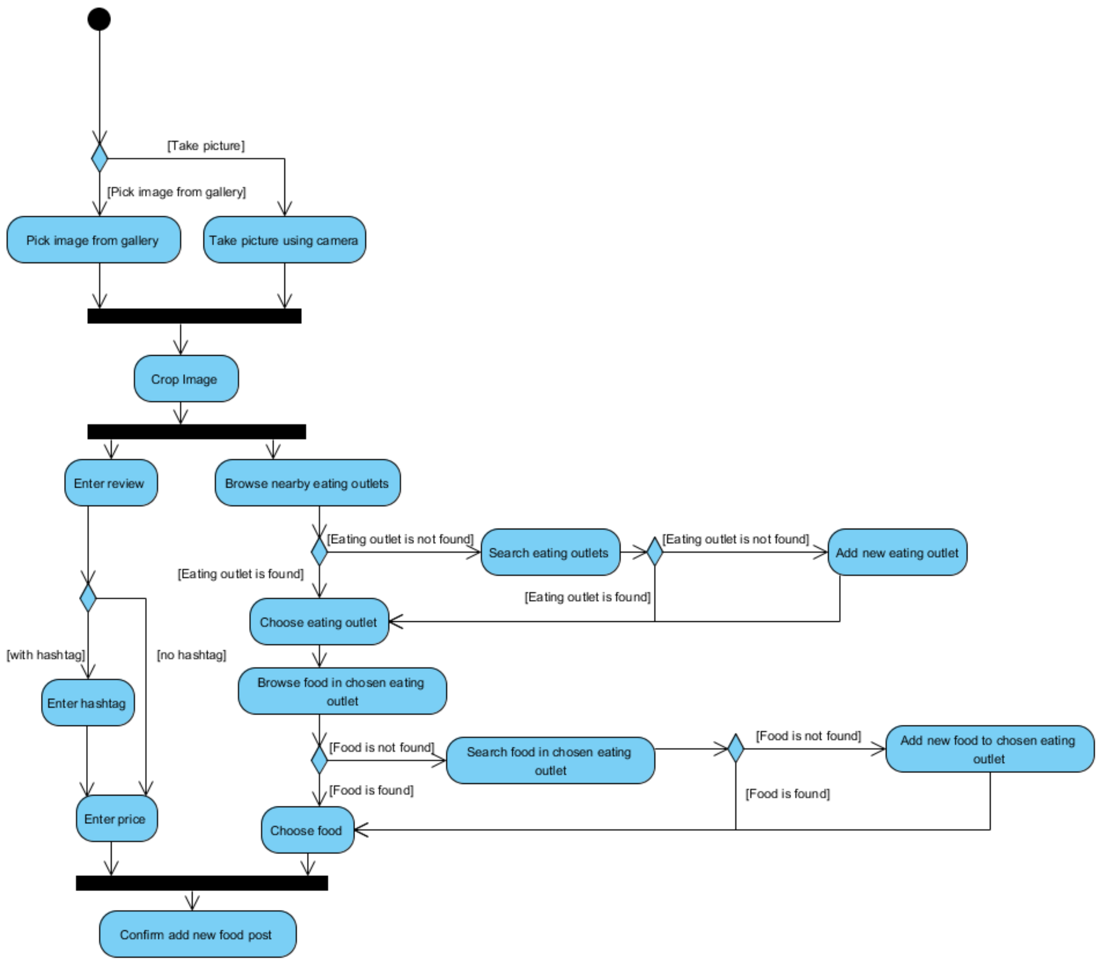


Figure 4.3: Add New Food Post Activity Diagram

User has the choice of taking picture using camera or picking image from a gallery. Then the picture will be cropped by user into square shape. After that, user needs to enter the review, price, eating outlets and food. Finally, user needs to confirm the adding of the food post.

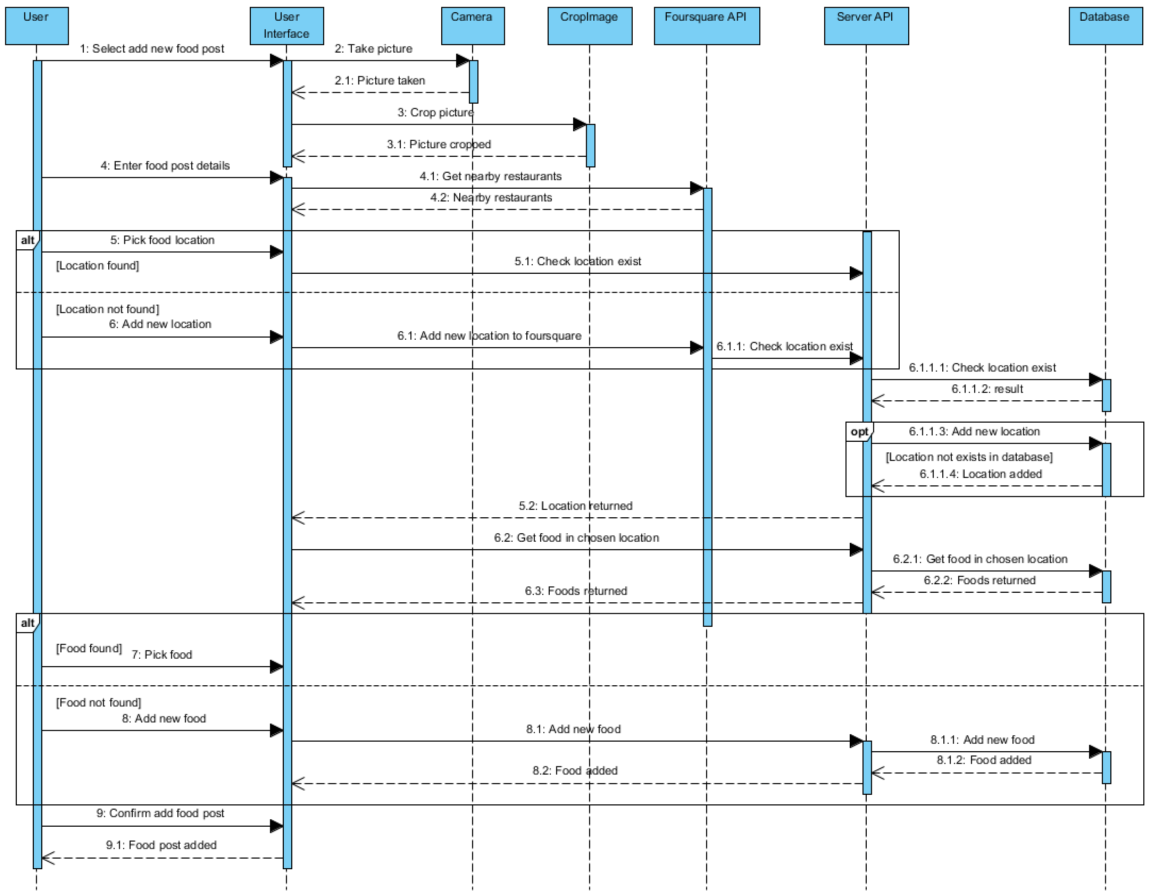


Figure 4.4: Add New Food Post Sequence Diagram

Figure 4.4 shows how the entities in the systems interact with each other when user adds a new food post:

1. User takes a picture.
2. The picture is cropped.
3. The user enters food post details
4. The application gets the nearby restaurants from Foursquare API
5. If the user finds the restaurant he wants, the user will pick the restaurant. Else, the user adds a new restaurant to the Foursquare API.
6. Then, the chosen restaurant is checked whether it is already exists in the database. If it does not exist, the new restaurant is added to the database.
7. After the restaurant is chosen, the application gets all foods in the chosen restaurant.
8. If the user finds the food name he wants, the user will pick the food. Else, the user adds new food to the chosen restaurant.

9. The user confirms the “add new food” post.
10. Food post is added to the database.

4.3.2 Add Hashtag to Food Post

A hashtag is a word or a phrase prefixed with the symbol #. It can be used to group similar topic. A user can search for hashtags and look for the group of topics. For example, a user can include the hashtag #seafood in the food post so that other can search for the #seafood and get all the food post including the hashtag #seafood.

Regular expression is used to extract all hashtags from a food post and store it to the database. The regular expression format used to extract the hashtags is:

```
/#(\w+)/u
```

When user submit a food post, the comment for the food post will be passed into the regular expression matcher to search for the hashtag on the server side. The food_post_hashes database table is used to store the food post that contains the hashtag. If hashtag is found in the comment, the food post is stored in the food_post_hashes table with the hashtag. Figure 4.5 shows the program flow of adding hashtag to food post. The square area of the diagram shows that user can enter hashtag to the food post optionally.

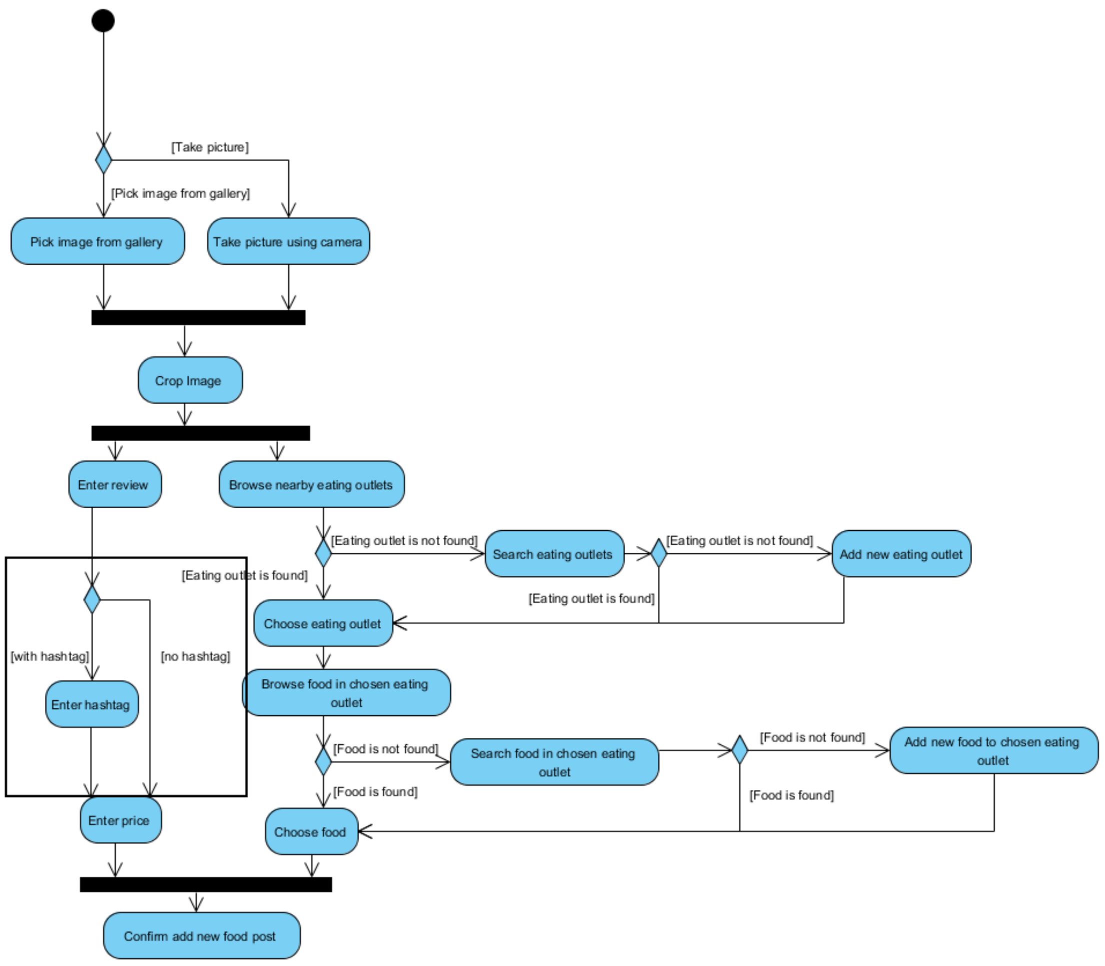


Figure 4.5: Add Hashtag to Food Post Activity Diagram

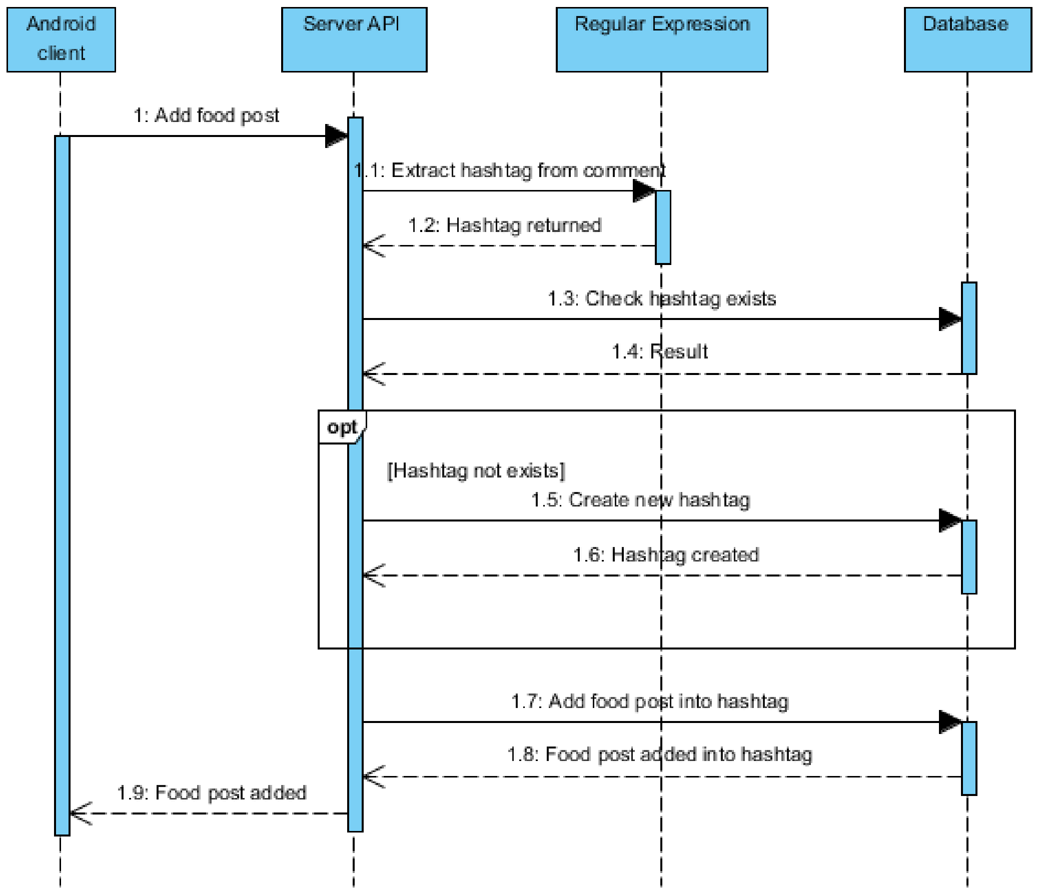


Figure 4.6: Add Hashtag to Food Post Sequence Diagram

Figure 4.6 shows the interaction model of “add hashtag to food post” interaction.

1. When the food post is added, the Server API extracts the hashtag from comment.
2. Then, the extracted hashtag is checked whether it exists in the database.
3. If it does not exist, the application creates a new hashtag in the database.
4. Then, a new row is added to the food_post_hashes table with food_post_id and hash_id as the id of the food post and hashtag respectively.

4.3.3 “Like” Food

User can like food. The number of “likes” of the food will be shown so that other users can know how many users “like” the food. A user just needs to press the “like” button below the food picture in order to “like” a food as shown in Figure 4.7.

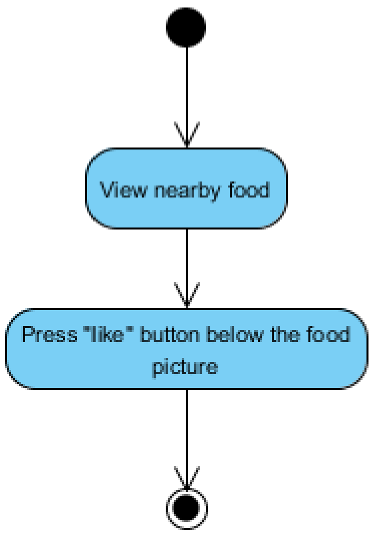


Figure 4.7: “Like” Food Activity Diagram

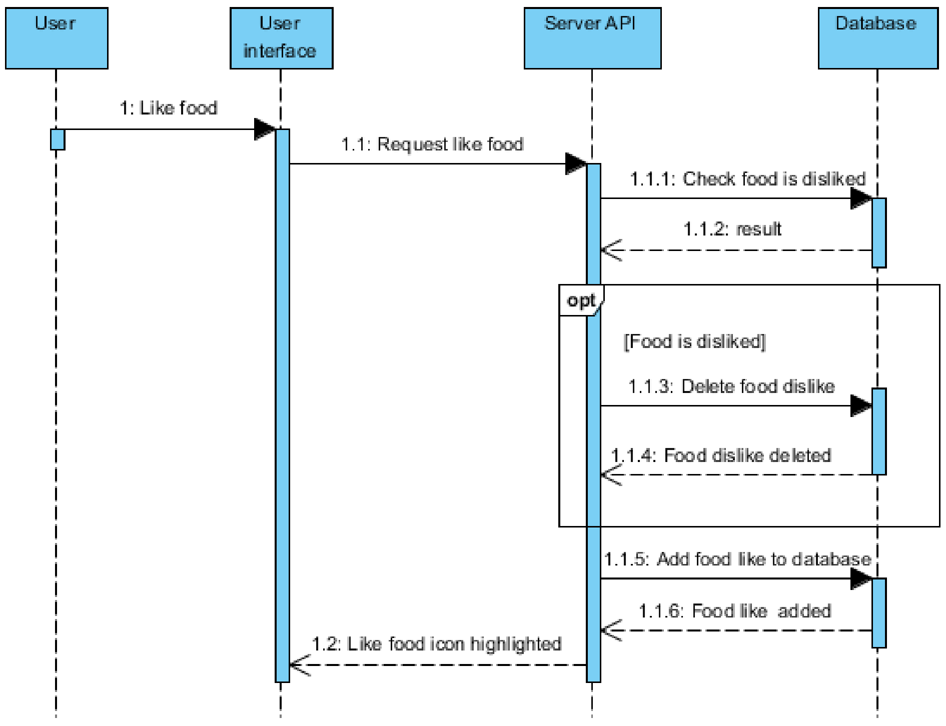

User
User interface
Server API
Database
1: Like food
1.1: Request like food
1.1.1: Check food is disliked
1.1.2: result
opt
[Food is disliked]
1.1.3: Delete food dislike
1.1.4: Food dislike deleted
1.1.5: Add food like to database
1.1.6: Food like added
1.2: Like food icon highlighted

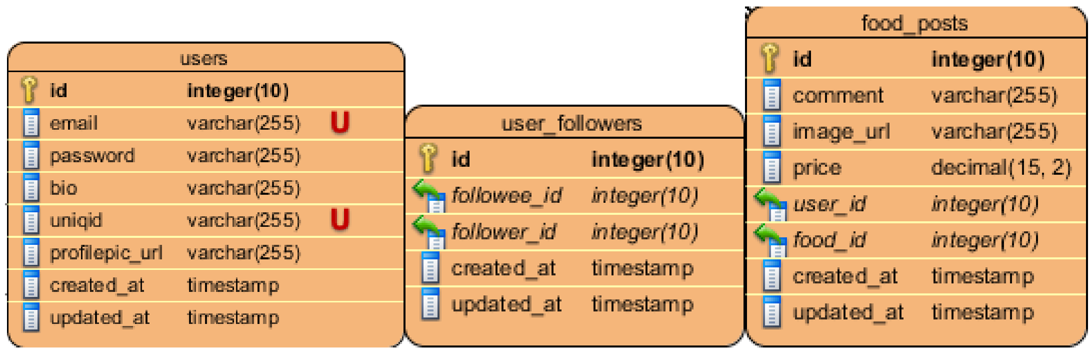
Figure 4.8: “Like” Food Sequence Diagram

Figure 4.8 shows the interaction model of user “likes” a food such as follows:

1. When a user likes a food, the mobile application calls the “like food request” in Server API.
2. The Server API checks whether the food is “disliked” by the user. If the food is disliked by the user, the food will be “undisliked”.
3. Then, the food “like” is added to database.

4.3.4 Follow User

A user can follow other users. After the follow relationship is established, any new food reviews posted by the followed user will be shown in a user’s following page. Three database tables as shown in Figure 4.9 are needed to get the food posts of the followed users. The user_followers table is used to store the following between users. Join between users and user_followers table can get all followees of a user. Then, join between the followees and the food_posts table can get food posts of the followee.



users		
id	integer(10)	
email	varchar(255)	U
password	varchar(255)	
bio	varchar(255)	
uniqid	varchar(255)	U
profilepic_url	varchar(255)	
created_at	timestamp	
updated_at	timestamp	

user_followers	
id	integer(10)
followee_id	*integer(10)*
follower_id	*integer(10)*
created_at	timestamp
updated_at	timestamp

food_posts	
id	integer(10)
comment	varchar(255)
image_url	varchar(255)
price	decimal(15, 2)
user_id	*integer(10)*
food_id	*integer(10)*
created_at	timestamp
updated_at	timestamp

Figure 4.9: Database Tables for Follower System

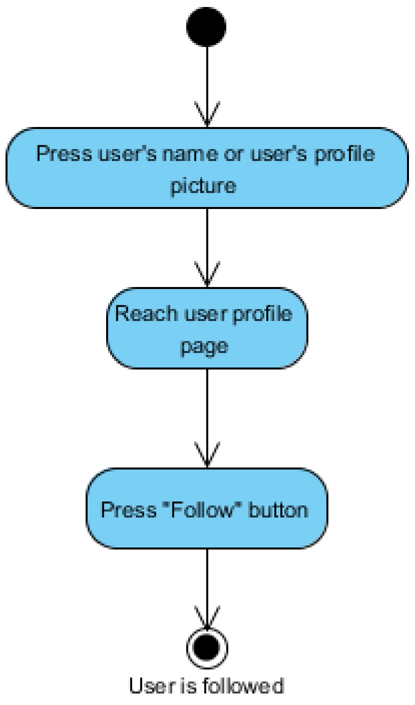


Figure 4.10: Follow User Activity Diagram

Figure 4.10 shows program flow of “follow user”. User can follow other users by going to their user profile page and presses the “Follow” button.

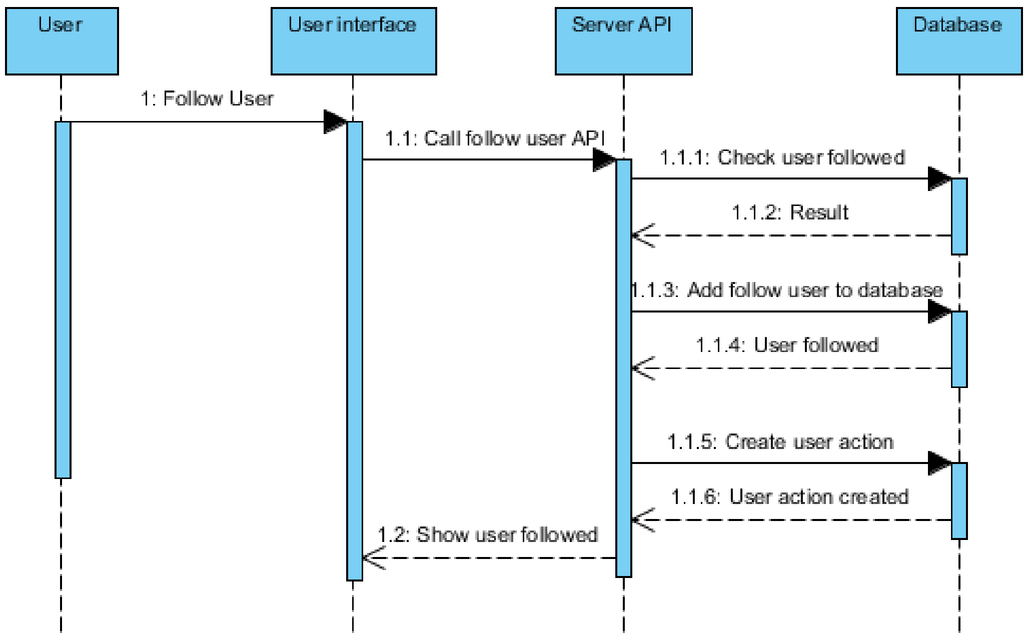

User
User interface
Server API
Database
1: Follow User
1.1: Call follow user API
1.1.1: Check user followed
1.1.2: Result
1.1.3: Add follow user to database
1.1.4: User followed
1.1.5: Create user action
1.1.6: User action created
1.2: Show user followed

Figure 4.11: Follow User Sequence Diagram

Figure 4.11 shows the interaction model of “user follows another user” interaction.

1. When user follows another user, the mobile application calls the follow user API in Server API.
2. The Server API checks whether the users is already followed.
3. If the user is not followed yet, it will be added to the database.
4. Then, a “follow user action” tag will be created in the database to denote that the user has followed another user.

4.3.5 Show nearby Food

User location and the search radius are needed to get nearby food. Haversine Formula is used to calculate the distance between two longitude-latitude points on the Earth. It is used to calculate the distance between the food and the user. It is recommended to calculate short distances (U.S. Census Bureau, 2001). For two points with latitudes latitude1 and latitude2 and longitudes longitude1 and longitude2 on the Earth, the distance between two points can be calculated as follows:

$$longitude_{distance} = longitude2 - longitude1$$
$$latitude_{distance} = longitude2 - latitude1$$
$$a = \left(\sin\left(\frac{latitude_{distance}}{2}\right)\right)^2 + \cos(latitude1) * \cos(longitude2) * \left(\sin\left(\frac{longitude_{distance}}{2}\right)\right)^2$$
$$c = 2 * atan2(sqrt(a), sqrt(1 - a))$$
$$distance = R * c$$

where R = radius of the Earth, the radius of the Earth is 6366 kilometres or 3956 miles.

However, the formula uses more processing power than normal SQL statement. Calculation of distance of all restaurants in the database and the user can take a long time. Therefore, only the food that is located inside the square area is calculated using the formula to improve the query time as shown in Figure 4.12. As shown in Figure 4.12, the restaurants which are represented as red and yellow colour circles within the square area around the user are filtered. Then, only those filtered restaurants are used to calculate the distance between the user and restaurants. Finally, food in the restaurants with distance less than the search radius which is marked with red colour will be the user's nearby food.

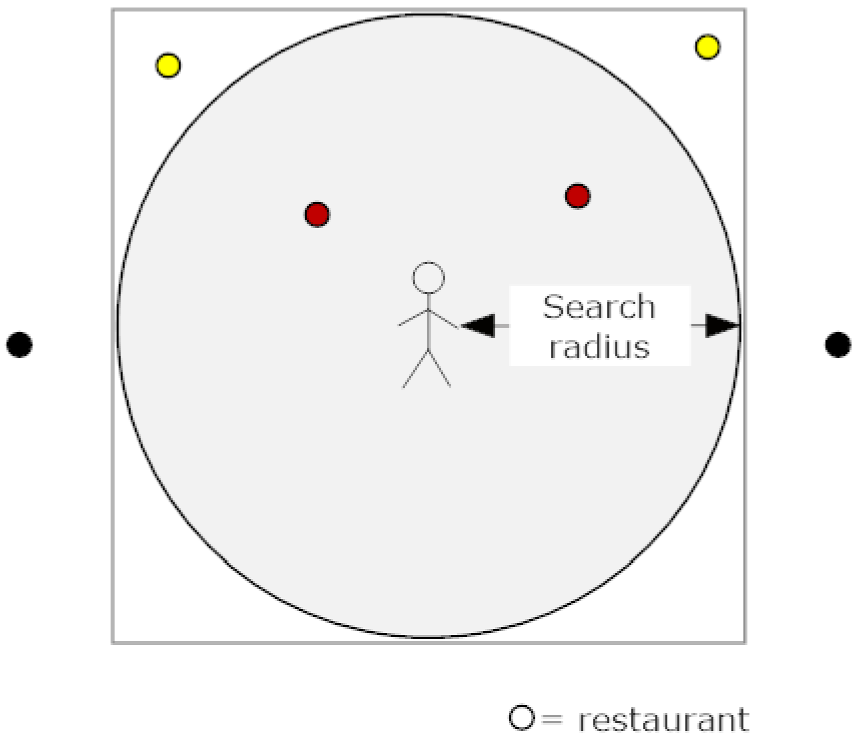


Figure 4.12: Get Nearby Restaurants

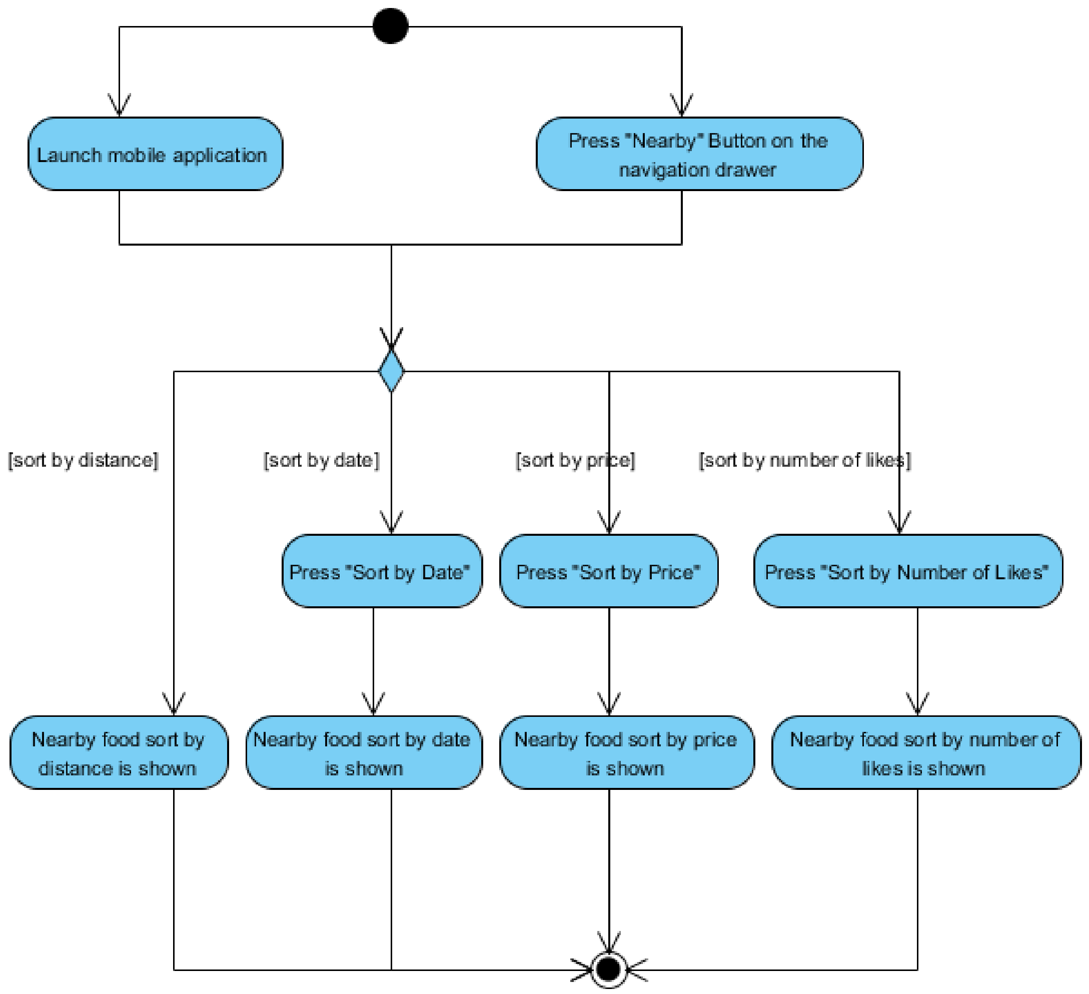

Launch mobile application
Press "Nearby" Button on the navigation drawer
[sort by distance]
[sort by date]
[sort by price]
[sort by number of likes]
Press "Sort by Date"
Press "Sort by Price"
Press "Sort by Number of Likes"
Nearby food sort by distance is shown
Nearby food sort by date is shown
Nearby food sort by price is shown
Nearby food sort by number of likes is shown

Figure 4.13: Show Nearby Food Activity Diagram

Figure 4.13 shows the program flow of “show nearby food” interaction. Nearby food will be shown immediately when the mobile application is launched. If a user has gone to other page, the user can go to “nearby food” page by pressing the “Nearby” button on the navigation drawer. Nearby food will be sorted by the food post created date by default. User can choose the sorting type by pressing the “Sort by Date”, “Sort by Price” or “Sort by Number of Likes” buttons to sort the nearby food.

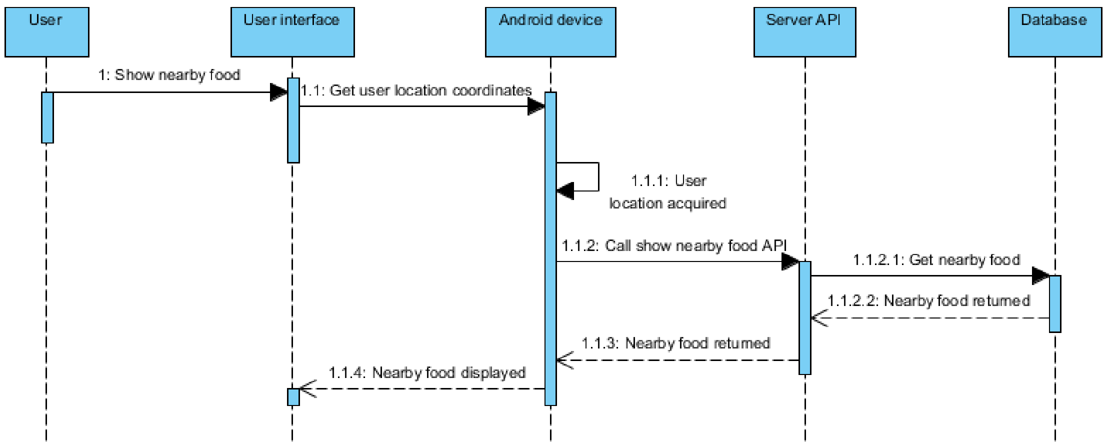

User
User interface
Android device
Server API
Database
1: Show nearby food
1.1: Get user location coordinates
1.1.1: User location acquired
1.1.2: Call show nearby food API
1.1.2.1: Get nearby food
1.1.2.2: Nearby food returned
1.1.3: Nearby food returned
1.1.4: Nearby food displayed

Figure 4.14: "Show Nearby Food" Sequence diagram

Figure 4.14 shows the interaction model of "showing nearby food".

1. When a user wants to show nearby food, the mobile application will get the user's location coordinates from the mobile device.
2. After the location is acquired, the mobile application calls the "show nearby food" API in the Server API.
3. Then the Server API returns the nearby food to the mobile application.
4. Then, nearby food are displayed on the mobile application.

4.4 Conclusion

In the prior sections of this report, data and interaction models of the system have been elaborated. Entity relationship diagrams are used to model the data model. Sequence diagrams are used to show the interaction of the entities of the system and activity diagrams are used show the flow of the mobile application. In the following chapter, the testing results of the mobile application will be presented and discussed in details.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Introduction

In this chapter, the testing results of the system runtime will be shown. Features of the mobile application will be explained by the screenshots of the mobile application. Besides, user acceptance testing were also carried out.

5.2 Show Nearby Food

Figure 5.1 shows the user nearby food. User can “likes” or “dislikes” the food by pressing the “like” or “dislike” icon. User can sort the nearby food by pressing the “Sort” button on upper right of the “Nearby” page as shown on Figure 5.2. By giving the option to user to sort the nearby food, user can look for the most popular, most recent, nearest or the cheapest food around them.

Besides, pressing the food picture or food name bring a user to the food detail shown in Figure 5.3. Pressing the restaurant name shows the user the restaurant detail as shown in Figure 5.4.

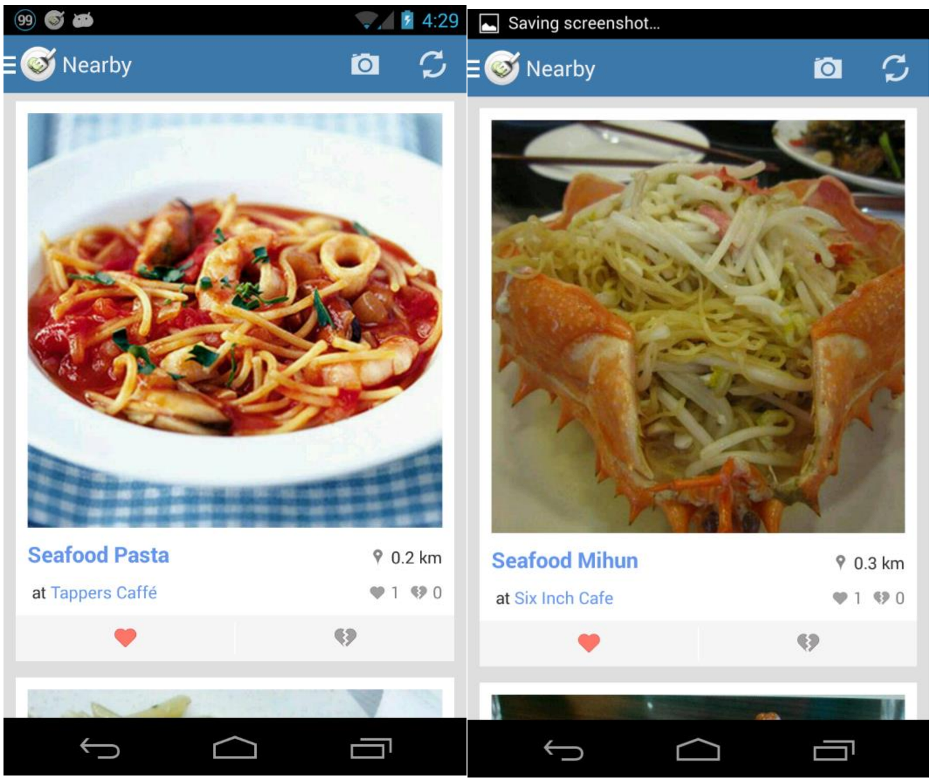


Figure 5.1: Nearby Foods Page

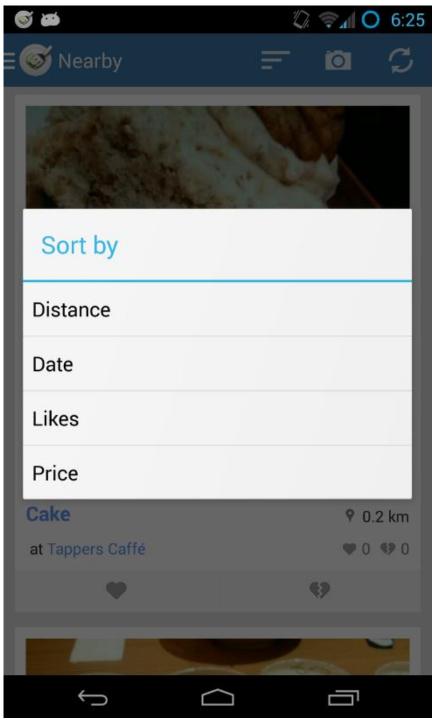


Figure 5.2: Sort Nearby Food

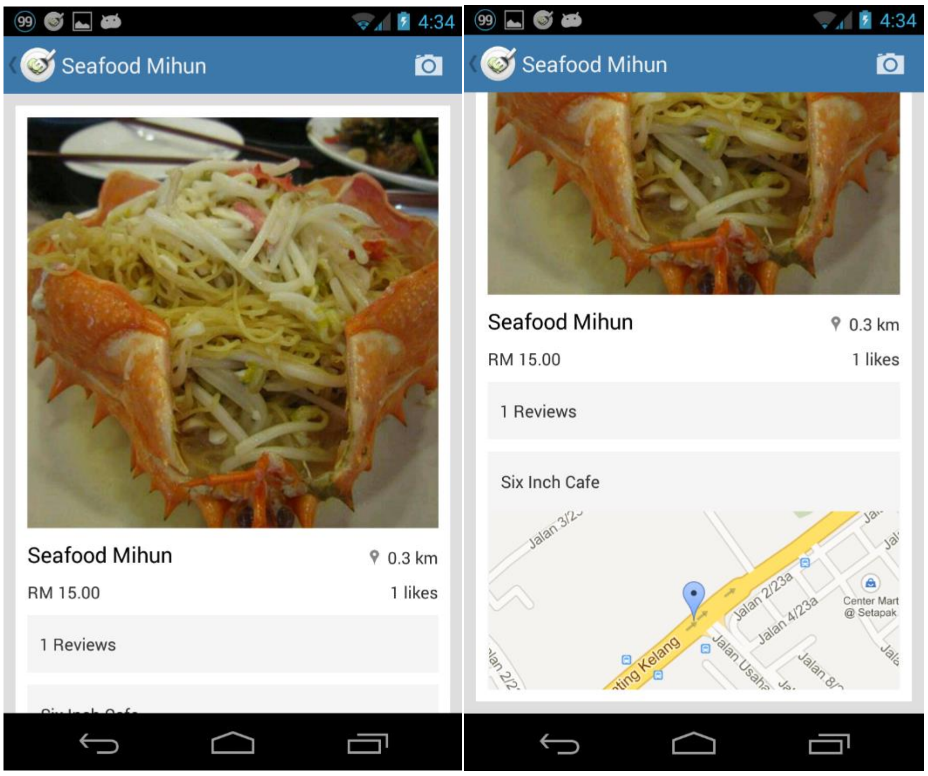


Figure 5.3: Food Details Page

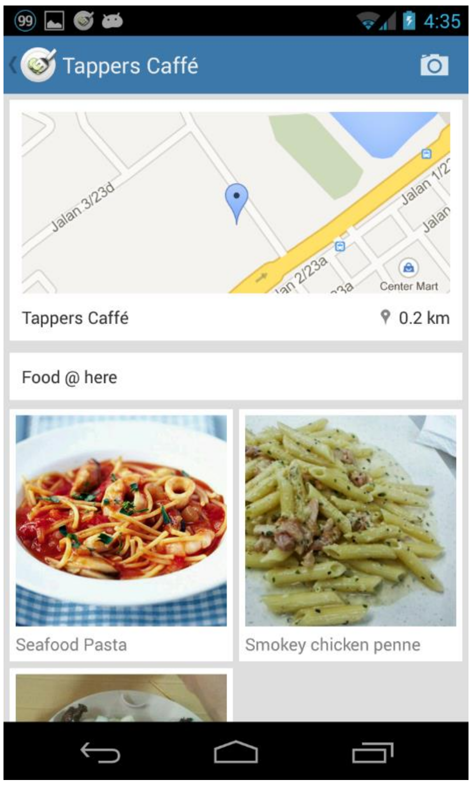


Figure 5.4: Restaurant Details Page

In the food detail page, a restaurant's location map and the number of reviews are shown. In the restaurant detail page, the restaurant's location map and all the food in that restaurant are shown. Pressing both the restaurant's location map in the food detail and restaurant detail page would link the user to an interactive map as shown in Figure 5.5. User can zoom and pan the interactive map. Pressing the restaurant name on the interactive map lets the user choose the map application to navigate to the restaurant as shown in Figure 5.5.

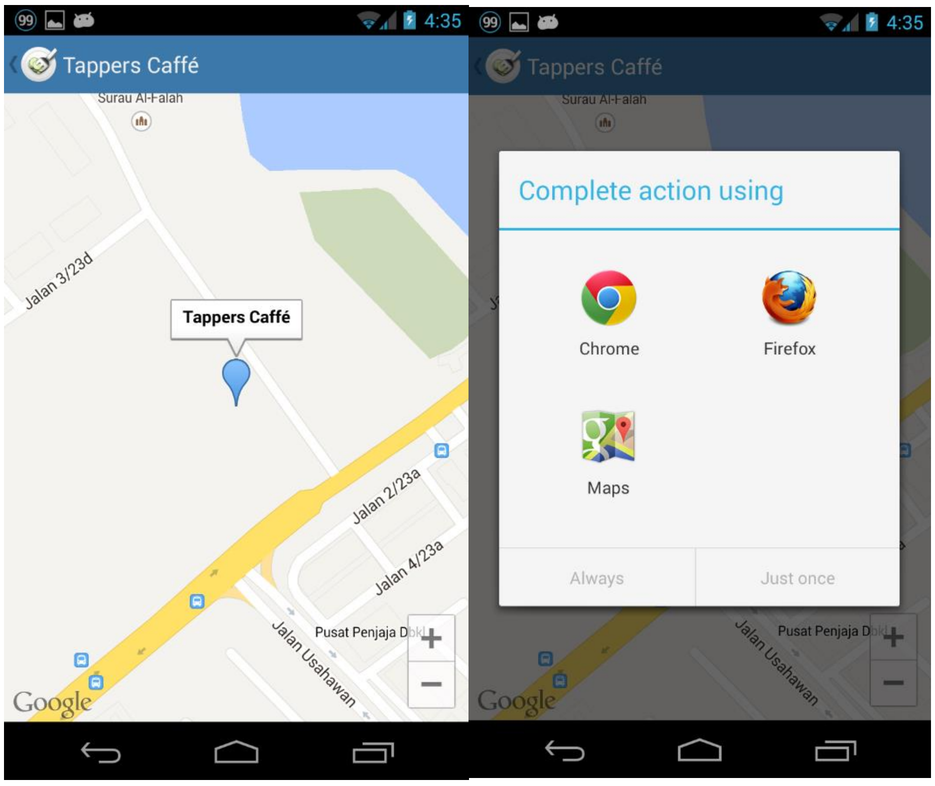


Figure 5.5: Restaurant Interactive Map

5.3 Add New Food Post

User can add new food post by pressing the camera icon on the top right of the application. Pictures can be taken by camera or selected in the gallery as shown in Figure 5.6. After the picture is ready, the user needs to enter comment, price, pick location and food name. User can pick from a list of locations that are acquired from Foursquare API. If the location is not in the list, the user can search the location using Foursquare API. If the location is not in Foursquare, the user can add a new restaurant as shown in Figure 5.7. After the user picks a location, food name needs to be picked. The user can pick the food name from a list of foods in the chosen location or add new food to the chosen location as shown in Figure 5.8.

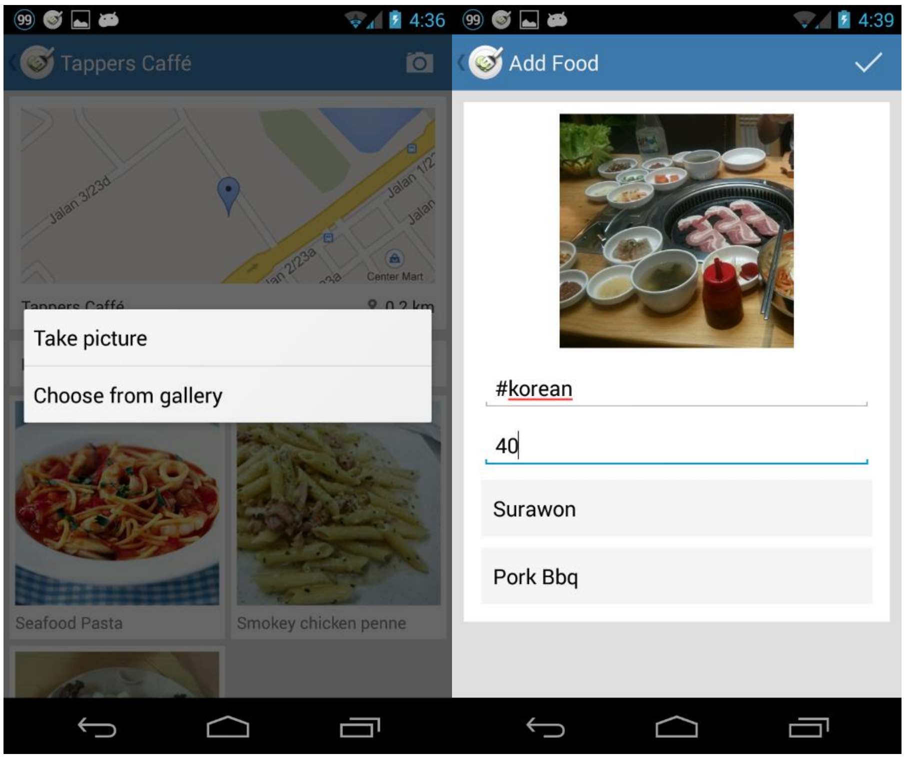
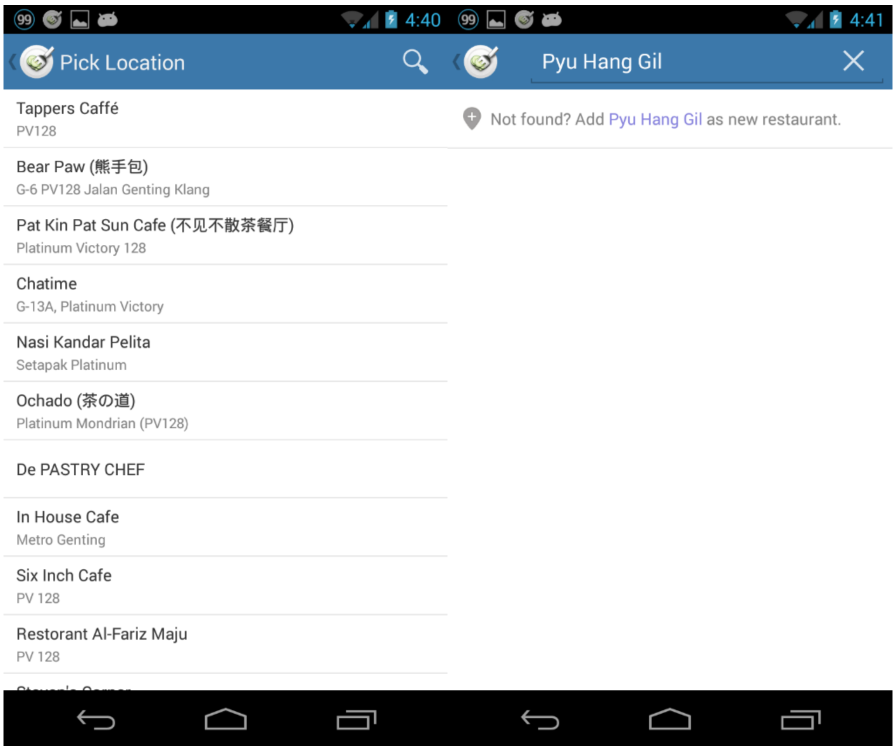
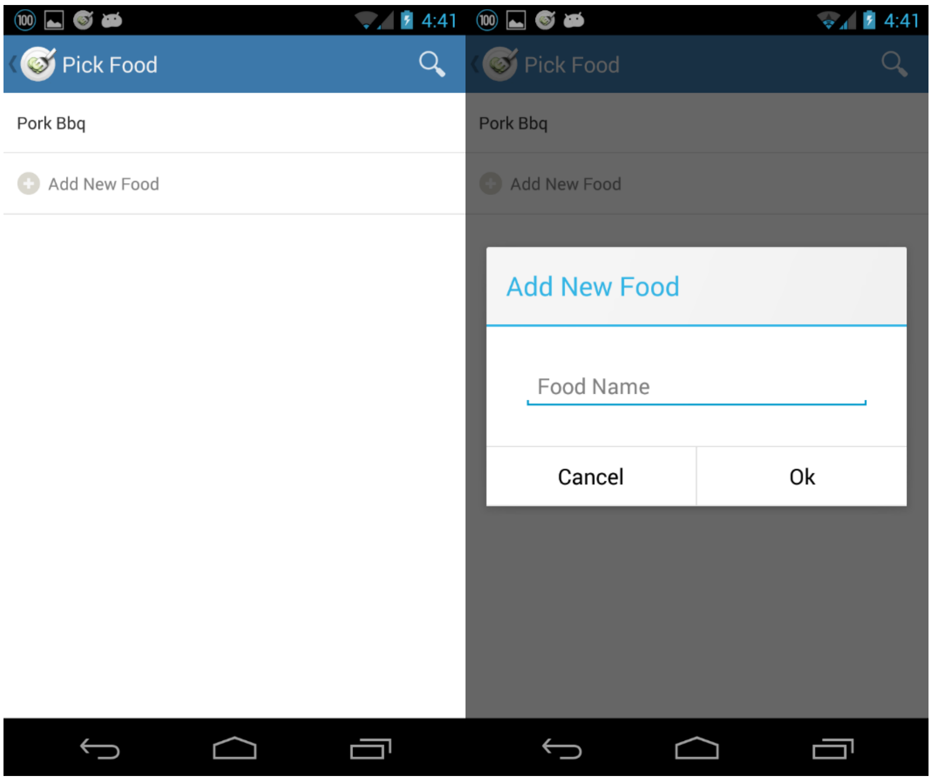


Figure 5.6: Add New Food Post


4:40
Pick Location
Tappers Caffé
PV128
Bear Paw (熊手包)
G-6 PV128 Jalan Genting Klang
Pat Kin Pat Sun Cafe (不见不散茶餐厅)
Platinum Victory 128
Chatime
G-13A, Platinum Victory
Nasi Kandar Pelita
Setapak Platinum
Ochado (茶の道)
Platinum Mondrian (PV128)
De PASTRY CHEF
In House Cafe
Metro Genting
Six Inch Cafe
PV 128
Restorant Al-Fariz Maju
PV 128

4:41
Pyu Hang Gil
Not found? Add Pyu Hang Gil as new restaurant.

Figure 5.7: Pick and Search Location


4:41
Pick Food
Pork Bbq
Add New Food

4:41
Pick Food
Pork Bbq
Add New Food
Add New Food
Food Name
Cancel
Ok

Figure 5.8: Pick or Add Food

5.4 Hashtag

On the client side, when showing of the food posts, the client side also searches for the hashtag. If hashtag is found, the hashtag is made into a link that links to another page that shows all food posts containing the hashtag as shown in Figure 5.9.

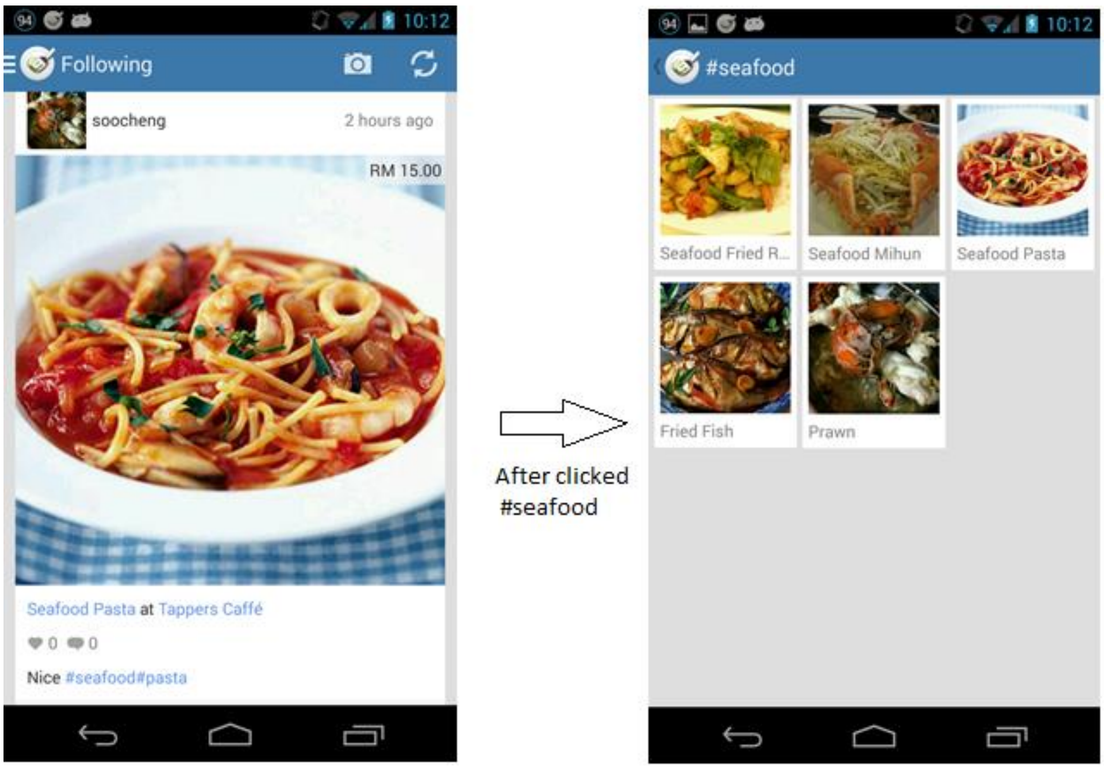


Figure 5.9: Food Post with Hashtag

5.5 Following Page

Figure 5.10 shows the users that a user is following. Their food posts are shown on the current user's following page as shown in Figure 5.11.

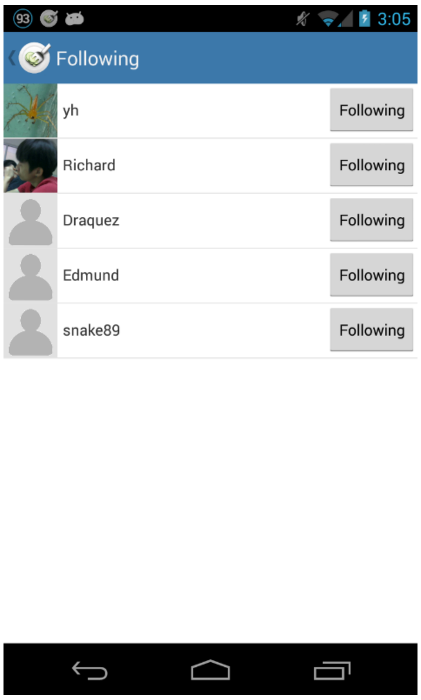


Figure 5.10: My Following Users

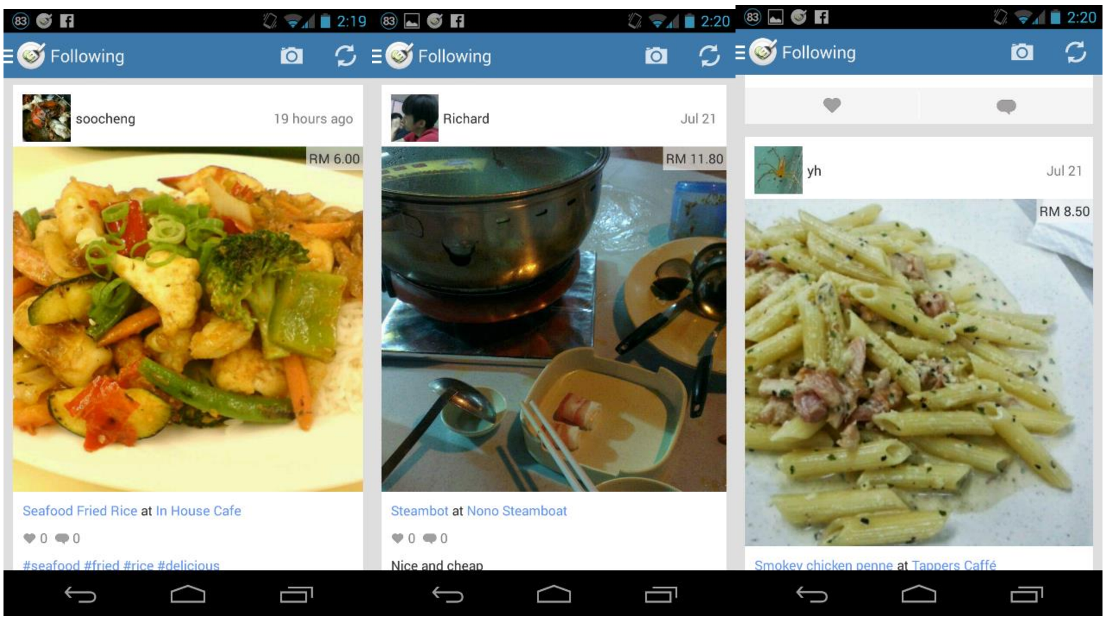

Following
soocheng 19 hours ago
RM 6.00
Seafood Fried Rice at In House Cafe
#seafood #fried #rice #delicious
Richard Jul 21
RM 11.80
Steambot at Nono Steamboat
Nice and cheap
yh Jul 21
RM 8.50
Smokey chicken penne at Tappers Caffé

Figure 5.11: Following Page

5.6 User Profile Page

“User Profile” page lets users to view their details such as profile picture, username, biography, number of following, number of followers such as shown on Figure 5.12.

Besides, users also can edit their profiles. When a user slides to the right, the user’s previous food posts will be shown such as Figure 5.13. It allows the user to view his food posts. A user is also allowed to change his profile picture, biography and email address in the “Edit Profile” page. Figure 5.14 shows the “Edit Profile” page.

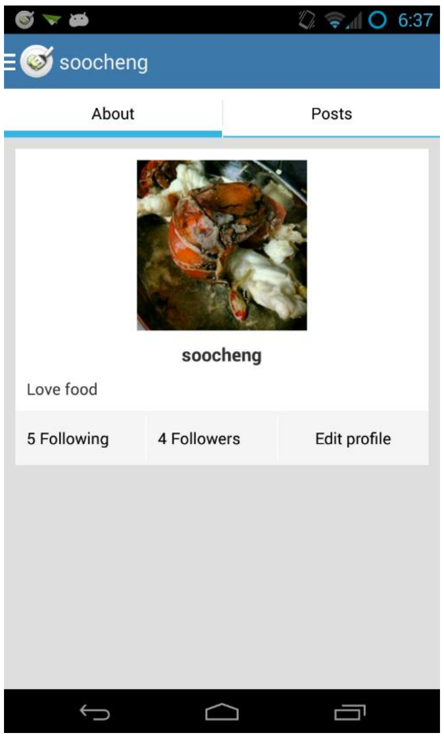


Figure 5.12: “User Profile” page

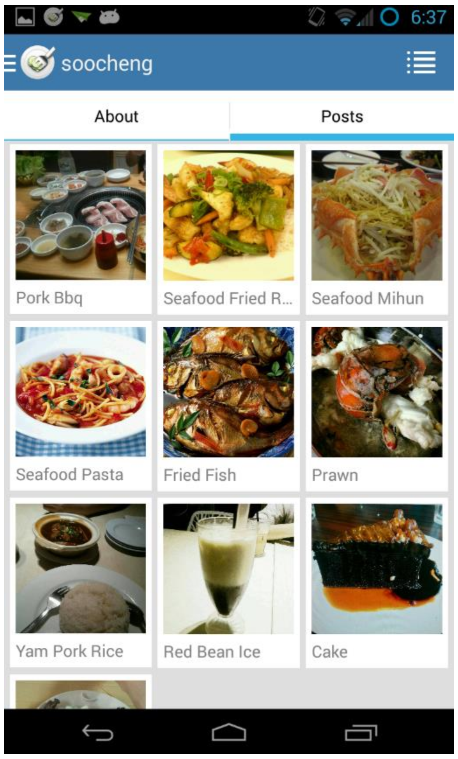
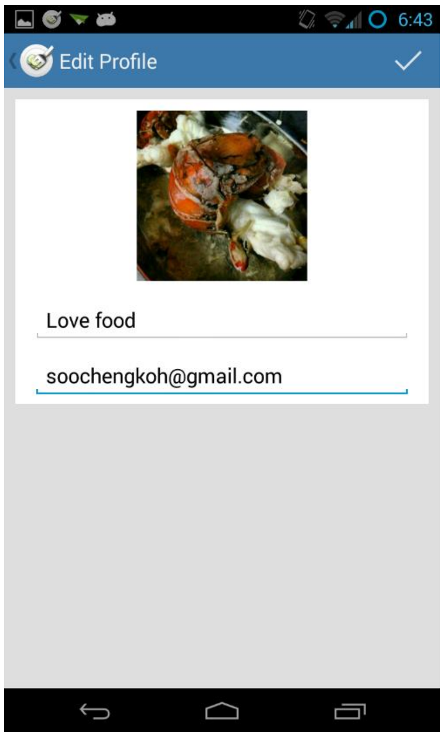


Figure 5.13: User's Food Posts



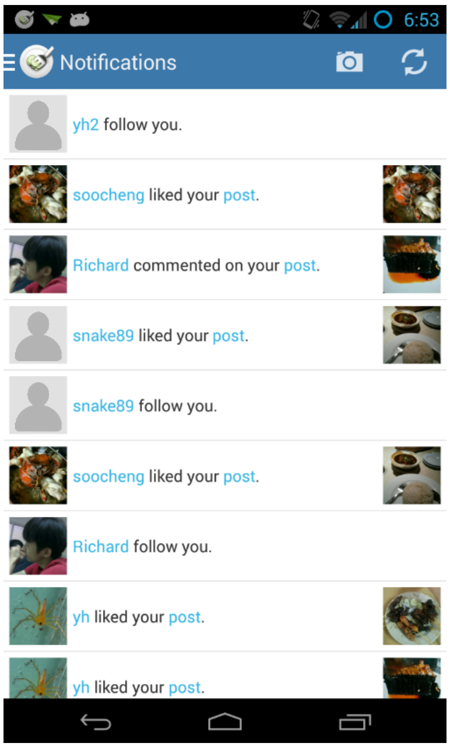
Love food

soochengkoh@gmail.com

Figure 5.14: "Edit Profile" page

5.7 Notification

"Notification" page shows the actions performed on the user such as the users who are following him, liked or commented on his posts. The user_actions table in database is used to show the notifications for the user. The actions that have receiver_id equal to the user's user_id are the actions performed on the user. Figure 5.15 shows the "Notifications" page. Pressing the username or the user's profile picture will link to the user's profile page. Pressing the food's picture will link to the food post.



yh2 follow you.
soocheng liked your post.
Richard commented on your post.
snake89 liked your post.
snake89 follow you.
soocheng liked your post.
Richard follow you.
yh liked your post.
yh liked your post.

Figure 5.15: "Notification" page

5.8 Search

Search function is available in the mobile application. It allows a user to search for either user, hashtag, restaurant or food easily. Users can search for their friends or families by using the search user function as shown on Figure 5.16. By searching hashtag, user can find group(s) of similar food. For example, user can search #seafood to look for food posts that are related to seafood as shown on Figure 5.17.

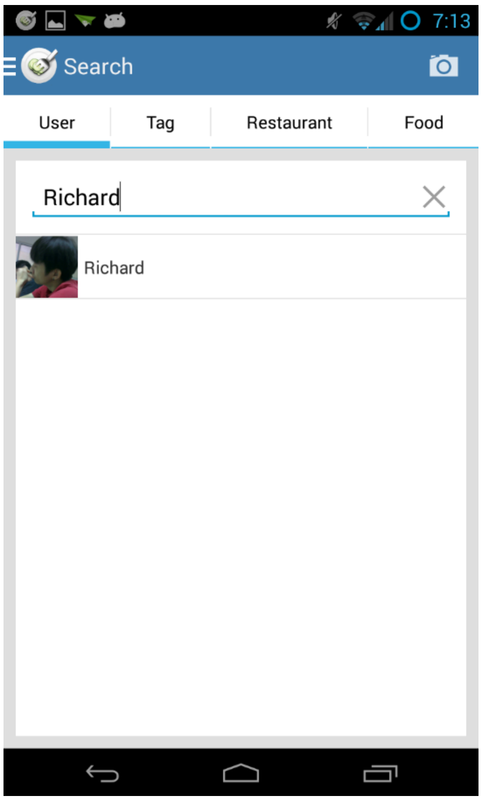


Figure 5.16: Search User

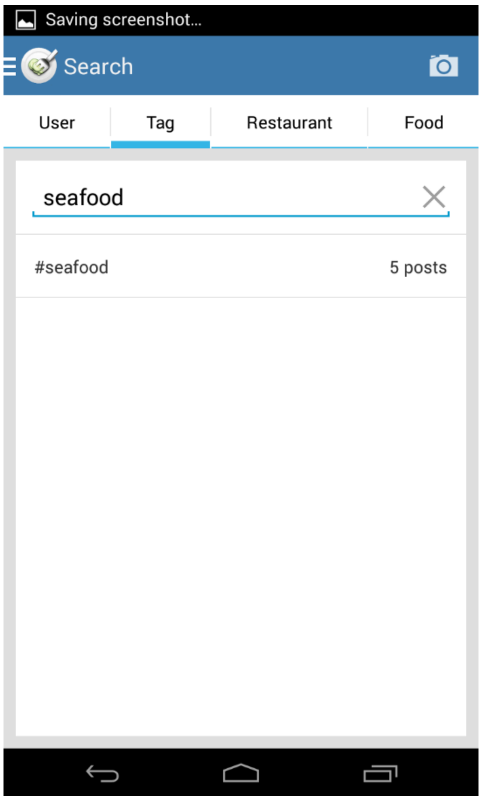
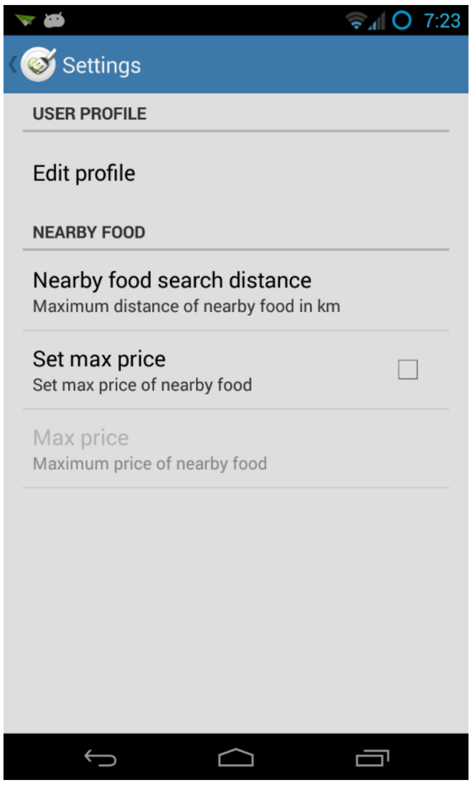


Figure 5.17: Search Tag

5.9 Settings

A user is allowed to control the maximum distance that is considered for determining the nearby food in the “Setting” page. When a user is walking, he can set the maximum distance to a shorter distance. When a user is driving, he may want to set the maximum distance to a longer distance. Other than that, user can also set the maximum price so that food that has higher price than the maximum price will not be shown in the nearby food. It is useful when user has a budget for his meal. Figure 5.18 shows the “Setting” page.


7:23

Settings

USER PROFILE

Edit profile

NEARBY FOOD

Nearby food search distance
Maximum distance of nearby food in km

Set max price ☐
Set max price of nearby food

Max price
Maximum price of nearby food

Figure 5.18: “Setting” page

5.10 Testing Cases and Results

Testing is carried out when the system is being developed to validate or verify that the system meets user requirements. Test cases and test results of different types of testing are presented and discussed in the following sections.

5.10.1 Content Testing

Content testing is carried out to verify the mobile application that there are no grammatical or spelling error in the content.

Test case ID: 1

Test case name: Grammar and spelling check

Description: Check every page for grammar or spelling error

Step	Step Description	Expected Result	Actual Result	Remarks
1	Browse through every page and check for grammar or spelling error	No grammar or spelling error found	Found some grammar and spelling error	Corrected the grammar and spelling error

Table 5.1: Test Case for Grammar and Spelling Check

Table 5.1 shows the test case for grammar and spelling check. Every page of the mobile application is checked for grammar or spelling error. Grammar and spelling errors are corrected if found.

5.10.2 User Interface Verification

User Interface testing is used to ensure that the user interface is user friendly and works as specified.

Test case ID: 2

Test case name: Design guideline check

Description: Check whether the mobile application follows mobile application interface design guideline

Step	Step Description	Expected Result	Actual Result	Remarks
1	Check whether texts are readable or not	Texts are readable, font size is not too small, font and link colour has high contrast with background colour,	Some background colour has low contrast with font colour	Changed the background colour
2	Check whether all sections are well aligned or not	Every section has its own division which clearly separate with big whitespace from others	Some sections are close to each other	Increase the white space between sections
3	Check whether all the buttons stay at the same position at every page	Same buttons should stay at the same position at different page.	Same button stays at the same position at different page.	

Table 5.2: Test Case for Design Guideline Check

Table 5.2 shows the test case for mobile application design guideline check. The user interface of the mobile application is checked so that it complies with the design guideline. User can easily learn to use the mobile application and avoid any possible error caused by the user interface such as clicking on wrong button.

5.10.3 Navigation Testing

Navigation testing is used to ensure that all the links are working as expected and no broken links.

Test case ID: 3
Test case name: Hyperlink check
Description: Check whether all hyperlink links to the correct page

Step	Step Description	Expected Result	Actual Result	Remarks
1	Press all the hyperlink	All hyperlink links to the correct page	All hyperlink links to the correct page	

Table 5.3: Test Case for Hyperlink Check

Table 5.3 shows that all hyperlinks are tested to ensure that all hyperlinks link to the correct page.

Test case ID: 4
Test case name: Hyperlink semantic correctness check
Description: Make sure all hyperlink names are meaningful and clearly describes where the hyperlink is going

Step	Step Description	Expected Result	Actual Result	Remarks
1	Check all the hyperlink names	All hyperlink names are suitable and correct	All hyperlink names are suitable and correct	

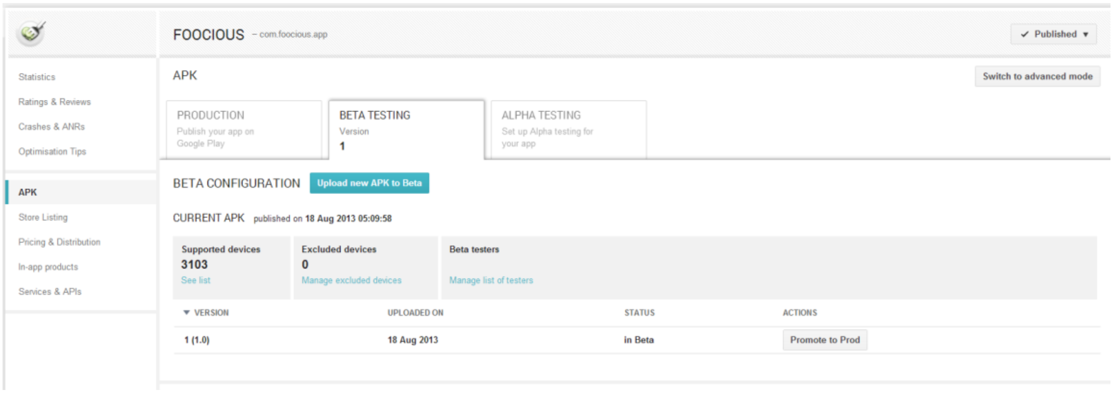
Table 5.4: Test Case for Hyperlink Semantic Check

Table 5.4 shows that all hyperlinks are checked semantically. All hyperlinks should be named correctly and suitably so that user can understand where he is going.

5.10.4 User Acceptance Testing

User acceptance testing is carried out to verify that the system meets user expectation. Google Play Store provides the beta testing feature to test mobile application with a group of users.

The mobile application is published to the Google Play Store for beta testing as shown in Figure 5.19. 10 users were invited to the Google+ community named “foocious beta” to participate in the beta testing. Only users that had joined the community can download the application from Google Play Store.



CURRENT APK published on 18 Aug 2013 05:09:58

Supported devices	Excluded devices	Beta testers
3103	0	
See list	Manage excluded devices	Manage list of testers

VERSION	UPLOADED ON	STATUS	ACTIONS
1 (1.0)	18 Aug 2013	in Beta	Promote to Prod

Figure 5.19: Google Play Store Beta Testing

After users had joined the beta testing, users are asked to test the application by following the test cases. The following tables shows the results of the testing. There will be rating given by user range from 1 to 5. A “1” means the user is “very dissatisfied” and a “5” means user is “very satisfied” with the outcome against expected result.

Test case ID: 5

Test case name: Test register module

Description: Test whether user can register

Step	Step Description	Expected Result
1	Enter username, password and email. Press "register" button	Register user and take user to the "Nearby" page
2	Leave any of username, password or email empty. Press "register" button	Error message "Please fill in all the fields" appears.
3	Enter existing username. Press "register" button	Error message "Username is used" appears.

Table 5.5: Test Case for Register Module

Username	Rating(1-5)
aaron	5
matthew	5
cheah	4
richard	5
chaihi	5
edmund	5
shashi	5
leonard	4
herng	4
ong	4
Average	4.6
Standard Deviation	0.5164

Table 5.6: Test Result for Test Case ID 5

Table 5.5 shows that the register module is tested by asking users to register with different ways. Error messages should be shown if one of the fields is not entered, username or email does not exist. Table 5.6 shows that the average rating for the register module is 4.6 which shows the users are satisfied with the outcome.

Test case ID: 6

Test case name: Test follower module

Description: Test whether user can follow other user and followed users' food posts shows on the "following" page or not.

Step	Step Description	Expected Result
1	Press "Follow" button on other user's "User Profile" page.	User should be followed. "Follow" button changes to "Unfollow" button
2	Press "Following" button on the navigation drawer to go to the "Following" page.	Followed user's food posts are shown.

Table 5.7: Test Case for Follower Module

Username	Rating(1-5)
aaron	5
matthew	4
cheah	5
richard	5
chaihi	4
edmund	4
shashi	5
leonard	3
herng	5
ong	5
Average	4.5
Standard Deviation	0.70711

Table 5.8: Test Result for Test Case ID 6

Table 5.7 shows the test case for follower module. User is asked to go to other user's profile and follows other user. Then, the followed user's food post should be on the user's "following" page. Table 5.8 shows that the average rating for the follower module is 4.5 which shows the users are satisfied with the outcome.

Test case ID: 7

Test case name: Test "like/dislike food" module

Description: Test whether user can like food and dislike food.

Step	Step Description	Expected Result
1	Press "Like" button below food picture on "Nearby" page	Food is liked. "Like" button changes to "Unlike" button. Like's count increases.
2	Press "Dislike" button below food picture on "Nearby" page	Food is disliked. "Dislike" button changes to "Undislike" button. Dislike's count increases.

Table 5.9: Test Case for "Like/Dislike Food" Module

Username	Rating(1-5)
aaron	5
matthew	5
cheah	5
richard	4
chaihi	5
edmund	5
shashi	4
leonard	5
herng	5
ong	4
Average	4.7
Standard Deviation	0.48305

Table 5.10: Test Result for Test Case ID 7

Table 5.9 shows the test case for “like/dislike food” module. User is asked to “like” or “dislike” food. When the food is liked or disliked, the button should change to “unlike” or “undislike”. Table 5.10 shows that the average rating for the “like/dislike food” module is 4.7 which shows the users are satisfied with the outcome against the expected result.

Test case ID: 8

Test case name: Test “add food post” module

Description: Test whether user can add food post.

Step	Step Description	Expected Result
1	Press “camera” button on top right side of the mobile application	“Take picture” and “Choose from gallery” options will be shown
2	Press “Take picture” button	Camera will be launched
3	Take picture	Picture is taken. User will be asked to crop the picture taken.
4	Crop picture	Picture will be cropped. User will be asked to enter food post’s details such as comment, price, location and food name.
5	Enter food details and press “Confirm” button	Food post is added. User will be taken to “Following” page and the added food post will be shown.

Table 5.11: Test Case for “Add Food Post” Module

Username	Rating(1-5)
aaron	5
matthew	4
cheah	5
richard	4
chaihi	5
edmund	4
shashi	5

leonard	4
herng	4
ong	3
Average	4.3
Standard Deviation	0.67495

Table 5.12: Test Result for Test Case ID 8

Table 5.11 shows the test case for "add food post" module. User is asked to add a food post. A user needs to take a picture, crop the picture, enter food details and upload the food post. Table 5.12 shows that the average rating for the "add food post" module is 4.3 which shows the users are satisfied with the outcome.

Test case ID: 9

Test case name: Test "notification" module

Description: Test whether user can view notification.

Step	Step Description	Expected Result
1	User A follows user B	User B's notification should show user A has followed user B.
2	User A comments on user B's food post	User B's notification should show user A has commented on user B's food post.
3	User A likes user B's food post	User B's notification should show user A has liked user B's food post.

Table 5.13: Test Case for Notification Module

Username	Rating(1-5)
aaron	5
matthew	4
cheah	4
richard	5
chaihi	4
edmund	4

shashi	4
leonard	4
herng	4
ong	4
Average	4.2
Standard Deviation	0.42164

Table 5.14: Test Result for Test Case ID 9

Table 5.13 shows the test case for "notification" module. User is asked to follow other user, "like" and comment on other user's food post to check that whether notifications show up on other user's "notification" page. Table 5.14 shows that the average rating for the "notification" module is 4.2 which shows the users are satisfied with the outcome.

Test case ID: 10

Test case name: Test "comment" module

Description: Test whether user can comment food post.

Step	Step Description	Expected Result
1	Press "Comment" button below food post picture	User will be taken to "Comment" page.
2	Enter comment and press "Add" button	Comment is added to the food post.

Table 5.15: Test Case for Comment Module

Username	Rating(1-5)
aaron	5
matthew	4
cheah	4
richard	5
chaihi	5
edmund	5
shashi	5

leonard	5
herng	5
ong	4
Average	4.7
Standard Deviation	0.48305

Table 5.16: Test Result for Test Case ID 10

Table 5.15 shows the test case for “Comment” module. User is asked to comment on other user’s food post. After a user comments on the food post, the comments should show up on the food post’s comment list. Table 5.16 shows that the average rating for the “comment” module is 4.7 which shows the users are satisfied with the outcome.

Test case ID: 11

Test case name: Test “Nearby food” module

Description: Test whether nearby food is shown and sorted correctly.

Step	Step Description	Expected Result
1	Press “Nearby” button on the navigation drawer	User will be taken to “Nearby” page. Nearby food will be shown
2	Press “Sort by Likes” button	Foods will be sorted by number of likes
3	Press “Sort by Price” button	Foods will be sorted by price
4	Press “Sort by Date” button	Foods will be sorted by date
5	Press “Sort by Distance” button	Foods will be sorted by distance

Table 5.17: Test Case for “Nearby Food” Module

Username	Rating(1-5)
aaron	5
matthew	4
cheah	5
richard	5
chaihi	5
edmund	5

shashi	5
leonard	5
herng	5
ong	4
Average	4.8
Standard Deviation	0.42164

Table 5.18: Test Result for Test Case 11

Table 5.17 shows the test case for “Nearby food” module. A user is asked to go to the “Nearby” page where nearby food should be shown on the “Nearby” page. Then, a user should be able to sort nearby food by distance, date, number of like or price. Table 5.18 shows that the average rating for the “Nearby food” module is 4.8 which show the users are satisfied with the outcome.

The average ratings for all test cases are generally more than 4 and the standard deviations are less than 1 which means the users generally highly rate the system and the ratings are quite consistent across different users.

5.11 Conclusion

Runtime features of the mobile application such as showing nearby food, hashtag, follower system, adding food post had been discussed in this chapter. Screenshots of the mobile application and test results on the system show that the features are working according to requirements.

After beta testing is carried out for the mobile application, there are around 30 foods added to mobile application around Genting Klang area. With the data contributed by the beta testers, the mobile application can help user to find the most popular, cheapest, most recent or nearest food around Genting Klang area.

CHAPTER 6

CONCLUSION

This project aims to provide a social network platform which allows users to share their food reviews with everyone so that it can assist users to make decision on the eating outlet and food considering the preferred locations, taste and price of the food as reviewed by other users.

Crowdsourcing is used to collect the food reviews from users so that users can share their food reviews with everyone and location-based service is used to connect users who are near to each other and find out what is the popular or cheapest food in an area. Comment, like and follower modules are added into the mobile application to improve the interaction between user and the application, it can increases the time that user spend on application, thus, it makes user more likely to contribute to the application.

This project is categorized into two parts which are a client side application and a backend server. Android native application acts as the client side and PHP is used to develop the backend server. Laravel PHP framework is used to ease the writing of PHP code on the server side. Foursquare Venue API is used to access the wealth information of nearby eating outlets and Google Maps API is used to generate the eating outlets' location map easily.

The database and interaction design of this projects are discussed in Chapter 4 and features of the mobile application are explained by the screenshots of the mobile application in Chapter 5.

In conclusion, the mobile application can only help user to find the most popular, cheapest, most recent or nearest food around Genting Klang area now because there are only food reviews around Genting Klang area. However, if the application user base continues to expand to other area, the mobile application will be able to provide information about the eating outlets in other area too.

In the future, it is proposed that this project can be enhanced in the following ways:

1. Able to search for eating outlets or food at other area rather than only nearby eating outlets or food.
2. Able to re-share food review.
3. Able to save food or eating outlets to a wish list
4. Show available promotions at an eating outlet
5. Create a website as client side so that users can read food reviews using web browser.
6. Connect the mobile application with Facebook, activities such as “like food review” or “add food review” can be shared to Facebook automatically.

REFERENCES

Beck, K., Beedle, M., Bennekum, A.v., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D. (2001) *Manifesto for Agile Software Development*, [Online], Available: http://agilemanifesto.org/ [30 March 2013].

Brabham, D.C. (2008) 'Crowdsourcing as a Model for Problem Solving', *The International Journal of Research into New Media Technologies*, vol. 14, p. 75.

DB-Engines Ranking (2013), April, [Online], Available: http://db-engines.com/en/ranking [4 April 2013].

Location Data Is Transforming Mobile (2013), 21 February, [Online], Available: http://www.businessinsider.com/location-data-is-transforming-mobile-2013-2 [2 March 2013].

Olenski, S. (2013) *Is Location Based Advertising The Future Of Mobile Marketing And Mobile Advertising?*, 17 January, [Online], Available: http://www.forbes.com/sites/marketshare/2013/01/17/is-location-based-advertising-the-future-of-mobile-marketing-and-mobile-advertising/ [2 April 2013].

Our Mobile Planet: United States (2012), May, [Online], Available: http://www.google.com/think/research-studies/our-mobile-planet-united-states.html [February 28 2013].

Rovick, P. (2011) *Location Based Services_Mobile Apps*, 17 September, [Online], Available: http://www.slideshare.net/provick1/whitepaper-mobile-phone-location-based-apps-rovick17sep11 [1 April 2013].

The foursquare Platform, [Online], Available: https://developer.foursquare.com/overview/ [26 February 2013].

US Census Bureau (2012), [Online], Available: http://www.census.gov/geo/ZCTA/zctafaq.html#Q20 [1 March 2013].

Wasserman, T. (2011) *What's Behind the Food Photography Trend?*, 9 May, [Online], Available: http://mashable.com/2011/05/09/foodtography-infographic/ [3 March 2013].