

CZ2006 Software Engineering Lab #1

Requirements Elicitation

SS6

Group Members:   
Chua Kok Liang   
Grace Mok Jie Qi  
Jozua Heng Yi Jie  
Nicholas Yeo Ming Jie  
Royce Ang Jia Jie  
Shahrin Chua Zong Da

Contents

[Functional Requirements 3](#_Toc32965739)

[Non-Functional Requirements 5](#_Toc32965740)

[Data Dictionary 6](#_Toc32965741)

[UI Mockups 8](#_Toc32965742)

[Initial Use Case Model 12](#_Toc32965743)

[Use Case 1: Displaying One-for-One Deals 12](#_Toc32965744)

[Use Case 2: Selecting One-for-One Deals 13](#_Toc32965745)

[Use Case 3: Finding a Match 15](#_Toc32965746)

[Use Case 4: Using a Chatroom 17](#_Toc32965747)

[Use Case Diagram 19](#_Toc32965748)

# Functional Requirements

1. System must be hosted on an Android OS device with Android 4.0 and above
2. System must have access to an Internet Connection
3. System must have access to Location Services
4. System must communicate with the backend server
5. System requires the user to login to use the system
   1. If the user does not have an account, the system must allow the user to create an account
      1. During the registration process, if a username has already been taken by another user, a message must appear to tell the user to enter another username until a unique username is entered
      2. Users must verify their email account to successfully create the account
6. System must allow users to edit their account details such as username, email, password and profile picture
   1. If user enters an already existing username, the system must prompt the user to enter another username
7. System must be able to retrieve information regarding all available one-for-one deals
   1. Information includes Name, Date, Vendor, Location, Category, and Terms and Conditions of deals
8. System must be able to display all retrieved one-for-one deals
   1. System must be able to allow the user to filter deals based on Date, Location, and Category
   2. System must be able to allow the user to filter deals based on keywords
9. Users must be able to select a deal
   1. User must specify their preferred Location of the deal
      1. User must select at least 1 Location from a list of given location in Singapore by clicking radio buttons
   2. System must notify the user that they have been put on a waitlist
10. System must be able to show the user all the waitlists they are currently in
11. System must match two users who select the same deal with at least one identical location choice
12. System must notify both users once they have been successfully matched
    1. User A and User B must be able to see each other’s ratings
    2. Upon successfully matching 2 users, the system must create a chatroom for the two users
       1. System must allow the two users to chat with each other to set a date and time to meet up
       2. At the bottom of the chat room, there will be a button to allow users to indicate that the meet-up is successful or unsuccessful
       3. When both users click the option that the meet-up is successful, the other user’s details will be archived under “previous matches” and both users will be able to rate one another
       4. When both users click the option that the meet-up is unsuccessful, the chatroom will be deleted but they will not be able to rate one another
          1. System must prompt the user to ask if they want to be rematched by the system

# Non-Functional Requirements

|  |  |
| --- | --- |
| **Usability** | 1. The registration process should be simple enough such that it should not take more than 10 minutes for the user 2. Login for new users should be immediate after registration and email verification 3. Main menu must be accessible on all pages 4. All pages must have a function that allows users to go back to previous pages 5. The system must be user friendly for people who are colour blind    1. Colour blind users must be able to clearly read all displayed text and information 6. Help messages must be displayed in the local language according to the user’s choice |
| **Reliability** | 1. Passwords must never be visible at any time during the registration and login process and thereafter unless toggled by users 2. Upon failing to update the account details, previous account details are not changed    1. User must be informed of the failure via email 3. When a deal expires or changes, the system must reflect these changes when the page refreshes 4. The system must not match the same users again if they have already rejected each other previously for the chosen deal 5. The system must not be unavailable to users for more than 24 hours |
| **Performance** | 1. After a system reboot or update, the full system functionality must be restored within 5 minutes 2. After an account is created, the details of the account should be recorded in the Backend Server within 30 seconds 3. The system must be able to handle 1000 users at a time 4. All queries from the Backend Server should be completed within 30 seconds |
| **Supportability** | 1. The system must be accessible to all android smartphone users above Android System 4.0 (IceCreamSandwich) 2. Users can use the system only if they are using the latest version    1. Users using an outdated version will be prompted to update their system to the latest version |

# Data Dictionary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| **User** | A user refers to one who uses the application to find a match for one-for-one deals. |
| **System** | A system refers to the “1Pair” application on Android. |
| **Account** | An account refers to all the relevant information (such as profile picture, username, password and email) and an unique identification number tagged to a registered user. |
| **One-For-One deal** | A one-for-one deal refers to a deal offered by various vendors in Singapore where users can get two items for the price of one. |
| **Vendor** | A vendor refers to a shop or franchise who is offering one-for-one deal(s). |
| **Terms and Conditions** | Terms and Conditions refers to the rules provided by the vendors regarding their one-for-one deals. These Terms and Conditions are to be abided by in order for the user to claim the one-for-one deal. |
| **Deal Location** | A deal location refers to the area(s) where a one-for-one deal is available at. |
| **User Preferred Location** | A user preferred location refers to the area(s) where a user wants to meet-up at. |
| **Match** | A match refers to a pair of users that are grouped together based on their preferred location for a selected one-for-one deal. |
| **Chat Room** | A chat room refers to a platform where matched users can plan their meetup. |
| **Page Refresh** | A page refresh refers to whenever a user moves from one page to another within the application. |
| **Waitlist** | A waitlist refers to a dynamic queue where users are added to when they click the “Find a match” button for a given deal. |
| **Category** | A category refers to a classification of the deals available on the application. The four categories are Food, Entertainment, Retail and Others. |
| **Archive** | An archive refers to the chat history between matched users. A chat history is archived after both users click on the “We have met” button at the bottom of the chatroom. |
| **Ratings** | A rating refers to a score given by one user to another after meeting up based on their experience from a scale of 0 to 5 stars. |
| **User’s Overall Rating** | A user’s overall rating refers to an average rating across all the scores given by other users. |
| **Registration** | A registration refers to the process of setting up an account. |
| **Backend Server** | A backend server refers to a remote online database that stores information such as account details and deals. |
| **Query** | A query refers to a request to retrieve relevant data from the Backend server. |

# UI Mockups

Figure 1.2 Homepage

Figure 1.1 Log-in Page

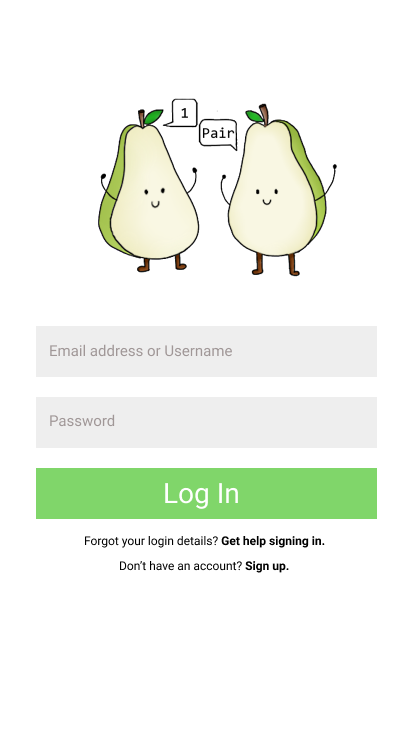
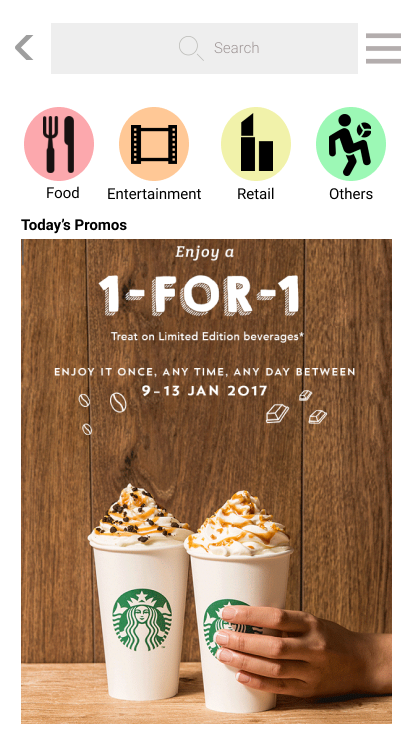
 

Figure 1.4 Filtering Deals

Figure 1.3 Browsing Deals

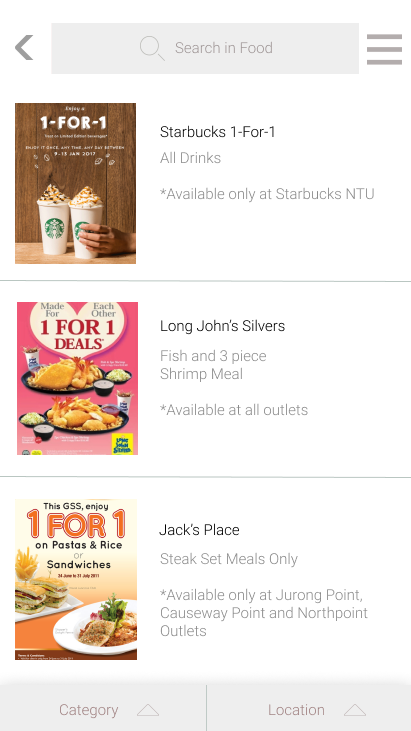
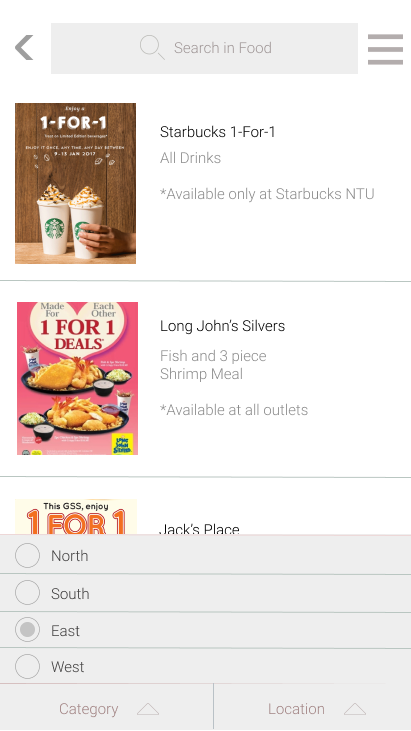
 

Figure 1.6 Deal

Figure 1.5 Menu

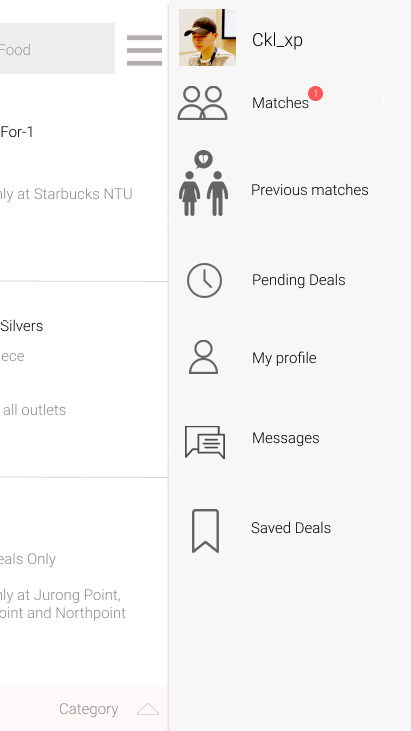
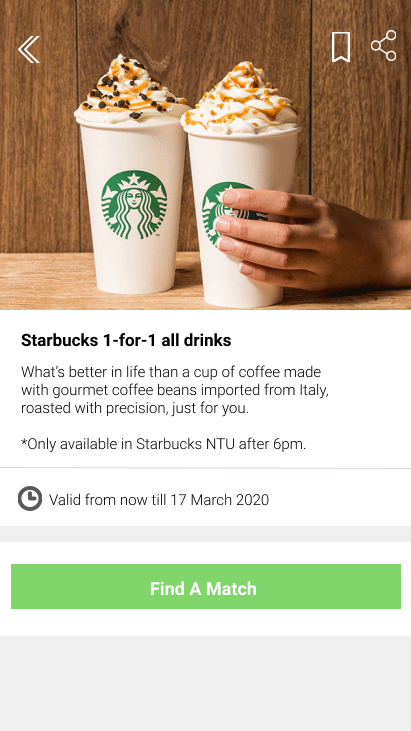
 

Figure 1.8 Found a Match

Figure 1.7 Selecting a Deal

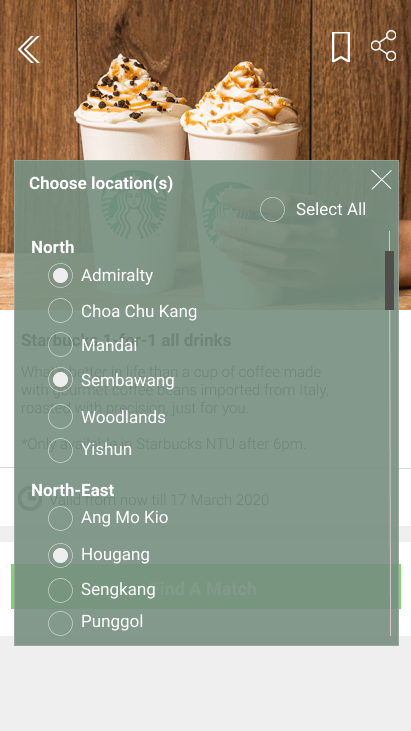
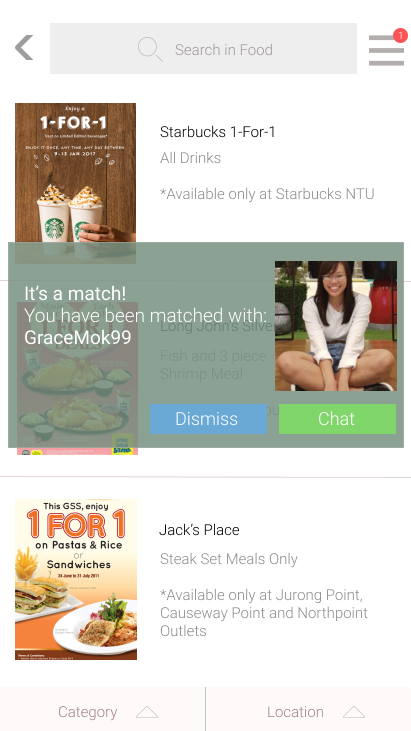
 

Figure 1.10 Ratings

Figure 1.9 Chatroom

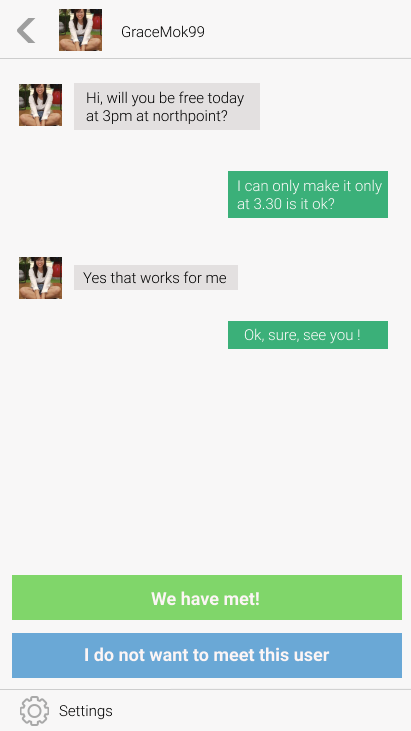
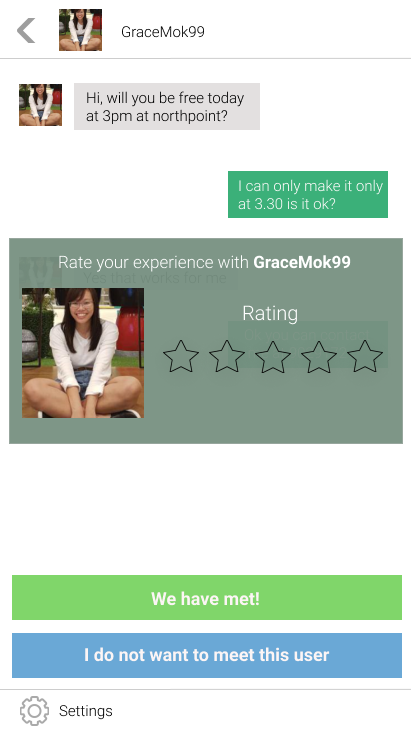
 

Figure 1.12 Pending Deals

Figure 1.11 Matches

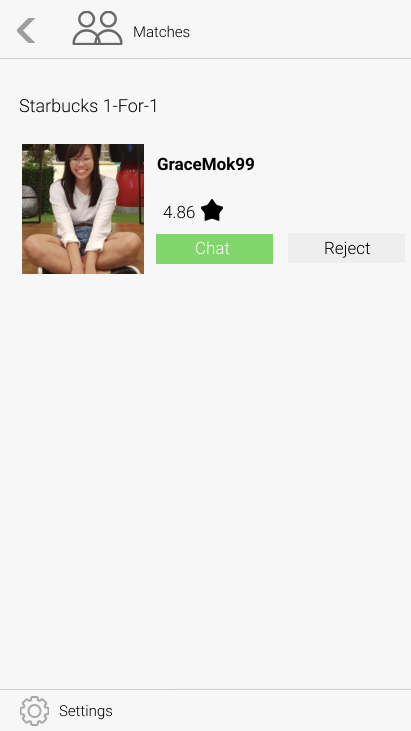
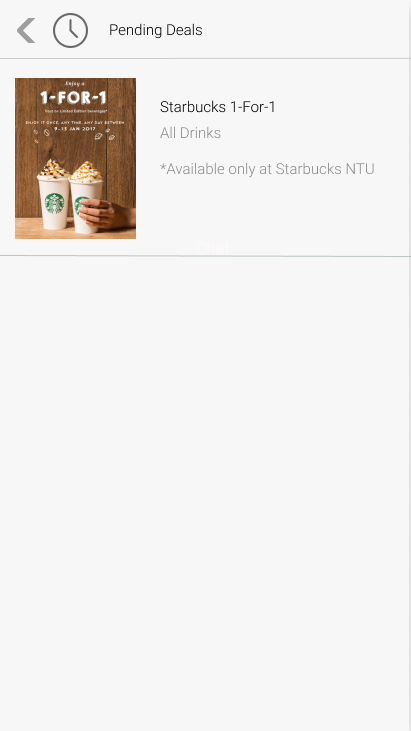
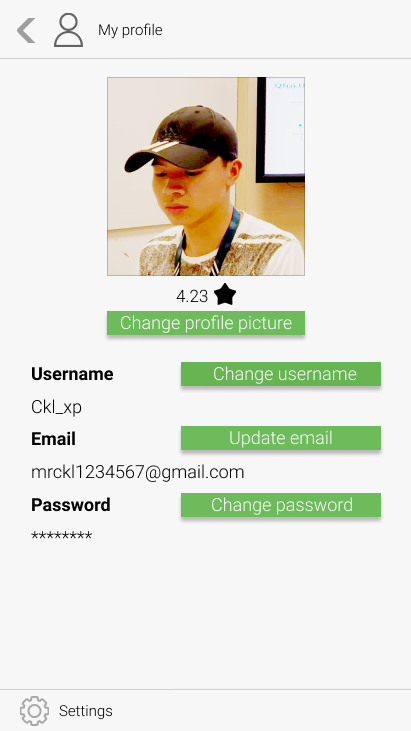
 

Figure 1.13 Profile Page



# Initial Use Case Model

## Use Case 1: Displaying One-for-One Deals

|  |  |
| --- | --- |
| Actor: | System, User, Backend Server |
| Description: | System displaying all valid one-for-one deals in the system. |
| Entry-Conditions: | 1. User must be logged in. 2. System must be connected to the backend server. |
| Exit-Conditions: | 1. All valid one-for-one deals are fully displayed. |
| Priority: | - |
| Frequency of Use: | - |
| Flow of Events: | 1. User logs in. 2. The system queries the backend server for the name, date, vendor, location, category, and terms and conditions of deals of all valid one-for-one deals. 3. The system displays all retrieved information. |
| Alternative Flows: | AF-S2: If there are no valid one-for-one deals   1. the system will display a page that says “No Deals Found”. |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | - |
| Assumptions: | The order of displayed data is in order of how they are being retrieved. |
| Notes and Issues: | - |

## Use Case 2: Selecting One-for-One Deals

|  |  |
| --- | --- |
| Actor: | System, User, Backend Server |
| Description: | User browses through a filtered list of displayed one-for-one deals based on their selected preferences and select a desired deal. |
| Entry-Conditions: | 1. User must be logged in. 2. System must be connected to the backend server. 3. Deals must be displayed in the system. |
| Exit-Conditions: | 1. User placed in waitlist of selected deals on the server |
| Priority: | - |
| Frequency of Use: | - |
| Flow of Events: | 1. User logs in. 2. Displaying One-for-One Deals (Use Case #1) has been invoked. 3. User is given the option to select filters such as name, category or location for deals that are to be displayed. 4. System displays the deals based on step 3. 5. User selects the preferred one-for-one deal. 6. System will display a list of available locations for the particular deal in Singapore in the form of radio buttons. 7. User selects the preferred location(s). 8. System acknowledges the user’s selection by adding the user on the waitlist on the server. 9. System notifies the user that the user has been put on the waitlist. 10. User dismisses the notification. |
| Alternative Flows: | AF-S6: If there are no preferred locations   1. The user exits the location selection page. 2. The system returns to step 3. |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |

## Use Case 3: Finding a Match

|  |  |
| --- | --- |
| Actor: | System, User, Backend Server |
| Description: | The system matches two users who have the same deal and with at least one identical location. |
| Entry-Conditions: | 1. User must be placed on the waitlist on the server 2. System must have at least two users on the waitlist. |
| Exit-Conditions: | 1. User has been successfully matched with another user from the waitlist. 2. Both users that are matched are removed from the waitlist. |
| Priority: | - |
| Frequency of Use: | - |
| Flow of Events: | 1. Selecting One-for-One Deals (User Case #2) has been invoked. 2. System retrieves the waitlist from the server. 3. System matches two users on the waitlist who have selected at least one identical location. 4. System removes the two users from the waitlist. 5. System allocates the two users a private chat room. 6. System notifies both users that they have been successfully matched. 7. User clicks on the “Chat” button to enter the chatroom. |
| Alternative Flows: | AF-S3: If there are no users in the waitlist to be matched.   1. The user will remain in the current waitlist. 2. System will continue to find a match until successfully executed.   AF-S6: If User clicks on “Dismiss” button   1. User stays on currently displayed page. |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |

## Use Case 4: Using a Chatroom

|  |  |
| --- | --- |
| Actor: | System, User, Backend Server |
| Description: | Allocates a private chat room for two successfully matched users to plan their meet-up. |
| Entry-Conditions: | 1. Two users have been successfully matched. |
| Exit-Conditions: | 1. When the two users have selected either the option of successful meet-up or unsuccessful meet-up. |
| Priority: | - |
| Frequency of Use: | - |
| Flow of Events: | 1. Finding a Match (Use Case #3) has been invoked. 2. System will display one user’s ratings to the other user. 3. System will display two interactive buttons: “We have met!” and “I do not want to meet this user” at the bottom of the screen. 4. Users plan their meet-up in the chatroom. 5. After a successful meet-up, both users select the “We have met!” button. 6. Rating a User (Use Case #5) will be invoked. 7. System closes the chatroom. 8. The conversation in the chatroom will be archived in the backend server. |
| Alternative Flows: | AF-S5: If a user selects “I do not want to meet this user” button   1. System will prompt the user if the user would like to be rematched with another user by clicking either of the “Yes” or “No” buttons. 2. If the user selects “Yes”, the prompt closes and Finding a Match (Use Case #3) will be invoked. 3. If the user selects “No”, the prompt closes and the user returns to Selecting a Deal (Use Case #2). 4. The chatroom will be deleted. |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |

## Use Case Diagram

