NANYANG TECHNOLOGICAL UNIVERSITY

School of Computer Science & Engineering
CZ2006 - Software Engineering

Project Name - Fitrition

Fitrition

Seminar Group: BCS2
Team: Dream Data

Name	Matriculation Number
Cogal Jacques Tracy Pasaol	U2022175B
Tan Hong Fan Merzen	U2022182E
Tan Kiang Hwee Jeremy	U2021940D
Teresa Zhang Han Yu	U2022886C
Wang Sishi	U2021695B
Yam Hui Jing	U2021656A
Yeow Ming Xuan	U2022461J

APRIL, 2022

Table of Contents

1. Product Description

1.1 Purpose

The Fitrition application aims to promote healthy living among its users. As more Singaporeans are living a secluded life as work from home (WFH), obesity problems are getting more common. To align healthcare with the smart nation goal in Singapore, our team has decided to come up with the application - Fitrition to bridge the gap. This application emphasises healthy eating, physical activity and nurturing a healthy lifestyle.

Effectively, Fitrition serves three main purposes. They are as follows:

(1) It facilitates the searching of nearby eateries and fitness facilities based on the GPS location of an Android mobile device,
(2) It tracks the exercise routine and diet of the users, ensuring the users are on track in achieving the goals set, and stay motivated
(3) It allows the users to motivate each other to stay on track in the healthy lifestyle by sharing their social status to their friends.

1.2 Scope

The users are to register a user account to use the Fitrition application. After logging in, they can search and view the details of nearby eateries as well as fitness facilities from the map. Information of the eateries and fitness facilities will be collected through the application with the assistance of Google Places API. Their choice of eateries and fitness facilities can be conveniently added into the calendar page which records their exercise routine and diet. As they accomplish each tiny step, their activities will be shared with their friends, creating an encouraging fitness community.

1.3 Users and stakeholders

The stakeholders of this project consist of the Android mobile users, Fitrition and fitness facilities.

1) **Android mobile users** will utilise the application to locate nearby locations where they can practise healthy eating and active lifestyle. They can track the activities using a calendar and share the achievements with their friends.

2) **Fitrition** strives to bring the locations closer to Android mobile users by providing a smooth and seamless platform for digital interaction between the facilities and mobile users. In addition, the motivation of sharing achievements will encourage Android mobile users to continue using the Fitrition application.

3) **Eateries and fitness facilities** are promoted in Fitrition to encourage the users to lead a healthier lifestyle by opting for nutritious meals and being active.

1.4 Assumptions and constraints

1) Users should have a GPS-enabled Android device capable of running the application.
2) Users should have Internet access in order to use the application.

1.5 Constraints

1) The Fitrition application currently supports Android OS running Android 8.0 Oreo(API level 26) and above.
2) The application is currently available only in English.
3) The application supports only 100 simultaneous connections to the cloud database, up to 1 GB of data storage and 10 GB per month of data download.

1.6 Initial UI Mockups

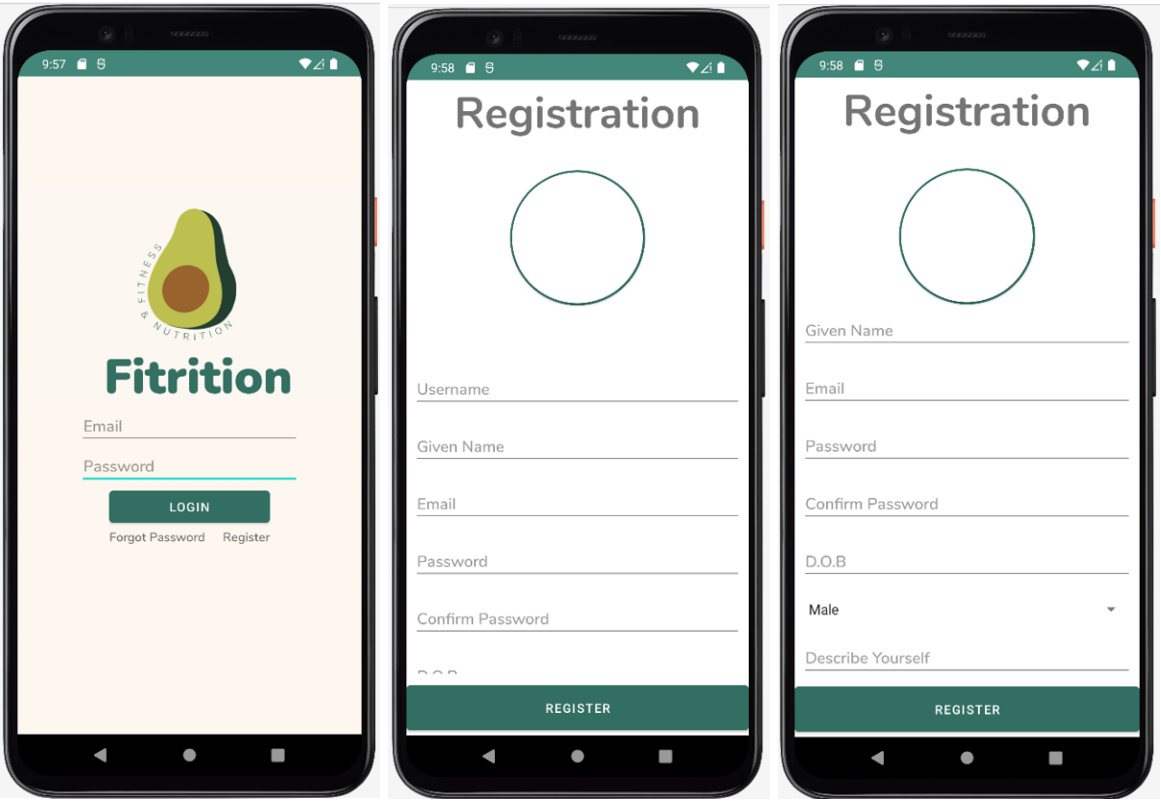


Figure 1.1: Login and Registration Page

Figure 1.1 shows the user interface for login and registration. The user interface layout is kept simple and concise, similar to the design for other applications. This provides a higher comfort level while registering for a new account and logging in to the application.

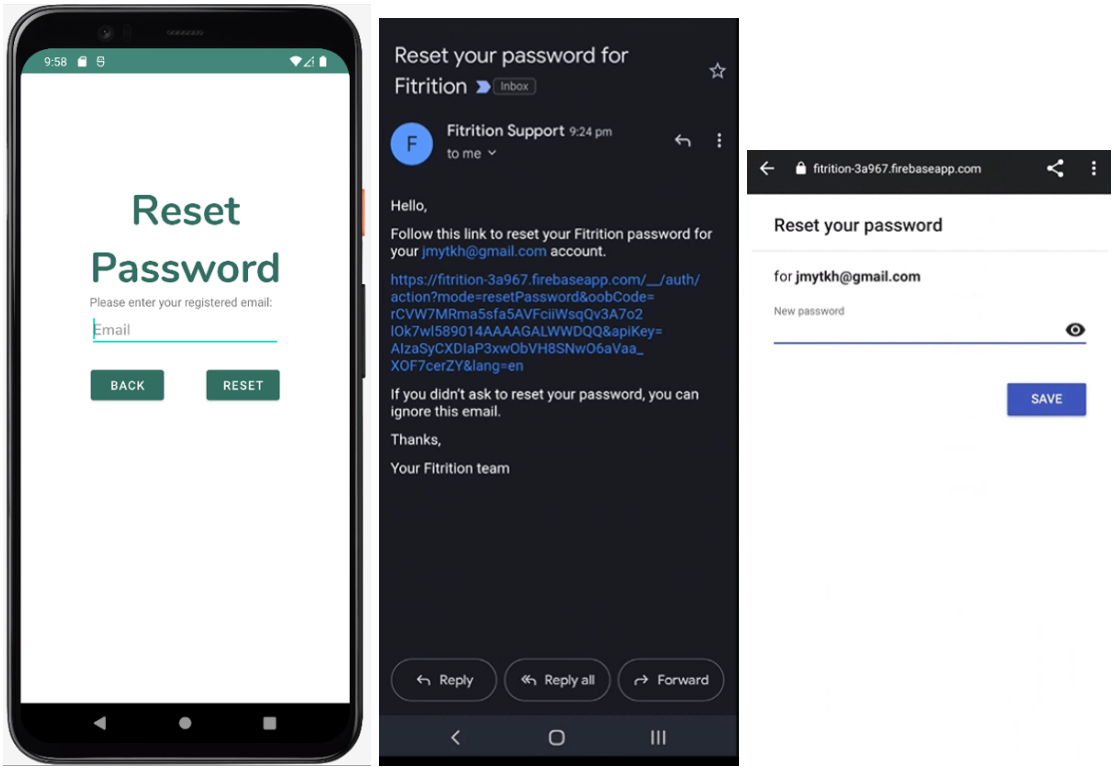


Figure 1.2: Forget Password Page

Figure 1.2 shows the user interface for Forget Password Page. The forget password page will request the user to input their email address. A link to change password will be subsequently sent to the user via the email. Upon clicking on the link, the page prompting for the new password will appear and the user will be able to set a new password for his account.

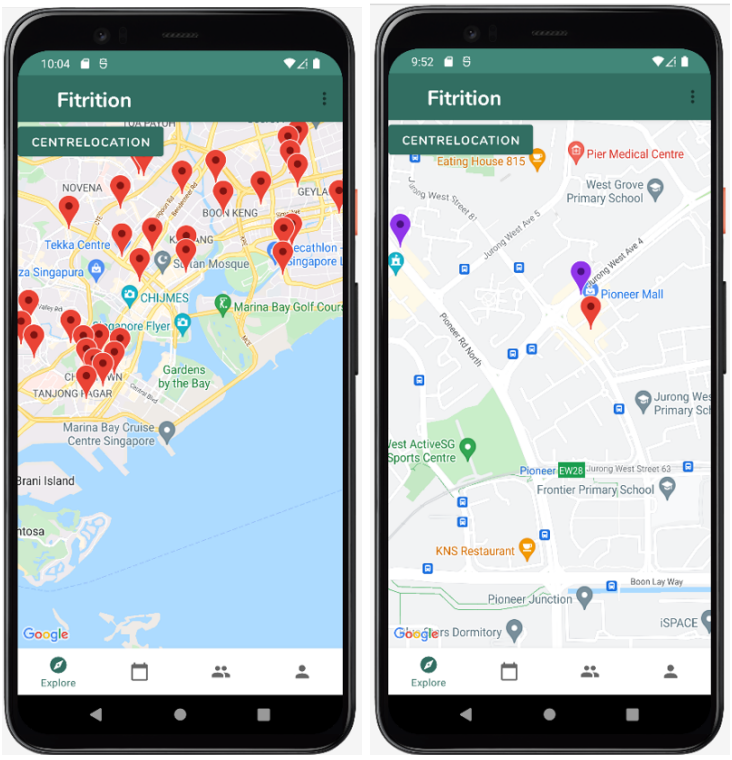


Figure 1.3: Explore Page

Figure 1.3 shows the user interface for the Explore Page. The navigation bar below allows the user to access the pages like Calendar Page, Social Page and Profile Page. The Explore Page will display a Singapore map and it will be zoomed to the locations according to the user's GPS location upon clicking on 'CentreLocation'. The map markers indicate the nearby eateries and fitness facilities. They are colour-coded such that the blue marker represents the user's location, the red markers represent the eateries while the purple markers represent the fitness facilities.

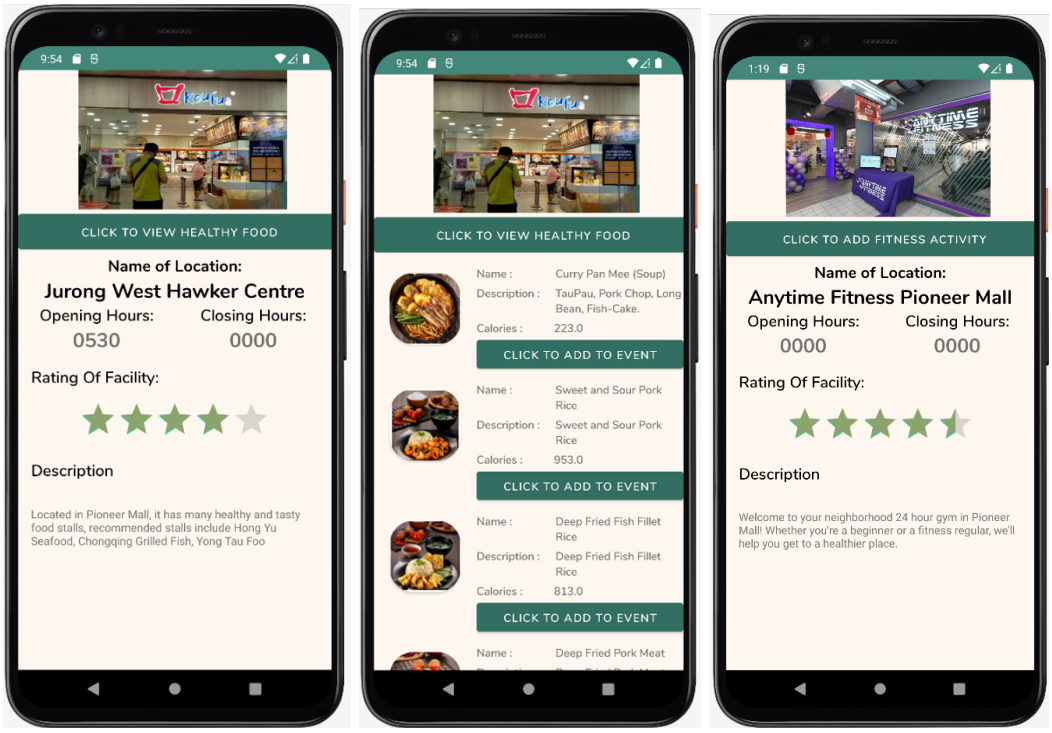


Figure 1.4: Facilities Information Page

Figure 1.4 shows the information of facilities after the user clicks on the markers, including the healthy food options the eateries offer. Important information such as name of location, opening and closing hours, ratings and a brief description of the location will also be displayed.

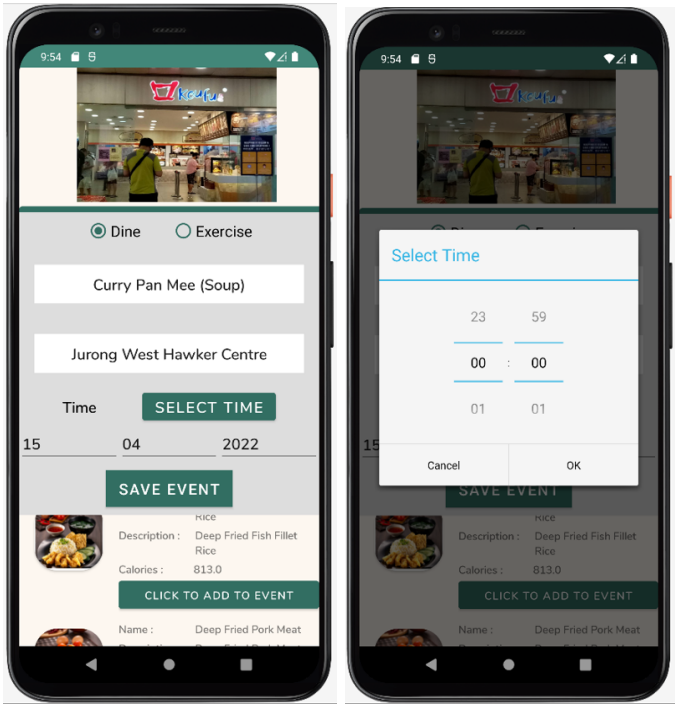


Figure 1.5: Add Event from Explore Page

Figure 1.5 shows how the button ‘click to add to event’ can conveniently add the dining or exercise activity into the calendar event. It also redirects to the phone’s native google maps to provide directions to the facility.

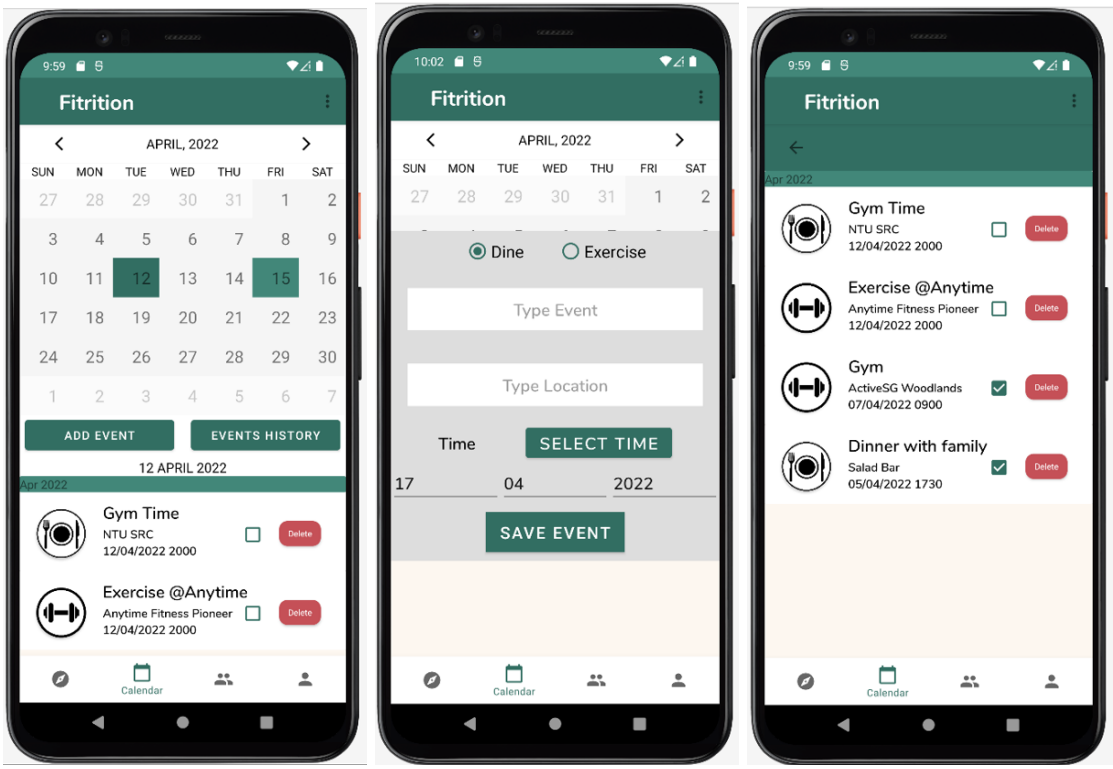


Figure 1.6: Calendar Page

Figure 1.6 shows the user interface for Calendar Page. The page displays all their events in the calendar, as well as creating and deleting events. Checkboxes appear for each event to allow the user to track their completion of the activities. Event history is also available in chronological order, for the user to track their past activities.

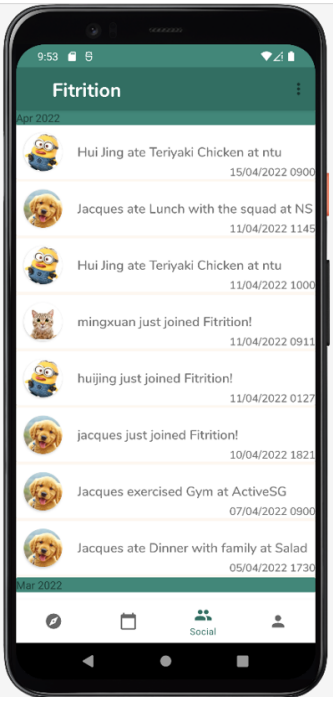


Figure 1.7: Social Page

Figure 1.7 shows the user interface for Social Page. The page displays the user's friends' status updates. The friend's profile page (Figure 1.11) will be shown upon clicking on the status.

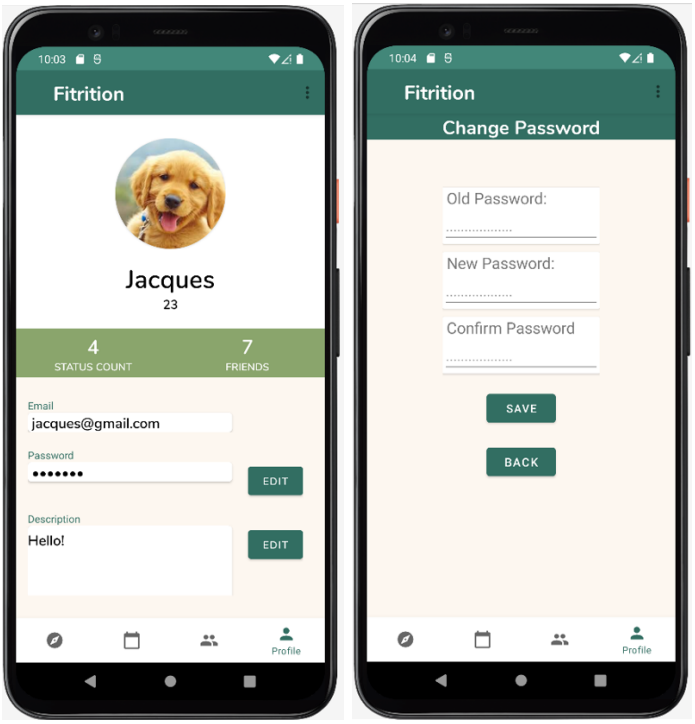


Figure 1.8: Profile Page

Figure 1.8 shows the user interface for Profile Page. The page displays the detailed information of the user such as name, age, status count and number of friends. Editing of information is also allowed.

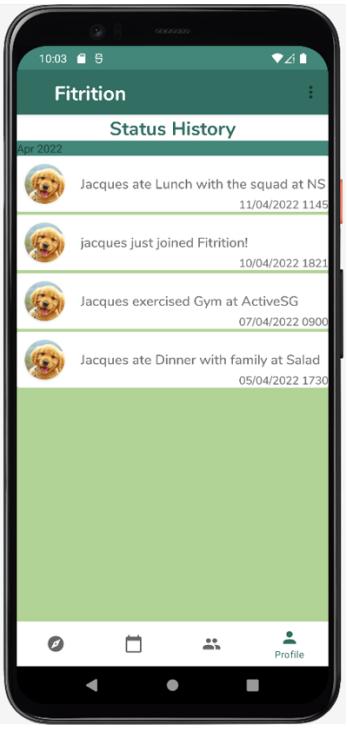


Figure 1.9: Status History Page

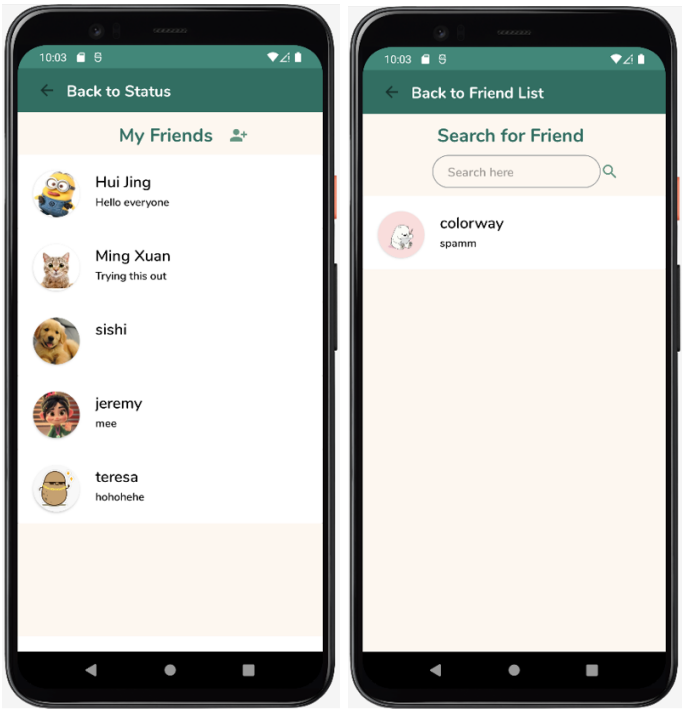


Figure 1.10: My Friends Page

Both Status History and My Friends Pages can be accessed from the Profile Page. Status History Page displays the past status of the user (Figure 1.9) while My Friends Page shows the list of friends the user is connected with (Figure 1.10).

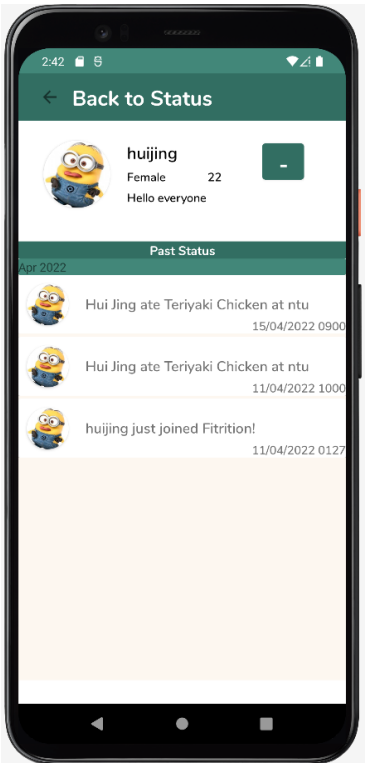


Figure 1.11: Friends' Profile Page

Figure 1.11 shows the user interface of Friends' Profile Page. The user can be brought to this page through Social Page and Profile Page while adding friends. Actions such as adding and disconnecting with friends can be completed at this page.

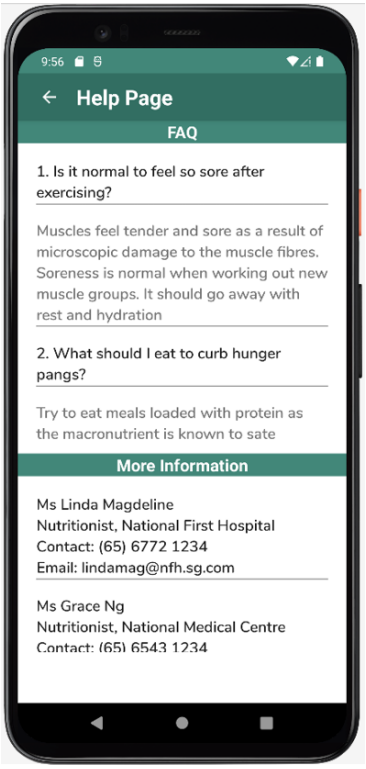


Figure 1.12: Get Help Page

Figure 1.12 shows the user interface for Get Help Page. This page displays professional health and fitness advice. It consists of answers to frequently asked questions as well as contact information of professionals to help the user in their fitness journey.

2. Functional Requirements

1. Register

1.1. The user must be able to register for a new account via our system registration.

1.1.1. Information to include:

1.1.1.1. Profile photo
1.1.1.2. Username
1.1.1.3. Name
1.1.1.4. Valid Email Address
1.1.1.5. Password & Confirm Password
1.1.1.6. Date of Birth
1.1.1.7. Gender
1.1.1.8. Personal Description

1.2. The system must validate all required fields have been filled up.

1.2.1. The system must ensure that the username is not taken by other users in the database system.

1.2.2. The system must ensure that the length of the username is at least 6 characters.

1.2.3. The system must ensure that the email address is not taken by other users in the database system.

1.2.4. The system must ensure that the length of the password is at least 6 characters.

2. Log In

2.1. The user must be able to login using their email address and password.

2.1.1. The system must validate that both fields have been filled up.

2.1.2. The system must validate that the email address is in a valid email format consisting of a local-part, @symbol and domain name.

2.1.3. The system must validate that the user email address and password combination exists in the database.

2.2. If the login information is wrong,

2.2.1. The system must display an error message indicating that the input is invalid

2.2.2. The system must prompt the user to re-enter the fields.

2.3. The system must redirect the user to the application's calendar page.

2.4. The user must be able to request a change of password should he forget his password.

2.4.1. The system must send a password reset email upon the user's request for change of password to the registered email.

2.4.2. The system must validate that the email address exists in the database

3. Navigation Bar

3.1. Upon clicking on the 'Explore' tab, the system must redirect the user to the Explore page.

3.2. Upon clicking on the 'Calendar' tab, the system must redirect the user to the Calendar page.

3.3. Upon clicking on the 'Social' tab, the system must redirect the user to the Social page.

3.4. Upon clicking on the 'Profile' tab, the system must redirect the user to the Profile page.

3.5. Upon clicking on the 'Help' under the drop-down menu, the system must redirect the user to the Get Help page.

3.6. Upon clicking on the 'Log Out' under the drop-down menu,

3.6.1. The system must end the user's session.

3.6.2. The system must redirect the user to the Login page.

4. Explore Page

4.1. The system must ask the user's permission to access his device's location at the first runtime.

4.2. The system must display a map with markers of eateries and fitness facilities as well as a 'CentreLocation' button.

4.3. Upon clicking on the 'CentreLocation' button, the system must display the zoomed in map from the user's location with eateries and fitness facilities within 2km, according to the user's current GPS location.

4.3.1. The blue marker shall represent the user's location.

4.3.2. The red markers shall represent the eateries.

4.3.3. The purple markers shall represent the fitness facilities.

4.4. The user must be able to drag the map manually with 1 finger to pan the map around.

4.5. The user must be able to zoom in and out the map manually by having 2 fingers to pinch and expand the intended areas.

4.6. The user must be able to select the markers on the map.

4.7. Upon clicking on each marker, the system must display the information of the eateries or fitness facilities.

4.7.1. Information includes:

4.7.1.1. Photo

4.7.1.2. Name

4.7.1.3. Opening and Closing Hours

4.7.1.4. Rating

4.7.1.4.1. Rating must range from 1-star to 5-stars

4.7.1.4.2. The system must display ratings in steps of 0.5 (e.g. 4.5 stars).

4.7.1.5. Description

4.8. The user must be able to view healthy food options for eateries.

4.8.1. Information includes:

4.8.1.1. Name of dish

4.8.1.2. Description of dish

4.8.1.3. Calories of dish

4.9. The user must be able to add the event from facilities to the calendar after inputting the following information:

4.9.1. Options: Dine or Exercise
4.9.2. Name of event
4.9.3. Location
4.9.4. Time
4.9.5. Date

5. Calendar Page

5.1. The system must be able to display the calendar in a monthly format relative from the current date.

5.2. Upon clicking on 'Add Event', the user must be able to add an event upon clicking on a date.

5.2.1. Options: Dine or Exercise
5.2.2. Name of event
5.2.3. Location
5.2.4. Time
5.2.5. Date

5.3. Upon clicking on 'Events History', the user must be able to view their events history.

5.4. Upon clicking on 'Delete', the user must be able to delete their events from the calendar.

5.5. The user must be able to tick the checkbox an hour before the event time.

5.5.1. The system must display the updated status in the user's friends' Social Page (in Section 6) and status history of his Profile Page (in Section 7)

6. Social Page

6.1. The system must display the status of the users' friends in chronological order.

6.2. Upon pulling down on the screen/refreshing, the system must display the updated status.

6.2.1. Information includes:

6.2.1.1. Profile photo
6.2.1.2. Name
6.2.1.3. Activity completed
6.2.1.4. Date posted
6.2.1.5. Time posted

6.3. Upon clicking on each status post, the user must be able to view the friend's profile page.

7. Profile Page

7.1. The system must display the following information:
- 7.1.1. Profile photo
- 7.1.2. Name
- 7.1.3. Age
- 7.1.4. Number of status count
- 7.1.5. Number of friends
- 7.1.6. Email address
- 7.1.7. Encrypted password
- 7.1.8. Short description

7.2. The user must be able to edit the following information:
- 7.2.1. Profile photo
- 7.2.2. Email address
- 7.2.3. Encrypted password

7.3. Upon clicking on 'Status Count', the user must be able to view the user's status history.

7.4. Upon clicking on 'My Friends', the user must be able to view the user's list of friends.

7.5. Upon clicking on 'Add Friends', the user must be able to view a list of friends in the database.

7.6. Upon clicking on a specific friend, the user must be able to view the friend's profile page.
- 7.6.1. Upon clicking on '+', the user must be able to add the friend.
- 7.6.2. Upon clicking on '-', the user must be able to disconnect with the friend.

7.7. The user must be able to search for friends via the search bar.
- 7.7.1. The system must validate that the user inputs a valid username
  - 7.7.1.1. Upon submitting an invalid username, the system must display an error message indicating that no such user exists in the database.
  - 7.7.1.2. Upon submitting a valid username, the system must display a success message.

8. Get Professional Help Page

8.1. The system must display a list of commonly asked questions with their answers.

8.2. The system must display the following information available for the user to contact for help.
- 8.2.1. Name of Professionals
- 8.2.2. Professional's Organisations
- 8.2.3. Mobile Number of Professionals
- 8.2.4. Email Address of Professionals

9. Interface with other systems

9.1. The system must be able to retrieve location from the user's device via Google's Geolocation API.

9.2. The system must be able to retrieve the list of eateries and fitness facilities from Google Places API.

9.3. The system must be able to display a map consisting of all the eateries and fitness facilities within a 2km distance from the user.

9.4. The system must be able to authenticate user access privilege via Firebase Auth.

9.5. The system must be able to retrieve user information via Firebase RealTimeDatabase.

9.6. The system must be able to retrieve friend information via Firebase RealTimeDatabase.

9.7. The system must be able to update user information via Firebase RealTimeDatabase.

9.8. The system must be able to update friend information via Firebase RealTimeDatabase.

2.1 Use Case Diagram

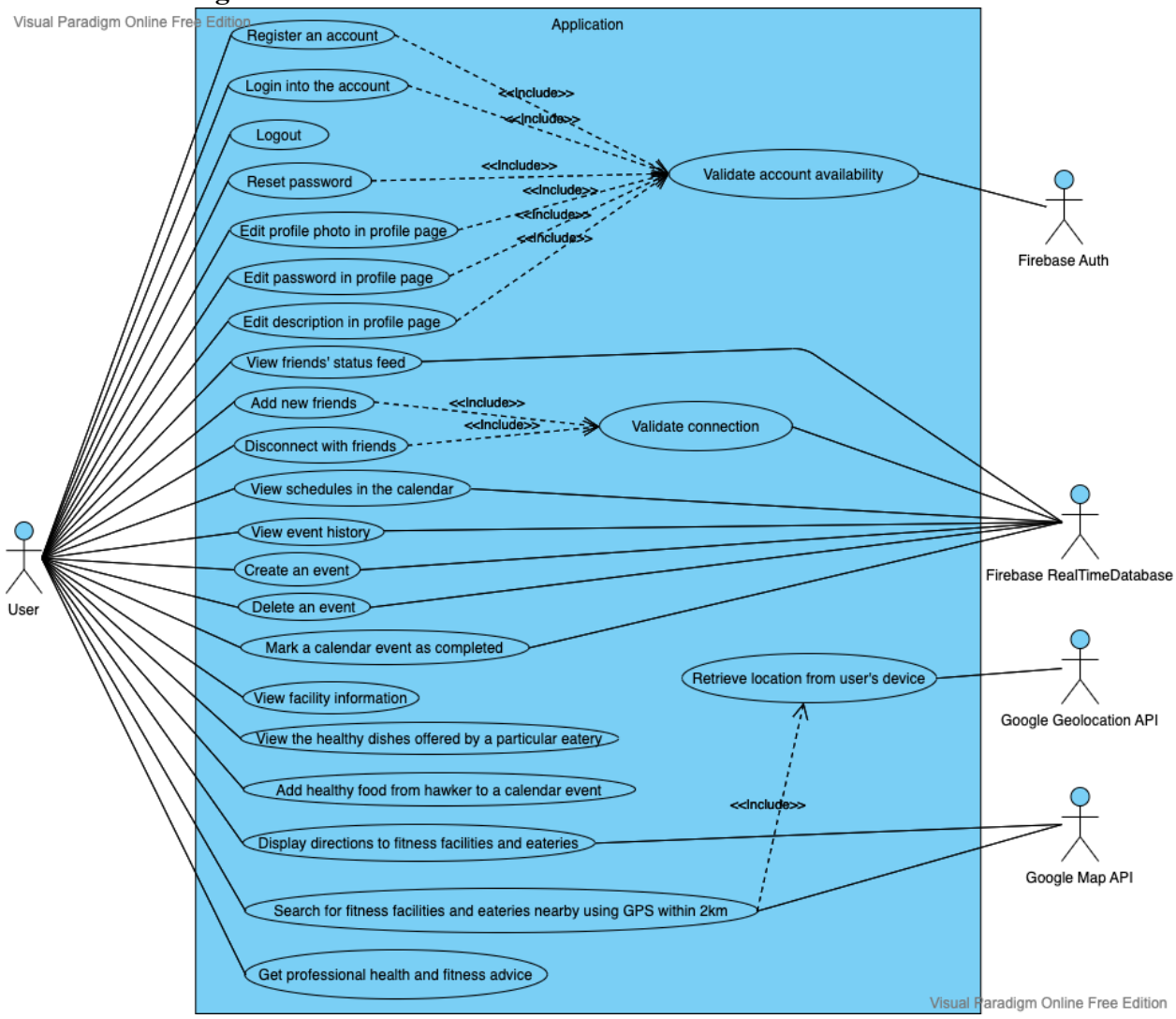


Figure 2.1: Use Case Diagram

2.2 Use Case Summary

Use Case ID	Description
UC001	Register an account
UC002	Login into the account
UC003	Logout
UC004	Reset password
UC005	View friends' status feed
UC006	Add new friends
UC007	Disconnect with friends
UC008	View schedules in the calendar
UC009	View event history
UC010	Create an event
UC011	Delete an event
UC012	Search for fitness facilities and eateries nearby using the GPS within 2km
UC013	View facility information (Rating, description, etc)
UC014	View the healthy dishes offered by a particular eatery
UC015	Add event from explore page to a calendar event
UC016	Display directions to fitness facilities and eateries
UC017	Get professional health and fitness advice
UC018	Edit profile photo in profile page
UC019	Edit password in profile page
UC020	Edit description in profile page
UC021	Mark an event as completed

2.3 Use Case Descriptions

Use Case ID:	**UC001**		
Use Case Name:	Register an account		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	16-04-22

Actor:	Individual User
Description:	The user can register a new account
Preconditions:	1. The user has downloaded and started the system
Postconditions:	1. The system has created the User account 2. The system explains to the user why the account could not be created
Priority:	Low
Frequency of Use:	Low
Flow of Events:	1. The user selects “Register” at the login page. 2. The system redirects to the account creation page and requests for the following inputs: a. Profile Photo b. Username c. Name d. Valid Email Address e. Password f. Confirm Password g. Date of Birth h. Gender i. Description 2. User inputs the required fields 3. The system validates the password and confirm password fields by confirming they are identical. 4. If the information provided is verified, the system sends a request to the database server to create the user account 5. The system redirects the user to the explore page
Alternative Flows:	AF-S1: If the input is invalid for any of the following reasons. Invalid email address format, Password that is less than 8 characters, Email address is already registered with an account in the database server or any text box left blank. 1. The system will display an error message

	2. The system directs user back to the account creation page in step 2
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC002**		
Use Case Name:	Login into the account		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	16-04-22

Actor:	Individual Users
Description:	Perform a login procedure to allow user data and preferences to be saved in the device and in the cloud server
Preconditions:	1. The user is not logged in
Postconditions:	1. The user is logged in and redirected to the explore page
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The user inputs the email address and password at the login page 2. The system will validate information with the database server 3. If the information is verified, the system will load the appropriate user profile into the device 4. The system redirects the user to the explore page
Alternative Flows:	AF-S3: If username or password is not valid: 1. The system displays an error message as a toast message 2. The user goes back to step 1 AF-S3:If the email address input does not exist in the database: 1. The system will display the appropriate error message for 2 seconds 2. The system will require the user to the “register new account” 3. The user returns to step 1 and checks the input
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC003**		
Use Case Name:	Logout		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	16-04-22

Actor:	Individual User
Description:	The user can log out of the system
Preconditions:	1. The user is logged in
Postconditions:	1. User logs out and return to the login page
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The user clicks the more options button at the top right corner of the app 2. The system shall display a pop up window with the log-out option 3. The user clicks on the logout button 4. The system will end the user session from the system and direct the user to the login page
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC004**		
Use Case Name:	Reset password		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	16-04-22

Actor:	Individual User
Description:	The user forgets password
Preconditions:	1. The user has an account registered
Postconditions:	1. The system sends an email detailing the steps to reset the password to the user's registered email address
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. The user clicks on the "Forgot Password" button at the login page 2. The system redirects the user to the reset password page and requests the user to input their email address 3. The user inputs their email address 4. The system will verify the email address with the account information in the database 5. If the system verifies the email address, the system will send an email to the user detailing the steps to reset the password 6. The user exits the system and looks up the email to reset their password 7. The user will be shown a confirmation page upon successful password reset
Alternative Flows:	AF-S5: If the email address does not exist in the database 1. The system will display an the appropriate error message 2. The system redirects to step 3
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC005**		
Use Case Name:	View friends' status feed		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	16-04-22

Actor:	Individual User
Description:	Social Page is filled with cards which detail the individual user's and friends' pertinent activities.
Preconditions:	1. The user has logged in
Postconditions:	1. The system displays status cards to the user
Priority:	Medium
Frequency of Use:	High
Flow of Events:	1. The user selects social page icon from bottom bar 2. Social page is displayed with list of cards in chronological order detailing friend's activities
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	Individual User and friends are actively using the app
Notes and Issues:	-

\

Use Case ID:	**UC006**		
Use Case Name:	Add new friends		
Created By:	Sishi	Last Updated By:	Sishi
Date Created:	01-04-22	Date Last Updated:	15–04-22

Actor:	Individual User
Description:	The user adds a new friend to be able to view their social feed
Preconditions:	1. The user and their friends both must have an account 2. The user and the user they want to be friends with must not already be friends
Postconditions:	1. The user has a new friend
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The user clicks on “Friends” at the Profile page 2. The system redirects the user to the “My Friends” page 3. The user clicks on the “add new friend” icon on the “My Friends” page 4. The system will redirect the user to the “Search for Friend” page 5. The user clicks on a card displaying a foreign user’s profile from a list of cards 6. The system will redirect to the foreign user’s profile 7. The user clicks on “+” icon on the foreign user’s profile 8. The system will add the foreign user to the user’s friend list in the database. The system will add the user to the foreign user’s friend list in the database. 9. The system will display “-” instead of “+” icon to the user on the newly added friend’s profile
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC007**		
Use Case Name:	Disconnect with friends		
Created By:	Sishi	Last Updated By:	Sishi
Date Created:	15–04-22	Date Last Updated:	15–04-22

Actor:	Individual User
Description:	The user disconnects with a friend
Preconditions:	1. The user and the user to be disconnected must already be friends 2. The user must be at the friend's profile page
Postconditions:	1. The user and the other user are no longer friends
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The user clicks on "-" icon on the current friend's profile page 2. The system will prompt the user with a confirmation dialogue message "Are you sure?" 3. If the user clicks on "Yes", the system will remove the current friend from the user's friend list in the database. 4. The system will display "+" instead of "-" icon to the user on the removed friend's profile
Alternative Flows:	AF-S3: If the user clicks on "No" 1. The user and current friend remain friends. 2. The user goes back to step 1
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC008**		
Use Case Name:	View schedules in the calendar		
Created By:	Jeremy	Last Updated By:	
Date Created:	01-02-22	Date Last Updated:	

Actor:	Individual User
Description:	The user views the following in a calendar format: 1. Events
Preconditions:	1. The user is logged in
Postconditions:	1. The system displays calendar and events for the day selected
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The user selects the Calendar Page from bottom navigation bar 2. The user selects the date of activity to display events for that day
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC009**		
Use Case Name:	View event history		
Created By:	Jeremy	Last Updated By:	
Date Created:	01-02-22	Date Last Updated:	

Actor:	Individual User
Description:	The user is able to view the history of all events created locally in a chronologically ordered list.
Preconditions:	1. The user is logged in 2. The user is at calendar page
Postconditions:	2. The system displays calendar and events for the day selected
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The system uses the included use case “View schedules in the calendar” to arrive at the calendar page. 2. The user clicks on the “Events History” Button 3. The system displays all events created by the local user in chronological order.
Alternative Flows:	-
Exceptions:	-
Includes:	8: View schedules in the calendar
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC010**		
Use Case Name:	Create an event		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	24-01-22

Actor:	Individual User
Description:	The user creates an event through the calendar page
Preconditions:	1. The user is at the calendar page
Postconditions:	1. Event is saved to the database attache to the current user 2. Checkbox of event added is not ticked
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The system uses the included use case "View schedules in the calendar" to arrive at the calendar page. 2. The user selects the month and date of activity from the calendar interface. 3. The user selects the "Add Event" button on the calendar page 4. The system displays a pop-up for input 5. The system requests the input of the following information from the user: a. The name of the event b. The location of the event c. The time of the event in 24 hr format d. The date of the event in "DDMMYYYY" format 6. The user inputs the necessary information and clicks the "Save Event" button 7. The system validates the input 8. If the input is valid, the system saves the activity to the device and database server is updated 9. The user is returned to the calendar page
Alternative Flows:	AF-S6: If the user taps outside of the pop-up 1. The system will discard the event 2. The system returns to step 4 AF-S8: If the input is Invalid: 1. System will display the appropriate error message and prompt the user for the specific input values that is required to fill in properly 2. The system returns to step 5
Exceptions:	-

Includes:	8: View schedules in the calendar
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC011**		
Use Case Name:	Delete an event		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	24-01-22

Actor:	Individual User
Description:	The user records down the food consumed and calories intake
Preconditions:	1. There is at least one event to delete
Postconditions:	1. Event is removed from local device and database document
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The system uses the included use case "Viewing schedules in the calendar to display the current schedule" 2. The user selects a date from the calendar interface 3. The system displays all events for the date selected 4. The user selects the "Delete Event" button 5. The event will be removed from the database and the phone 6. The system will display the message "Event deleted successfully"
Alternative Flows:	-
Exceptions:	-
Includes:	8: Viewing schedules in the calendar
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC012**		
Use Case Name:	Search for Fitness Facilities and Eateries nearby using the GPS within 2km		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	24-01-22

Actor:	Individual User
Description:	The user is shown fitness facilities, restaurant and eateries within 2km radius
Preconditions:	1. GPS must be on with permission access granted to the system 2. The user is logged in
Postconditions:	1. Map displays the facilities within 2km from the user's location
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The user selects the "Centre Location" button 2. Map zooms in on the user's location 3. Markers representing facilities within 2km from the user's GPS location are shown
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC013**		
Use Case Name:	View facility information (Rating, description, etc)		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	24-01-22

Actor:	Individual User
Description:	The user views facility information
Preconditions:	1. The user is logged in
Postconditions:	1. System displays facility information
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The system uses the included use case "Search for fitness facilities and eateries nearby using the GPS with 2km" to hone the map to the user location and to display markers. 2. The user clicks on a red or purple map marker 3. The system displays a page with detailed information of the facility and reviews of the facility:
Alternative Flows:	-
Exceptions:	-
Includes:	12: Search for Fitness Facilities and Eateries nearby using the GPS within 2km
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC014**		
Use Case Name:	View the healthy dishes offered by a particular eatery		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	24-01-22

Actor:	Individual User
Description:	The user can read detailed information of healthy food options provided by eatery
Preconditions:	1. The user is logged in 2. Map marker clicked by the user clicked belongs to an eatery
Postconditions:	-
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The system uses the included use case “View facility information” to display the information of the eatery 2. The user clicks on the “Click to view healthy food” button 3. The system will display the user a scrollable list of the food items that are available with its required information
Alternative Flows:	-
Exceptions:	-
Includes:	13: View facility information (Rating, description, etc)
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC015**		
Use Case Name:	Add event from explore page to a calendar event		
Created By:	Jeremy	Last Updated By:	Jeremy
Date Created:	24-01-22	Date Last Updated:	24-01-22

Actor:	Individual User
Description:	The user records down the food consumed and calories intake
Preconditions:	1. The user is logged in
Postconditions:	1. Food Journal is saved to the device and backed up to the database server 2. Checkbox is not ticked
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The system uses the included use case "View the healthy dishes by a particular eatery" to display healthy food options provided by eatery 2. The user selects the "Click to add to event" button on the calendar page 3. An add event pop-up appears for the user to input planned event details 4. The system requests the input of the following information from the user: a. The time of the event in 24 hr format using a time picker b. The date of the event in "DDMMYYYY" format 5. The user inputs the necessary information 6. If the user selects "Save Event", the system saves the activity to the device and database server is updated 7. The user is returned to the view of healthy dishes page
Alternative Flows:	AF-S5: If the user taps outside the add event pop-up or uses the phone back button. 1. The system discards the Food Journal entry 2. The system directs the user back to the page displaying the healthy food offered by a particular eatery in step 1
Exceptions:	-
Includes:	14: View the healthy dishes offered by a particular eatery
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC016**		
Use Case Name:	Display directions to fitness facilities and eateries		
Created By:	Jeremy	Last Updated By:	
Date Created:	01-04-22	Date Last Updated:	

Actor:	Individual User
Description:	The user gets directions to the desired fitness centre or hawker centre from its current location
Preconditions:	1. GPS is turned on
Postconditions:	1. Android system is at "Google Map" with directions outlined to get the user from current location to the desired facility
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1) The system uses the included use case "Search for fitness facilities and eateries nearby using the GPS with 2km" to hone the map to the user location and to display markers. 2) The user selects the map marker of the desired location 3) The user selects the back button to return to the map 4) The user clicks the directions logo found at the bottom right corner of the map 5) The user is directed to Google Maps with directions from its current location
Alternative Flows:	-
Exceptions:	-
Includes:	12: Search for Fitness Facilities and Eateries nearby using the GPS within 2km
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC017**		
Use Case Name:	Getting Professional Health and Fitness Advice		
Created By:	Ming Xuan	Last Updated By:	
Date Created:	24-01-22	Date Last Updated:	24-01-22

Actor:	Individual User
Description:	The user gets professional health and fitness advices
Preconditions:	1. The user is logged in
Postconditions:	1. The system displays the information which consists of the professional advice
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The user selects the icon with a vertical 3 dot on the right of the top bar 2. The system displays the dropdown box, with the option to “help” 3. The user clicks on the option “Help” 4. The user is directed to the “Get Professional Help” page 5. The system displays a list of frequently asked questions (FAQ) and answers 6. The user will scroll down the FAQ list to discover more FAQs 7. The system displays contact information of professionals at the bottom 8. The user will scroll down the contact information list to discover more contact information
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC018**		
Use Case Name:	Edit profile photo in profile page		
Created By:	Jeremy	Last Updated By:	
Date Created:	01-04-22	Date Last Updated:	

Actor:	Individual User
Description:	The user wishes to edit their profile picture
Preconditions:	1. The user is logged in
Postconditions:	1. The user profile picture is updated
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. The user clicks on the profile page via bottom navigation bar 2. The system redirects the user to the profile page 3. The user clicks on profile photo 4. The system shows an image selection picker where the user may choose from their recent photos or search for their photo in another app 5. The user selects a photo 6. The system will prompt a confirmation message 7. If the user selects “Yes” 8. The system prompts a toast message “Uploading” 9. The system uploads the updated profile picture to the Firebase Database 10. The user is directed back to the profile page with the updated profile picture
Alternative Flows:	AF-S7: If the user selects “No” or the user taps outside of the confirmation box: 1. The profile picture is not updated 2. The user is directed back to the profile page at step 1
Exceptions:	
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC019**		
Use Case Name:	Edit password in profile page		
Created By:	Jeremy	Last Updated By:	
Date Created:	01-04-22	Date Last Updated:	

Actor:	Individual Users
Description:	The user wishes to change their password
Preconditions:	1. The user is logged in
Postconditions:	1. The system displays the message “password changed”
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. The user clicks on the profile page via bottom navigation bar 2. The system redirects the user to the profile page 3. The user clicks on the edit button next to the password tab 4. The system will display a page requiring the user to input the following a. Old password b. New password c. Confirm password 5. The user selects the “Save” button 6. The system will validate the inputs provided 7. The system will update the password into the server 8. The system will prompt a toast message “Password Changed” 9. The user selects the “Back” button 10. The user is directed back to the profile page
Alternative Flows:	AF-S4: If the user selects “Back” instead of “save” 1. The system discards the changes 2. The system directs the user back to the profile page in step 1 AF-S5: If the user input is invalid, the system will prompt a appropriate error message at the input textbox 1. The user is returned to step 4
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC020**		
Use Case Name:	Edit description in profile page		
Created By:	Hui Jing	Last Updated By:	Hui Jing
Date Created:	16-04-2022	Date Last Updated:	16-04-2022

Actor:	Individual Users
Description:	The user wishes to edit their description
Preconditions:	1. The user is logged in
Postconditions:	1. The system displays the message "description updated"
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. The user clicks on the profile page via bottom navigation bar 2. The system redirects the user to the profile page 3. The user clicks on the edit button next to the description tab 4. The system will display a page requiring the user to input the new description 5. The user selects the "Save" button 6. The system will update the description into the server 7. The system will prompt a toast message "Description Updated" 8. The user selects the "Back" button 9. The user is directed back to the profile page
Alternative Flows:	AF-S4: If the user selects "Back" instead of "save" 1. The system discards the changes 2. The system directs the user back to the profile page in step 1
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	**UC021**		
Use Case Name:	Mark an event as completed		
Created By:	Teresa	Last Updated By:	
Date Created:	16-04-2022	Date Last Updated:	16-04-2022

Actor:	Individual User
Description:	The user ticks the checkbox for the event that user has completed
Preconditions:	1. The checkbox is not ticked 2. The even has never been completed
Postconditions:	1. The system will generate a status associated with the event for the user to be seen in friends' social page and user status history
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. The system uses the included use case "View schedules in the calendar" to arrive at the calendar page. 2. The user ticks the checkbox for a particular event 3. System validates that the event ticked is within 1 hour of the time set 4. If the time requirement is met, the checkbox will be ticked
Alternative Flows:	AF-S4: If the time requirement is not met because the event tick is more than an hour into the future 1. The system displays an error message as a toast message 2. Return to step 1
Exceptions:	-
Includes:	8: View schedules in the calendar
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

2.4 Class Diagram

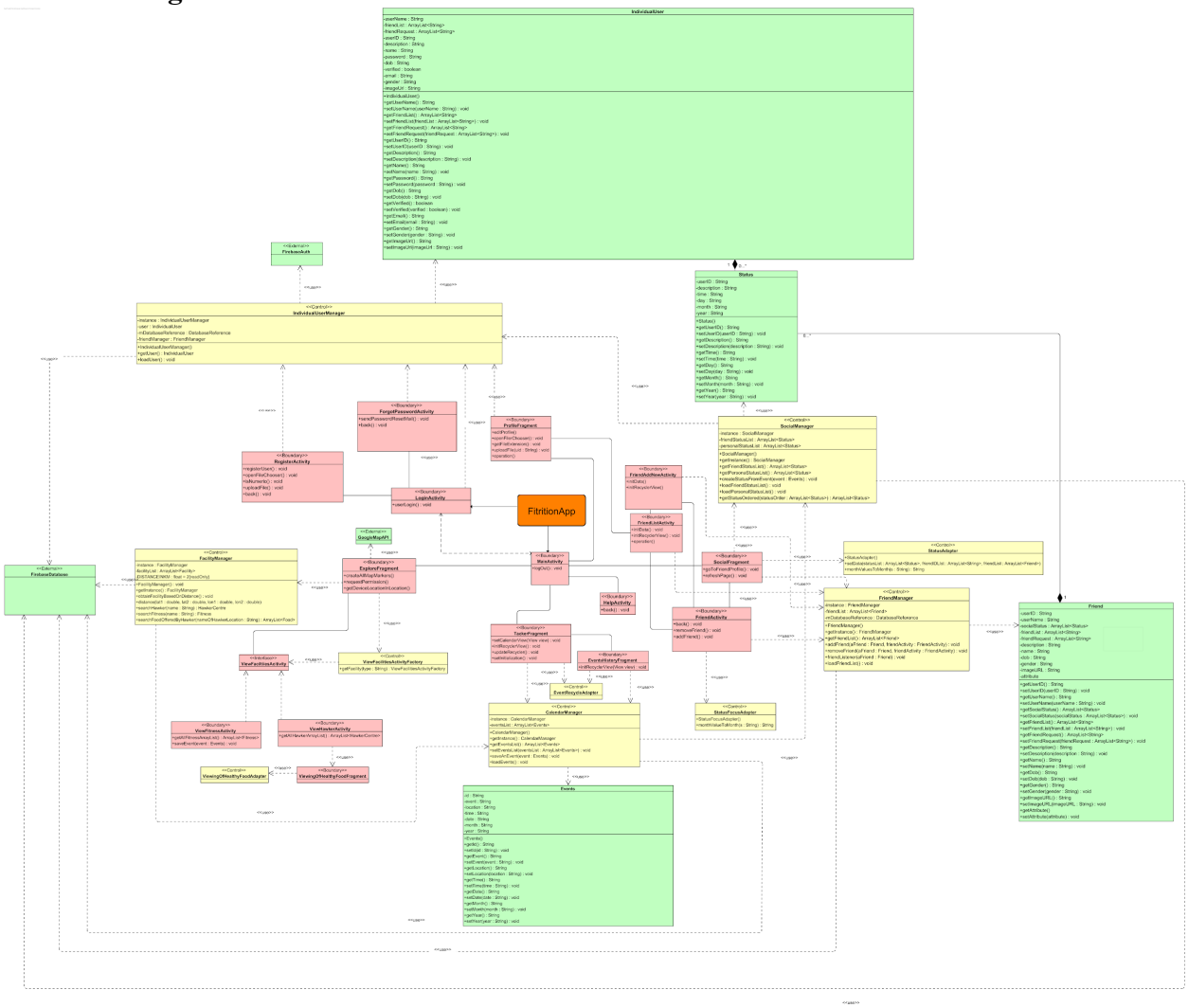


Figure 2.2: Class Diagram

2.5 Sequence Diagram

1) Register an account (UC001)

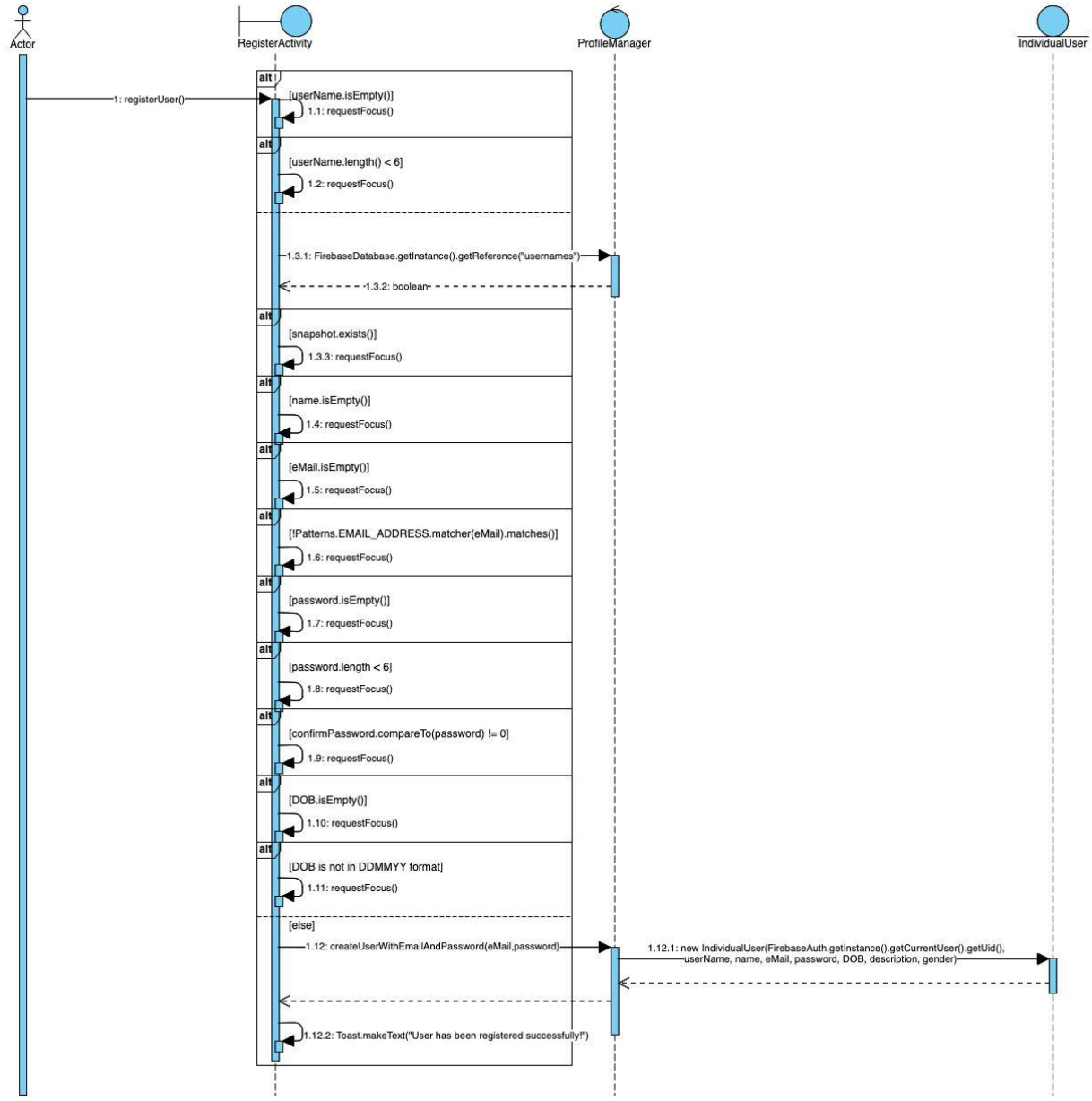


Figure 2.3: Sequence Diagram for UC001

2) Reset password (UC004)

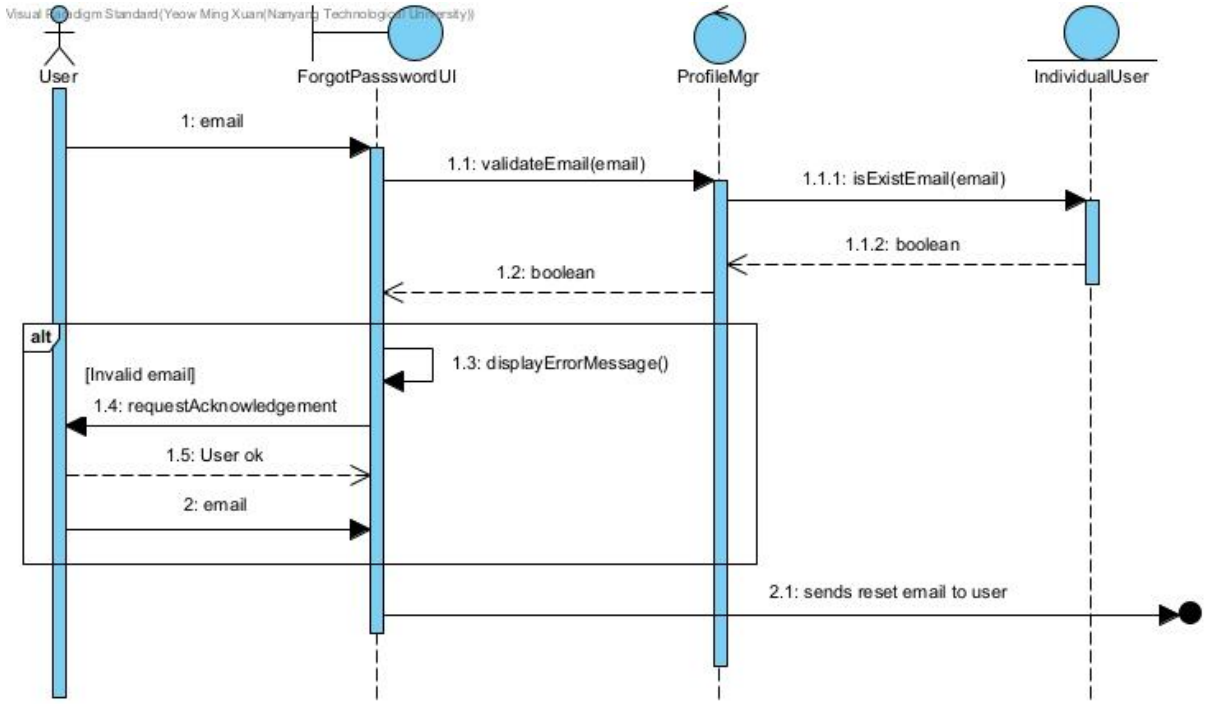


Figure 2.4: Sequence Diagram for UC004

3) View friends' status feed (UC005)

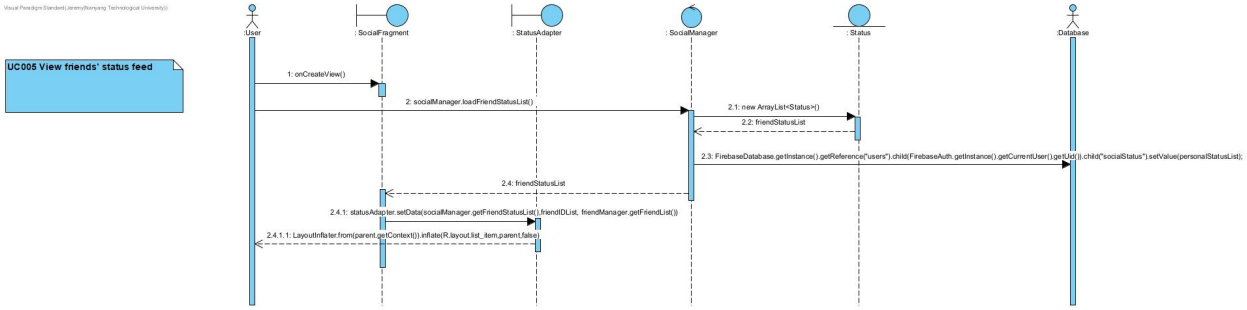


Figure 2.5: Sequence Diagram for UC005

4) Add new friends (UC006)

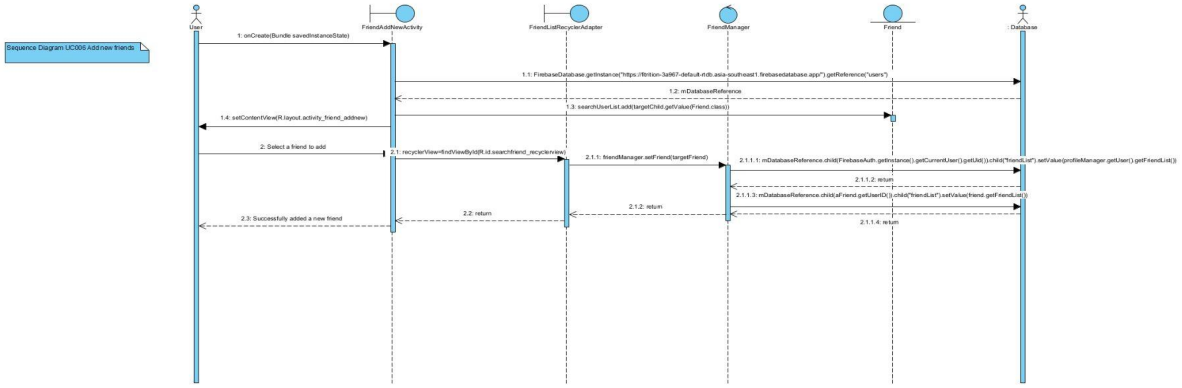


Figure 2.6: Sequence Diagram for UC006

5) Create an event (UC010)

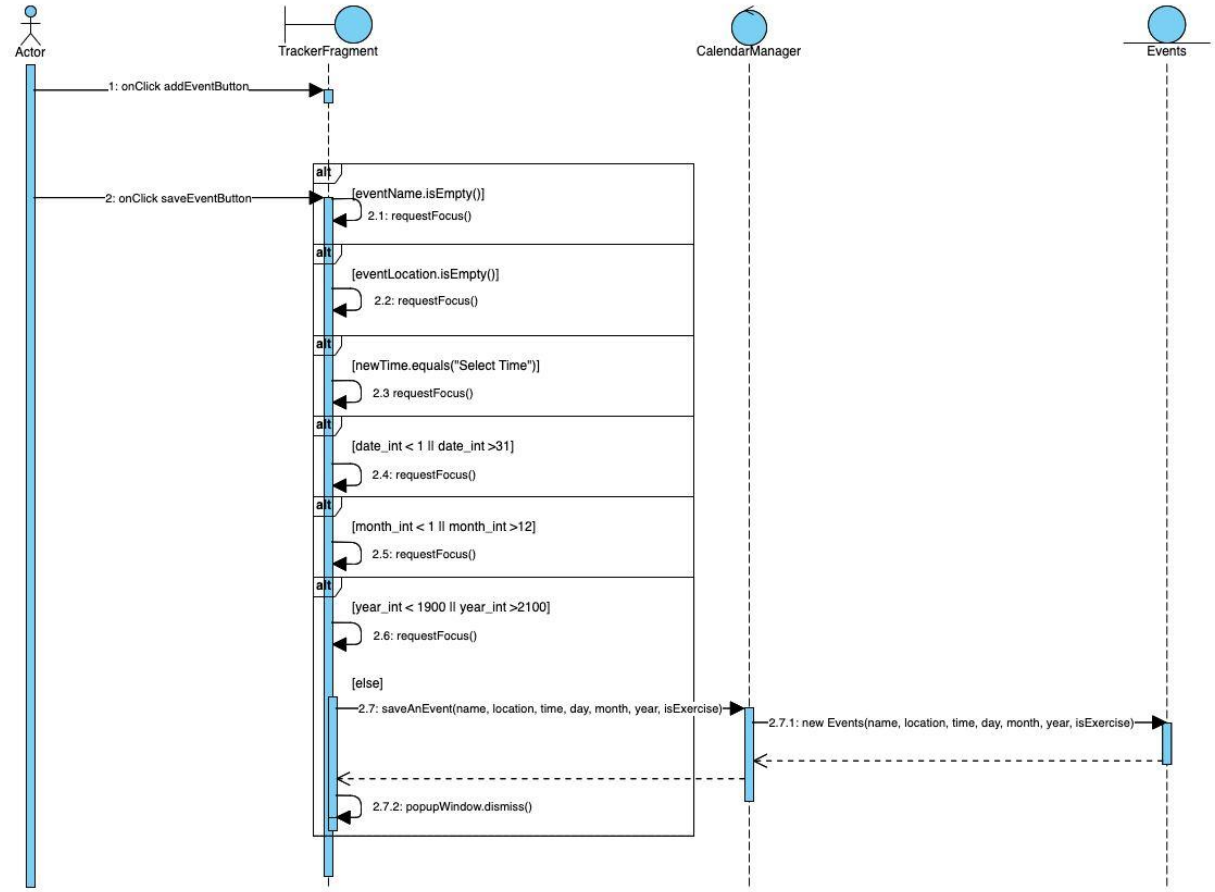


Figure 2.7: Sequence Diagram for UC010

6) Search for fitness facilities and eateries nearby using the GPS within 2km (UC012)

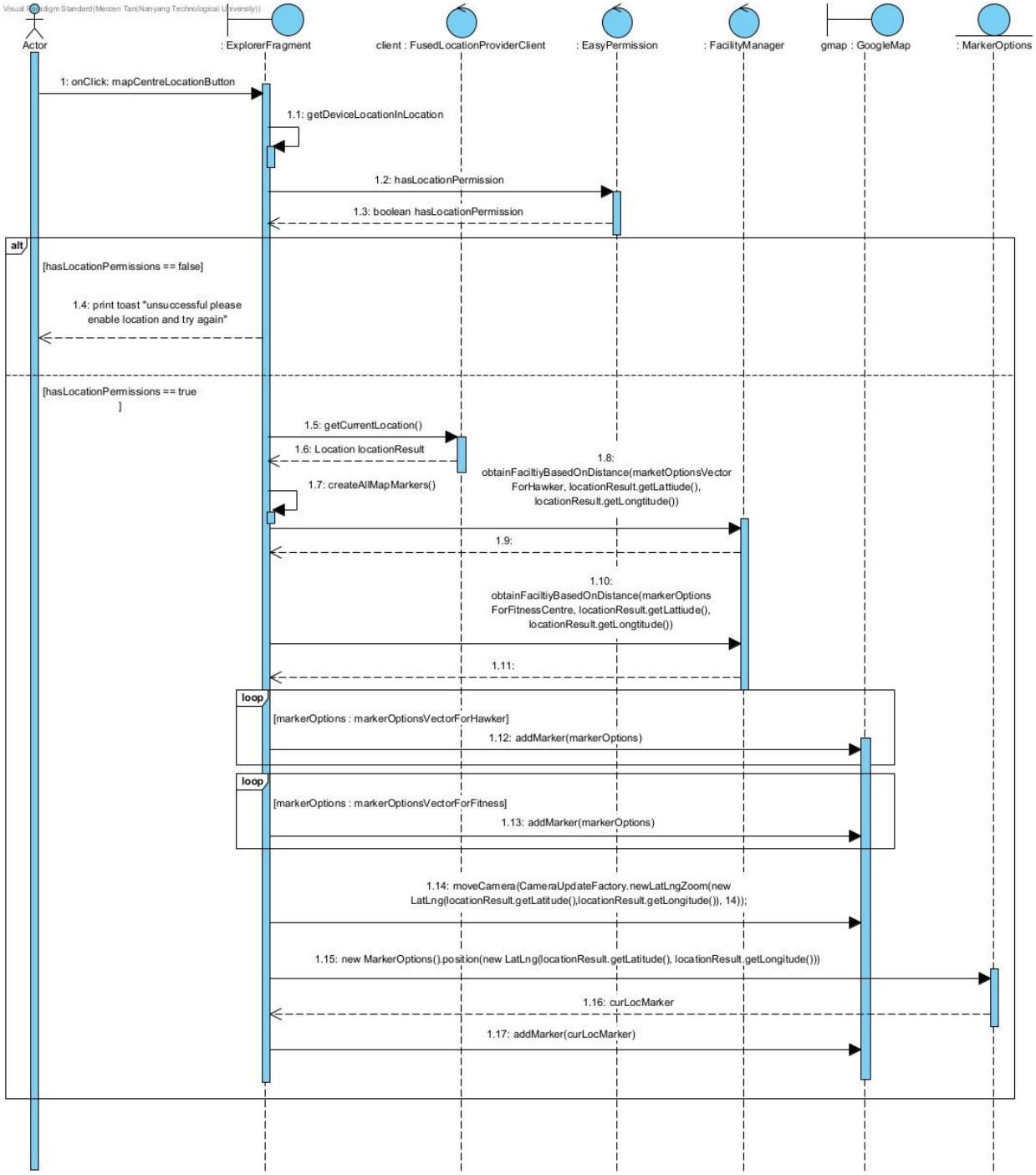


Figure 2.8:Sequence Diagram for UC012

7) View facility information (Rating, description, etc) (UC013)

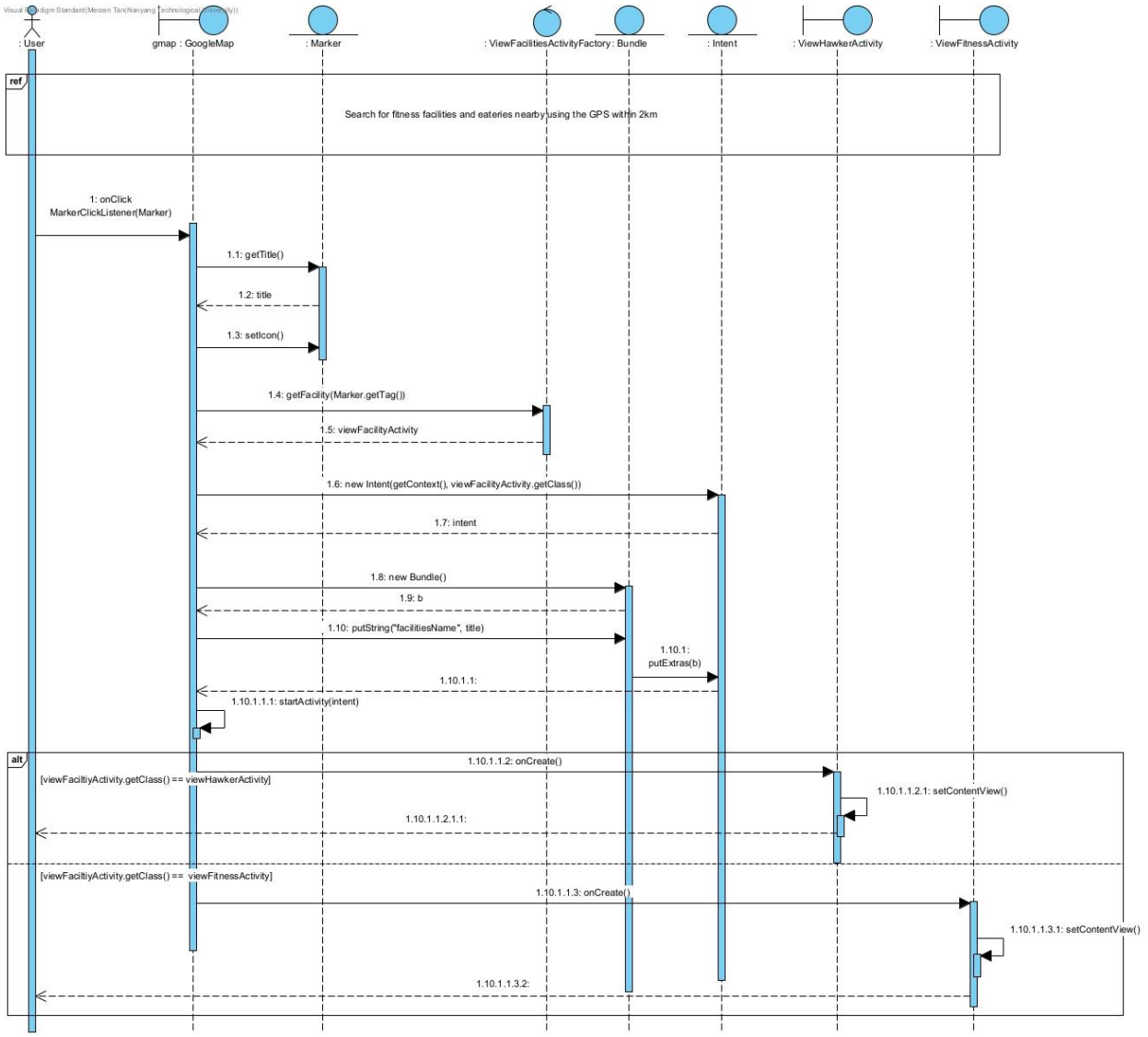


Figure 2.9: Sequence Diagram for UC0013

8) View the healthy dishes offered by a particular eatery (UC014)

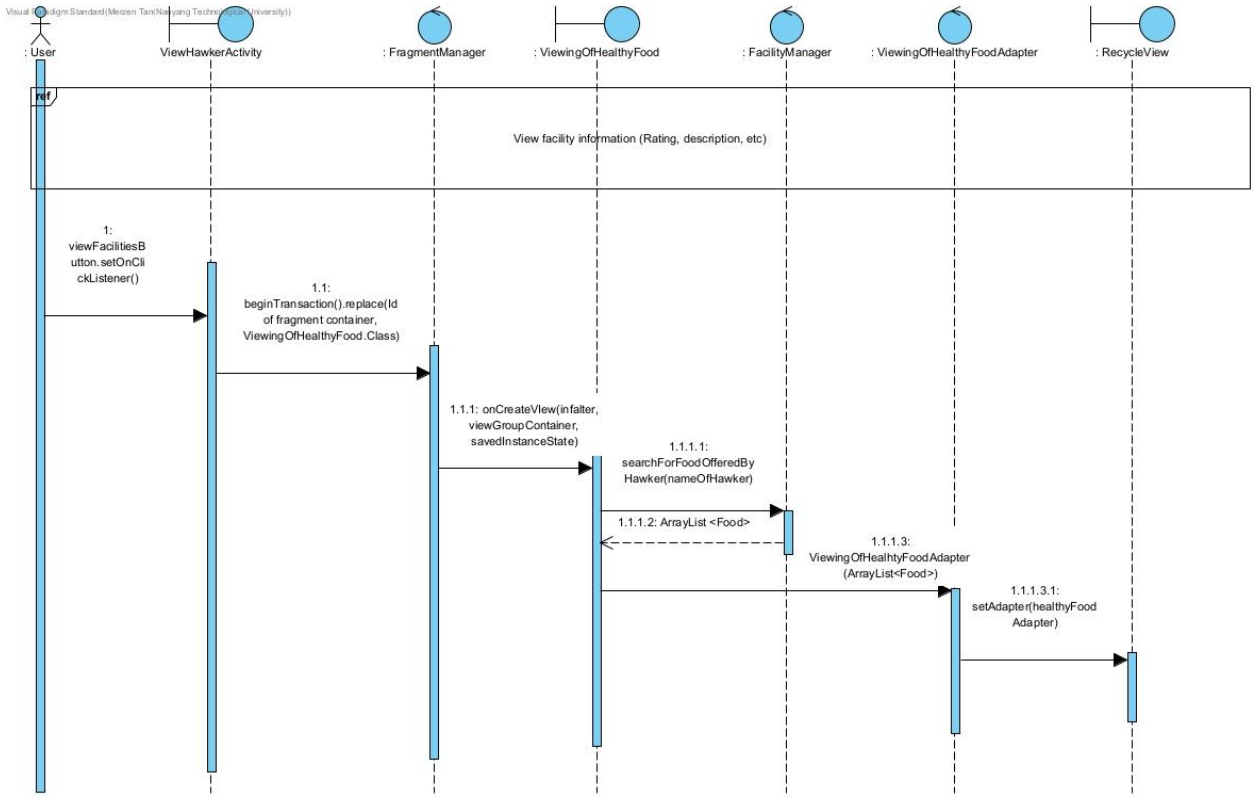


Figure 2.10: Sequence Diagram for UC0014

9) Add event from explore page to a calendar event (UC015)

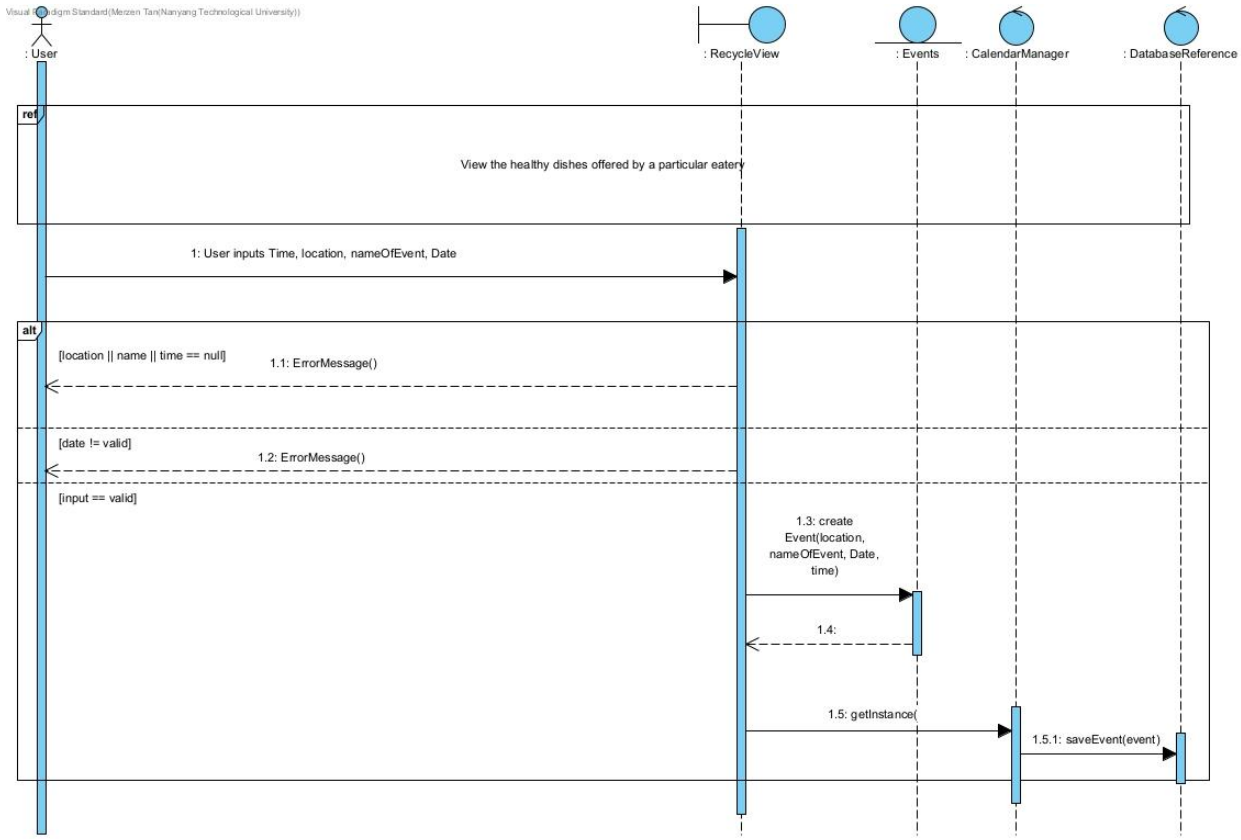


Figure 2.11: Sequence Diagram for UC015

2.6 Dialog Map

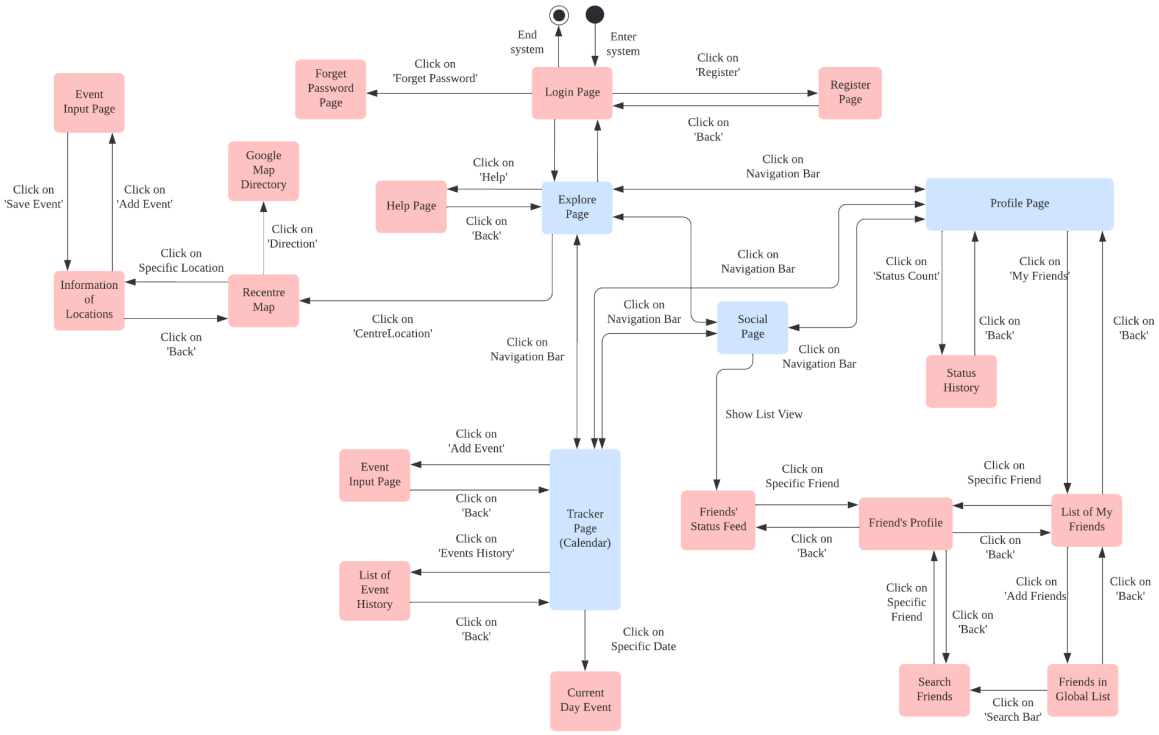


Figure 2.12: Dialog Map

3. Non-functional Requirements

3.1 Performance Requirements

- The system must not crash when the user opens the application.
- The application must acknowledge all user queries and return the necessary display results within 5 seconds.
- Querying of the Google Geolocation API must take less than 7 seconds to respond.
- The application must be able to support at – least 40 queries per minute.
- In case of failure of the application, the platform must be available to the users again within 2 hours.
- The Social page must reflect any new updates from friends within 5 minutes of it happening.
- After a system reboot, the application must fully restore functionality within 2 minutes.
- The system must be able to support internal locus of control.
    - To give the user the sense of control of events occurring and the system will behave as what they expect.
    - Fast responsive time in order to prevent the user from experiencing any lag or latency.
- The system must reduce short-term memory load.

3.2 Security Requirements

- The application utilises the user's location from Google's Geolocation API to denote their current location on the map. No location information is stored in the application's system for safety purposes.
- The application shall not share the user's application usage history with third-party companies.
- The user's account information will not be shared with any third-party companies.
- The system will mask the password field in order to prevent any potential shoulder surfing.

3.3 Usability Requirements

- The application will provide accurate and up-to-date information for all facilities.
- The application must be available to users for at – least 95% of the day.
- The buttons for the major features of the application will be made into icons for better usability and UX.
- The user must be able to understand how to use the various features of the application within the first 5 minutes of seeing the interface.
- The user should be able to reach their profile within 3 clicks.
- The mobile application must be compatible with all devices that are currently running Android 8.0 and above.
- The system must offer informative feedback.
    - To provide necessary feedback to the user when invalid inputs are detected.
    - To display an appropriate error message when certain processes fail.
- The system must strive for consistency.
    - A consistent sequence of actions is required for similar situations.

- A consistent visual layout must be adopted in the application (e.g. labels, fonts, and colours).

3.4 Extendibility Requirements

- The system must be able to maintain with little or no downtime occurring.
- To facilitate easy data access using other platforms, the data collected will be separated from the system and stored in an online database.

4. Interface Requirements

4.1 User

Fitrition works on all types of users who would like to lead healthier lifestyles in terms of physical exercises and food options.

Currently, the Fitrition application does not support special diet filters of nearby eateries (e.g. Vegetarian, Halal, etc).

4.2 Hardware

Fitrition requires to work on the hardware devices with location service enabled to locate the user's current location.

4.3 Software

Fitrition is being designed to work on Android Devices.

4.4 Communication

Fitrition will be accessed over the Internet. All features will be accessible through the application.

5. Architecture Design

5.1 System Architecture Diagram

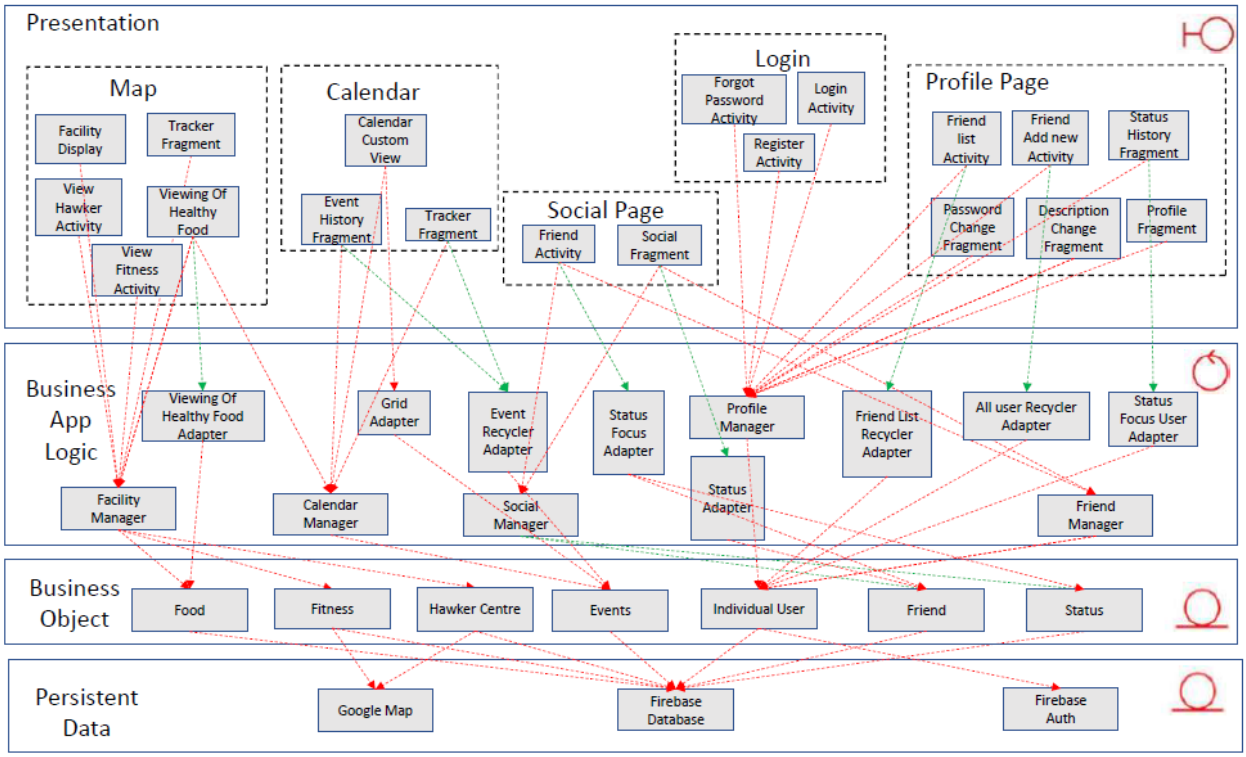


Figure 3.1: System Architecture Diagram

5.2 Design Pattern

1) Strategy Pattern

Problem:

The strategy pattern is ideal for situations where a set of objects should be interchangeable. In this project, the .xml layouts displayed depend on the type of the map marker clicked. (Figure 3.2) Hence the *ViewFacilityActivity* object should be interchangeable depending on what the user clicks to facilitate the upgrading of the application in the future.

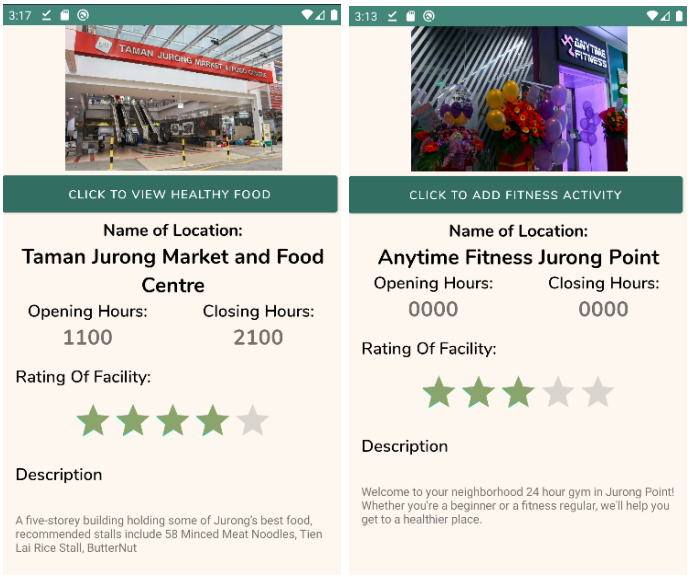


Figure 3.2: Layout for eatery (left) and fitness facility (right)

Solution:

The *ViewFacilitiesActivity* interface is created to interact with the explorer fragment boundary class. (Figure 3.3) It allows switching between the two different activity objects, namely *ViewFitnessActivity* and *ViewHawkerActivity*, as they implement the *onCreate()* function in the interface. Therefore, during runtime, the appropriate *OnCreate()* method will be executed depending on the marker chosen by the user. This enables easy extension of new facilities in the future. For example, if a new *ViewMedicalClinics* activity is to be added in the future, the new activity object can also implement the *onCreate()* function in the interface. Minimal changes are required to be made to the explorer fragment because the implementation of the *onCreate()* method is encapsulated.

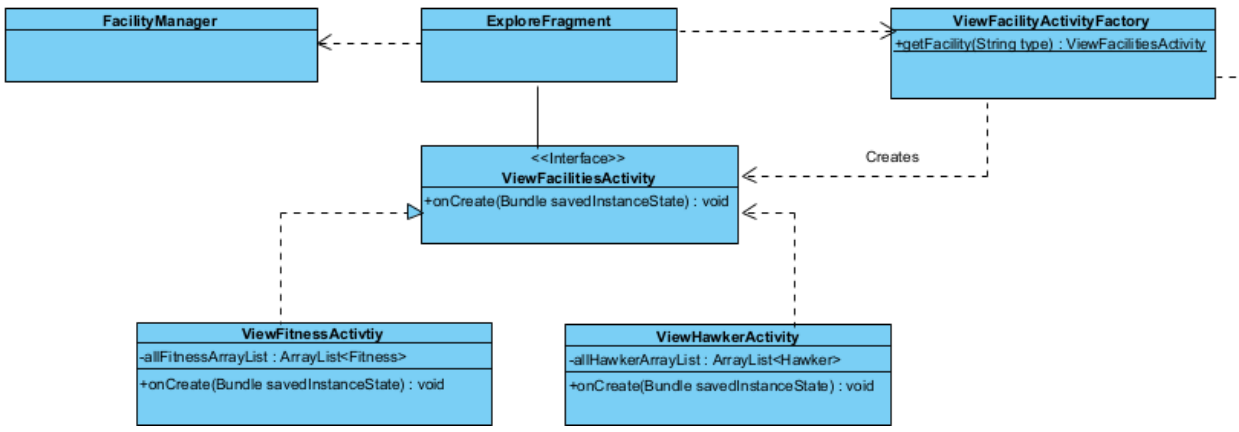


Figure 3.3: Class Diagram that focuses on *ViewFacilitiesActivity* interface

2) Factory Pattern

Problem:

Strategy Pattern is insufficient as the particular *ViewFacilitiesActivity* object to be instantiated will only be known at runtime.

Solution:

The factory method was adopted by defining a *ViewFacilitiyActivityFactory*, which consists of a static getFacility() to instantiate the different *ViewFacilitiesActivity* objects. (Figure 3.4)

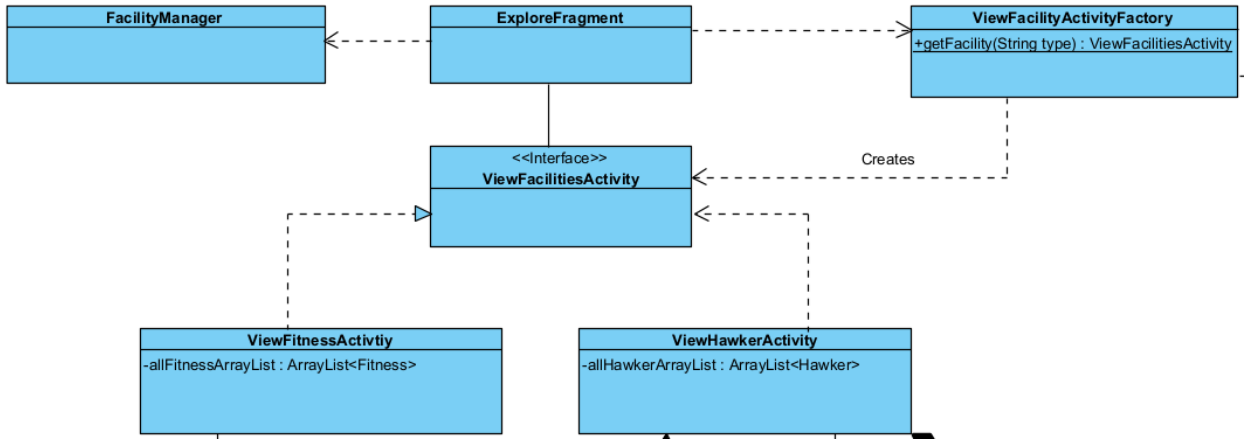


Figure 3.4: Class Diagram that focuses on *ViewFacilitiyActivityFactory*

During runtime, when the explorer fragment wants to use a particular view class, it will call the static method to create the object and retrieve the concrete class. The context can then interact with the concrete product but using the methods of the interface. This reduces the amount of code changes required when a new facility is added. By localising the logic to instantiate the view object, the class selection and object creation are decoupled from the explorer fragment where the object is used. This allows for greater flexibility in object creation.

6. Data Dictionary

Term	Description
User	A user is a person who is using the application to achieve their personal fitness goals and to find facilities that suit their goals in Singapore.
System	A system refers to the Fitrition mobile application.
Facilities	Consist of fitness facilities and eateries.
Eatery	Any place in Singapore where a user can eat - cafes, restaurants, hawker centres, canteens, etc.
Fitness facility	Any place in Singapore where a user can exercise - parks, gym, etc.
Map marker	A map marker is an indication of where the facilities are, on the map. Map markers consist of 4 colours, purple denotes fitness facilities, red denotes eatery, blue denotes your current location and green denotes the current marker that is pressed.
Rating	Rating is a measure of users' satisfaction on a particular area ranging from 1 to 5 stars where 1 depicts very unsatisfactory and 5 depicts very satisfactory.
Event	Refers to a fitness / dining event that can be added either through the calendar page or the map page.
Professionals	Professionals are those who are accredited, such as nutritionists and personal trainers.
GPS	Global Positioning System which will detect a user's current location. It is used interchangeably with Wi-Fi positioning systems.
Search	Search is a feature that allows the users to search friends based on their given input.
Shoulder surfing	Shoulder Surfing is a type of social engineering technique where another person spies on the user when they are typing their credentials to obtain their password.

7. Testing

7.1 Black Box Testing

Equivalence class:

1) Username

Valid class	6 <= input <= 15 characters, input is correct username
Invalid class	Input < 6 characters (lowest of lowest), input > 15 characters (highest of highest); input is wrong username

Boundary value:

1) Username

Valid boundary values	{6, 15}, correct username
Invalid boundary value	{5, 16}, incorrect username

Username	**length**	**Expected Result**	**Actual Result**
testin	6	User has been registered successfully!	User has been registered successfully!
test1	5	Username needs to be at least 6 characters long!	Username needs to be at least 6 characters long!
testing12345678	15	User has been registered successfully!	User has been registered successfully!
testing123456789	16	Username needs to be at most 15 characters long!	Username needs to be at most 15 characters long!

Equivalence class:

2) Password

Valid class	6 <= input <= 20 characters; input is correct password
Invalid class	Input < 6 characters (lowest of lowest), input > 20 characters (highest of highest); input is wrong password

Boundary value:

2) Password

Valid boundary values	{6, 20}, correct password
Invalid boundary value	{5,21}, incorrect password

Password	**length**	**Expected Result**	**Actual Result**
123456	6	Successful login	Successful login
12345	5	Password need to be at least 6 characters long!	Password need to be at least 6 characters long!
passwordpassword1234	20	Successful login	Successful login
passwordpassword12345	21	Password need to be at most 20 characters long!	Password need to be at most 20 characters long!

1. Registration

a) Generic cases

Test ID	Scenario	Expected Results	Actual Results
1	Register with valid inputs in all fields	The system displays the main menu for user to continue the operation	The system displays the main menu for user to continue the operation
2	Register with incomplete fields	The system prompts the user to fill up the required fields for registration	The system prompts the user to fill up the required fields for registration
3	Register with taken username	The system prompts the user to select another username	The system prompts the user to select another username
4	Register with password mismatch	The system prompts the user to re-enter the password	The system prompts the user to re-enter the password
5	Register with invalid D.O.B	The system prompts the user to re-enter their DOB in the valid range	The system prompts the user to re-enter their DOB in the valid range

b) Specific cases (Email address)

Test ID	Email Address	Expected Results	Actual Results
1	test@gmail.com	Approve	Approve
2	test	Reject	Reject
3	test@abcd	Reject	Reject

c) Specific cases (Combination)

The following inputs are discrete values and have only 2 equivalence classes:

Equivalence class for D.O.B

1. Valid equivalence class = {DDMMYYY, where DD = 1 to 31, MM = 1-12, YYYYY = any 4 digit number }
2. Invalid equivalence class = {123, … }

Equivalence class for Username

1. Valid equivalence class: {length of username >=6 and unique}
2. Invalid equivalence classes: {length of username < 6 or non- unique}

Equivalence class for Name

1. Valid equivalence class = {any string, “testuser”, …}
2. Invalid equivalence class = {empty string}

Equivalence class for Email

1. Valid equivalence class = {String of format XX@XX.com, where XX is any character}
2. Invalid equivalence class = {“abc”, … }

Equivalence class for Confirm Password

1. Valid equivalence class = {String that matches the password field}
2. Invalid equivalence class = {all other strings}

Equivalence class for Password

1. Valid equivalence class = {length of password >= 6}
2. Invalid equivalence class = {length of password < 6}

As the number of possible cases in each of these equivalence classes are large, we'll only be choosing 1 representative value for each equivalence class

Username	**Given Name**	**Email**	**Password**	**Confirm Password**	**D.O.B**	**Expected Result**	**Actual Result**
testuser	test	user@gmail.com	testpass	testpass	12122000	User has been registered successfully!	User has been registered successfully!
testuser	test	user@gmail.com	testpass	testpass	12122000	Username is already taken!	Username is already taken!
test	test	user@gmail.com	testpass	testpass	12122000	Username needs to be at least 6 characters long!	Username needs to be at least 6 characters long!
testuser	Empty(“”)	user@gmail.com	testpass	testpass	12122000	Name is required!	Name is required!
testuser	test	Empty(“”)	testpass	testpass	12122000	Email is required!	Email is required!
testuser	test	user@gmail.com	Empty(“”)	testpass	12122000	Password is required!	Password is required!
testuser	test	user@gmail.com	testpass	Empty(“”)	12122000	Password and confirmation	Password and confirmation

						do not match!	do not match!
testuser	test	user@gmail.com	testpass	testpass	Empty(“”)	Please enter valid date in DDMMYYY format	Please enter valid date in DDMMYYY format
testuser	test	user@gmail.com	testpass1	testpass2	12122000	Password and confirmation do not match!	Password and confirmation do not match!
testuser	test	user@gmail.com	testpass2	testpass1	12122000	Password and confirmation do not match!	Password and confirmation do not match!

2. Login

a) Generic cases

Test ID	**Scenario**	**Expected Results**	**Actual Results**
1	Login with valid account username and password	The system displays the main menu for user to continue the operation	The system displays the main menu for user to continue the operation
2	Login without valid credentials	The system prompts the user to enter the credentials again	The system prompts the user to enter the credentials again
3	Login with incomplete fields	The system prompts the user to fill up the required fields for logging in	The system prompts the user to fill up the required fields for logging in
4	Login without filling up minimum character requirements	The system prompts the user to fill up the required fields for logging in	The system prompts the user to fill up the required fields for logging in

b) Specific cases (Combination)

Email	**Password**	**Expected Results**	**Actual Results**
user@gmail.com	testpass	Successful login	Successful login
wrong@gmail.com	testpass	Failed to login! Please check your credentials	Failed to login! Please check your credentials

Empty(“”)	testpass	Email is required!	Email is required!
user@gmail.com	wrongpass	Failed to login! Please check your credentials	Failed to login! Please check your credentials
user@gmail.com	Empty(“”)	Password is required!	Password is required!

3. Get user’s current location

Test ID	Scenario	Expected Results	Actual Results
1	User allowed location permissions for the app	The system displays the main menu for user to continue the operation	The system displays the main menu for user to continue the operation
2	User denied location permissions for the app	The system prompts the user to allow permissions again every time they switch back to the explore page	The system prompts the user to allow permissions again every time they switch back to the explore page

4. Add event from explore page to a calendar

a) Generic cases

Test ID	Scenario	Expected Results	Actual Results
1	Create event with all details filled in	Event successfully created	Event successfully created
2	Create event with incomplete fields	The system prompts the user with the respective message(s) to fill up the required field(s)	The system prompts the user with the respective message(s) to fill up the required field(s)
3	Create event with invalid Time (not between 0000 and 2359)	Time is not valid, please enter number in range of 0000 to 2359	Time is not valid, please enter number in range of 0000 to 2359

b) Specific cases (Combination)

Type Event	**Type Location**	**Event Time**	**Expected Results**	**Actual Results**
Nasi Lemak	*NTU Canteen*	Between 0000 and 2359	Successful event creation	Successful event creation
Nasi Lemak	*NTU Canteen*	Empty(“”) / Out of range	Time is not valid, please enter a valid time.	Time is not valid, please enter a valid time.
Nasi Lemak	Empty(“”)	Between 0000 and 2359	Location is not valid. Please enter a valid location.	Location is not valid. Please enter a valid location.
Nasi Lemak	Empty(“”)	Empty(“”) / Out of range	Location is not valid. Please enter a valid location.	Location is not valid. Please enter a valid location.
Empty(“”)	*NTU Canteen*	Between 0000 and 2359	Name is not valid. Please enter the event name.	Name is not valid. Please enter the event name.
Empty(“”)	*NTU Canteen*	Empty(“”) / Out of range	Name is not valid. Please enter the event name.	Name is not valid. Please enter the event name.
Empty(“”)	Empty(“”)	Between 0000 and 2359	Name is not valid. Please enter the event name.	Name is not valid. Please enter the event name.
Empty(“”)	Empty(“”)	Empty(“”) / Out of range	Name is not valid. Please enter the event name.	Name is not valid. Please enter the event name.

5. Calendar Page

a) Text box with “Type Event” hint

Test ID	**Text Enter**	**Expected Results**	**Actual Results**

1	Lunch with Martha	Successfully Saved	Successfully Saved
2	Empty(“”)	Name is not valid. Please enter the event name.	Name is not valid. Please enter the event name.

b) Text box with “Location” hint

Test ID	**Text Enter**	**Expected Results**	**Actual Results**
1	North Spine	Successfully Saved	Successfully Saved
2	Empty(“”)	Location is not valid. Please enter a valid location.	Location is not valid. Please enter a valid location.

c) Numeric Textbox with “Event time” hint

Test ID	**Text Enter**	**Expected Results**	**Actual Results**
1	0000	Successfully Saved	Successfully Saved
2	2359	Successfully Saved	Successfully Saved
3	-0001	Time is not valid. Please enter number in range of 0000 to 2359	Time is not valid. Please enter number in range of 0000 to 2359
4	2400	Time is not valid. Please enter number in range of 0000 to 2359	Time is not valid. Please enter number in range of 0000 to 2359

d) Numeric Textbox with “DD” hint

Test ID	**Text Enter**	**Expected Results**	**Actual Results**
1	1	Successfully Saved	Successfully Saved
2	31	Successfully Saved	Successfully Saved
3	0	Date is not valid. Please enter number in range of 1 to 31	Date is not valid. Please enter number in range of 1 to 31
4	32	Date is not valid. Please enter number in range of 1 to 31	Date is not valid. Please enter number in range of 1 to 31

e) Numeric Textbox with “MM” hint

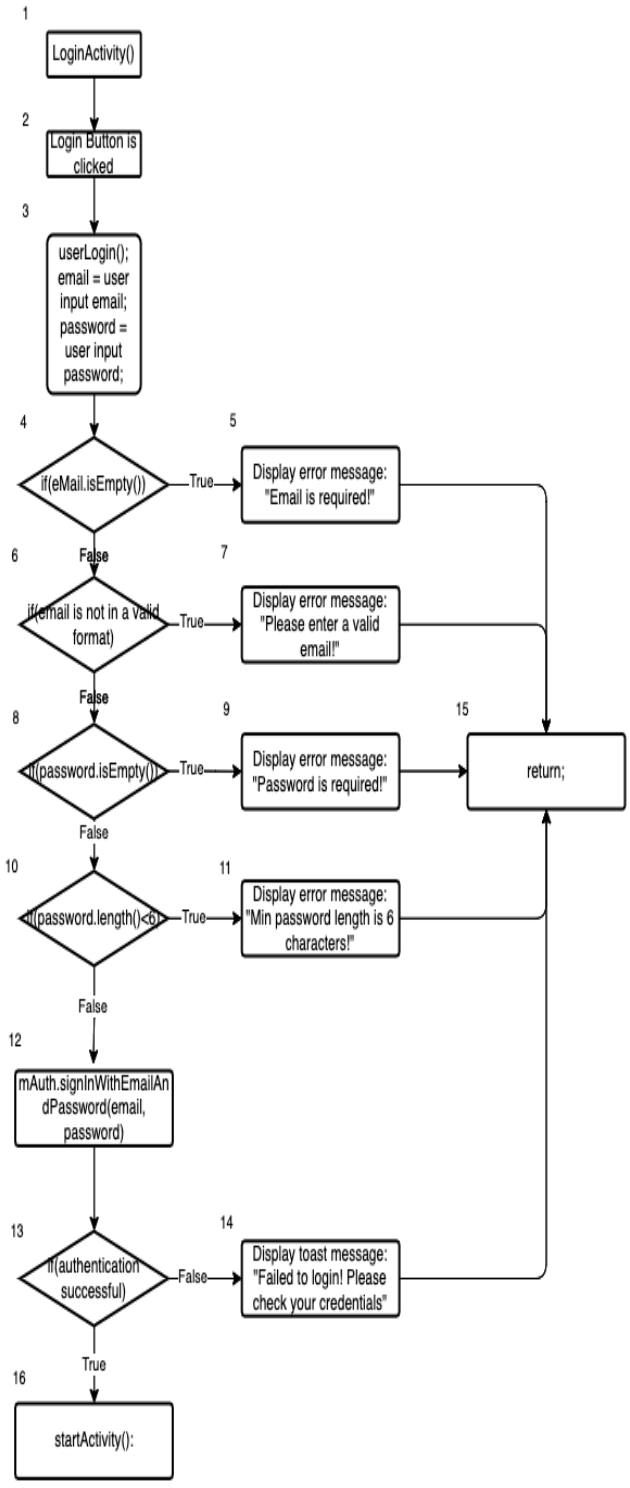
Test ID	Text Enter	Expected Results	Actual Results
1	1	Successfully Saved	Successfully Saved
2	12	Successfully Saved	Successfully Saved
3	0	Month is not valid. Please enter number in range of 1 to 12	Month is not valid. Please enter number in range of 1 to 12
4	13	Month is not valid. Please enter number in range of 1 to 12	Month is not valid. Please enter number in range of 1 to 12

f) Numeric Textbox with “YYYY” hint

Test ID	Text Enter	Expected Results	Actual Results
1	1900	Successfully Saved	Successfully Saved
2	2100	Successfully Saved	Successfully Saved
3	1899	Year is not valid. Please enter number in range of 1900 to 2100	Year is not valid. Please enter number in range of 1900 to 2100
4	2101	Year is not valid. Please enter number in range of 1900 to 2100	Year is not valid. Please enter number in range of 1900 to 2100

7.2 White Box Testing

1. Control Flow Test for LoginActivity()



Cyclomatic Complexity

Taking Cyclomatic complexity:

|decision points| + 1 = 5 + 1 = **6**

Basic Paths

I. Baseline path: 1, 2, 3, 4, 6, 8, 10, 12, 13, 16
II. Basic path 2: 1, 2, 3, 4, 5, 15
III. Basic path 3: 1, 2, 3, 4, 6, 7, 15
IV. Basic path 4: 1, 2, 3, 4, 6, 8, 9, 15
V. Basic path 5: 1, 2, 3, 4, 6, 8, 10, 11, 15
VI. Basic path 6: 1, 2, 3, 4, 6, 8, 10, 12, 13, 14, 15

Test Cases

I. email = “teresa_zhang@gmail.com” ; password: “12345678” ;

II. email = “” ; password = “1234678 ” ;

III. email = “teresa_zhang” ; password = “1234678” ;

IV. email = “teresa_zhang@gmail.com” ; password = “” ;

V. email = “teresa_zhang@gmail.com” ; password = “1234”;

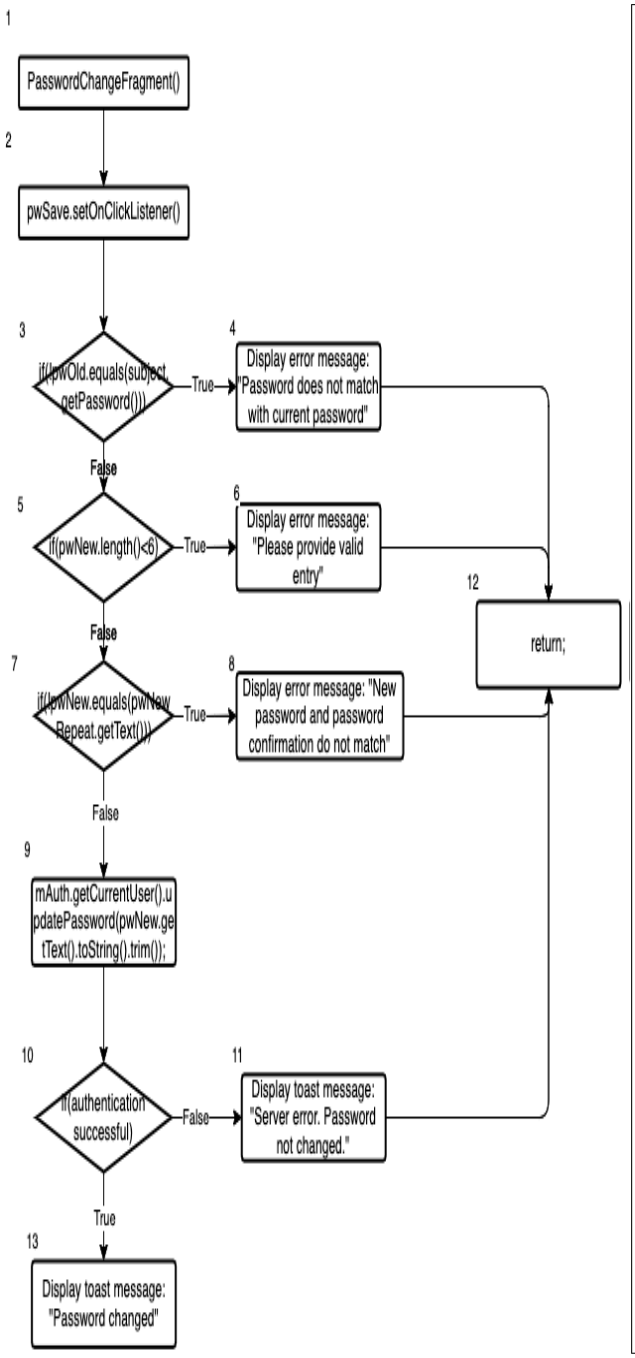
VI. email = “teresa_zhang@gmail.com” ; password = “87654321” (wrong password);

Real Execution Paths

I. 1, 2, 3, 4, 6, 8, 10, 12, 13, 16
II. 1, 2, 3, 4, 5, 15
III. 1, 2, 3, 4, 6, 7, 15
IV. 1, 2, 3, 4, 6, 8, 9, 15
V. 1, 2, 3, 4, 6, 8, 10, 11, 15
VI. 1, 2, 3, 4, 6, 8, 10, 12, 13, 14, 15

Figure 4.1: White Box Testing for LoginActivity()

2. Control Flow Test for PasswordChangeFragment()



Cyclomatic Complexity

Taking Cyclomatic complexity:

|decision points| + 1 = 4 + 1 = **5**

Basic Paths

I. Baseline path: 1, 2, 3, 5, 7, 9, 10, 13
II. Basic path 2: 1, 2, 3, 4, 12
III. Basic path 3: 1, 2, 3, 5, 6, 12
IV. Basic path 4: 1, 2, 3, 5, 7, 8, 12
V. Basic path 5: 1, 2, 3, 5, 7, 9, 10, 11, 12

Test Cases

I. password: “12345678” ; new password: “246810”; confirm password: “246810”

II. password: “87654321” (wrong password); new password: “246810”; confirm password: “246810”

III. password: “12345678” ; new password: “2468”; confirm password: “2468”

IV. password: “12345678” ; new password: “246810”; confirm password: “246811”

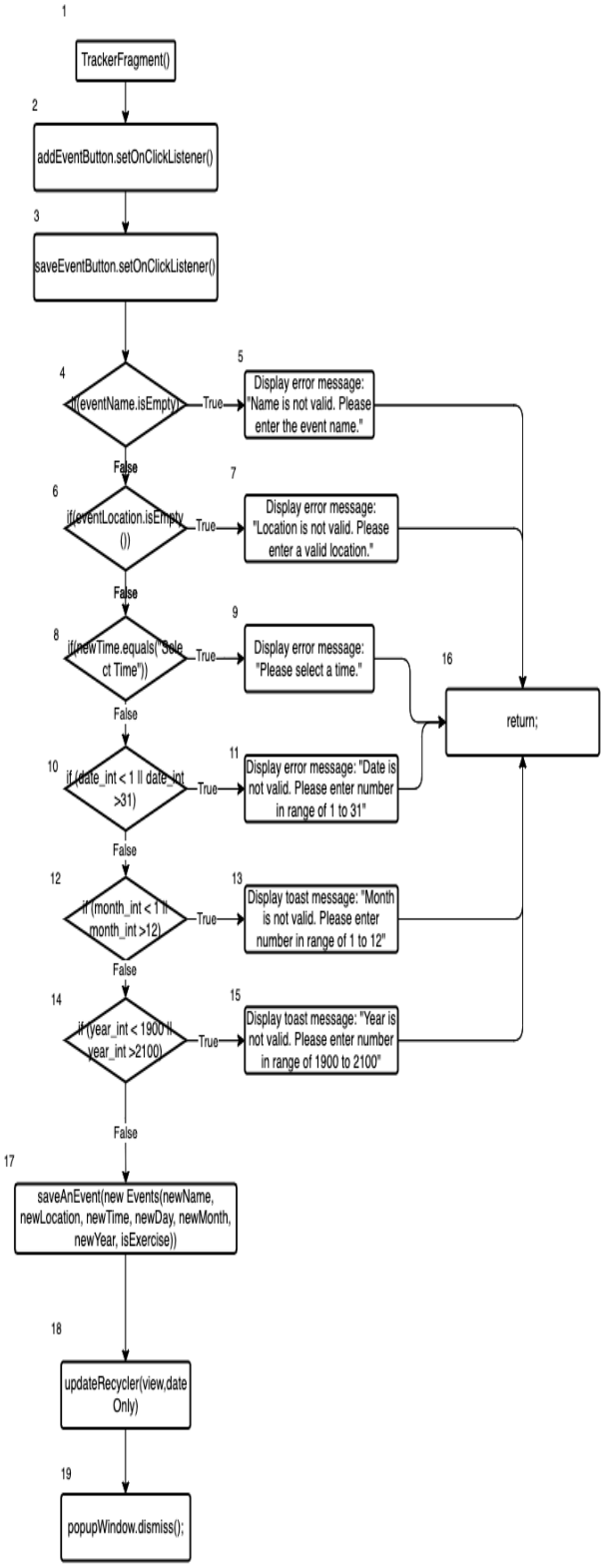
V. password: “12345678” ; new password: “246810”; confirm password: “246810”; No Internet;

Real Execution Paths

I. 1, 2, 3, 5, 7, 9, 10, 13
II. 1, 2, 3, 4, 12
III. 1, 2, 3, 5, 6, 12
IV. 1, 2, 3, 5, 7, 8, 12
V. 1, 2, 3, 5, 7, 9, 10, 11, 12

Figure 4.2: White Box Testing for ChangeFragment()

3. Control Flow Test for TrackerFragment()



Cyclomatic Complexity

Taking Cyclomatic complexity:

|decision points| + 1 = 6 + 1 = 7

Basic Paths

I. Baseline path: 1, 2, 3, 4, 6, 8, 10, 12, 14, 17, 18, 19
II. Basic path 2: 1, 2, 3, 4, 5, 16
III. Basic path 3: 1, 2, 3, 4, 6, 7, 16
IV. Basic path 4: 1, 2, 3, 4, 6, 8, 9, 16
V. Basic path 5: 1, 2, 3, 4, 6, 8, 10, 11, 16
VI. Basic path 6: 1, 2, 3, 4, 6, 8, 10, 12, 13, 16
VII. Basic path 7: 1, 2, 3, 4, 6, 8, 10, 12, 14, 15, 16

Test Cases

I. event = "Lunch with Teresa"; location = "NTU Canteen 14"; time = "12:30"; date = "16042022";

II. event = ""; location = "NTU Canteen 14"; time = "12:30"; date = "16042022";

III. event = "Lunch with Teresa"; location = ""; time = "12:30"; date = "16042022";

IV. event = "Lunch with Teresa"; location = "NTU Canteen 14"; time = "Select Time"; date = "16042022";

V. event = "Lunch with Teresa"; location = "NTU Canteen 14"; time = "12:30"; date = "32042022";

VI. event = "Lunch with Teresa"; location = "NTU Canteen 14"; time = "12:30"; date = "16132022";

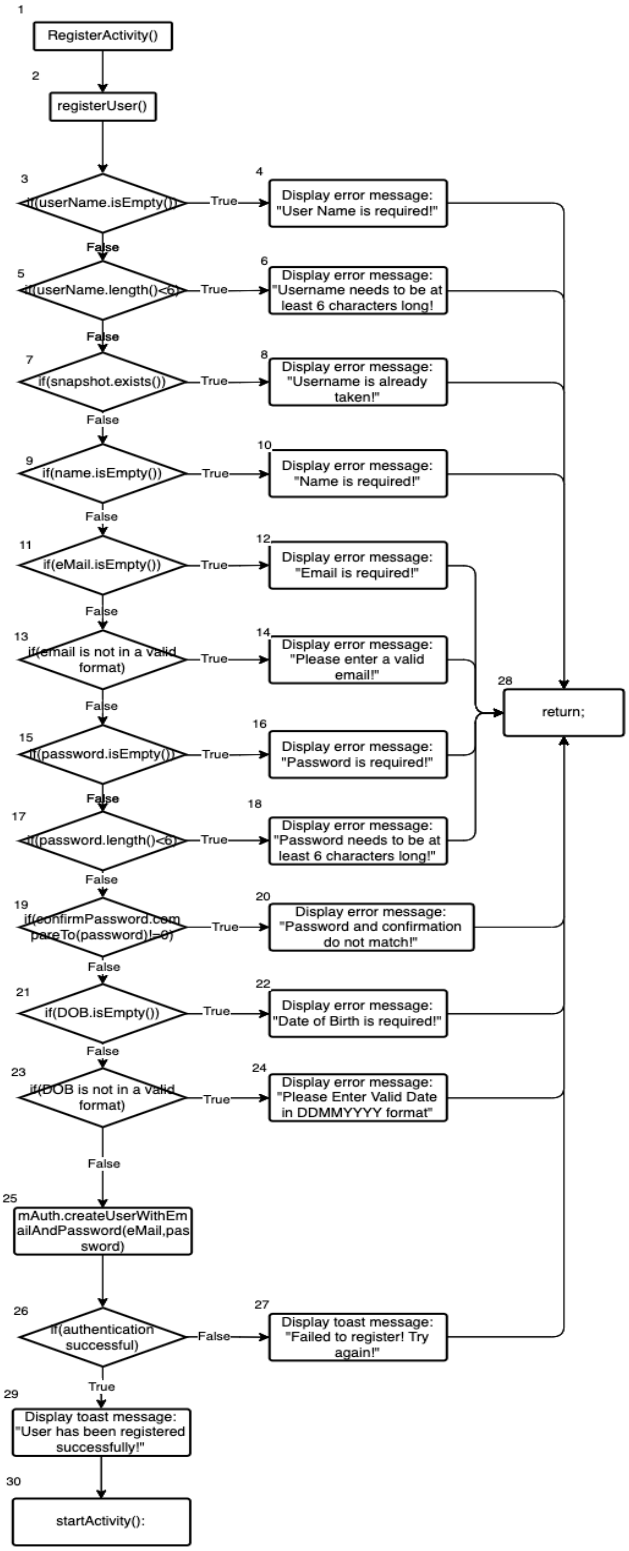
VII. event = "Lunch with Teresa"; location = "NTU Canteen 14"; time = "12:30"; date = "16042111";

Real Execution Paths

I. 1, 2, 3, 4, 6, 8, 10, 12, 14, 17, 18, 19
II. 1, 2, 3, 4, 5, 16
III. 1, 2, 3, 4, 6, 7, 16
IV. 1, 2, 3, 4, 6, 8, 9, 16
V. 1, 2, 3, 4, 6, 8, 10, 11, 16
VI. 1, 2, 3, 4, 6, 8, 10, 12, 13, 16
VII. 1, 2, 3, 4, 6, 8, 10, 12, 14, 15, 16

Figure 4.3: White Box Testing for TrackerFragment()

4. Control Flow Test for RegisterActivity()



Cyclomatic Complexity

Taking Cyclomatic complexity:

|decision points| + 1 = 12 + 1 = **13**

Basic Paths

I. Baseline path: 1, 2, 3, 4, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 26, 29, 30
II. Basic path 2: 1, 2, 3, 4, 28
III. Basic path 3: 1, 2, 3, 5, 6, 28
IV. Basic path 4: 1, 2, 3, 5, 7, 8, 28
V. Basic path 5: 1, 2, 3, 5, 7, 9, 10, 28
VI. Basic path 6: 1, 2, 3, 5, 7, 9, 11, 12, 28
VII. Basic path 7: 1, 2, 3, 5, 7, 9, 11, 13, 14, 28
VIII. Basic path 8: 1, 2, 3, 5, 7, 9, 11, 13, 15, 16, 28
IX. Basic path 9: 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 18, 28
X. Basic path 10: 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 20, 28
XI. Basic path 11: 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 22, 28
XII. Basic path 12: 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 24, 28
XIII. Basic path 13: 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 26, 27, 28

Real Execution Paths

I. 1, 2, 3, 4, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 26, 29, 30
II. 1, 2, 3, 4, 28
III. 1, 2, 3, 5, 6, 28
IV. 1, 2, 3, 5, 7, 8, 28
V. 1, 2, 3, 5, 7, 9, 10, 28
VI. 1, 2, 3, 5, 7, 9, 11, 12, 28
VII. 1, 2, 3, 5, 7, 9, 11, 13, 14, 28
VIII. 1, 2, 3, 5, 7, 9, 11, 13, 15, 16, 28
IX. 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 18, 28
X. 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 20, 28
XI. 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 22, 28
XII. 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 24, 28
XIII. 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 26, 27, 28

Figure 4.4: White Box Testing for RegisterActivity()

5. Test Cases

I. username = “Healthy123”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12345678”; DOB = “19122001”;

II. username = “”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12345678”; DOB = “19122001”;

III. username = “123”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12345678”; DOB = “19122001”;

IV. username = “Healthy” (taken); name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12345678”; DOB = “19122001”;

V. username = “Healthy123”; name = “”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12345678”; DOB = “19122001”;

VI. username = “Healthy123”; name = “Teresa”; email = “”; password = “12345678”; confirm password = “12345678”; DOB = “19122001”;

VII. username = “Healthy123”; name = “Teresa”; email = “teresa_zhang”; password = “12345678”; confirm password = “12345678”; DOB = “19122001”;

VIII. username = “Healthy123”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “”; confirm password = “12345678”; DOB = “19122001”;

IX. username = “Healthy123”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “1234”; confirm password = “1234”; DOB = “19122001”;

X. username = “Healthy123”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12349876”; DOB = “19122001”;

XI. username = “Healthy123”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12345678”; DOB = “”;

XII. username = “Healthy123”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12345678”; DOB = “19/12/2001”;

XIII. username = “Healthy123”; name = “Teresa”; email = “teresa_zhang@gmail.com”; password = “12345678”; confirm password = “12345678”; DOB = “19122001”; No Internet;

8. Appendix

For more information and detailed demo of Fitrition app, please refer to the YouTube link below for our video demo:

https://youtu.be/2Q1OQhK_vM4