**SC2006 Software Engineering**

****

**Lab Group SSP3**

|  |  |
| --- | --- |
| Group Members | |
| Name | Matriculation No. |
| Chew Zhi Qi | U2120266D |
| Benjamin Chung Zhi Yong | U2021629B |
| Bong Jia Hui | U2121174D |
| Gan Hao Yi | U2120769F |
| Ji Hanyi | U2022188F |
| Ryu Hyunsun | U2122335L |

**Table of Contents**

[1. Introduction 4](#_Toc1090650209)

[1.1. Purpose 5](#_Toc775834026)

[1.2. Document Conventions 5](#_Toc1651569775)

[1.3. Users and Stakeholders 5](#_Toc336764248)

[1.5. Product Scope 5](#_Toc677689337)

[2. Overall Description 6](#_Toc623285434)

[2.1. Product Perspective 7](#_Toc1770820364)

[2.2. Product Functions 7](#_Toc821410908)

[2.3. User Classes and Characteristics 7](#_Toc308688818)

[2.4. Operating Environment 7](#_Toc892186842)

[2.5. Design and Implementation Constraints 7](#_Toc2051955543)

[2.6. User Documentation 8](#_Toc1910781691)

[2.7. Assumptions and Dependencies 8](#_Toc564343637)

[3. External Interface Requirements 8](#_Toc358646428)

[3.1 Design Pattern 9](#_Toc1717567245)

[3.2 User Interfaces 9](#_Toc2102803509)

[3.3 Hardware Interfaces 17](#_Toc1457845220)

[3.4 Software Interfaces 17](#_Toc934972339)

[3.5 Communications Interfaces 18](#_Toc1232766452)

[4. System Features 18](#_Toc537936448)

[4.1. Login 19](#_Toc474197632)

[4.2. Register 19](#_Toc863359172)

[4.3. Reset / Change Password 19](#_Toc384326084)

[4.4. Uploading Profile Picture 20](#_Toc768900704)

[4.5. Find a Location 20](#_Toc1988781942)

[4.6. Get Direction 21](#_Toc439829837)

[4.7. View Rating 21](#_Toc828838737)

[4.8. Post a Comment 21](#_Toc596753891)

[4.9. View / Update Favourites List (Add / Remove) 21](#_Toc1160109106)

[5. Other Nonfunctional Requirements 22](#_Toc419784520)

[5.1 Performance Requirements 23](#_Toc1752445531)

[5.2 Safety Requirements 23](#_Toc1503379170)

[5.3 Security Requirements 23](#_Toc10531782)

[5.4 Software Quality Attributes 23](#_Toc1898452650)

[5.4.1 Reliability 23](#_Toc1362124855)

[5.4.2 Maintainability 23](#_Toc661362057)

[5.4.3 Reusability 23](#_Toc1919990555)

[5.4.4 Portability 23](#_Toc150074552)

[5.4.5 Supportability 23](#_Toc851576646)

[5.4.6 Usability 23](#_Toc114758086)

[5.5 Business Rules 24](#_Toc2081268714)

[5.6 Data Dictionary 24](#_Toc1417813385)

[Appendix A. Use Case Descriptions 25](#_Toc606319188)

[Appendix B: Analysis Models 44](#_Toc219787549)

[Use Case Diagram 45](#_Toc305518138)

[Dialog Map 46](#_Toc1143429562)

[Sequence Diagrams 46](#_Toc1138115950)

[System Architecture 50](#_Toc1164722385)

[Class Diagram/Conceptual Model 51](#_Toc1260320222)

[Appendix C: Test Cases 51](#_Toc1734363780)

[Black Box Testing 52](#_Toc476907566)

[Checking if the username is changed correctly. 55](#_Toc2099054437)

[Username 55](#_Toc1889902198)

[Expected Output 55](#_Toc1394813869)

[Result 55](#_Toc551236899)

[No entry 55](#_Toc1939850301)

[“Username field is empty” 55](#_Toc1660226143)

[Pass 55](#_Toc1726960618)

[Cownic 55](#_Toc1282159050)

[“Username updated”, reflected in the Home page. 55](#_Toc282146372)

[Username is updated. Pass 55](#_Toc1016862592)

[Test case 7: Comment word limit. Comments should not be empty and cannot exceed 3000 characters. 56](#_Toc923106110)

[White Box Testing 56](#_Toc1900776888)

# Introduction

## Purpose

Our mobile application POI.sg enables the Singapore public, especially those aged between 18-59 who wants to adopt a healthy lifestyle (defined as target users) to:

1. Find exercising locations and restaurants that sell healthy food in their vicinity based on device’s GPS location.
2. Share and view others’ comments and ratings on different exercising locations/restaurants, and make informed decisions based on distance and others’ reviews

## Document Conventions

|  |  |  |  |
| --- | --- | --- | --- |
| Conventions | Font Size | Font Type | Font Style |
| Headings | 18 | Times New Roman | Bold |
| Sub-Headings (H2) | 14 | Times New Roman | Bold |
| Sub-Headings (H3) | 11 | Times New Roman | Italic |
| Normal Text | 11 | Times New Roman | None |
| Important Terms | 11 | Times New Roman | Bold |

## 1.3. Users and Stakeholders

The stakeholders of this project consist of restaurants, sports venues, Android mobile users and the POI.sg software team. The information about restaurants and sports venues will be collected through the dataset retrieved by Data.gov.sg. Android mobile users will utilize the application to search for restaurants that sell healthy food or various sports venues in their vicinity or specific locations of interest. In addition, users will be able to share and view others’ reviews and ratings and make faster and better decisions.

**1.4. Intended Audience**

The software requirement document is intended for developers, users and testers. Developers can review the application to improve the application capability with the help of this document. They can modify the features or provide updates to the application easily by looking at the overall description, system features and non- functional requirements. This allows them to have a better understanding of the application and cater to the needs of the users. New features can be implemented more efficiently by analyzing the interfaces and analysis models. Testers can use this document to provide periodic maintenance and testing. The test cases in the document can make debugging more manageable. Design and implementation constraints can be further eliminated. Safety and security requirements can be strengthened as well.

## 1.5. Product Scope

The application is connected to an online cloud database which contains data about various exercising locations and restaurants that sell healthy food around Singapore. Users can browse through the application to view locations in their vicinity or search for facilities / restaurants near their specific locations of interest. It is also an interactive platform where target users can communicate their opinions regarding the different exercising locations and restaurants. Users can also login to bookmark locations as their favorites.

# Overall Description

## Product Perspective

POI.sg is a mobile application that will be used on smartphones with Android operating system. The idea for this application comes from wanting to provide especially those aged 18-59 who wants to adopt a healthy lifestyle to find preferred exercising locations, to comment on and rate different exercising locations and to make exercise-related decisions based on distance.

## Product Functions

The system must provide, at a minimum, the following functions in accordance with the other requirements described within this SRS document.

* Register and login as unique user of the application
* Resetting of password if the user forgets his/her password
* Various settings: Reset Username, Reset Password, Upload Profile Picture, Modify personal info (e.g. age, weight, height, etc.)
* Find exercise locations and healthy eateries near the user.
* Leave and view Comments under a certain Location
* Sort locations by alphabetical order or by ratings.
* Save frequent/ preferred locations as Favorites.

## User Classes and Characteristics

We generate 4 user classes.

1. People who exercise/eat healthily regularly.
2. People who are looking to start exercising/eating healthily.
3. People who are looking for a nearby location to exercise and eat healthily conveniently.
4. People who want to exercise / eat healthier but lack/don’t know the proper ways and methods.

There is 1 user class for the application:

Registered User:

-Able to login

-Able to reset password

-Able to view healthy food and exercise locations

-Able to view ratings and descriptions for location

-Able to view account information

-Able to view and leave comments

-Able to change profile picture

-Able to change username

-Able to change password

-Able to delete account

-Able to sign out

## Operating Environment

The application will be running on Android Studio. It will be using Firebase to store data such as users information and description of courses.

## Design and Implementation Constraints

The application is currently available only in English.

The application depends on the datasets stored on the cloud. There are no live updates to the dataset.

## User Documentation

The application is simple to use. The flow of the application is highly straightforward. Users should be able to navigate it with ease. A video on the application usage is available in Appendix D and users may refer to it if needed.

## Assumptions and Dependencies

Users should have an Android device capable of running the application.

Users should have Internet access to use the application.

Users allow the application to access device location.

# External Interface Requirements

## 3.1 Design Patterns

**3.1.1 Facade Design Pattern**

We implemented a bottom navigation bar which allows users to navigate to different pages via a simple user interface. The NavigatorBar is a facade class which serves to aggregate the different Page classes of our application. This hides the complexities of the system and reduces dependencies on the classes which has unique functionalities. It also decouples the client from a complex system and users are able to easily access another page from any pages he/she is currently on.

**3.1.2 Singleton Pattern**

For each login, the user object is only created once. The user’s details is then passed to other classes and there is no need to re-instantiate the user object to get the attributes.

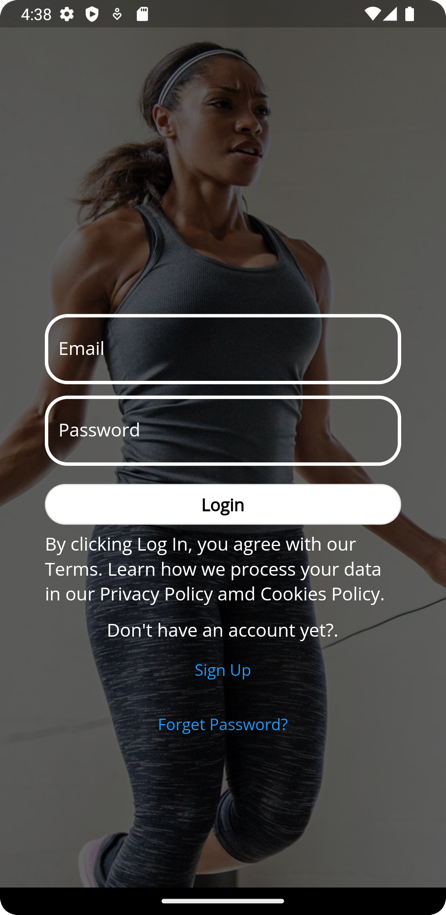
**3.1.3 Builder Pattern**

This is implemented in the Comment Section where each comment is a class. Each comment is built with the user’s username, comment and the date and time when each comment is posted. The final Comment section comprises of a list of comments.

## 3.2 User Interfaces

POI.sg is a highly interactive application and is simple to use. It has features such as buttons, icons, pop-up messages and text boxes. Every page is clear and consistent for users. Error messages will notify the user when there is an invalid entry and will prompt the user to retry. These interactive features improve the user's experience.

Every function in the application is activated based on the user's input. The screen layouts in the application that have a user interface include the Start Screen, Sign-Up Screen, Main Page Screen, Search Location Screen. Users can navigate the application based on their actions and interactions with the application.



Screen 1: Start Screen

The Start Screen has two buttons “Login” and “Sign Up”. New users will click on “Sign Up” and will be directed to the Sign Up Screen (Screen 2). Existing users will need to fill the input fields of Email and Password with their valid account data before clicking on “Login”, after which the user will be directed to the Main Page (Screen 3).

Graphical user interface, application

Description automatically generatedGraphical user interface, application

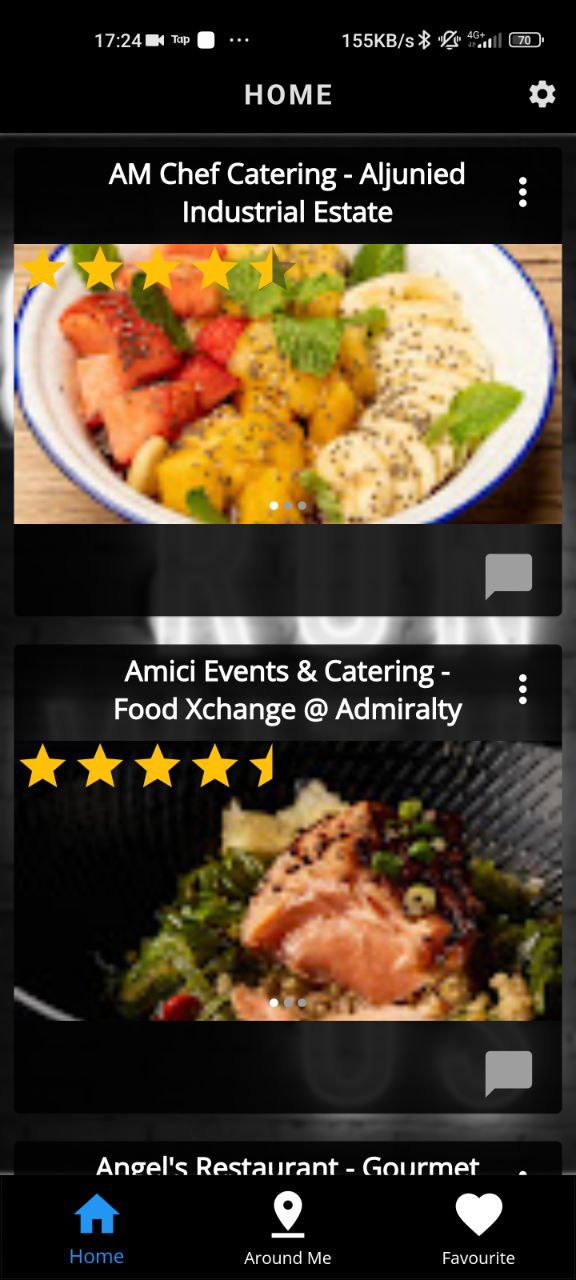
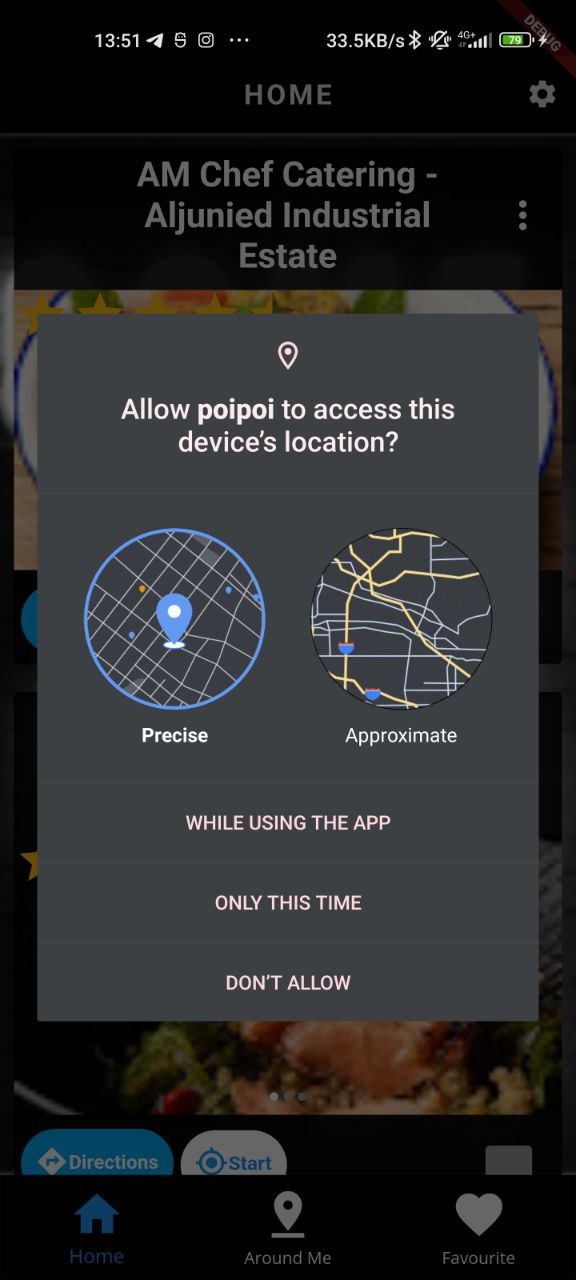
Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

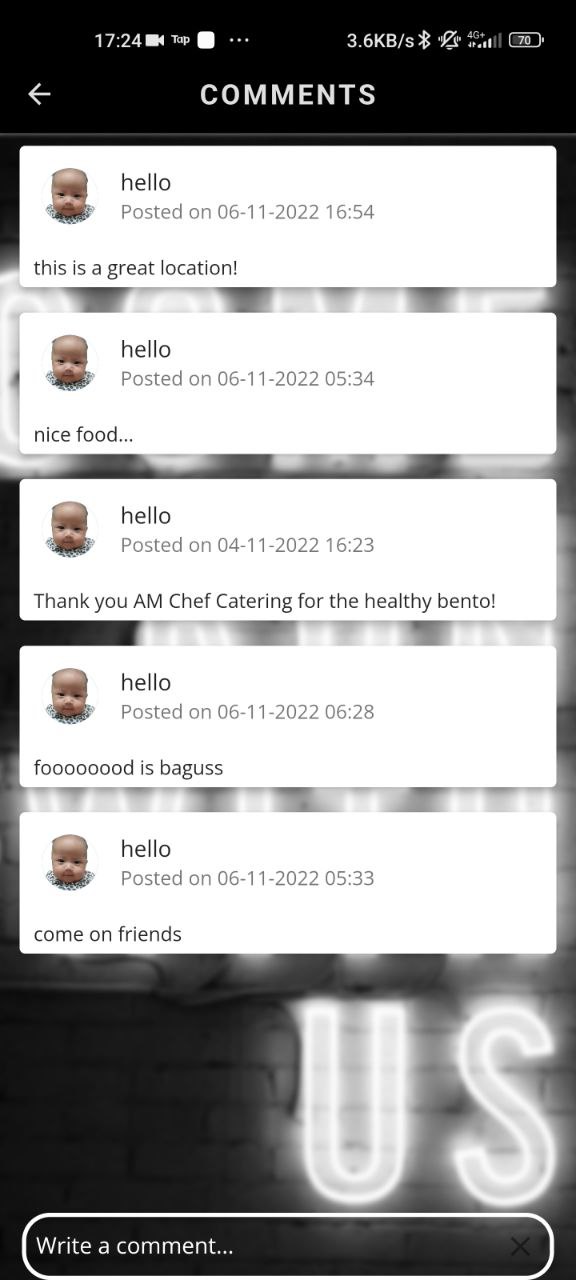
Screen 2: Sign Up Screen(s)

The Sign Up Screen(s) have input fields for Full Name, Email, Date of Birth, Gender and Password. The user will enter this information and the application will register the user. After the user input the valid data, the user clicks on “Continue” button. Once all the information has been submitted, the application will navigate the user to the Main Page Screen (Screen 3).



Screen 3: Main Page Screen

Just before entering the main screen, a pop up will appear to ask the user to allow access to device’s location. Then, the Main Screen shows the top ranked eateries/locations based on alphabetical order or in order of average rating. The navigation bar at the bottom of the application consists of the commonly used icons on the buttons for better users’ comprehension. The navigation bar contains the buttons to navigate to the Settings Screen, Register Screen and “Forgot Password” Screen. Users can navigate to other screens easily.



Screen 4: Comments Screen

The Comment Section shows the list of past comments for the specific location . User can also leave new comments.

A picture containing text, indoor

Description automatically generated

Screen 5: Favourites Screen

The Favourites Screen shows the favourite locations and eateries the user have previously saved and marked. The navigation bar at the bottom of the application consists of the commonly used icons on the buttons for better users’ comprehension. The navigation bar contains the buttons to navigate to the Settings Screen, Register Screen and “Forgot Password” Screen. Users can navigate to other screens easily.



Screen 6: AroundMe Screen

The AroundMe Screen shows all the locations and eateries within a radius of 4 km around the user’s current location. Users can select the ‘+’ icon on the left to show a list of the locations that match the markers on the screen. Users can select the ‘Ø’ icon on the left to hide the list of locations that matches the markers on the screen.

The right button switches the markers on the map and the list of locations between the exercise locations and the food locations.

Graphical user interface, text, application

Description automatically generated

Screen 7: Settings Screen

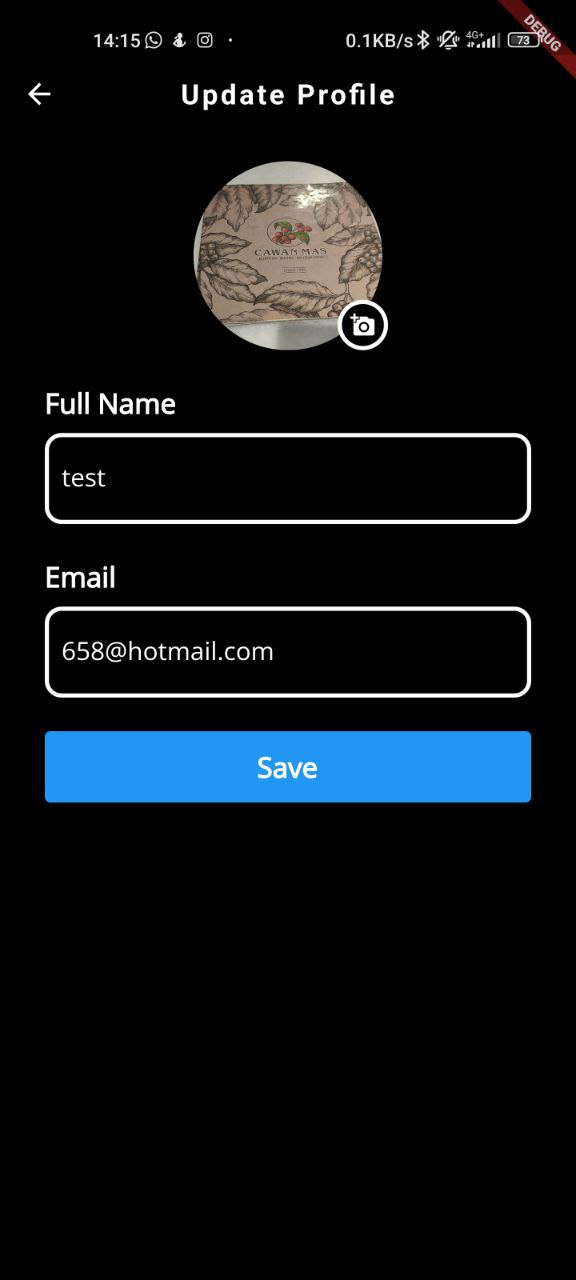
The Settings Screen includes buttons such as Upload Profile Picture, Change Password, Change Username and Delete Account. Users are also able to log out on this screen. The user interface layout is shown below.

Graphical user interface, application

Description automatically generated

Screen 8: Change Password Screen

The Change Password Screen allows the users to change their current password. The link will then be sent to their email to change their password.



Screen 9: Update Profile Screen

The Update Profile screen allows the users to change their name, profile picture and their email which is their user name. After the user clicks on save, the user’s name, username and profile picture would be updated if there were any changes made.

## 3.3 Hardware Interfaces

The hardware interface of the application is heavily dependent on the touch screen of a smartphone. The touch screen will detect the user's movement and decide the next course of action. The movements include tapping, swiping, and scrolling. It also includes typing inputs on a keypad. The application requires capacitive touch screen sensors. There are also server-side components that must be executed on server-class computers/cloud for the application features. Client-side components must execute on Android devices.

## 3.4 Software Interfaces

The application uses Firebase as the database for storing the user data.

Firebase Cloud Storage is utilized to store the user profile pictures.

Firebase Real-time Storage and Authenticator are used to store the user login database.

Firebase - FireStore Storage stores the module list, threads and comments.

The current version of the application is 1.0.

The libraries and dependencies used include:

* firebase-auth: 21.0.1
* firebase-firestore:24.0.1
* firebase-database:20.0.3
* firebase-storage:20.0.0

## 3.5 Communications Interfaces

The transfer of data between the application and database utilizes Firebase API. This ensures that the updates are instant and various users can communicate in real-time.

# System Features

## Login

* + 1. *Description and Priority*
       1. Users should be able to login and logout as they wish. Priority is high as users are unable to use the application if they are not logged in.
    2. *Stimulus/Response Sequences*

1. User will enter their login credentials (EMAIL & PASSWORD).

2. User clicks on the ‘LOGIN’ button.

3. System will validate the credentials with the database.

4. User is logged in if the credentials are valid.

5. If login credentials are valid, the user will be directed to the main page.

* + 1. *Functional Requirements*
       1. Database must be able to return the correct data when required and determine if a login is successful or unsuccessful.
       2. User must be able to choose to register an account if the account does not exist.
       3. Database must be able to validate the credentials when needed to.

## Register

* + 1. *Description and Priority*
       1. New users must sign up for a new account before being able to login. Priority is high as users would be unable to login without an account.
    2. *Stimulus/Response Sequences*

1. User clicks on the ‘REGISTER’ button.

2. User keys in full name, email, date of birth, gender, mobile number,

password, confirm password.

3. User clicks on the ‘Sign up’ button.

4. System will check whether the conditions are met to create a new user

5. System will create a new user in the database

6. User is then logged in

* + 1. *Functional Requirements*
       1. Email provided by the user must be unique and valid.
       2. Password provided by the user must be unique and meet the following conditions:
          1. Should contain at least one uppercase letter.
          2. Should contain at least one lowercase letter
          3. Should contain at least one number.
          4. Should contain at least one special character.
          5. Database must be able to store the new data into the existing database.
          6. Hashing will be done to ensure that users’ privacy is not compromised.

## Reset / Change Password

* + 1. *Description and Priority* 
       1. Existing users should be able to reset/change their passwords when they want to change them or forget them. Priority is medium as users do not often reset their passwords.
    2. *Stimulus/Response Sequences*

1. User clicks on the ‘FORGOT PASSWORD?’ button. / User clicks on ‘Change Password’ button on Settings Page.

2. User keys in the email that they use to register with the application

3. System will check if the email is registered with the application

4. System will then send a new password to the User’s email

* + 1. *Functional Requirements*
       1. Database will be able to send a “Reset Password” email to the user’s email address when requested and the user will be able to reset their password.
       2. Database will be able to update the new password that the user provided, which must satisfy the same password requirements.

## Uploading Profile Picture

* + 1. *Description and Priority*
       1. Users should be able to either set/change their profile picture. Priority is medium as users would not be changing their profile picture often
    2. *Stimulus/Response Sequences*

1. User clicks on the ‘UPLOAD PROFILE PICTURE’ button

2. User clicks on the ‘CHOOSE PHOTO’ button

3. User chooses a picture from the downloads file on their phone

4. User click on the ‘UPLOAD’ button

* + 1. *Functional Requirements*
       1. Database must be able to retrieve the correct profile picture for each unique user and display it on the user profile page.
       2. Database must be able to update the saved profile picture for each user when required.
       3. The picture provided by the user must be a valid type for the picture to be accepted.

## Find a Location

* + 1. *Description and Priority*
       1. Users should be able to browse through or search for locations (restaurants that sell healthy food/sports venues) posted by the civil authorities. Priority is high as it is one of the key functions of the application.
    2. *Stimulus/Response Sequences*

1. Users can browse through the locations near their vicinity displayed on the app

2. Users can select the location displayed then details, comments and ratings of the specific activity/location will be loaded.

* + 1. *Functional Requirements*
       1. The system must be able to retrieve the list of locations from the dataset near the user’s current location and display it on the map.
          1. Once the user clicks on the specific location on the map, information about the location retrieved from the dataset must be shown.

## Get Direction

* + 1. *Description and Priority*
       1. Users will be able to get the shortest route to the specific location. Priority is high as it is one of the key functions of the application.
    2. *Stimulus/Response Sequences*

1. Users click on the existing location/restaurants on the map.

2. Users will be directed to Google Maps screen

* + 1. *Functional Requirements*
       1. Once the user clicks on directions icon on a specific location, the application will direct to Google Maps screen.

## View Rating

* + 1. *Description and Priority*
       1. Users will be able to view ratings of an exercise location / healthy F&B establishment. Priority is high as it is one of the key functions of the application.
    2. *Stimulus/Response Sequences*

1. Users can view ratings under any exercise location/healthy F&B establishment on Home Page.

* + 1. *Functional Requirements*
       1. Database should be able to store ratings.

## Post a Comment

* + 1. *Description and Priority*
       1. Users should be able to post a new comment under any location. Priority is high as it is one of the key functions of the application.
    2. *Stimulus/Response Sequences*

1. Users can select the option to add comments under any exercise location / healthy F&B establishment. Users will be prompted to input their comments in the description box.

2. Users can post comments successfully if requirements are met

* + 1. *Functional Requirements*
       1. Comment section must display past comments if any
       2. Comment input section must not be empty when the user presses “SEND”.
          1. Word limit is restricted to 3000 characters
       3. Database must be able to store comments correctly under their respective exercise location / healthy F&B establishment.

## View / Update Favourites List (Add / Remove)

* + 1. *Description and Priority*
       1. Users should be able to view their favourite lists
       2. Users should be able to update their favourite lists by adding or removing items within the list. Priority is high as it is one of the key functions of the application.
    2. *Stimulus/Response Sequences*

1. Users can select the option to add the item (exercise activity / food location) to their favourites list when viewing this item

2. Users can select the option to view their favourite list.

3. Users can select the option to delete the item from their favourites list when viewing the whole list.

* + 1. *Functional Requirements*
       1. Database must be able to store the users’ favourite lists for viewing and updating purposes.

# Other Nonfunctional Requirements

## 5.1 Performance Requirements

1. The application shall load up within 5 seconds
2. The home screen shall be displayed within 1 second of opening the application.
3. The description of each location will be displayed within 3 seconds from clicking into a location
4. The comments under a particular activity/location shall take within 2 seconds to load
5. Posting of a new comment will take at most 2 seconds to do so.
6. All error messages will be displayed within 0.5 seconds

## 5.2 Safety Requirements

1. The passwords key in by the user is hashed using salt hashing to ensure security and privacy
2. User’s login credentials are encrypted in the database to ensure security and privacy to each user

## 5.3 Security Requirements

1. Software used to develop the application should be industry- recognized with a certain amount of credibility.
2. User has to login successfully to post threads and comments.
3. Users will be prompted to re-login if he/she has not logged in to the application in a long period of time

## 5.4 Software Quality Attributes

## 5.4.1 Reliability

1. The mobile application should run smoothly and should not crash frequently.
2. Application should handle little or no downtime.
3. There will be weekly update of data in the database to ensure the reliability of the database.
4. The application should be usable even when there is a maintenance of the backend by segregating the front-end of the application from the backend of the application.

## 5.4.2 Maintainability

1. The implementation, design of the application is done with consideration of the future maintenance fees in mind.
2. Documentation shall be done in a clear and concise manner to ensure future additions or maintenance can be done by anyone if they have the documentation in hand.

## 5.4.3 Reusability

1. Implementation of the application is done so in modules to allow reusing of these modules in other application implementations.

## 5.4.4 Portability

1. Mobile Application must be able to work on various sizes and resolutions of the mobile phone screen.

## 5.4.5 Supportability

1. The database shall store at most 1000 user records.
2. Application will be able to support at most 500 users using the application at the same time.
3. The system will be able to do update if there are new modules added / old modules removed each year

## 5.4.6 Usability

1. Action Buttons are obvious and are easy to spot on the page.
2. Mobile application will prompt the user for the correct / valid input whenever possible to ease the user navigation process.
3. Mobile should be easy enough for new / sporadic users to navigate around within a short period of time.

## 5.5 Business Rules

1. Only registered users will be able to post threads/comments in the mobile applications.
2. Only admin users can manually edit the database and do any kind of modifications to the data stored in the database.

## 5.6 Data Dictionary

|  |  |
| --- | --- |
| **Term** | **Description** |
| User | Any person registered on the application who wants to use the services |
| Email | Email used for registration and password reset |
| Username | Unique name for each user |
| Password | Unique string of characters that allows access to the application |
| Mobile Number | Phone Number of User |
| Birthdate | Birthdate of User |
| System | Database that stores query results |
| Location | Data stored in System about priority locations (Healthy Food / Exercising Areas) |
| Activity | Types of exercise available at Location |
| Filters | Keywords available for a User to edit / manipulate results to User’s preference. |
| Profile | The record stored in the system’s database about the user’s personal information |
| Profile Picture | The picture accompanying the profile to identify the user |
| Direction | The recommended route to the designated location |
| Rating | The classification of locations based on the user’s experience at the location |
| Comment | Remarks of locations based on the user’s experience at the location |
| Favorite List | A list containing locations marked by the user as “Favourite” |
| Settings | A tab that allows configuration of User’s account |
| Marker | A pointer on the Google Maps to display locations |

# Appendix A. Use Case Descriptions

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 1 | | |
| Use Case Name: | Login to User Account | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User (Initiating actor), Firebase |
| Description: | User needs to login to use the app. |
| Preconditions: | 1. User has downloaded the app 2. User has registered an account successfully 3. There is stable internet communication available |
| Postconditions: | 1. Application main menu page is loaded |
| Priority: | * High. Users would not be able to use the application if they are not logged in. |
| Frequency of Use: | * Whenever the user enters the app. |
| Flow of Events: | 1. User will enter his/her email address and password to log in. 2. User will click on the “LOGIN” button. 3. Application system will then verify the username and password with the database. 4. Upon successful login, the main menu page of the application will be shown |
| Alternative Flows: | AF-S1: If the user inputs incorrect password   1. If a user inputs the incorrect password, an error message “Email / Password entered is invalid, Please try again”. will be shown. 2. The user can input password again.   AF-S2: If the user inputs incorrect username   1. If a user inputs the incorrect username, an error message “Email / Password entered is invalid, Please try again”. will be shown. 2. The user can input username again.   AF-S3: If the user clicks on the “FORGOT PASSWORD” button   1. User is led to the Reset Password page. |
| Exceptions: | 1. The user can choose to forget the password or register an account (if the account does not exist). |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | 1. Login is in English |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 2 | | |
| Use Case Name: | Search for location(s) that provides sport facilities or healthy food options in user’s current vicinity | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User (Initiating actor), Google API, Geolocation API |
| Description: | User can see locations near him/her that are used for exercising or provide them with healthy food options. |
| Preconditions: | 1. User login successfully. |
| Postconditions: | 1. The AroundMe interface is loaded. 2. Relative positions of sports venues or eateries that sell healthy food near user’s location will be displayed on a map. |
| Priority: | * High. |
| Frequency of Use: | * Whenever the user loads the AroundMe tab for healthy food and exercise locations around him/her (on-demand) |
| Flow of Events: | 1. User selects the AroundMe tab on the main page. 2. The AroundMe page is loaded. 3. System searches for locations of interest from its database that are near to the user’s current location (radius = 4km). 4. AroundMe page displays a map of locations that provides sports facilities and healthy food options. 5. User selects one of the locations from the results obtained. 6. The details of selected location will be displayed. |
| Alternative Flows: | AF-S1: If the user has not allowed access to location, there would not be any locations displayed on the map. |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 3 | | |
| Use Case Name: | View directions to selected location | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User, Google Maps API |
| Description: | User can get recommended shortest route to the selected location. |
| Preconditions: | 1. User login successfully. 2. User selects the AroundMe tab. 3. A map of exercise or healthy food locations around the user is displayed. 4. The selected location can be found in the database. |
| Postconditions: | 1. Directions for shortest route to selected location is displayed. |
| Priority: | * High |
| Frequency of Use: | * Whenever the user clicks on ‘Get Direction’ to the selected location. |
| Flow of Events: | 1. User clicks on the ‘Get Direction’ button for the location selected. 2. System will direct the user to Google Maps. |
| Alternative Flows: | N/A |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 4 | | |
| Use Case Name: | Access to User Location | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User (Initiating actor) |
| Description: | User can allow access to location in the app. |
| Preconditions: | 1. User login successfully. |
| Postconditions: | 1. A confirmation of consent will be shown. |
| Priority: | * High |
| Frequency of Use: | * Whenever the user wants to query current location in the vicinity. |
| Flow of Events: | 1. If access to location has not been turned on, a pop up will be shown. 2. User can click on Disallow or Allow button. 3. A confirmation display will be shown. |
| Alternative Flows: | N/A |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 5 | | |
| Use Case Name: | Create Account | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User (Initiating actor), Firebase |
| Description: | Users can create an account. |
| Preconditions: | 1. User has downloaded the app.   User does not has registered an account with the email provided. |
| Postconditions: | 1. App main page is loaded |
| Priority: | * High. User would be unable to login without an account. |
| Frequency of Use: | * Low, when they first use the application |
| Flow of Events: | 1. User clicks on the “Register” button. 2. User will enter their full name, email address, date of birth, gender and password to create an account. 3. User will click on the “Register” button. 4. System will check if the requirements are valid to create a new user. 5. The new user registration will be successful.   The login page of the application will be shown. |
| Alternative Flows: | AF-S1: If the email key in is not in the correct format   1. Error message: “Valid email is required” is displayed. 2. User return to Step 2 to re-enter a valid email.     AF-S2: If the password fails the checker check   1. Error message “Password too weak” is displayed. 2. User return to Step 2 to re-enter stronger password.     AF-S3: If the mobile number key in is not 8 digits   1. Error message “Please enter a valid phone number” is shown. 2. User return to Step 2 to re-enter a valid mobile number.     AF-S4: If the email is already in use   1. Error message “User already registered with this email” is displayed. 2. User return to Step 2 to re-enter a valid email address.   AF-S5: If the password and confirm password does not tally   1. Error message “Password key-in is not the same” is displayed. 2. User return to Step 2 to re-enter password and confirmation password. |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 6 | | |
| Use Case Name: | Change Profile Picture | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User (Initiating actor), Firebase |
| Description: | User can upload/update their profile picture. |
| Preconditions: | 1. User login successfully. 2. User has downloaded the pictures of interest. |
| Postconditions: | 1. User’s profile picture will be uploaded successfully. |
| Priority: | * Low. |
| Frequency of Use: | * Whenever the user wants to change the profile picture |
| Flow of Events: | 1. User clicks on the “Settings” button. 2. The settings page will be loaded and the user can click on the ‘Upload profile picture’ button. 3. User will be directed to a “Upload your profile picture” page with the current profile picture shown. 4. Upon clicking the “Choose picture” button, the user can select a picture from the downloads page. 5. User will be directed back to the “Upload your profile picture” page with the chosen profile picture. 6. User clicks on the ‘Save’ button to confirm. |
| Alternative Flows: | N/A |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 7 | | |
| Use Case Name: | Change Password | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User (Initiating actor),Firebase |
| Description: | User can change their password. |
| Preconditions: | 1. User login successfully. / User forgets the password |
| Postconditions: | 1. “Password reset successful” message will be printed and the user will be directed back to the settings page. |
| Priority: | * Low. |
| Frequency of Use: | * Whenever the user wants to change the password. * Whenever the user forgets the password. |
| Flow of Events: | 1. User clicks on the “Settings” button. The settings page will be loaded. 2. Upon clicking the “Change password” button, a “Change your password” page will be loaded. 3. User will enter his/her email address to reset the password. 4. The validity of email will be checked. 5. Firebase system will then verify if the password is valid and update the new password. |
| Alternative Flows: | AF-S4: If the user does not enter a valid email   1. An error message “Email Not Found/ Invalid email” will show to prompt the user to check his/her email again |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 8 | | |
| Use Case Name: | Change Username | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User (Initiating actor), Firebase |
| Description: | Users can change their username |
| Preconditions: | 1. User login successfully. |
| Postconditions: | 1. The user will be directed back to the settings page. |
| Priority: | * Low |
| Frequency of Use: | * Whenever the user wants to change the username. |
| Flow of Events: | 1. User clicks on the “Settings” button. The settings page will be loaded. 2. Upon clicking the “Change username” button, a “Change your username” page will be loaded. 3. User will enter his/her new username and click the “Save” button to confirm. 4. User will be directed back to settings page |
| Alternative Flows: | N/A |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 9 | | |
| Use Case Name: | View Favourites | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User (Initiating actor), Firebase |
| Description: | Users can view locations/activities they indicated as favorites. |
| Preconditions: | 1. User login successfully. 2. User added locations to Favourites before. |
| Postconditions: | 1. A list of favorite locations/activities of a user will be shown. |
| Priority: | Low |
| Frequency of Use: | * Not Frequent. |
| Flow of Events: | 1. User clicks on the Favourites button on the main page of the app. 2. A list of locations and activities the user indicated as favorites prior will be shown. |
| Alternative Flows: | N/A |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | NIL |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 10 | | |
| Use Case Name: | Add to Favourites | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User, Firebase |
| Description: | User can add locations to favourites list |
| Preconditions: | User has logged in successfully. |
| Postconditions: | Favourites list has updated in database. |
| Priority: | Low |
| Frequency of Use: | Frequent |
| Flow of Events: | 1. The user will click on “Add to Favorites” button on a selected location from Home Page. |
| Alternative Flows: | N/A |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | N/A |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 11 | | |
| Use Case Name: | Remove from Favourites | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | User, Firebase |
| Description: | User can remove locations from favourites list |
| Preconditions: | User has logged in successfully. |
| Postconditions: | Favourites list has updated in database. |
| Priority: | Low |
| Frequency of Use: | Frequent |
| Flow of Events: | 1. The user will click on “Delete” button on a selected location on View Favorites page. |
| Alternative Flows: | N/A |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | N/A |

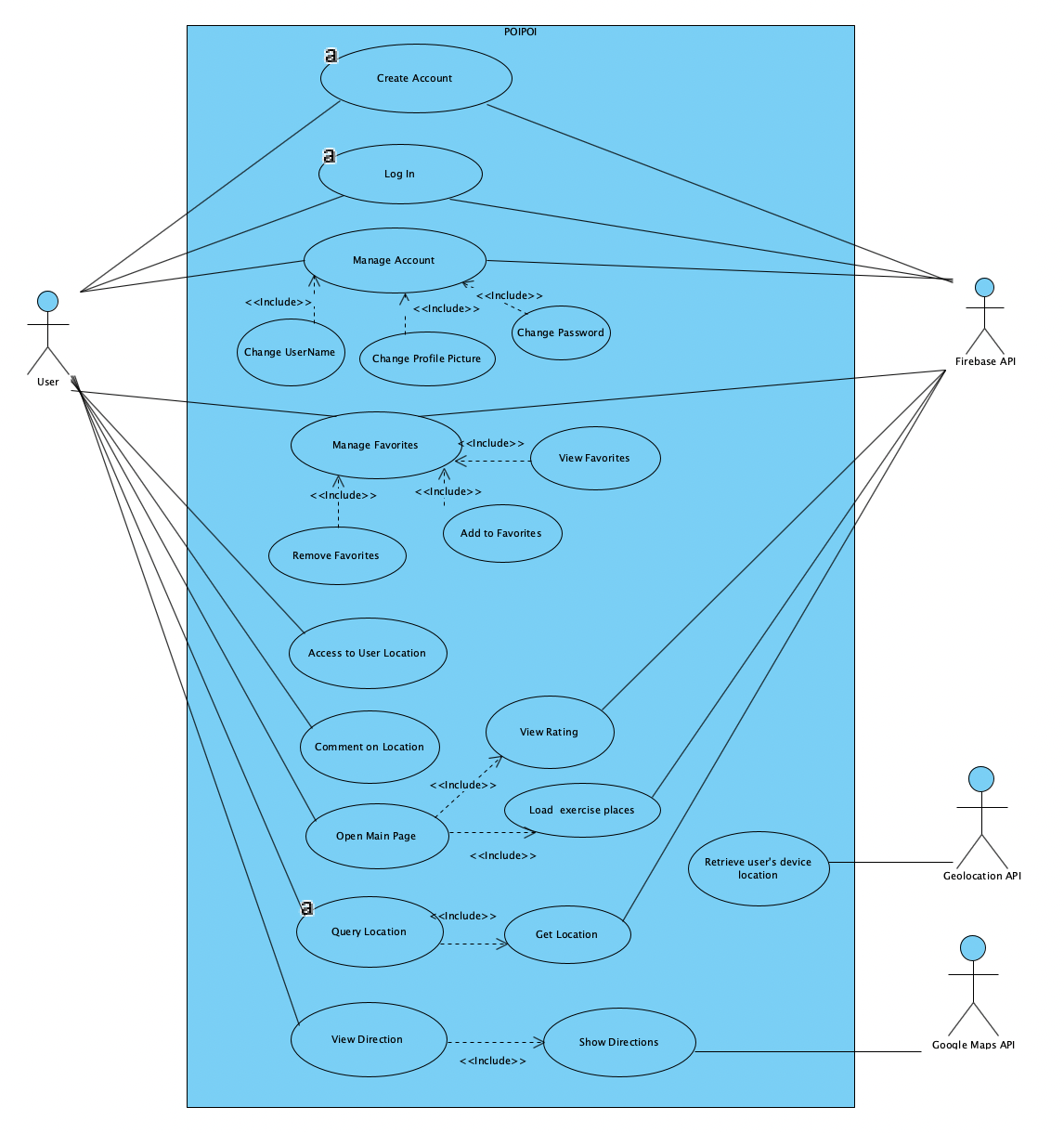
|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 12 | | |
| Use Case Name: | Open Main Page | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Actor: | | User, Firebase | | |
| Description: | | Whenever user logs in, the main page will be loaded. | | |
| Preconditions: | | User has logged in successfully. | | |
| Postconditions: | | Main page is successfully loaded. | | |
| Priority: | | High | | |
| Frequency of Use: | | High | | |
| Flow of Events: | | 1. After user logs in successfully, exercise locations and restaurants will be retrieved from the dataset. 2. The main page will display a list of sports venues / restaurants and their details. 3. The user will also be able to view their ratings. | | |
| Alternative Flows: | | AF-S1: If the user wants to view locations in alphabetical order, the list will be sorted accordingly.  AF-S2: If the user wants to view locations in order of their ratings, the list will be sorted accordingly. | | |
| Exceptions: | | N/A | | |
| Includes: | | N/A | | |
| Special Requirements: | | N/A | | |
| Assumptions: | | N/A | | |
| Notes and Issues: | | N/A | | |
| Use Case ID: | 13 | | | | |
| Use Case Name: | Comment on Location | | | | |
| Created By: |  | | Last Updated By: |  | |
| Date Created: |  | | Date Last Updated: |  | |

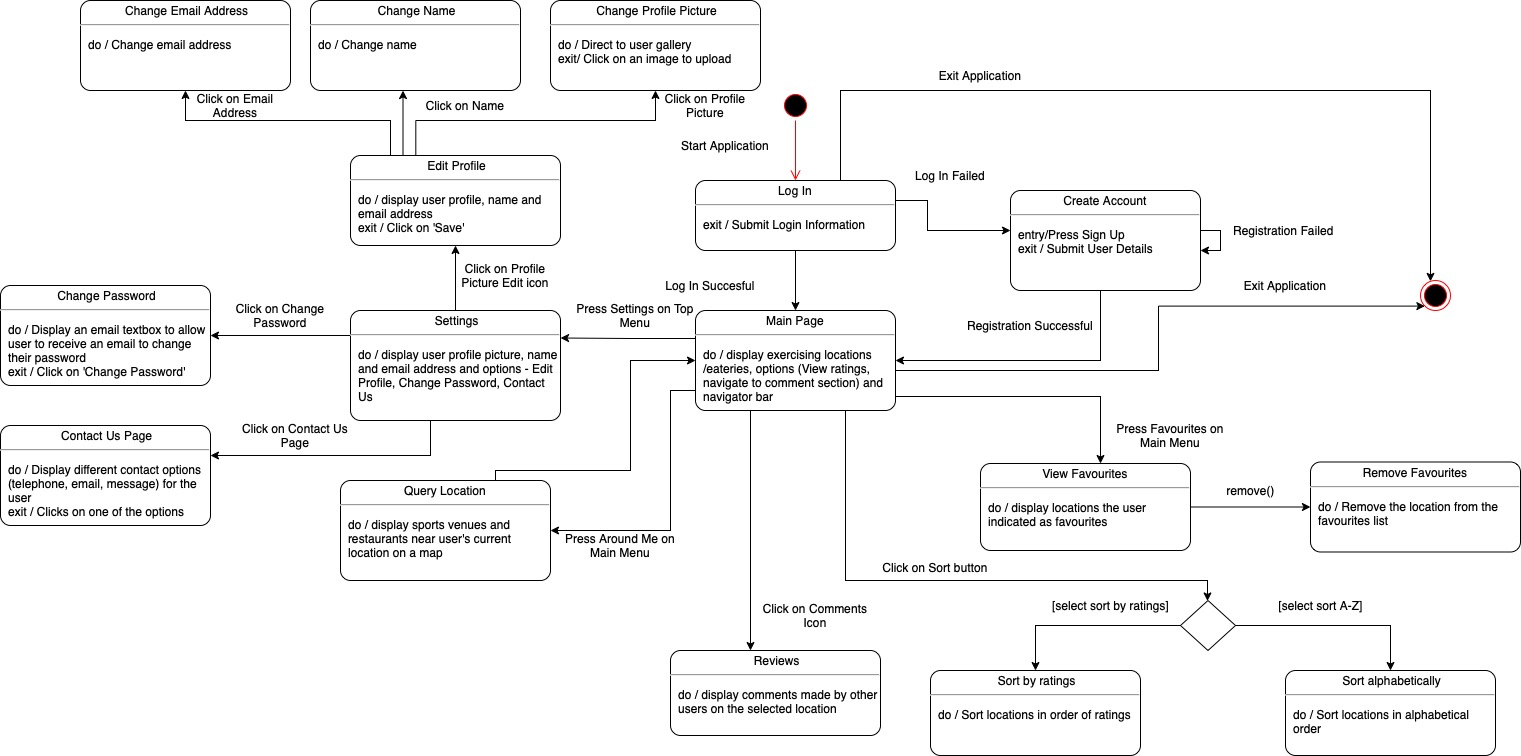
|  |  |
| --- | --- |
| Actor: | User, Firebase |
| Description: | Whenever user clicks on comments icon on a specific location, comments page will be loaded. |
| Preconditions: | 1. User has logged in successfully. 2. Main Page is successfully loaded |
| Postconditions: | User will be able to see their comments. |
| Priority: | High |
| Frequency of Use: | High |
| Flow of Events: | 1. After the main page has successfully loaded, the user can click on comments icon on a specific location. 2. The user will be able to view comments made by others. 3. The user will be able to type in the text box below. 4. After the user presses enter, the user will be able see their comment on the page. |
| Alternative Flows: | AF-S4 – If the user does not write anything, an error message “Please enter a comment” will be shown.  AF-S4 – If the user exceeds 3000 characters, an error message “Please limit comment length to under 3000 characters.” will be shown. |
| Exceptions: | N/A |
| Includes: | N/A |
| Special Requirements: | N/A |
| Assumptions: | N/A |
| Notes and Issues: | N/A |

# Appendix B: Analysis Models

## Use Case Diagram

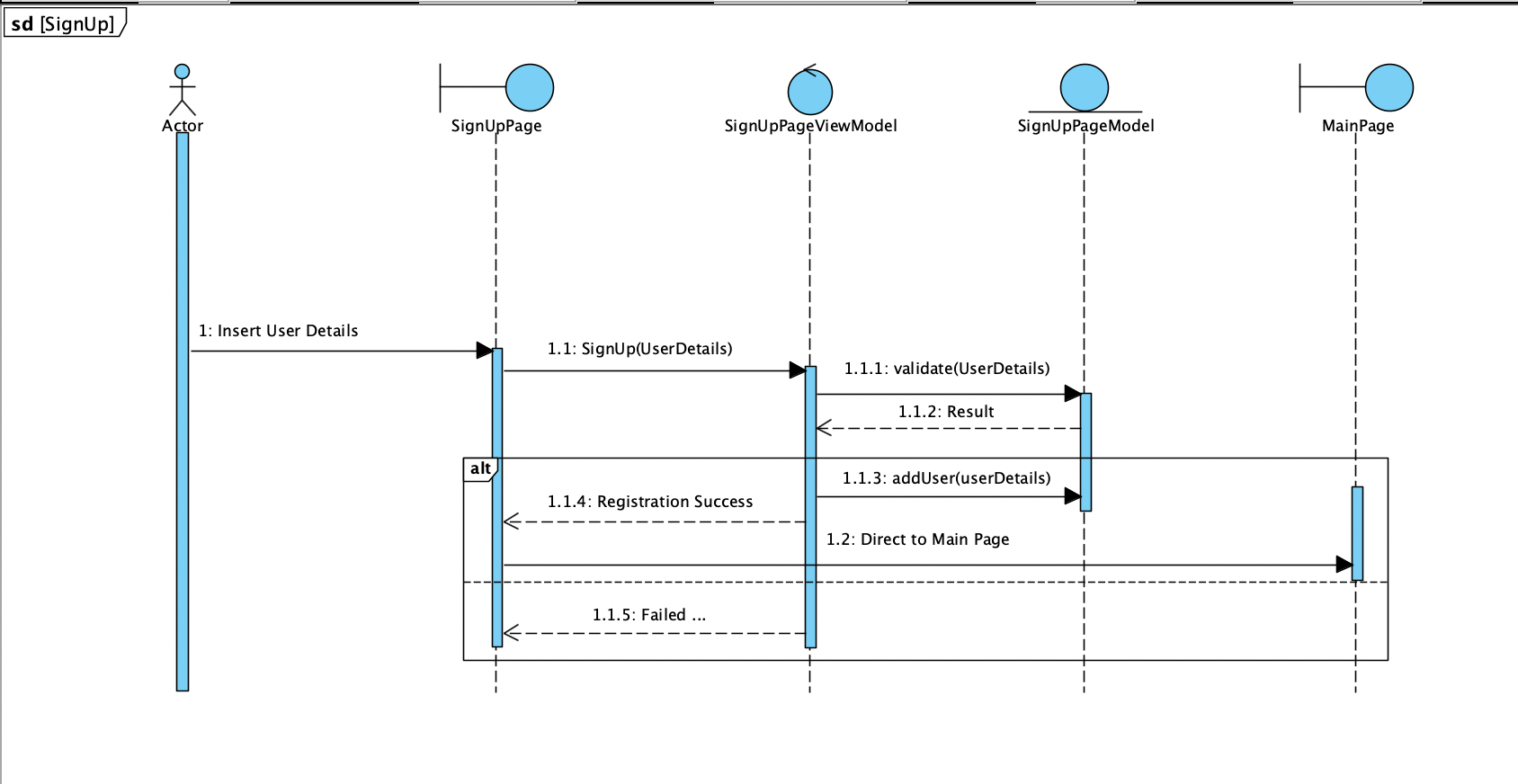


## Dialog Map

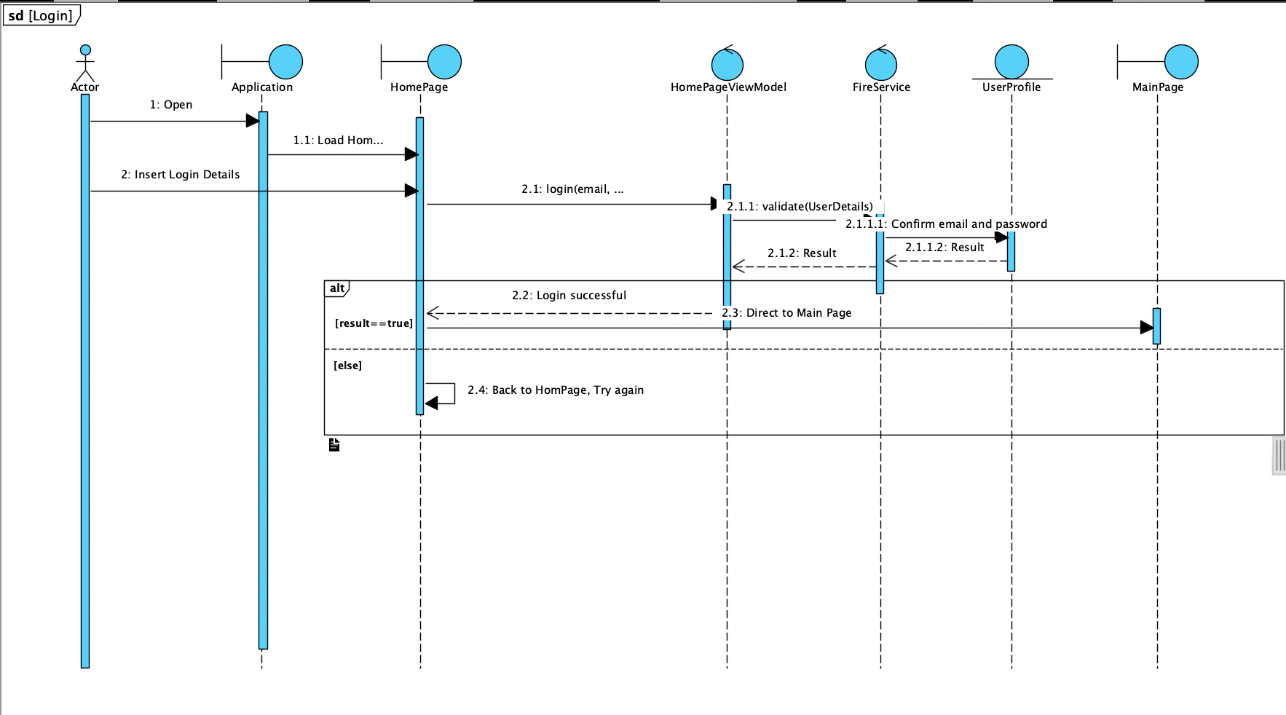


## Sequence Diagrams

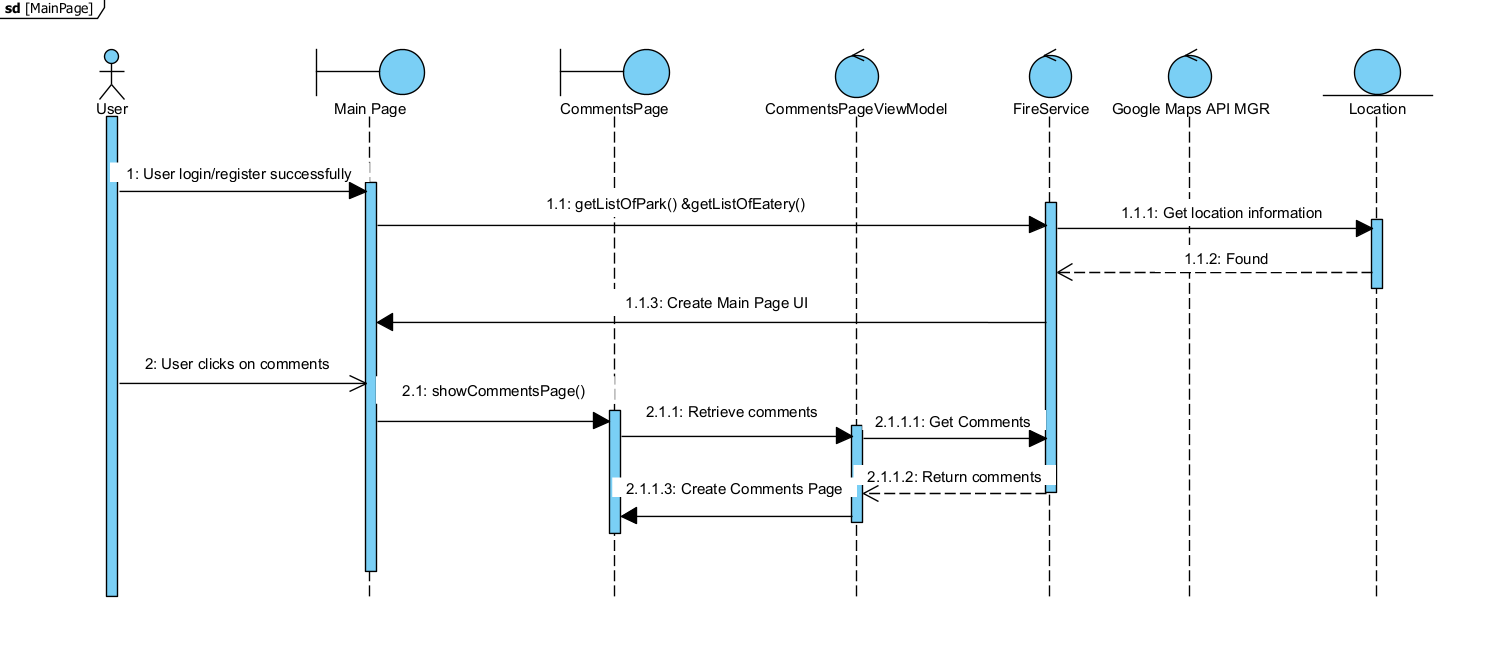
Sign Up (Registration)



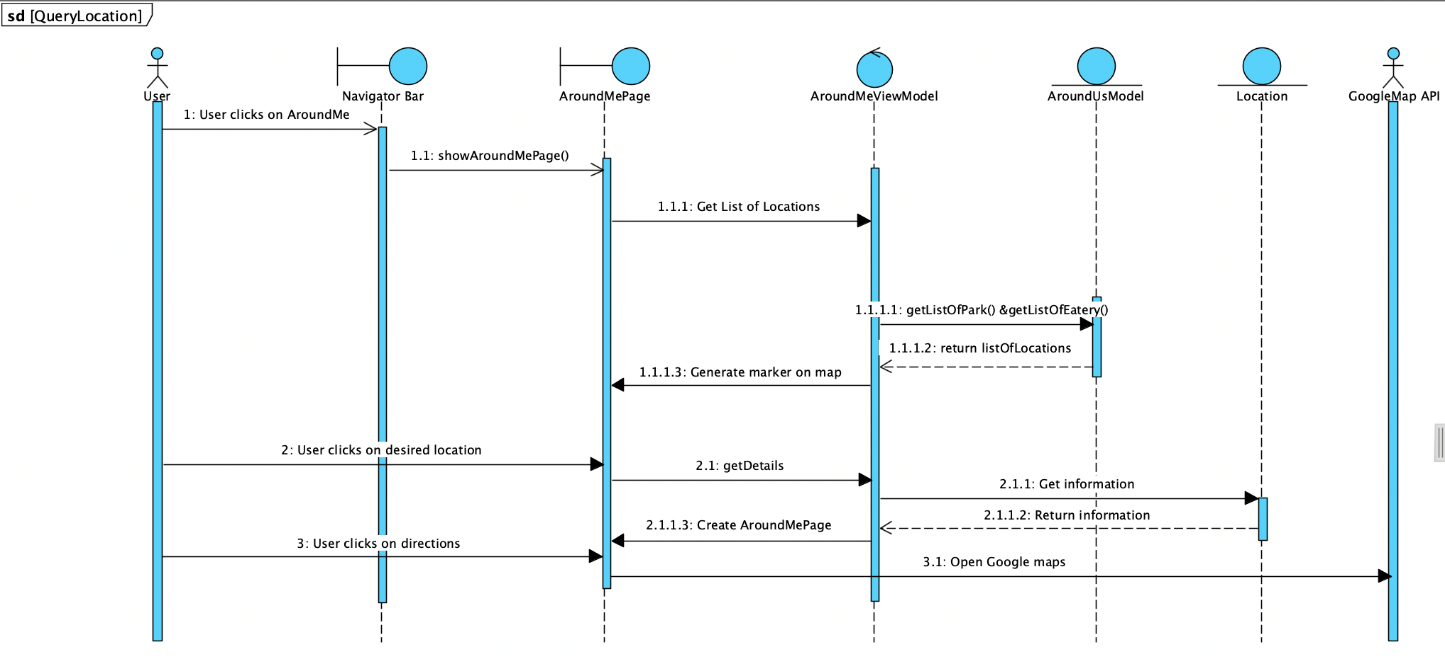
Login



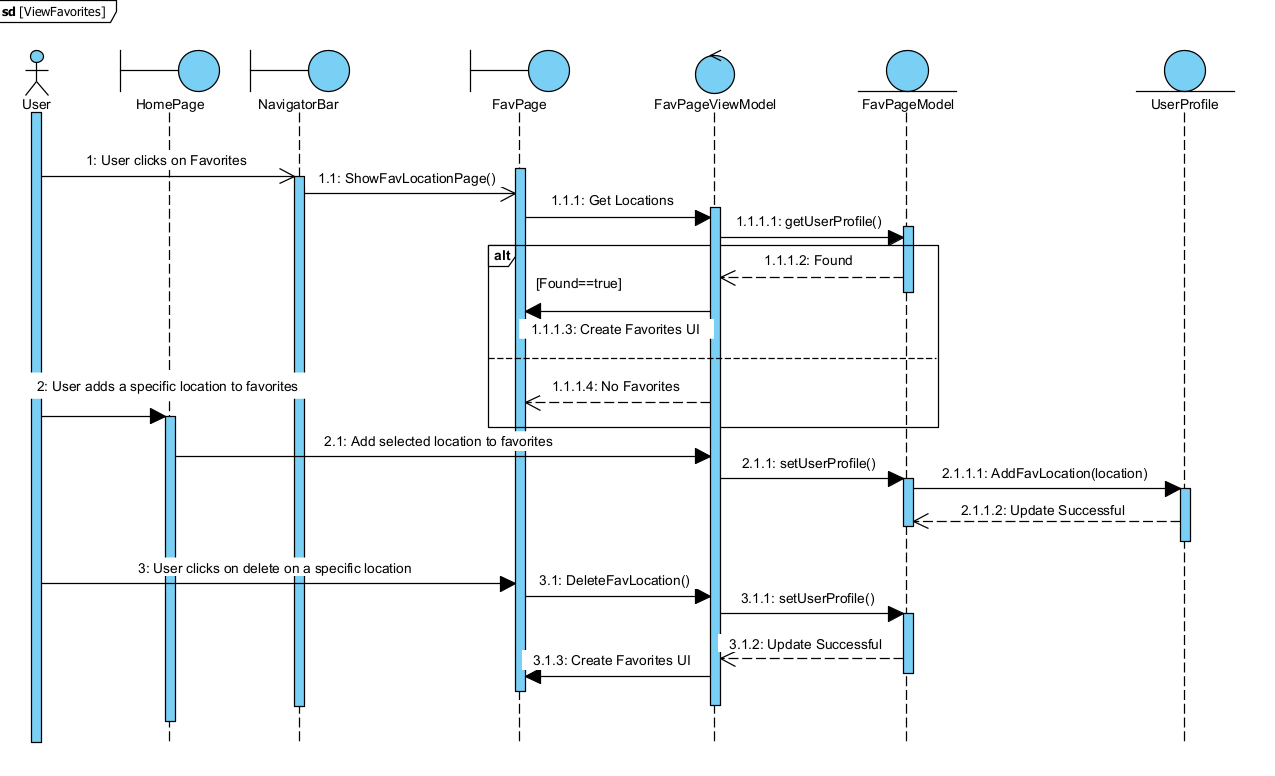
Main Page Loading & Comments



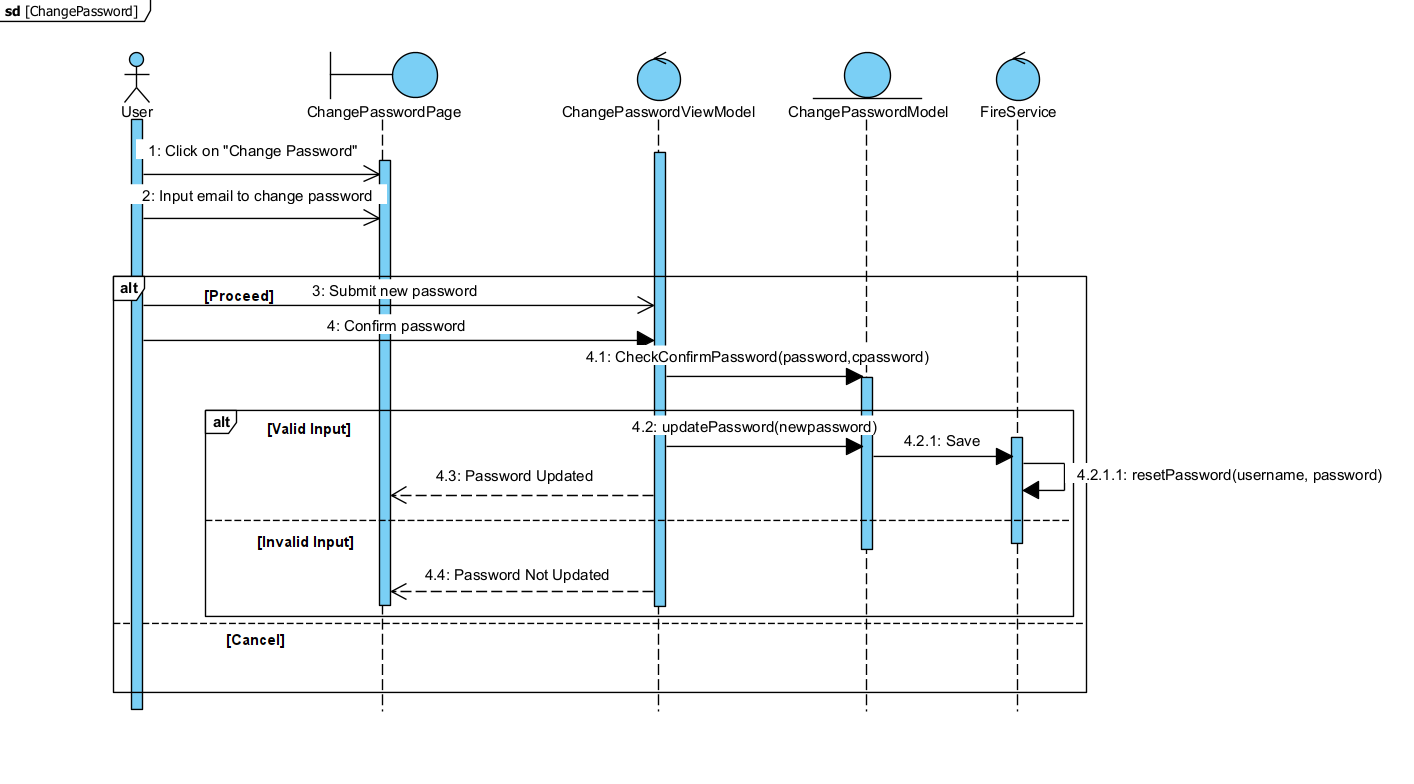
Query Location



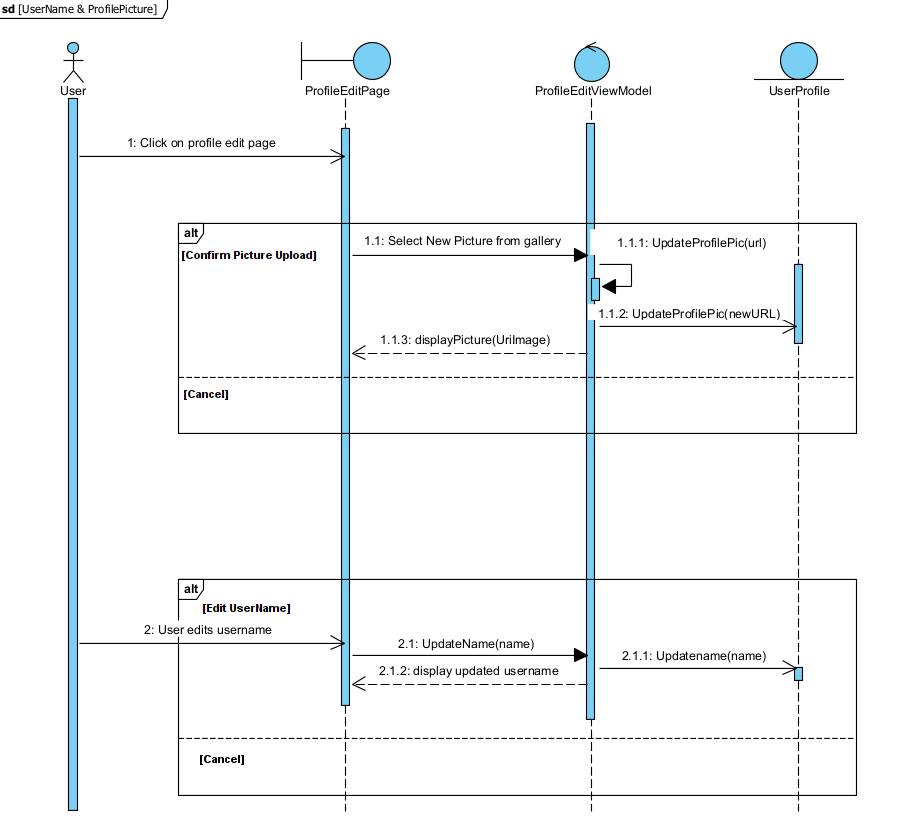
View Favorites



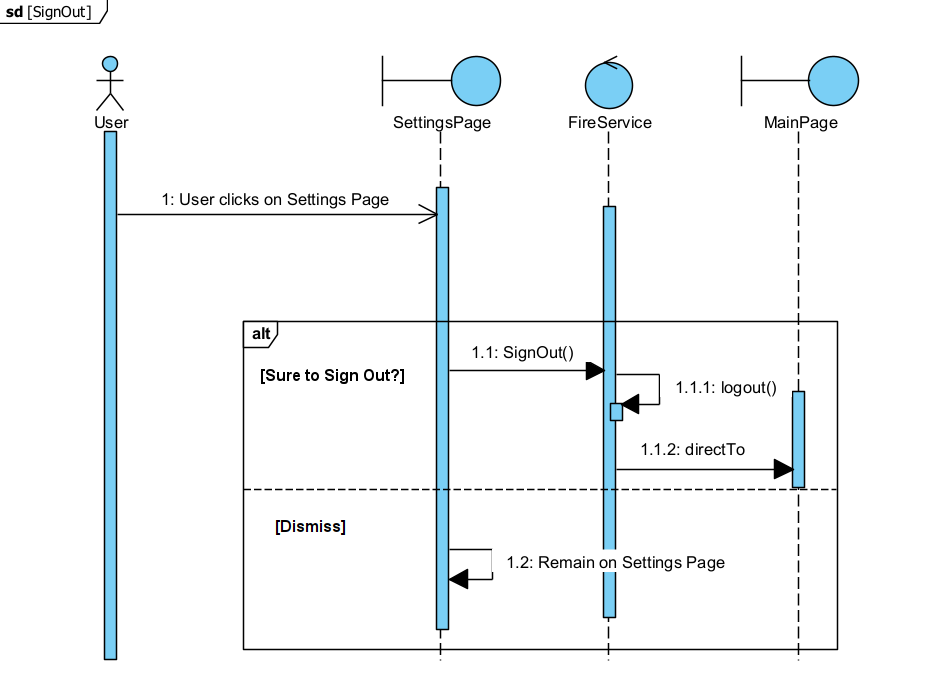
Change Password



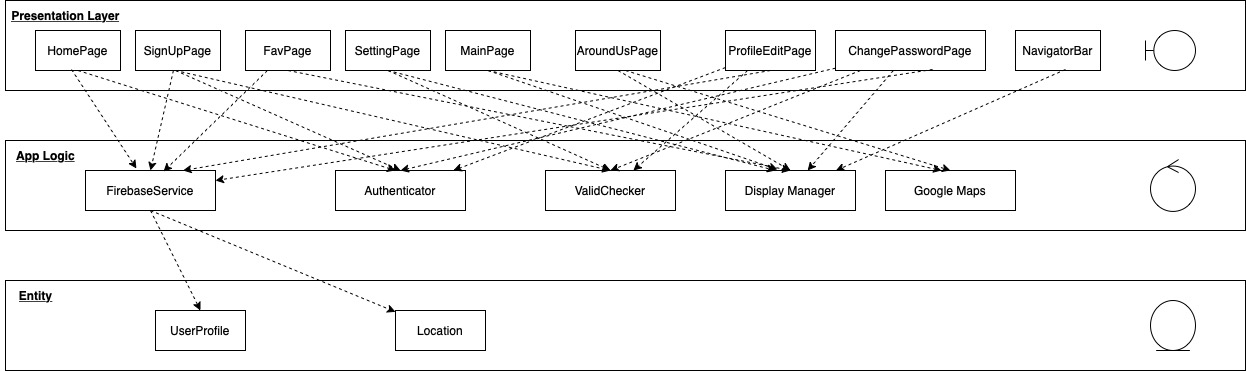
Change Username & Profile Picture



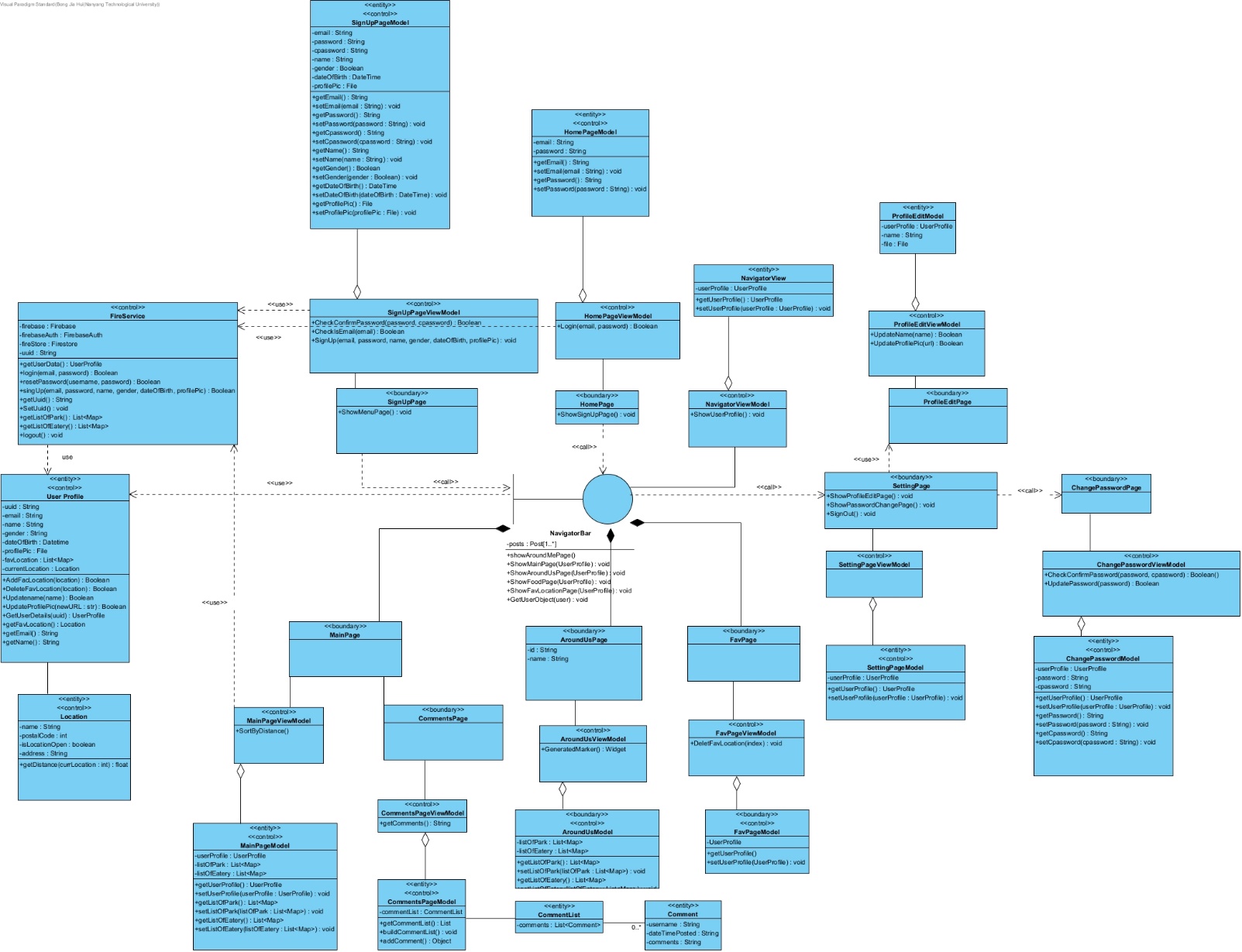
Sign Out



## System Architecture



## Class Diagram/Conceptual Model



# Appendix C: Test Cases

The two main types of testing used on our application is Black Box Testing and White Box Testing. Both testing methods were conducted on the features and functionalities of the application to ensure that the application does not contain any error or fault that may cause the system to produce an incorrect or unexpected result.

## Black Box Testing

Equivalence class testing was done for discrete values. We also did boundary value testing which is only applicable to a range of values.

1. **Login**

A. Generic cases

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id** | **Scenario** | **Expected Result** | **Actual Result** |
| 1 | Login with valid account username and password | The system displays the main page for the user | The system displays the main page for the user |
| 2 | Login without filling up all the required fields | The system prompts the user to fill up the required fields | The system prompts the user to fill up the required fields |
| 3 | Login without filling any fields | The system prompts the user to fill up the required fields | The system prompts the user to fill up the required fields |
| 4 | Login without valid credentials | The system prompts the user to enter the credentials again | The system prompts the user to enter the credentials again |

B. Specific Cases

Test Case 1: Login. Checking if email, password, and confirmation password are correct.

|  |  |  |  |
| --- | --- | --- | --- |
| Email | Password | Expected Exception | Result (FireBaseException caught) |
| ant@gmail.com | Anthony | “User-not-found” | Caught user-not-found, Pass |
| randomuser@gmail.com (Assuming this email is not registered in the app) | @Anthony123345 | “Wrong-password” | Caught wrong-password, Pass |
| No entry | No entry | “User-not-found”, “Wrong-password” | Caught user-not-found, Caught wrong-password, Pass |
| ant@gmail.com | @Anthony123345 | No exception | No exception, Pass |

1. **Registration**

A. Generic Cases

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id** | **Scenario** | **Expected Result** | **Actual Result** |
| 1 | Register with valid account username and password | The system displays the main page for the user | The system displays the main page for the user |
| 2 | User account already exists | The system will display an error message | The system will display an error message |
| 3 | Register with incomplete fields | The system prompts the user to fill up the required fields | The system prompts the user to fill up the required fields |
| 4 | Register with password mismatch | The system prompts the user to fill up the required fields | The system prompts the user to fill up the required fields |

B. Specific Cases

Test Case 2: SignUp. Checking if all the user details for creating an account is valid. Correct image link: [Profile Picture](https://firebasestorage.googleapis.com/v0/b/poi-poi-f8448.appspot.com/o/9Fw33zEZG8OG98sn1nP1FGcnw6a2?alt=media&token=abf3ae23-88b5-492f-be89-eb12800b8623)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Email | Password | Confirm password | Full Name | Date of birth | Gender | Profile Picture | Expected Output | Result |
| No entry | @Anthony123345 | @Anthony123345 | Ant | 03/06/2000 | Male | Correct image link | “Full Name is required” | Pass |
| ant@gmail.com(Assuming this email is already registered) | @Anthony123345 | @Anthony123345 | Ant | 03/06/2000 | Male | Correct image link | “An account already exists for that email” | Pass |
| ant@gmail.com | No entry | @Anthony123345 | Ant | 03/06/2000 | Male | Correct image link | “Please enter password” | Pass |
| ant@gmail.com | 123 | @Anthony123345 | Ant | 03/06/2000 | Male | Correct image link | “Please provide password more than 6 character” | Pass |
| ant@gmail.com | @Anthony123345 | No Entry | Ant | 03/06/2000 | Male | Correct image link | “Please enter confirm password” | Pass |
| ant@gmail.com | @Anthony123345 | @Anthony | Ant | 03/06/2000 | Male | Correct image link | “Password are not the same” | Pass |
| ant@gmail.com | @Anthony123345 | @Anthony123345 | No entry | 03/06/2000 | Male | Correct image link | “Please enter your name” | Pass |
| ant@gmail.com | @Anthony123345 | 31/7/1999 | Male | No entry | Male | Correct image link | “Please choose your date of birth” | Pass |
| ant@gmail.com | @Anthony123345 | 31/7/1999 | Male | 03/06/2000 | No Entry | Correct image link | “Gender not selected” | Pass |
| ant@gmail.com | @Anthony123345 | 31/7/1999 | Male | 03/06/2000 | No Entry | No selected image | “Please select your profile picture” | Pass |
| ant@gmail.com | @Anthony123345 | 31/7/1999 | Male | 03/06/2000 | Male | Correct image link | “User registered” and the user is directed to the Main page. | Pass |

1. **Change Password**

Checking if the password meets the valid requirements.

|  |  |  |
| --- | --- | --- |
| Password | Expected Output | Result |
| No entry | “New Password field is left empty” | Pass |
| hello | “Password enter is too weak. Please Try again. | Pass |
| jelly22fi$h | “Password Reset Successful” | Password reset. Pass |

1. **Change Username**

## Checking if the username is changed correctly.

|  |  |  |
| --- | --- | --- |
| Username | Expected Output | Result |
| No entry | “Username field is empty” | Pass |
| Cownic | “Username updated”, reflected in the Home page. | Username is updated. Pass |

1. **Favorites**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id** | **Scenario** | **Expected Result** | **Actual Result** |
| 1 | Add a location to favorites | The system saves the location into the favorites list | The system saves the location into the favorites list. Pass |
| 2 | Add a location already in a favorites list | The system will display an error message “The location is already in favorites” | Pass |
| 3 | View favorites | The system loads the list of user’s favorites and displays on the page. | Pass |
| 4 | View favorites but the list is empty | The system displays an error message. | Pass |
| 5 | Remove favorites | The system updates the favorites list and is removed from the favorites. | Pass |

1. **Get User’s current location**
2. Display user’s geo location on google map (Valid inputs)

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Input Latitude Longitude** | | **Expected output** | **Actual Output** |
| 1.3532919 | 103.6920974 | Hall of Residence 9 (Hall 9) | Hall of Residence 9 (Hall 9) |
| 1.3543025 | 103.6834343 | Hall of Residence 10 (Hall 10) | Hall of Residence 10 (Hall 10) |

1. **Get directions to a location**

|  |  |  |
| --- | --- | --- |
| **Directions** | **Expected Result** | **Actual Result** |
| Jalan Boon Lay Junction with Boon Lay Way | Same results as google (is it google) map | Same results as google map |
| Choz Catering Pte Ltd | Same results as google map | Same results as google map |

1. **Comments**

## Test case 7: Comment word limit. Comments should not be empty and cannot exceed 3000 characters.

|  |  |
| --- | --- |
| Number of characters for comment | Result |
| 1 | Accept |
| 3000 | Accept |
| 0 | Reject(Error message “Field is empty”) |
| 3001 | Reject(Error message “Please limit comment length to under 3000 characters”) |

1. **Profile Picture**

Test case 9: Change Profile Picture. Picture should update to new selected image.

|  |  |
| --- | --- |
| Number of characters for comment | Result |
| Valid image chosen | Accept, update profile picture |
| Invalid image chosen | Reject(Error message “Please choose an image”) |

## White Box Testing

White box testing was done to test if the flow of the application is as expected and to also test for alternative flows.

|  |  |
| --- | --- |
| **Test Case 1** | |
| **Test Case Type** | User Interface |
| **Description** | ‘LOGIN’ button on Home page leads to the Main page |
| **Expected Results** | Login page opens |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |

|  |  |
| --- | --- |
| **Test Case 2** | |
| **Test Case Type** | User Interface |
| **Description** | ‘Sign Up button on main page leads to the register page |
| **Expected Results** | Sign Up page opens, System will display ‘You can sign up now’ and registers new user. |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |

|  |  |
| --- | --- |
| **Test Case 3** | |
| **Test Case Type** | User Interface |
| **Description** | ‘Forget password button on main page leads to the register page’ |
| **Expected Results** | Forget password page opens, System will display ‘Please enter your email’ and user receives a link to reset password. |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |

|  |  |
| --- | --- |
| **Test Case 4** | |
| **Test Case Type** | User Interface |
| **Description** | ‘Able to navigate between pages e.g. Home, Around Me, Setting and Favorite’. No change of selection of tabs will result in no change in page shown. |
| **Expected Results** | Navigating to all pages without issue. |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |

|  |  |
| --- | --- |
| **Test Case 5** | |
| **Test Case Type** | User Interface |
| **Description** | When user key in any new comment and press the submit button in a location’s comment page. |
| **Expected Results** | The newly created comment will be posted on the page immediately |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |

|  |  |
| --- | --- |
| **Test Case 6** | |
| **Test Case Type** | User Interface |
| **Description** | Able see description of healthy food/exercise location and able to navigate to location with google map. |
| **Expected Results** | Both healthy food and exercise locations are shown on map and able to navigate to location. |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |

|  |  |
| --- | --- |
| **Test Case 7** | |
| **Test Case Type** | User Interface |
| **Description** | When user clicks on add to favorite location, location will be show on favorite page |
| **Expected Results** | The location will be added in favorite page |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |

|  |  |
| --- | --- |
| **Test Case 8** | |
| **Test Case Type** | User Interface |
| **Description** | When user clicks on delete favorite location, location will be removed on favorite page |
| **Expected Results** | The location will be removed in favorite page |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |

|  |  |
| --- | --- |
| **Test Case 9** | |
| **Test Case Type** | User Interface |
| **Description** | User will be able to change their name/profile picture/password.  1.‘'UPLOAD’ button in the “Change Profile Picture” page will change the user’s profile picture when the user selects a picture from their device gallery.  2. User will be able to change their password. After clicking on reset password button a link will be sent to the email.  3. ‘SIGN OUT’ button in the settings page will log the user out. |
| **Expected Results** | User able to change their name/profile picture/password  1. System will display “Profile Pic uploaded.”  2. User received email and able to change their password  3. User will be pushed to homepage. |
| **Process Diagram** |  |
| **Pass/Fail** | Pass |