**Table of Contents**

**Part I**

**Software Quality Assurance Plan (SQA) and Result**

[**1. Project Name : KAIDEE**](#_9qkild558lyt) **1**

[**2. Objective of this document**](#_3hw7hjnygpzw) **1**

[**3. Term definition**](#_mwgy6c2xn9xb) **1**

[**4. Summary scope of the project**](#_m6k3p27john2) **1**

[**5. Software quality assurance goal**](#_mewp9sjxkq6f) **2**

[**6. SQA policy**](#_vgd7qmmexgxr) **2**

[**7. Unit Testing**](#_6cx16cz1rdqz) **3**

[7.1 Strategy Method for Process and Product Quality Assurance for Unit Testing](#_k5oxugu297im) 3

[7.2 Quality Assurance Activity Implementation and Result](#_7bclw5pjlkmf) 3

[7.3 Act on result](#_uws8qbilqz41) 4

[**8. Integration Testing**](#_5y0c96elis7f) **5**

[8.1 Strategy Method for Process and Product Quality Assurance](#_m4wm8zimnzg3) 5

[8.3 Act on result](#_l54oud5cs6z7) 7

[**9. System Testing**](#_z3c7607s9dhk) **8**

[9.1 Strategy Method for Process and Product Quality Assurance](#_80hhjbw1lnw5) 8

[9.2 Quality Assurance Activity Implementation and Result](#_6o9bctuw2vy9) 9

[9.3 Act on result](#_12s6c54a4jzr) 26

[**10. User Interface Testing**](#_tzmb1tbgmalr) **27**

[**11. Design verification and validation**](#_68a18bqvjmta) **34**

**Part I**

**Software Quality Assurance Plan (SQA) and Result**

# **1. Project Name** : KAIDEE

# **2. Objective of this document**

The objective of this document is to clarify each step of the testing plan of our Kaidee Application. The testing plan will test the system starting from the component level to the integration of the entire system in order to make sure that each feature achieve a high quality result. This testing plan will include three types of test: unit test, integration test, and system test, and will be conducted by both the developer team and other members of the project on the entire system.

# **3. Term definition**

The followings are some terms and their definitions:

**Seller** – A user who has products that would like to sell on Kaidee platform. Sellers will be people who upload product information to Kaidee system. They will also receive money from Kaidee after buyers receive the product. Sellers can respond to buyers’ questions through chat platform on Kaidee application.

**Buyer** – A user who want to buy some products on Kaidee platform. Buyers will be people who transfer money to Kaidee after confirming to buy some products. They can also rate and comment on the seller they buy the product from after the purchasing process is completed. Buyers will be able to chat with sellers and provide their product of preference as well.

**First handed** – Products on the application that have not been used by sellers. Buyers will get the product in 100% brand new quality from sellers. This option can be chosen on filter feature when browsing for products. It comes along with a second handed option.

**Second handed** – Products on the application that might be used a few times by sellers. Buyers can buy the same product with a lower price. This option is in filter feature alongside with the first handed option.

**Wishlist** – Wishlist is a section where buyers (or users in general) can save some products that they are interested in.Users can also select the sort by button on the top right corner of the page to compare the products in the wish list. Users can choose to view their wishlist on their profile and also at the bottom of the main page.

**Hashtag** – Hashtag is a sub-category feature located in search feature. It is a word or a phrase describing a particular group of product. This is used for users to find what they like more specifically. Hashtag feature is also in “sell” feature, which is when sellers upload photos of their products on Kaidee. Sellers can also select/type in some hashtag words before uploading in order to help buyers find the product easier.

**Cart** – Cart is a section which will have the information of the product for buyers to review before they purchase that product. The cart will keep the information of only 1 product.

# 

# 4. Summary scope of the project

This project has been initiated to increase trust between sellers and buyers, and to attract young generation and female users to Kaidee through the use of knowledge from three subjects which are Software Engineering, Database Systems, and ICE Capstone.

In order to increase trust between buyers and sellers and prevent fraud, we came up with a new payment system which will make Kaidee become a middleman between buyers and sellers, together with a feedback system which will allow buyers to rate sellers. We also decided to attract young and female buyers by improving user interface, user experience, and other several features such as product recommendation, product comparison, and advanced search&filter.

# 5. Software quality assurance goal

**Requirements quality –** The correctness, completeness, and consistency of the requirements model will have strong influence on the quality of all work products that follow.

**Design Quality –** Every element of the design model should be assessed by the software team to ensure that it exhibits high quality and that the design itself conforms to requirements by using user interface testing. Moreover, good design quality should have low coupling and high cohesion in which it can be test by using verification and validation method.

**Code quality –** Source code and related work products must conform to local coding standards and exhibit characteristics that will facilitate maintainability.

**Quality control effectiveness –** A software team should apply limited resources in a way that has the highest likelihood of achieving a high quality result.

# 6. SQA policy

Software quality assurance policy aims to maintain an effective and efficient software quality assurance, to eliminate deficiencies and inaccuracies and to ensure high quality standards.

The software quality assurance policy is fundamental for all work undertaken by both the developer team and other members of the project on the entire system and should be implemented by all members in their work. To that effect the project shall:

**6.1** Maintain consistency in work method throughout in accordance with set policies, procedures, regulations and codes of practice and without significant deviation.

**6.2** Ensure that all policies, procedures, relevant regulations and codes of practice are implemented and systematically reviewed to reflect the values of project.

**6.3** Regularly monitor and measure the quality of its work methods, outputs and outcomes with a view to ensuring high quality standards, best value and continuous improvement.

Together, this will provide efficient results and error free product.

# 7. Unit Testing

## 7.1 Strategy Method for Process and Product Quality Assurance for Unit Testing

The main idea of unit testing is to particularly focus on single element with specific action taken by the user. Related path of execution and set of expected result will be main consideration according to the input action. This type of testing will be more focus on test case that will likely to lead to an error. The test case will generated based on the user action. The main purpose is to observe and see whether the system is able to handling the error properly so the user can proceed and take further action.

## 7.2 Quality Assurance Activity Implementation and Result

**Class Name: Registration**

**Class Overview and Objective**

This class let the user register for an account on Kaidee in order to buy or sell the product on Kaidee. The user will provide there 10 digits mobile phone number with their desired password. The mobile number represents the user identity.

**Interface**

Interface of this class consists of 3 text fields, mobile number, OTP and password and 2 button request for OTP and submit.

**Output**

The input fields stored as a distinct variable as String data type must be prepared to be sent, as an output, to the API:

* Phone Number
* Password

*Testing Action on : Registration Page*

|  |  |  |
| --- | --- | --- |
| **Path** | **User Action** | **Expected Result** |
| **1** | **User put in mobile number and new password but mobile number is not 10 digits then tap on submit** | The system will enter handling method which will not allow to proceed because the information of mobile number used for registration is critical data to confirm the identity and uniquely distinguish the user.  Thus, it is necessary to ensure that the system obtain this information correctly. User must make correction to their mobile input and attempt again. |
| **2** | **User put in valid 10 digits mobile number and new password but mobile number is already registered with Kaidee and tap on submit.** | The system will detected duplicated of user and will show the error “Invalid Phone Number” |
| **3** | **User put in valid 10 digits number and password then tap on submit.** | The registration process will be successful and the system will navigate the user to the product recommendation page. |

\*Path 1,2 are considered to be exception handling case.

**Class Name: Preference**

**Class Overview and Objective**

This class has purpose to obtain the product preferences of the user. Information about interested product categories and items condition(first hand, second hand) will be collected.

**Interface**

Interface of this class consists of 2 pickerviews, 1 switch, and 1 button. Picker views are for choosing interest product category and prefered item conditions. Switch is for choosing whether user wants to receive newsletter from Kaidee. Button is for confirm the choosing of preference and newsletter decision.

**Output**

The input fields stored as Int datatype for categories preference which will be keep as a match in the reference table. For example, choosing preference “mobile” maybe keep as int ‘0’ in the reference table. For the items condition, first hand and second hand, it will be keep as boolean value.

*Testing Action on : Preferences Page*

|  |  |  |
| --- | --- | --- |
| **Path** | **User Action** | Expected Result |
| **1** | **User choose their categories according to their preferences, items conditions, and choose whether he/she wants to receive newsletter from Kaidee. Then tap on “Submit”.** | User will be navigated to the product recommendation page. The product recommendation page will show the products according to the user preferences. The user will also get the newsletter sent to their email when Kaidee issued based on their choice. |
| **2** | **User doesn’t choose any preferences on categories and/or items conditions.** | User will be navigated to the product recommendation page but the result shown will be default. |

\*Note : There will be no error handling in this case because there is no way invalid input can occur.

## 7.3 Act on result

When the user is in registration process, if he/she put in the valid mobile number but doesn’t put in password and tap on submit button. In this case, it should be an error but we didn’t create the handling, the information of the registering user will be keep but the password will be stored as blank. So this violates the security requirements, only the user who register with their email and password can log-in to the system. If this is the case after registration is completed. Login from anywhere can be done without password(because it is stored as blank). With only email/mobile, the application login the user navigate the user to the product recommendation page.

This case of error is done by creating the exception handling just like the case of invalid mobile number input. The system will show up the pop-up notify the user that the password is required and not allow the user to proceed until he/she have fill in their new password.

Note that on the registration process we are not taking the OTP process into consideration as we have to contact the cellular service provider so here we are consider the 10-digit mobile number input with non-blank password to be valid input leaving the OTP field blank. OTP is made for further deployment of the company as needed.

# 8. Integration Testing

## 8.1 Strategy Method for Process and Product Quality Assurance

After performed unit testing, we will test the modules using integrated testing using bottom-up approach. The lowest level components are grouped with higher level components and then test whether the components can interact with each other according to the specification or not. The grouped components continue to group with higher level components forming cluster until it reached highest level component which resulting in the complete system.

**8.2 Quality Assurance Activity Implementation and Result**

For all the functions we have: log in/registration, product recommendation, advance search and filter, add product to wish list, add product to cart, payment confirmation, give feedback and rating; we will first test whether the type of value passing between the application and database is conform to the specification or not and the data is properly stored/load to and from the database or not. Next, we will check the flow of the data between the database and the method whether it is corresponding to the parameter and variable defined or not. At last, we will test that the data that is received from the user through user interface is correctly pass through the method and store in the database or not and also check that the data retrieved from the database is correctly shown on the user interface.

**8.2.1 Test the reliability of database of login function**

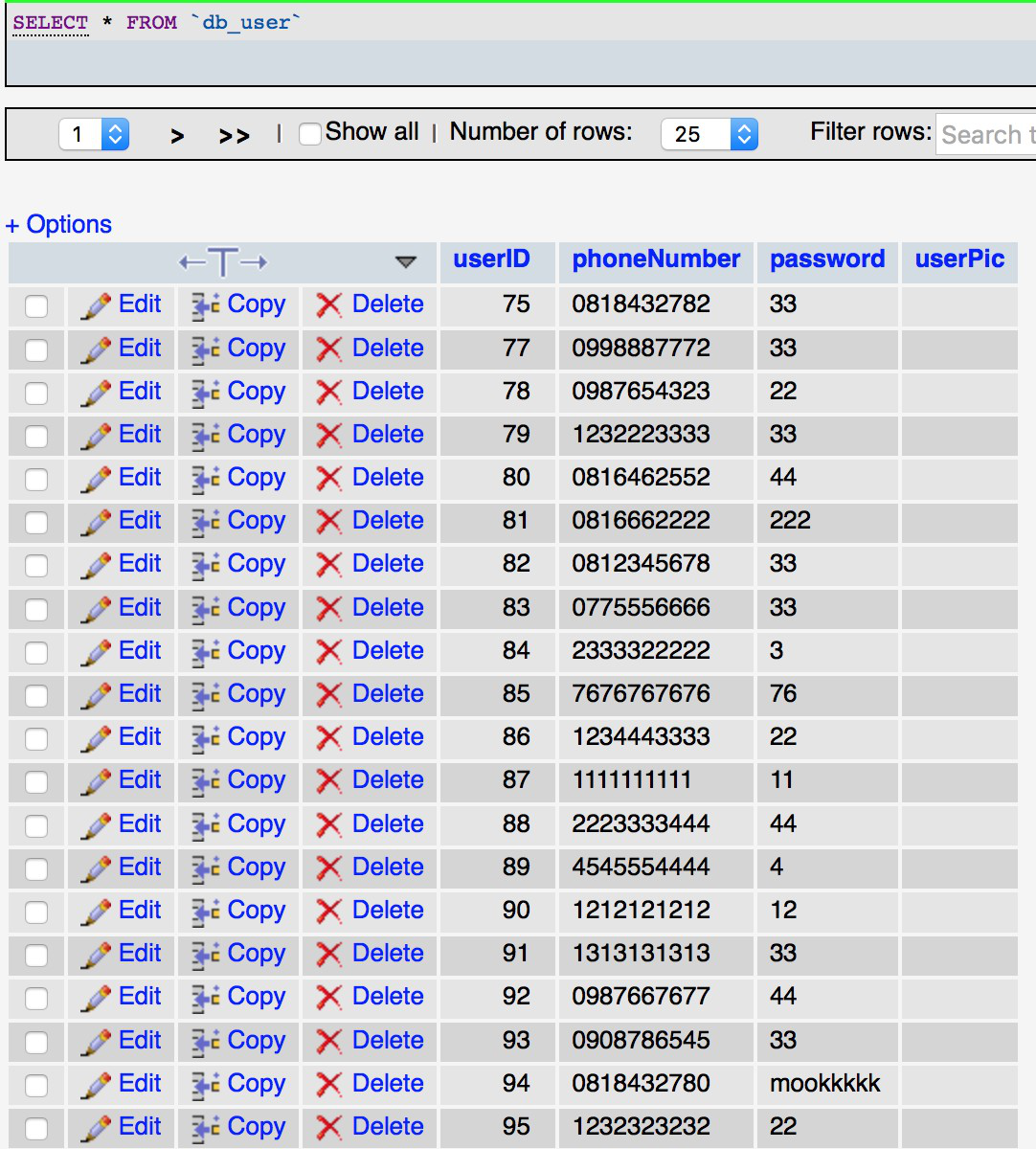
****

Figure 8.1 Database table and data retrieved from the database

The left picture shown above indicate that the value of data is correctly inserted into the database is the ‘user table’(phpMyAdmin) and into the right column and the right picture show that the data is retrieved correctly from the database(printed in the terminal or Command Prompt).

**8.2.2 Test data flow between database and method of login function**

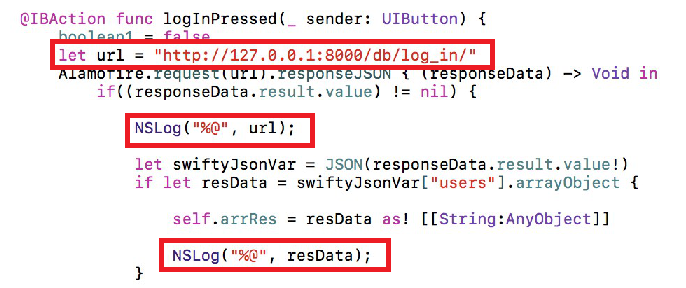


Figure 8.2programming code for getting information from database (Xcode)

****

Figure 8.3 200 code state successful data retrieved

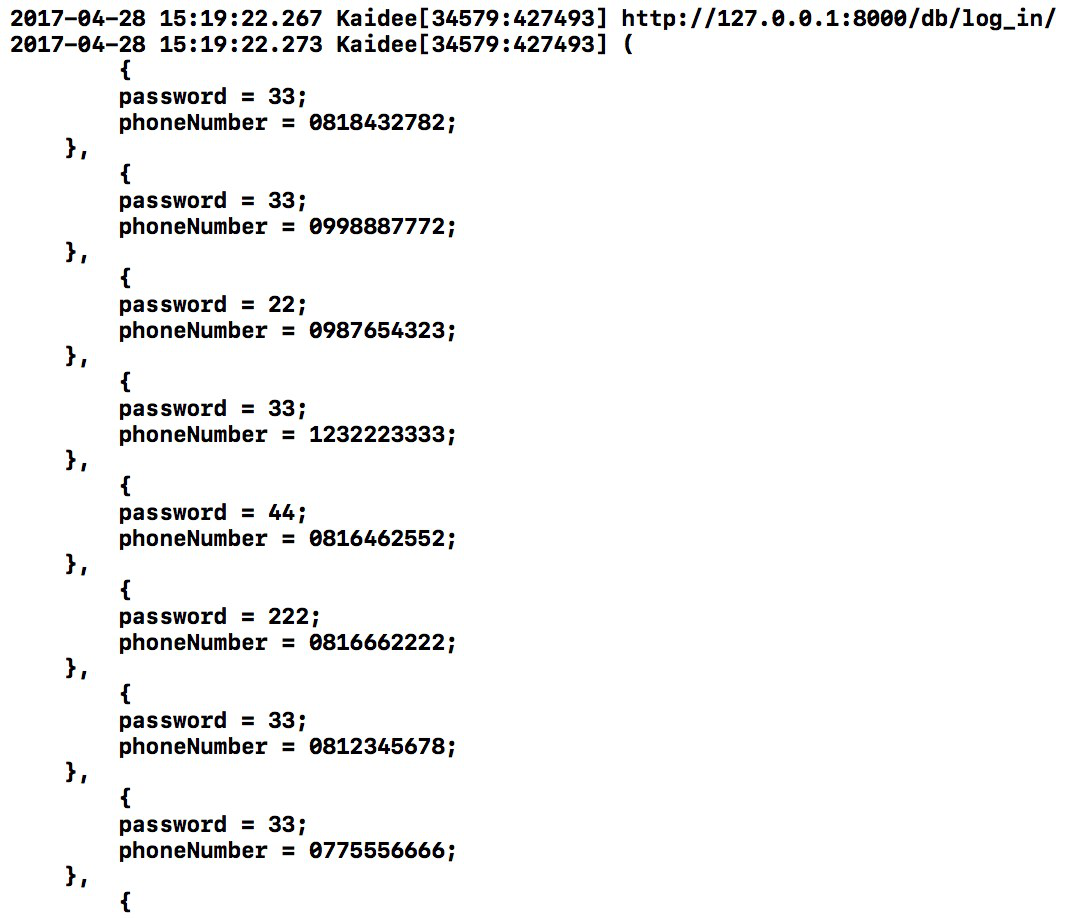
****

Figure 8.4Data shown in the Xcode program

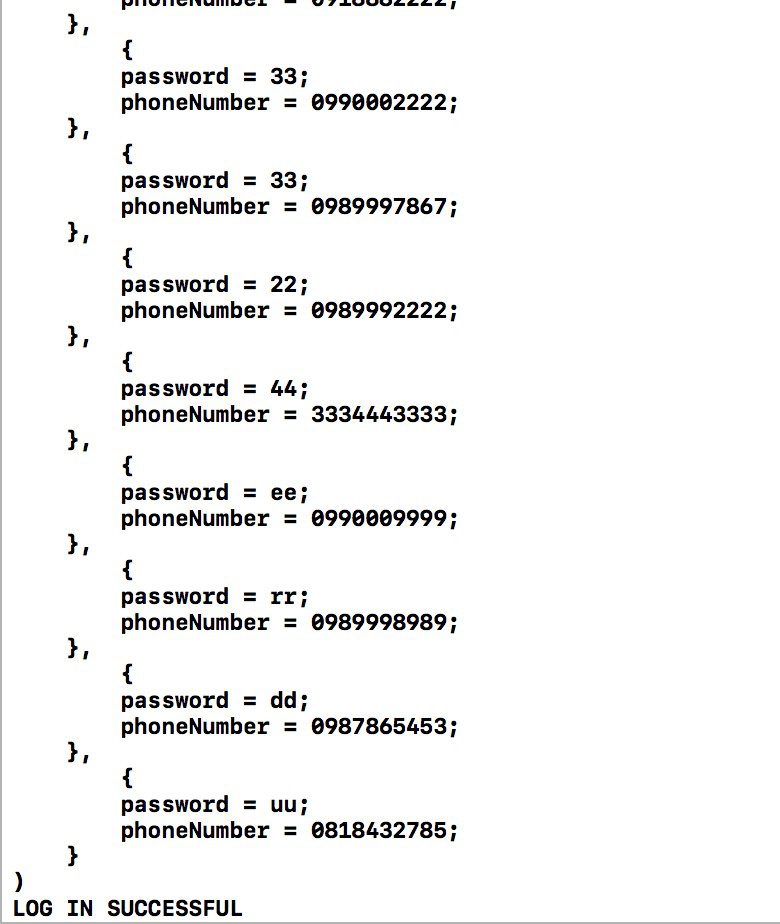
****

Figure 8.5Printed login status

For the second step, we check that the data can be retrieved from the database via the Xcode program in the login class.The GET command can be executed successfully since the database reply with the code ‘200’ which means success. The data printed in the program is corresponded to the data stored in the database. Also, we write a code to check if the login phone number and password is similar to one of those in the database the system is logged in with a ‘log in successful’ notification.

**8.2.3 Test with user interface**

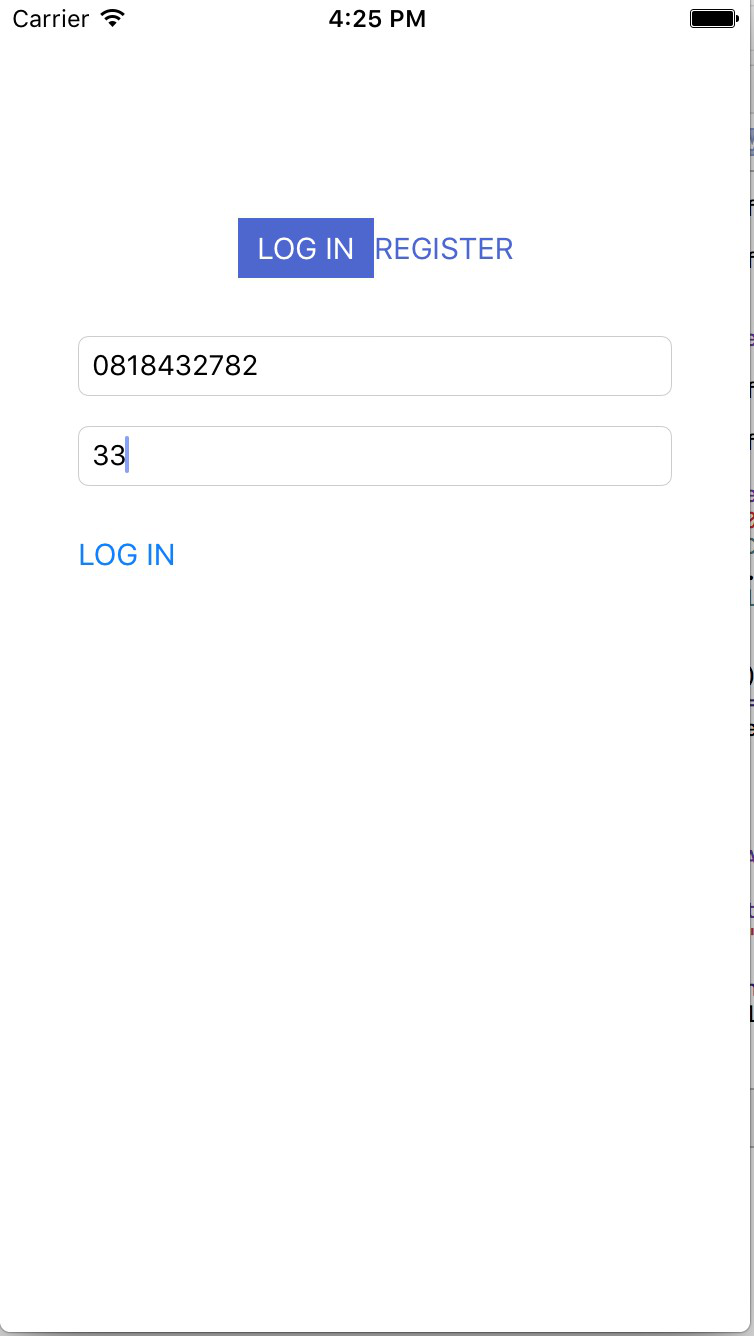
****

Figure 8.6 Login scenario

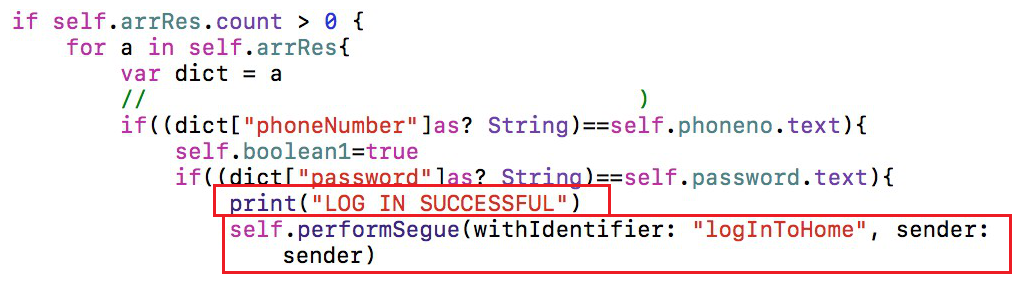
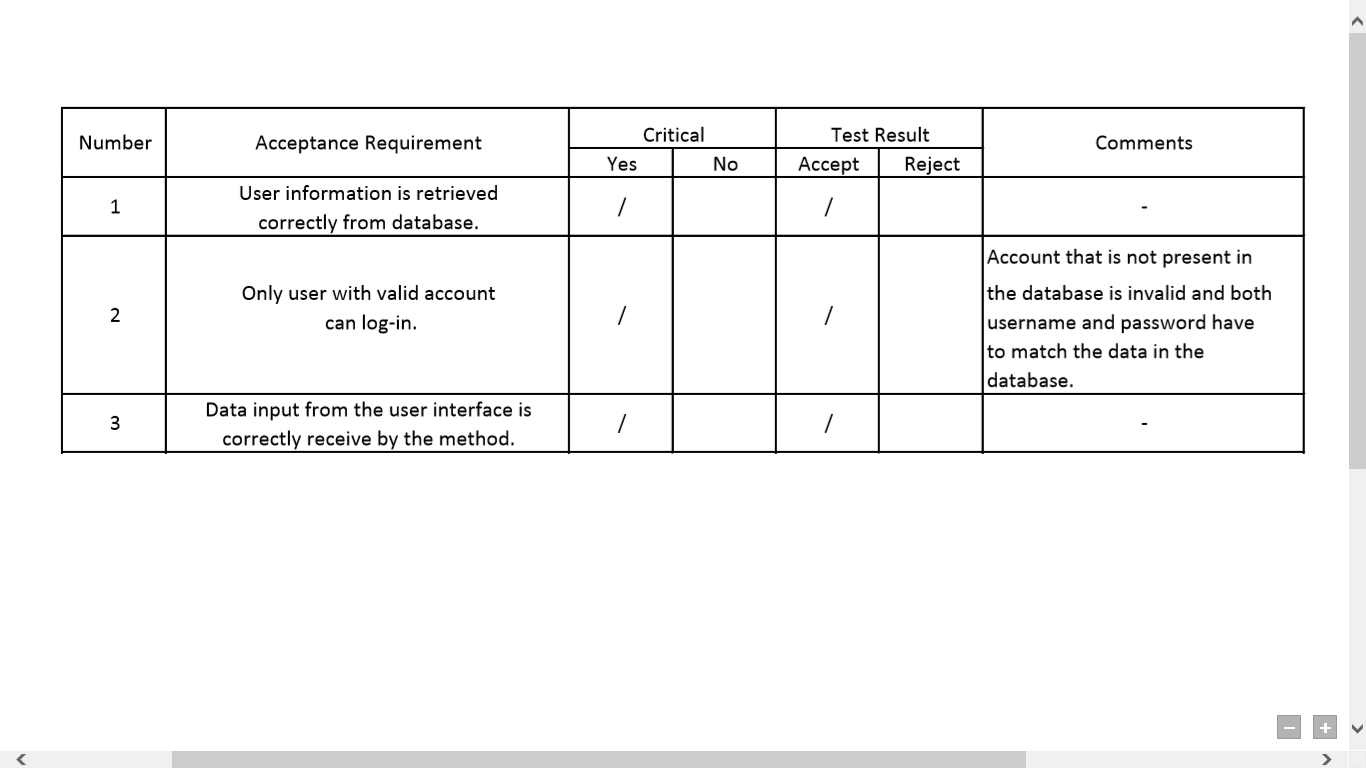


Figure 8.7Login successful code

For the last step, we test with the real scenario where users put in their phone number and password. The user interface can receive the data correctly and pass it into the method to check for the validity of the user account. The ‘log in successful’ tag is printed along with the logged-in user ID. The ‘home’ user interface will then be shown with user ID pass to the page.

## 8.3 Act on result

****

There is no non-conformance or deviation with the requirements.

# 9. System Testing

## 9.1 Strategy Method for Process and Product Quality Assurance

The goal of system testing is to ensure that all requirements are met for all components of the system. Systems testing examines the ability of the system to meet both functional and nonfunctional requirements.

**9.1.1 Functional Testing**

For functional testing, we decided to do the use case testing which is a functional black box testing technique by testing the end to end flow of both normal flows and alternative exceptional flows based on the user action and system action. User action is the steps which the user does (Request) and system action is the output of user action (response). Test cases are based on use cases. The expected result is system can perform according to user action.

**9.1.2 Non-Functional Testing**

Performance, usability, reliability, security, and operational requirements were considered. The main focus is placed on performance and usability. Therefore, two tests were developed to test performance and usability of components in the application under a particular workload.

The goal of performance testing is to determine the response time taken to complete each function. During this testing, system components will be monitored to verify the response time. The expected result is the response time must be less than 3 seconds for any transactions over the network, and the system database must be updated in real time.

The goal of usability testing is to identify any usability problems, collect qualitative and quantitative data and determine the participant's satisfaction with the product. Usability testing assesses how well the user interface supports the use cases. Usability tested whether the functional requirements were met by the system as well as the ease of use of the system.

## 

## 

## 

## 9.2 Quality Assurance Activity Implementation and Result

**9.2.1 Functional Testing**

1. **Use Cases and Functional Requirements**

|  |  |  |
| --- | --- | --- |
| **Use Case Name: Make Registration ID: 1** | | **Functional Requirement.** |
| **Normal Flow/Subflow** | **Alternative Exceptional flows** | **REQ-1:**  **Make Registration** |
| 1. The buyer/seller clicks on register button and puts in phone number | 1a. The buyer/seller puts in invalid phone number. | 1. The system allows user to make registration using mobile phone number.  1a. The system displays error message. |
| 2. The buyer/seller chooses category preferences. |  | 1.2 The system allows user to choose preferences for recommendation feature. |
|  | 2a. The buyer/seller doesn't want to choose product preferences. | 1.2a The user can skip the choosing preference process. |
| 3. The buyer/seller edits profile |  | 1.3 The system updates new information in the database. |
|  | 3a. The buyer/seller doesn't want to edit profile. | 1.3a The system doesn’t make change to the database. |
|  | | |
| **Use Case Name: View Recommendation ID: 2** | | **Functional Requirement.** |
| **Normal Flow/Subflow** | **Alternative Exceptional flows** | **REQ-2:**  **View Recommendation** |
| 1. The buyer views product recommendation. |  | 2.1 The system provides the personalized recommendation for products according to product category preferences selected during registration process. |
| 2. The buyer explores further. |  | 2.2 The user can click on the product to see more details. |
|  | 2a. The buyer isn't interested in recommended products. | 2.2a The user can ignore the recommended products and explore other things. |
|  |  |  |
| **Use Case Name: Search and Filter ID: 3** | | **Functional Requirement.** |
| **Normal Flow/Subflow** | **Alternative Exceptional flows** | **REQ-3:**  **Search and Filter** |
| 1.S-1 The buyer searches with hashtag (keyword). |  | 3.1 The system allows user to search products with keyword by matching the keyword to the hashtag that seller added when posting the product. |
| 1.S-2 The buyer filters products. |  | 3.2 The system provides an advanced filter which allows user to search product by category, subcategory, and range of price. |
|  |  |  |
| **Use Case Name: Compare Products ID: 4** | | **Functional Requirement.** |
| **Normal Flow/Subflow** | **Alternative Exceptional flows** | **REQ-4:**  **Compare Products** |
| 1. The buyer adds product to wishlist. |  | 4.1 The system allows user to add up to four different products. |
| 2. The buyer selects comparison criteria. |  | 4.2 The system allows user to order products by category or price. |
| 3. The buyer views compared product. |  | 4.3 The system displays products ordering by the chosen criteria. |
|  |  |  |
| **Use Case Name: Make Payment ID: 5** | | **Functional Requirement.** |
| **Normal Flow/Subflow** | **Alternative Exceptional flows** | **REQ-5:**  **Make Payment** |
| 1. The buyer orders product. |  | 5.1 The system brings the user to payment process. |
| 2. The buyer puts in money transfer information. |  | 5.2 The user can successfully add the payment information and the order is sent to the seller. |
|  | 2a. The verification of money transfer information has an error. | 5.2a The system displays an error message. |
|  | 2b. The buyer goes back to step 2 again | 5.2b The user can edit the payment information. |
| 3. The buyer confirms an order. |  | 5.3 The system brings the user to the feedback page after the user confirmed order. |
| 4. The buyer gives feedback on seller. |  | 5.4 The system can successfully post feedback on the seller’s profile. |
|  | 4a. The buyer doesn't want to give feedback. | 5.4a The user can skip the giving feedback process |

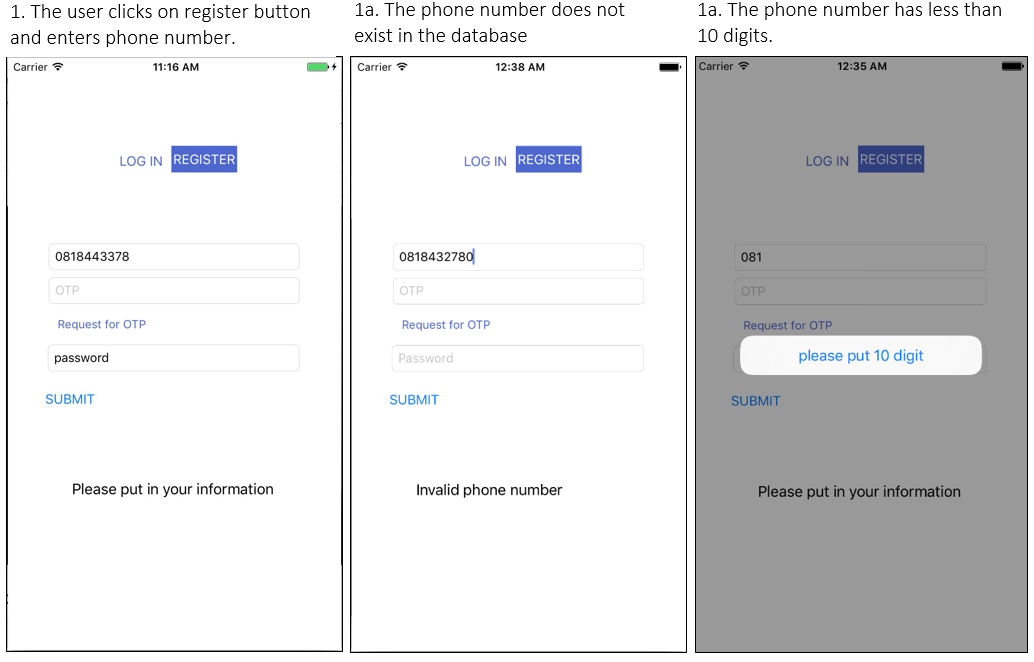
**2)** **Functional Testing and Example of Result**

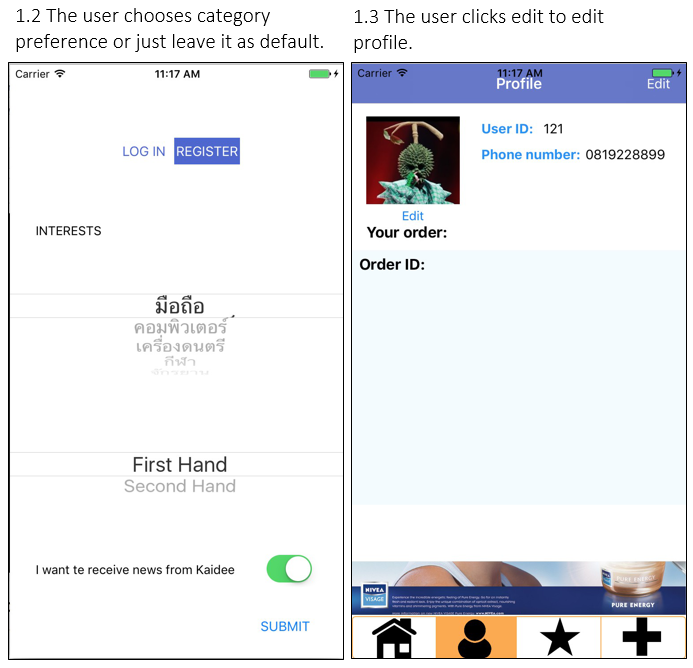
Each use case is performed according to the normal flow, subflow, and alternative exceptional flow written on its use case description. The results are then compared with the functional requirements of each use case that were proposed in the System and Software Requirement Specification Document. In order to pass each criterion, the result must conform to its requirement. Otherwise, there must be some action on the result in order to improve the system. The following table shows the result of Search and Filter use case testing. There is no Alternative Exceptional flows for this use case.

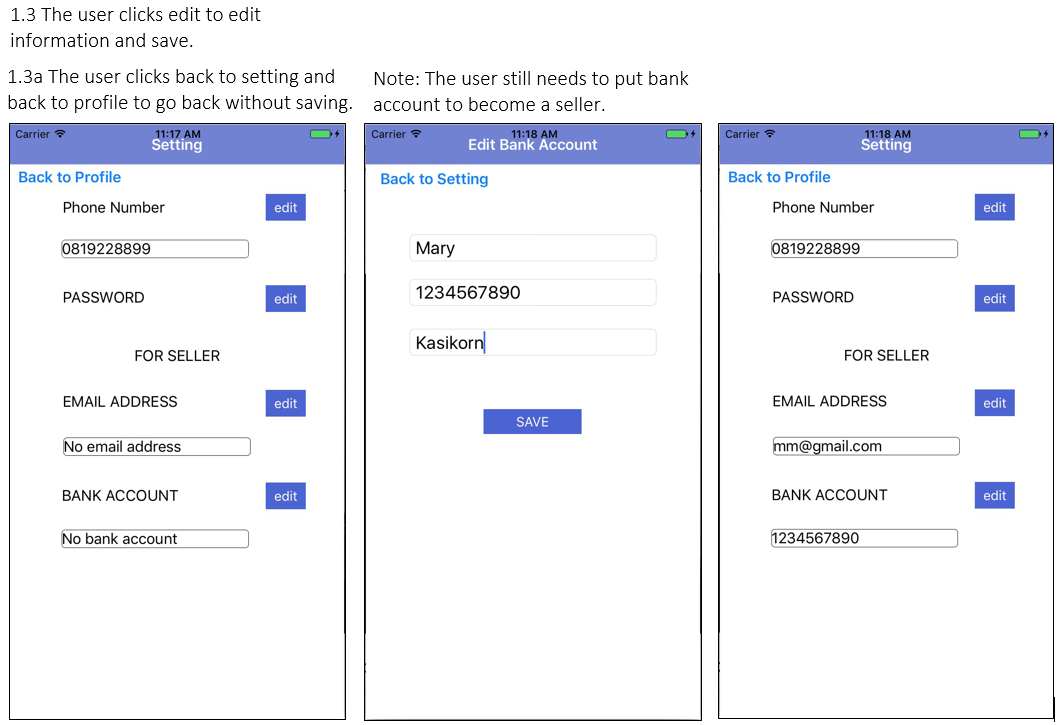
**2.1) Make registration use case.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name: Make Registration ID: 1 REQ-1: Make Registration** | | | |
| **Normal Flow/ Exceptional Flow** | **Expected Result** | **Test Result**  **(PASS/FAIL)** | **Note** |
| 1. The user clicks on register button and enters phone number.  **Exceptional Flow**  1a. The user enters invalid phone number. | **REQ-1** The system allows user to make a registration using mobile phone number.  **1a.** The system displays error message. | **PASS**  **PASS** | There are two types of validation check: check digit check and database check. |
| 2. The user chooses category preferences.  **Exceptional Flow**  2a. The user doesn't want to choose product preferences. | **REQ-1.2**The system let user choose preferences for recommendation feature.  **1.2a** The user can skip the choosing preference process. | **PASS**  **PASS** | When the user skips this process, the preference will be set to default value. |
| 3. The user edits profile  **Exceptional Flow**  3a. The user doesn't want to edit his/her profile. | **REQ-1.3** The system updates new information in the database.  **1.3a** The system doesn’t make change to the database. | **PASS**  **PASS** | The user still needs to put bank account to become a seller. |

**Example of result**

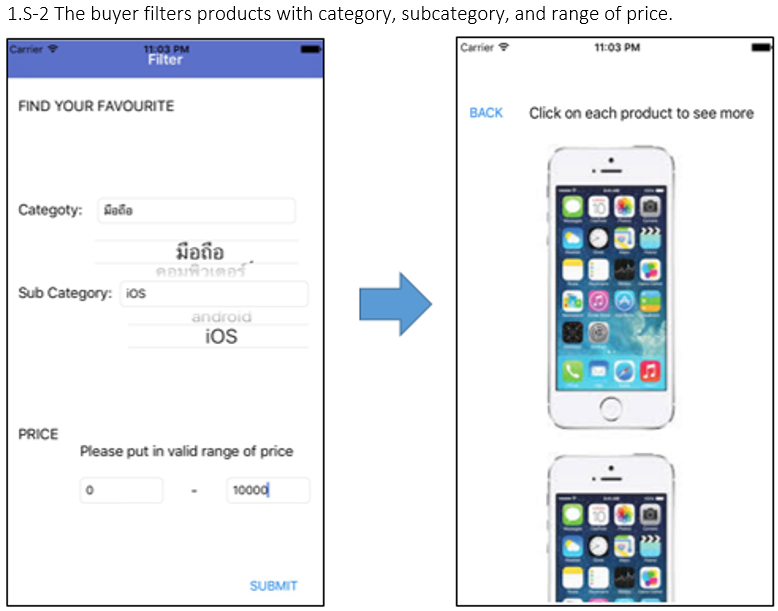
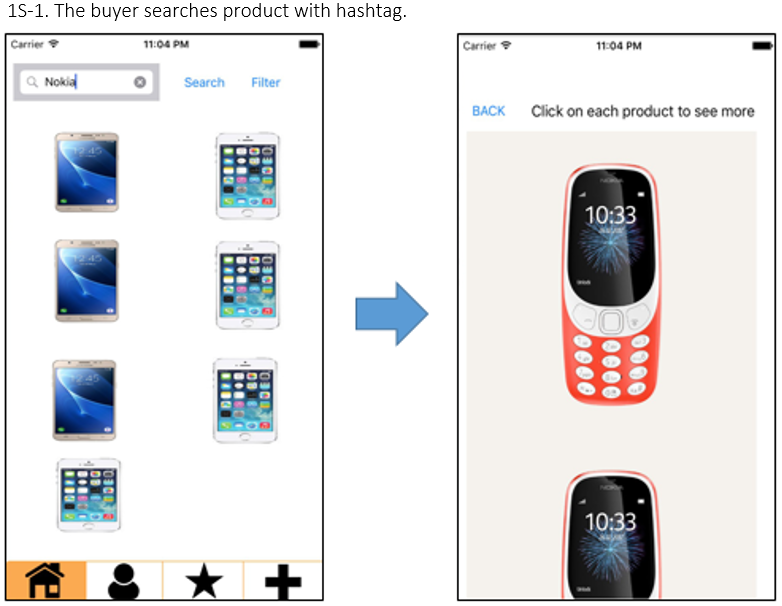






**2.2) Search and Filter use case**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name : Search and Filter ID: 3 REQ-3: Advanced Search & Filter** | | | |
| **Normal Flow/Subflow** | **Expected Result** | **Test Result**  **(PASS/FAIL)** | **Note** |
| **1.S-1** The buyer searches with hashtag. | **REQ-3.1** The system allows user to search products with hashtag. | **PASS** | The system displays result matched to the hashtag “Nokia”. |
| **1.S-2** The buyer filters products with category, subcategory, and range of price. | **REQ-3.2** The system allows user to search product with matched category, subcategory, and range of price. | **PASS** | The system displays result matched to category “มือถือ”, subcategory “ios” within range 0 - 10000 baht. |

**Example of result**

**9.2.2 Non-functional Testing**

1. **Non-Functional Requirements**

|  |  |
| --- | --- |
| **Non-functional Requirements** | |
| **1. Performance** | 1.1 Response time for any transactions over the network must be less than 3 seconds  1.2 The system database must be updated in real time. |
| **2. Usability** | 2.1 User interface supports the use cases and minimize user effort.  2.2 White space, color, fonts, graphics, and interface elements subtly highlight important content and convey interactivity. |
| **3. Reliability** | 3.1 The system displays the alert message to the user when it detects an absence of internet connection. |
| **4. Security** | 4.1 Password must be valid in order for user to log into the member system.  4.2 Only authorized administrators will be able to manage contents in the system. |
| **5. Operational** | 5.1 The system can perform on Apple iPhone with iOS 8.2 or higher.  5.2 The system can perform when connected to the internet.  5.3 The system will read and write to the main database in the system. |

1. **Non-Functional Testing and Example of Result.**

**2.1) Performance Testing**

Particular attention was placed on performance testing of the mobile iOS application. The speed test will be performed by recording the response time of each function. The selected use case for this testing was Search and Filter.

For advanced search, the response time is referred to the amount of time taken from the moment that a user click to on search button until the time that the application displays all products that have hashtag related to the search key.

The response time for the advanced filter is referred to the amount of time taken from the moment that a user click on the submit button after choosing preferences until the time that the application displays all products that match to the preferences.

The expected result that will conform to the requirement for this use case is that the response time for each task must be less than 3 seconds. Otherwise, the system need to be improved.

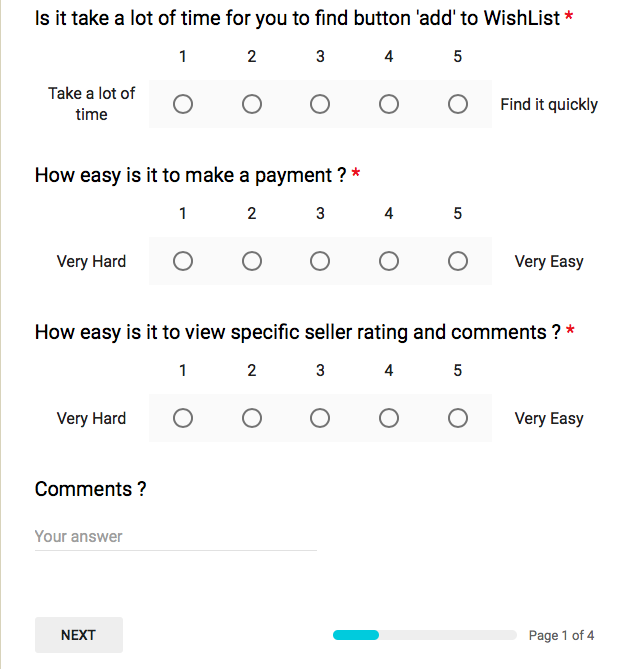
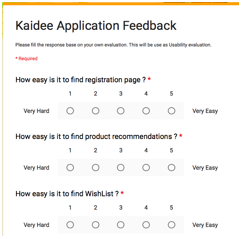
|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name : Search and Filter ID: 3** | | | |
| **Normal Flow/Subflow** | **Expected Result** | **Result**  **(Response Time)** | **Result**  **(PASS/FAIL)** |
| 1.S-1 The buyer search with hashtag (keyword). | 3.1 The response time for the system to display the products with matched hashtag is less than 3 seconds. | **1.65 sec.** | **PASS** |
| 1.S-2 The buyer filter products. | 3.2 The response time for the system to display the products with matched preferences is less than 3 seconds. | **1.13 sec.** | **PASS** |

**2.2) Usability Testing**

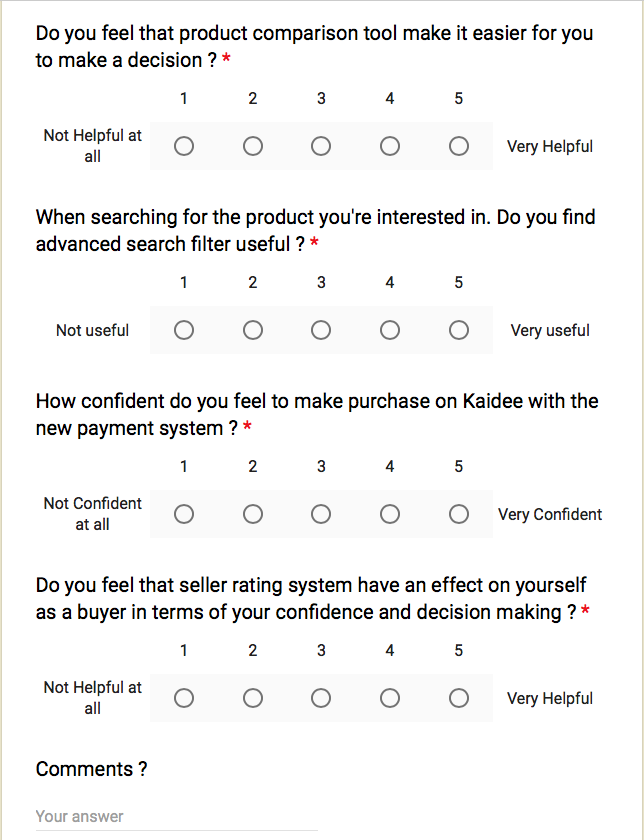
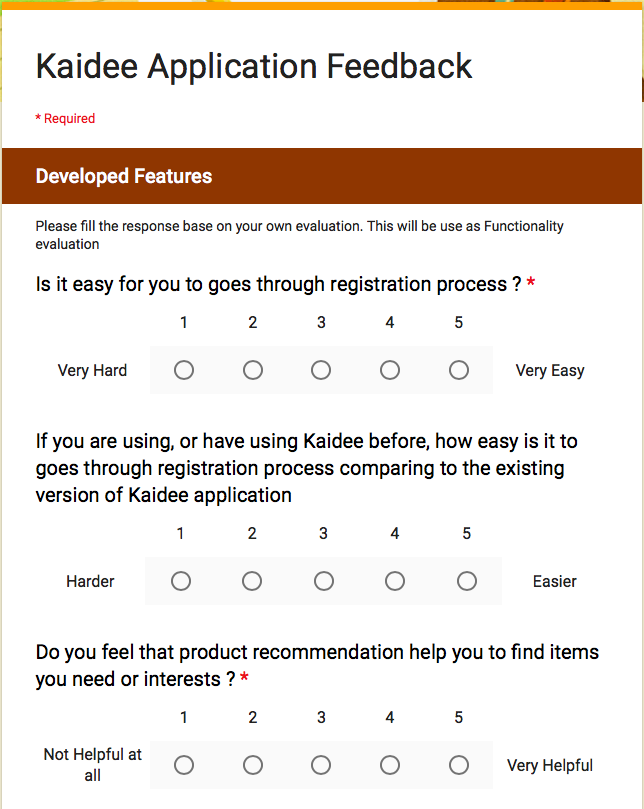
The test was conducted with representative users from different background. During a test, participants will try to complete typical tasks while observers watch, listen and takes notes. The participants will also be asked to do the questionnaires.

The questionnaires are divided into three main sections and one more additional section at the end to collect informations of participants. The first section of the questionnaire is to determine the ease of use of the application(evaluate in terms of usability). The following section is to evaluate functionality of the developed function and the last section is to evaluate on the overall application. The expected result is that the average score for each criterion in the three main sections is at least 80% of the total score.

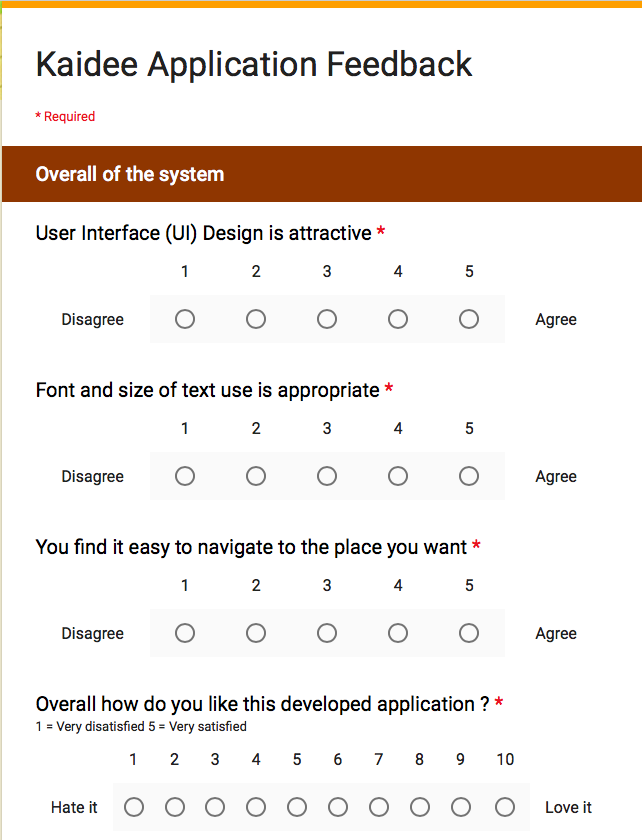
**First Section Questions**



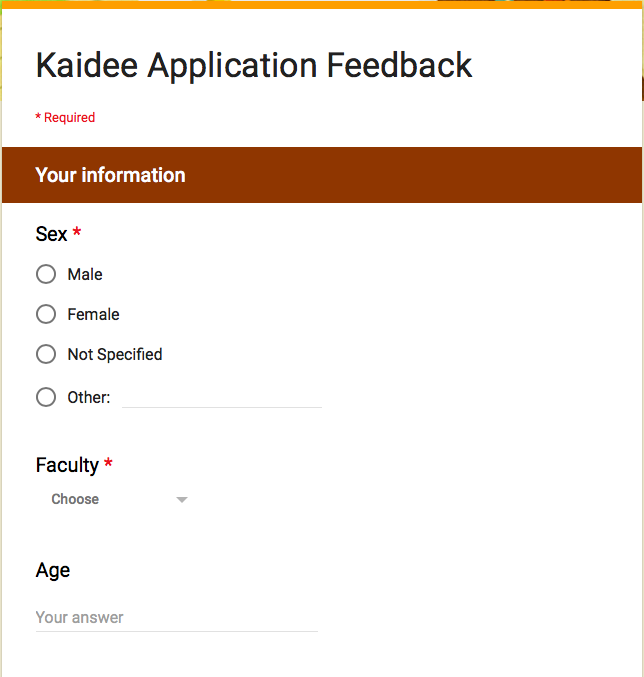
**Second Section Questions**



**Third Section Questions**

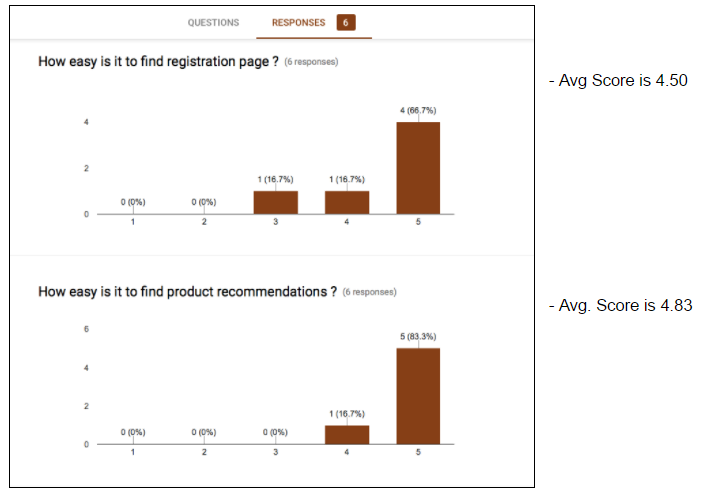


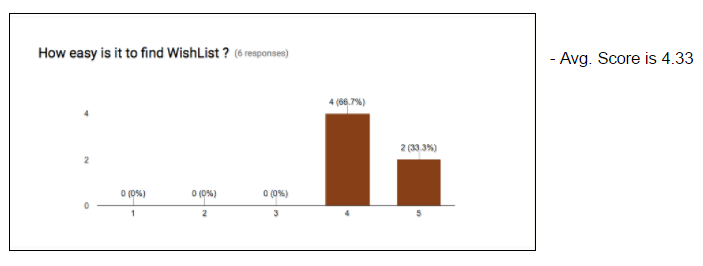
**Fourth Section Questions**

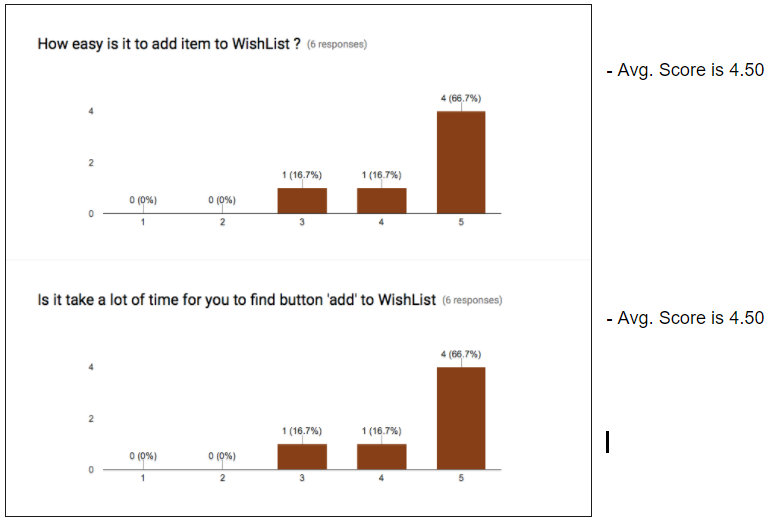


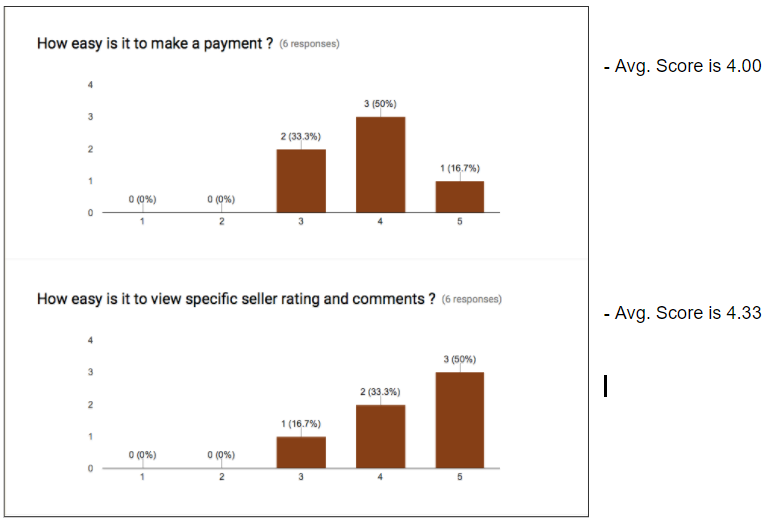
**Experimental Result**

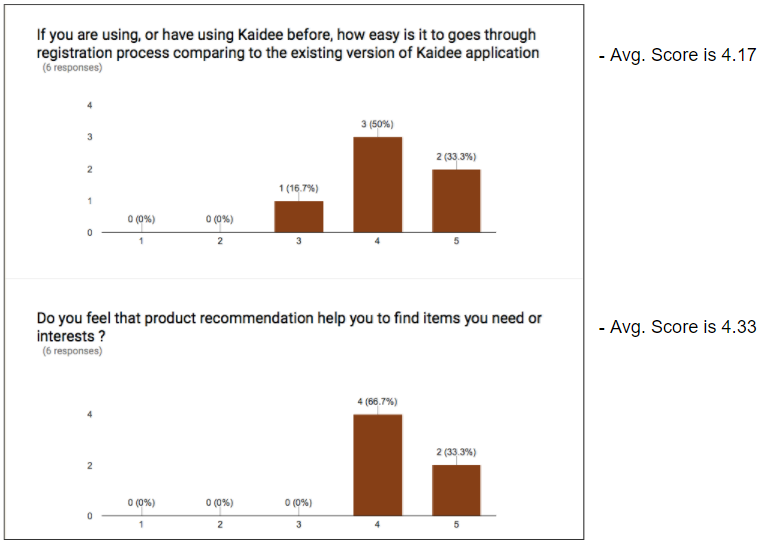
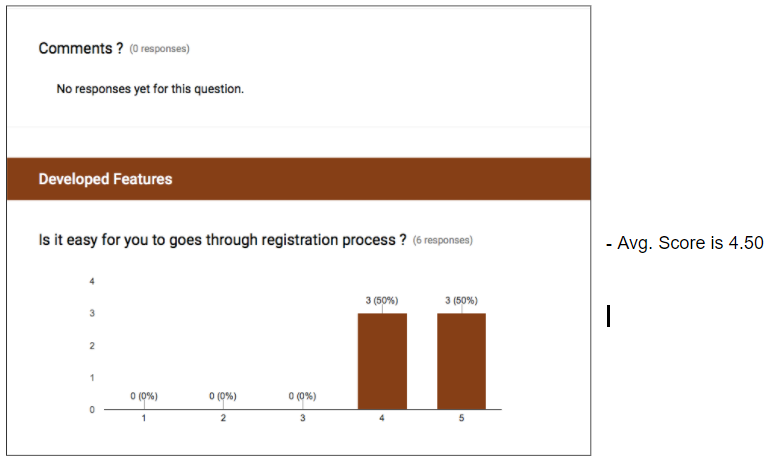
The satisfaction evaluation from those who have try to use application are shown below.

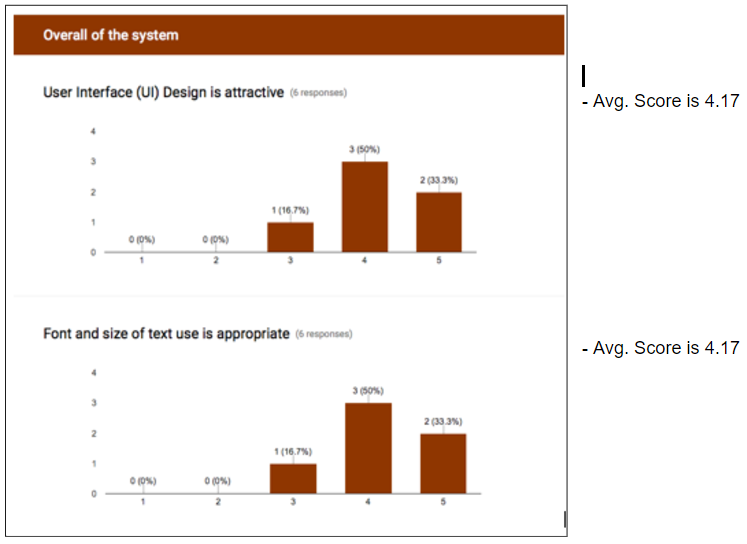
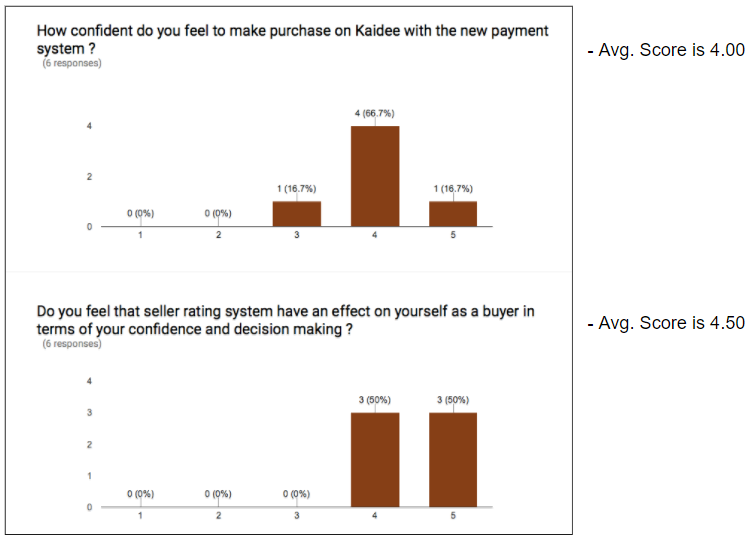
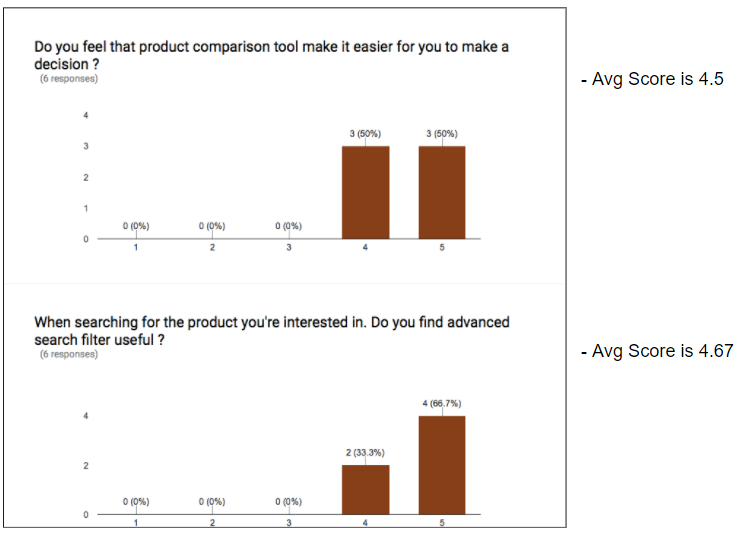


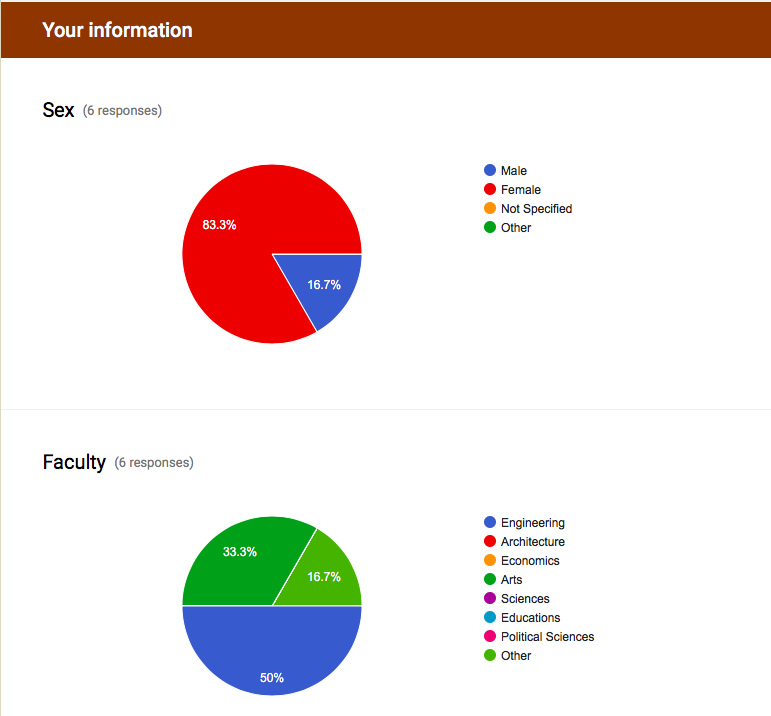


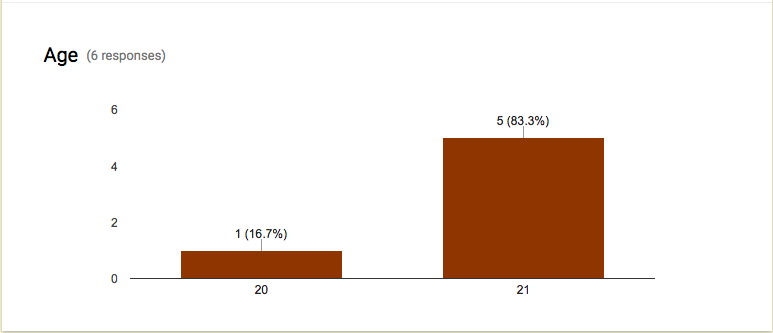












**Summary**

The result is satisfy as it conforms to the expected result stated before the test was performed that is the average score is at least 80% of the total score. Most of the participants seem to be satisfied with the new version of application developed as the response score are mostly around 4-5. Majority of the testers are female with age of 21 and are in faculty of engineering. There is no additional comments collected from the responder.

## 9.3 Act on result

According to the functional and non-functional testing, there are no deviation as the result of the test are all conformed to the requirements.

# 10. User Interface Testing

For the user interface testing, we decided to conduct the test by heuristic method which is comparing the set of principles for interface design we had agreed on during the design specification to the real application’s interface.

**10.1 Layout**

1.1 Like items are grouped together into areas with a natural intuitive flow from left to

right, and top to bottom.

1.2 The top area shows title of each page.

1.3 The middle area displays information

1.4 The bottom area contains four tab menus including home, profile, wishlist, and sell

tabs.

**Example of result :**

On profile page, there is title on the top area indicating profile page. The middle area shows profile picture, information of the user, and pending orders. The bottom area includes four main tab menus. The design conforms to the principle of layout of user interface design.

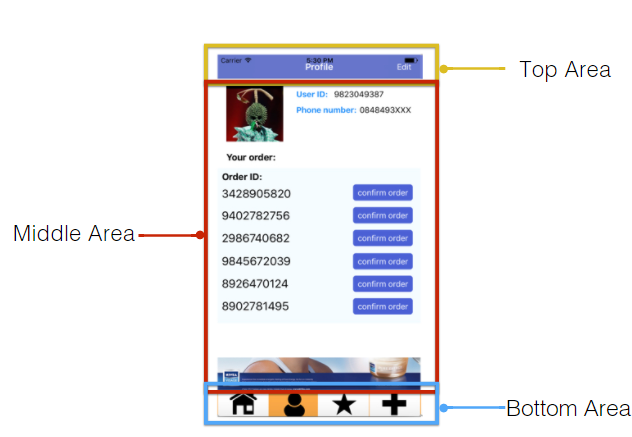


Figure 10.1 Profile Page

**10.2 Content Awareness**

2.1 There is title on every interface.

2.2 Contents displayed on each page has a clear representation and is related to the title

of each page.

2.2 Labeled buttons and labeled box input are provided.

**Example of result :**

The interface is identified by its title. There are labeled boxes to indicate what value should be input as well as an example of valid format of the input. Every button is labeled so the user knows what will happen after clicking at the button.

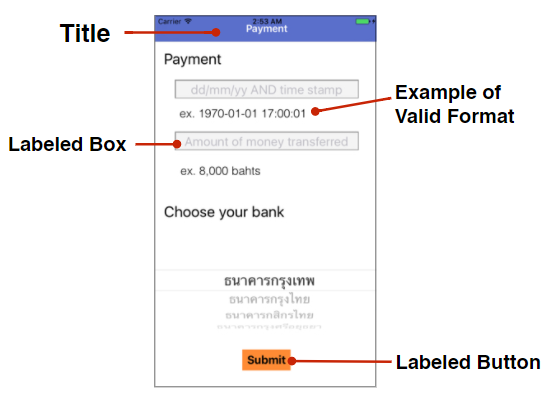


Figure 10.2 Payment Page

**10.3 Aesthetic**

3.1 All interfaces concern about the use of white space, colors, and fonts.

3.2 There are use of white space to separate contents.

3.3 Each page displays only essential information.

3.4 The theme colors include white, blue, grey, and orange which are main colors of Kaidee’s theme.

3.5 Image icons are used to make interfaces more attractive to users.

**Example of result :**

All interfaces are simple minimalist with the use of white space to separate contents. Four theme colors were applied and the font used was helvetica. There are also image icons such as product icons, tab icons, etc.

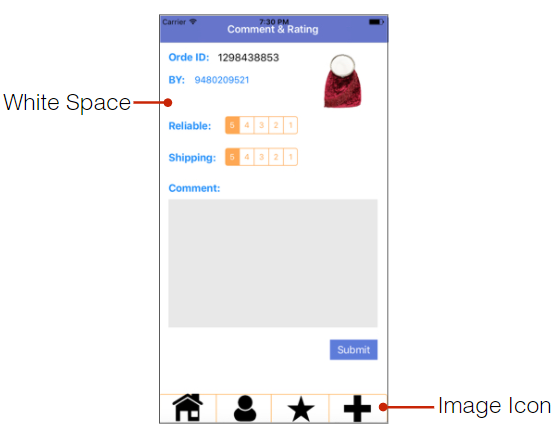


Figure 10.3 Feedback Page

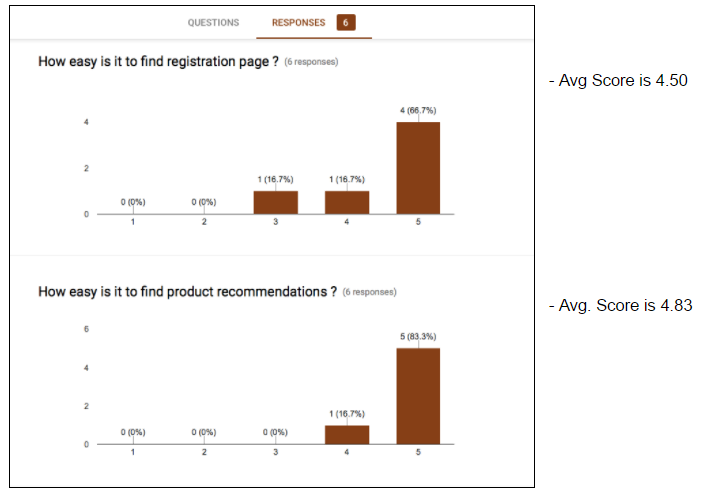
**10.4 User Experience**

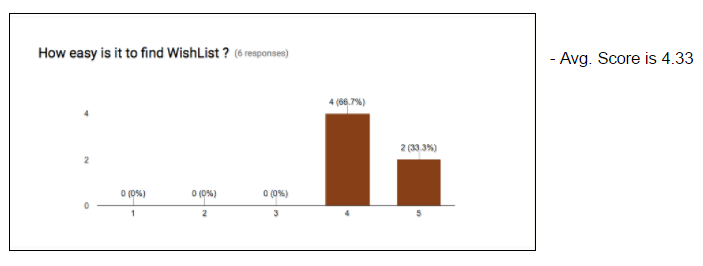
4.1 Each function is easy to be used and understood by infrequent users. This is proved by the result of formal usability testing.

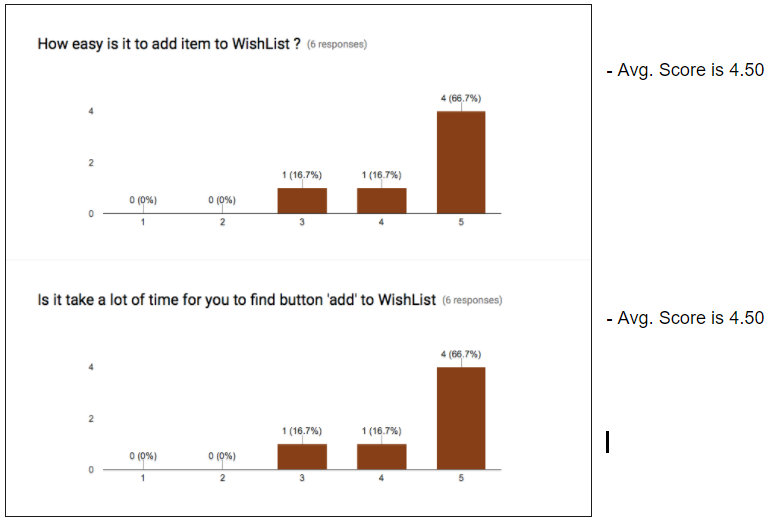
**Example of result :**

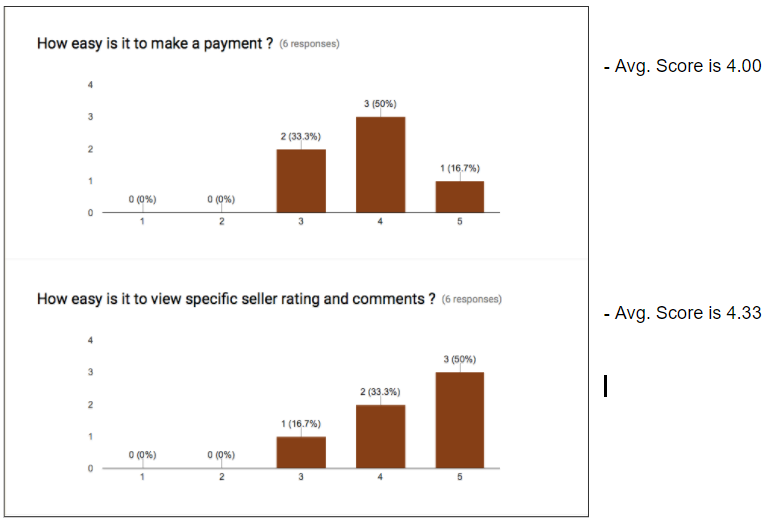
The usability testing was conducted to test for ease of learning and use. The expected result is the average score for each criterion is at least 80% of the total score. The evaluation of user’s satisfaction from those who have tried using the application are shown below.

( You can see the complete test’s activities and result on page 16.)









**10.5 Consistency**

5.1 Same template is applied for all interfaces.

**Example of result :**

The interfaces with similar function apply the same template.

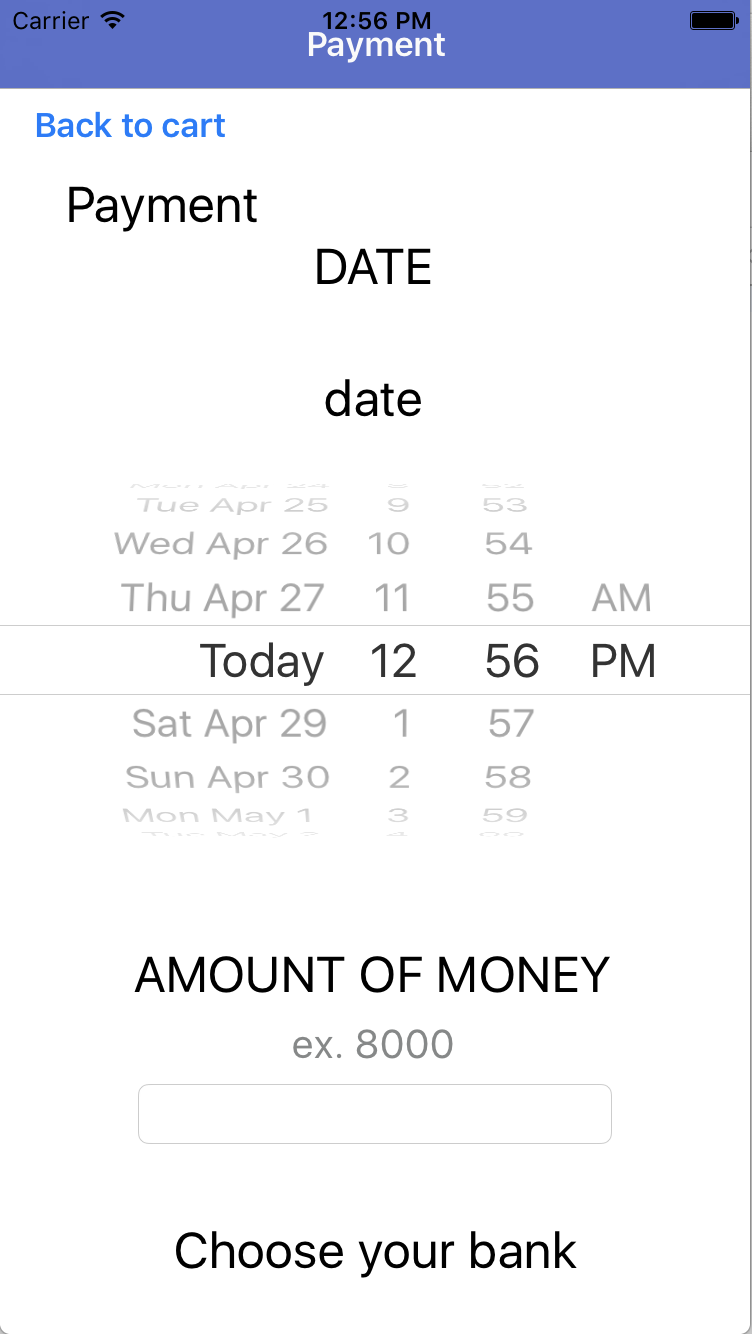
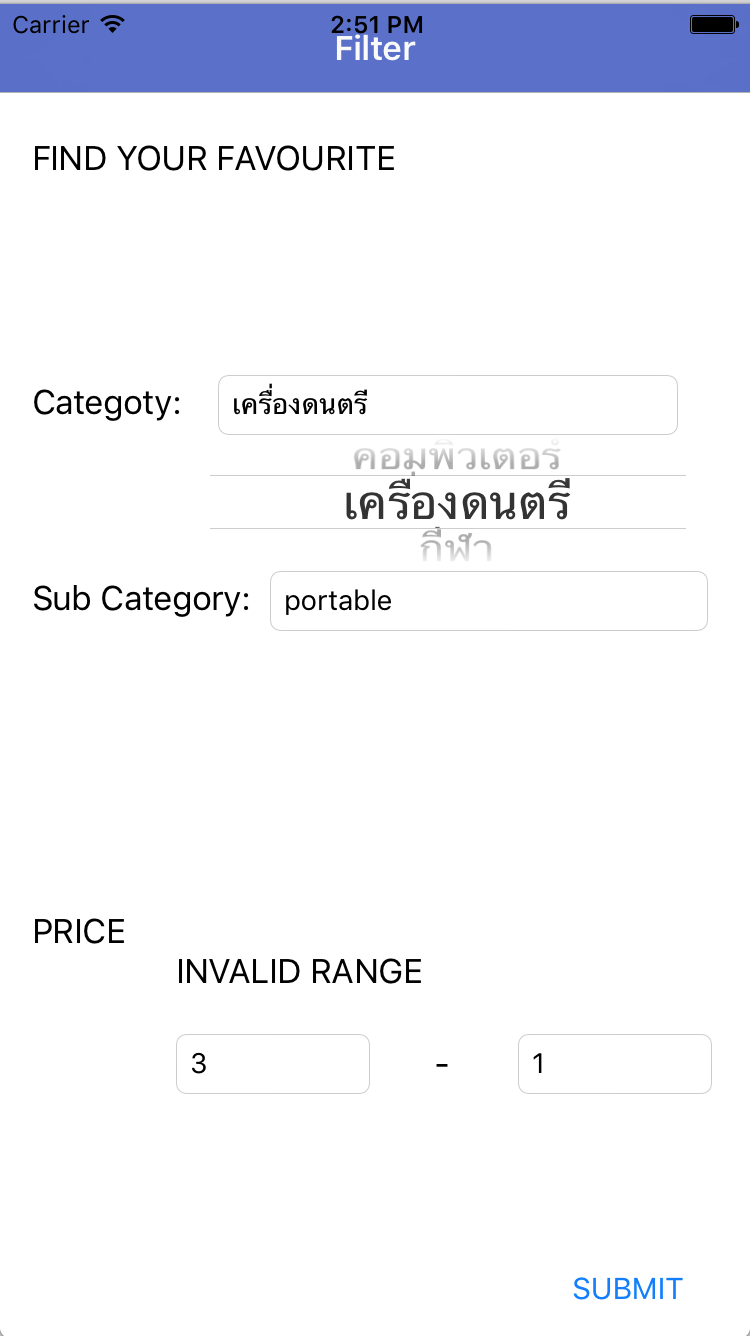


Figure 10.5 Filter Input Information page and Payment Information page

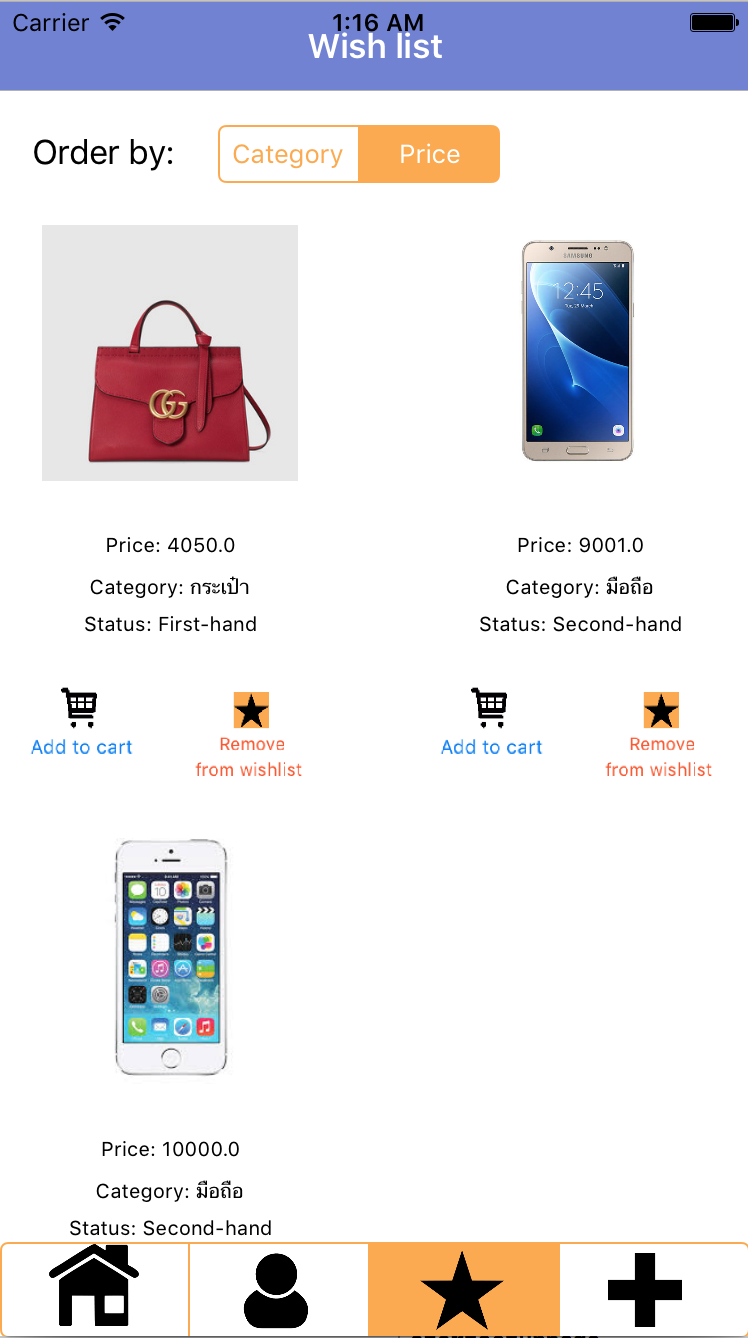
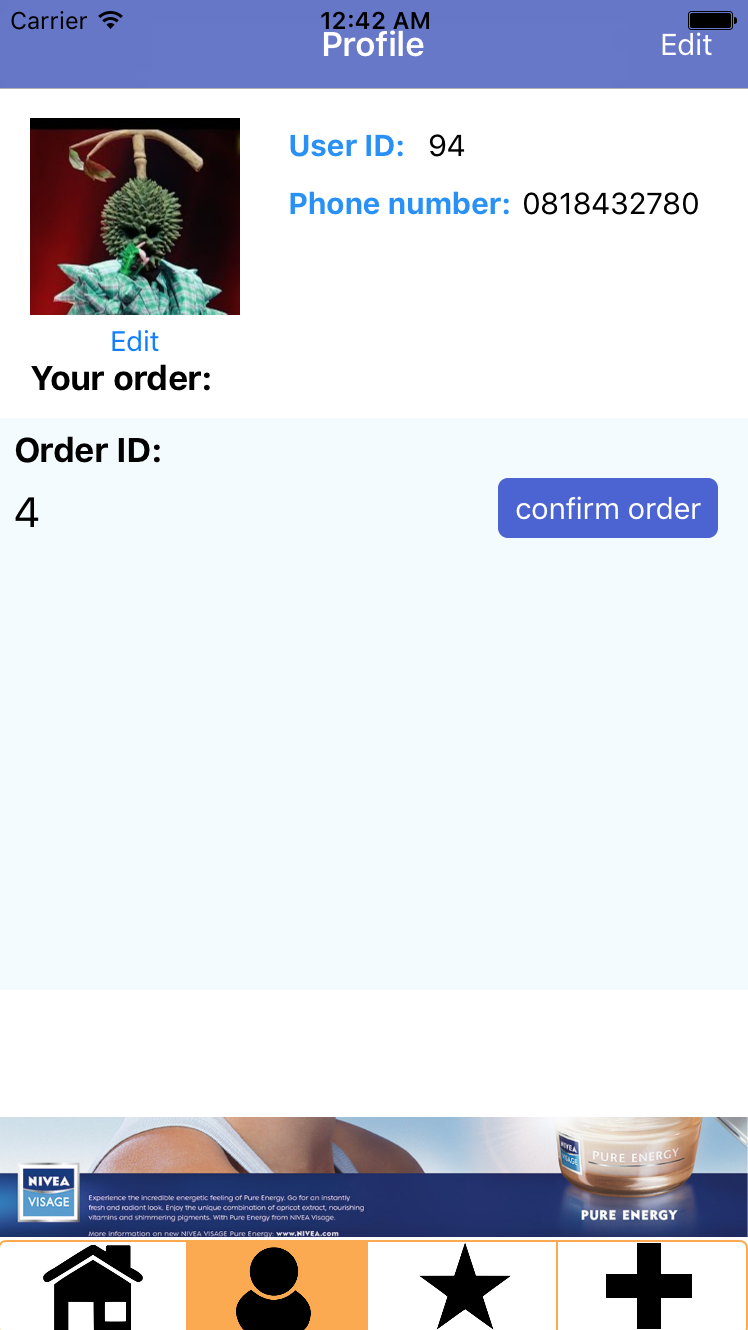


Figure 10.5 Profile Page and Wishlist Page.

**10.6 Minimal User Effort**

6.1 Most tasks can be performed within no more than three clicks from the starting menu

to the destination page.

**Example of result :**

The example below shows the process to add a product into the wishlist within 3 steps. First, clicking at the home button. Second, clicking at the product icon and the system will link to the product page. Third, clicking at the ADD TO WISHLIST button and the products will be in the wishlist.

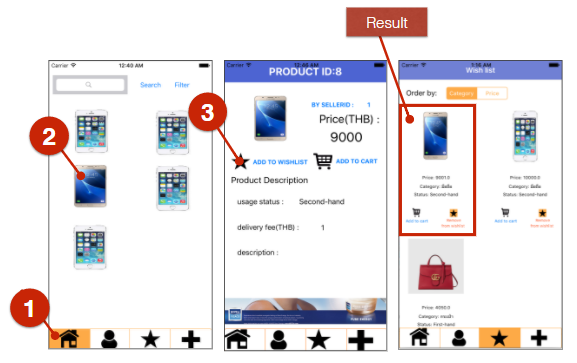


Figure 10.6 Add to wishlist process

6.2 The four main pages which are home, profile, wishlist, and sell are accessible from other pages so they help saving user’s time.

**Example of result :**

The user can click at each tab menus to immediately access to the 4 main pages.

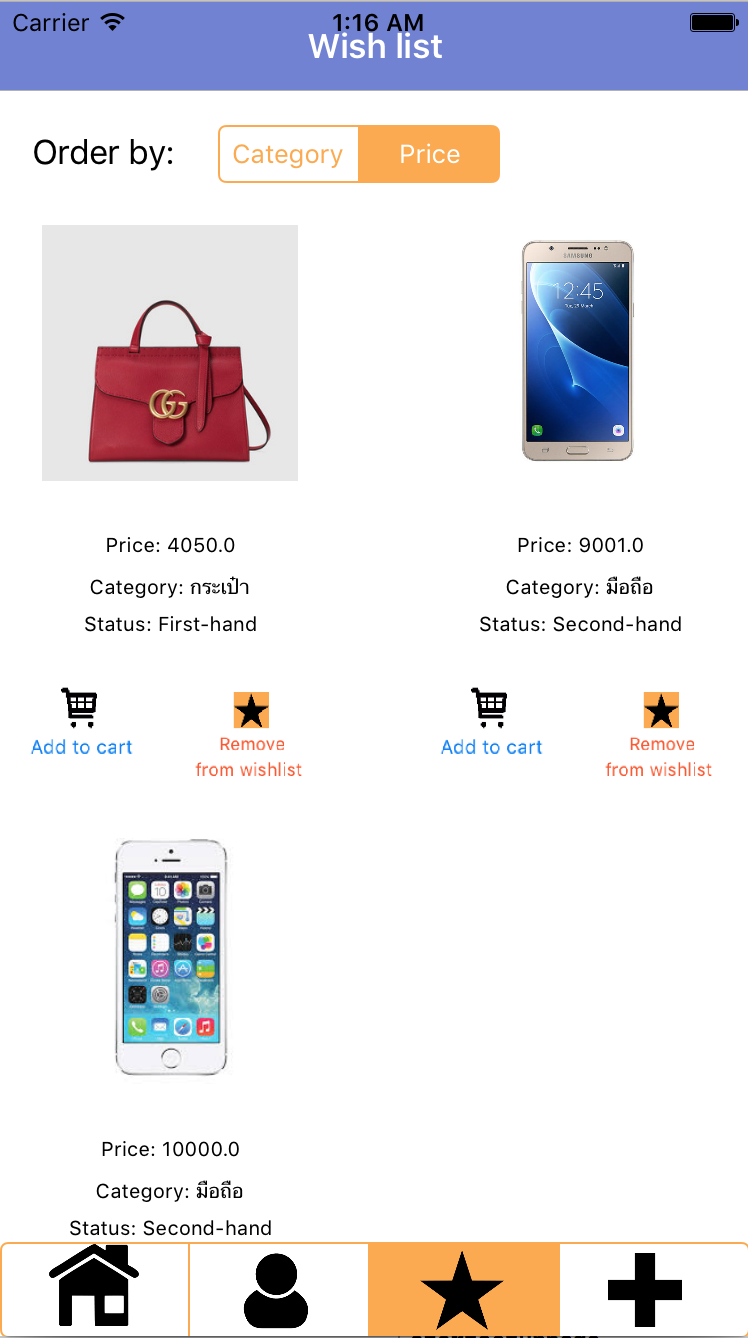
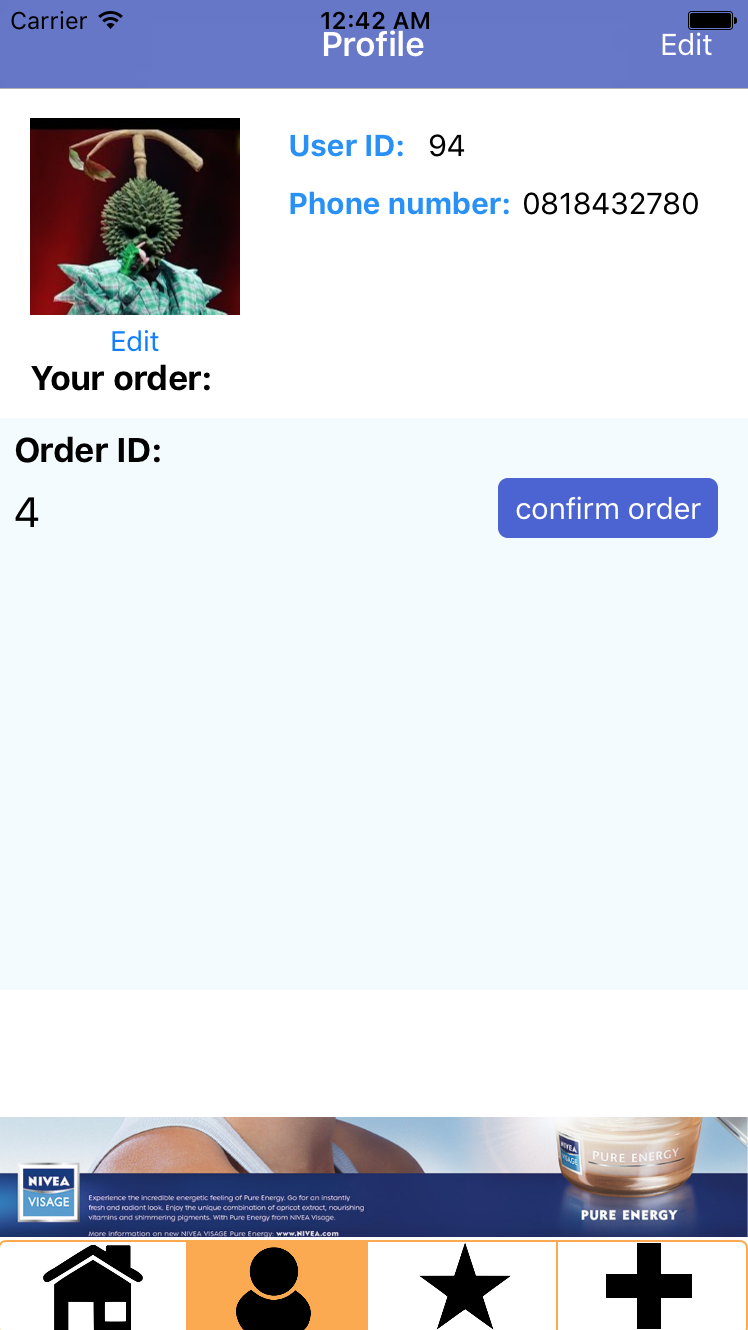
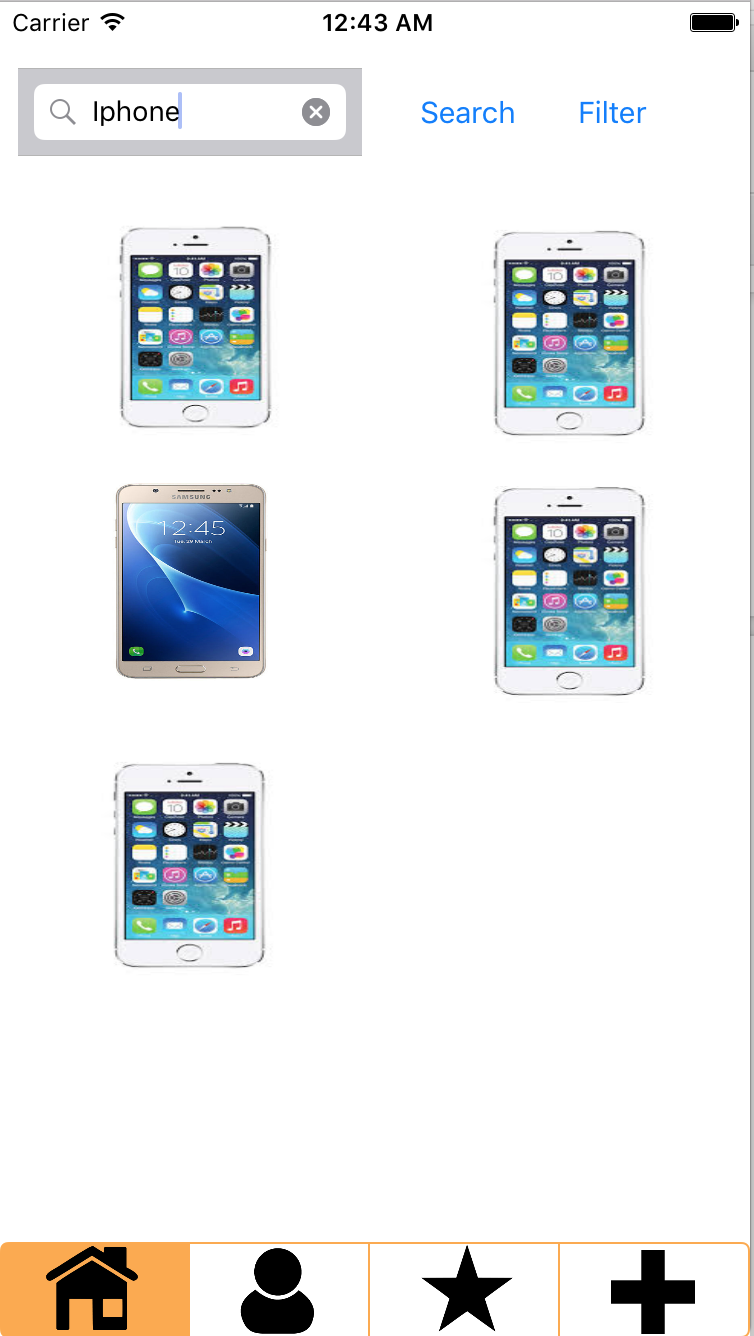


Figure 10.6.2 Home Page, Profile Page, Wishlist Page, and Product Page.

# 11. Design verification and validation

For the design verification and validation, we first balance the functional and structural models. For example, every class on a class diagram must be associated with at least one use case and vice versa.

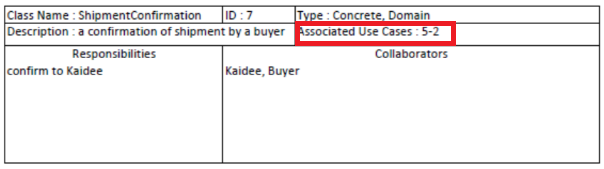


Figure 11.1 Class Description of ShipmentConfirmation class

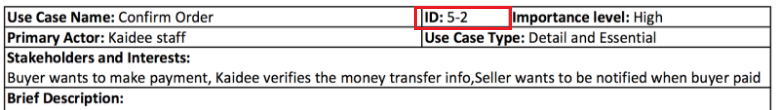


Figure 11.2 Use-case description of Confirm Order use-case

Next, we verify and validate the package diagram by making sure that all the identified packages must make sense from a problem domain point of view. For example, moneyTransfer, Order, shipmentConfirmation are all grouped together in order package since all of these class are accessed when users order a product.

At last, we verify and validate class and method design by walking through all models, ensuring that structural and functional models are consistent to each other (balancing both models) and role-play each scenarios and verify them with the constraints, contracts and, method specifications and confirm that the design still have low coupling and high cohesion properties. For example,all the precondition constraints need to be achieved before the method can execute.