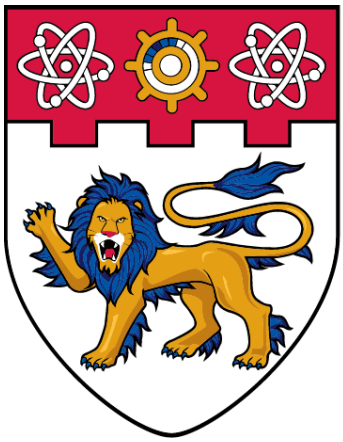


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

School of Computer Science and Engineering
CZ2006 DSS1

‘SportBuds’ application

Completed by: Rainers Rabbids

Clarissa Bella Jew U2021124G
Choy Xin Yun U2021811D
Khant Nyi Nyi U2022568J
Caleb Cheam U2021423G
Kam Rainer I-Wen U2020198J
Tan Yi Jie Joel U2021190G

Table of Contents

1. Introduction

1.1 Context

As the Singapore government slowly starts to ease restrictions after the COVID-19 pandemic, group activities are beginning to resume. However, after 2 years of safe-distancing measures and restrictions on playing sports, many people may not be used to forming a large group of people, especially with strangers, to play sports together. As such, our app was created with the purpose of helping people find new friends to play sports together with.

1.2 Purpose of SportBuds

The SportBuds team aims to provide users with an easily accessible and safe platform to organise and join sports activities, while making friends with fitness enthusiasts of similar interest.

Convenience

The SportBuds app aims to provide its users the ease of convenience through its one-stop platform. Users of the app are able to organise sport activities at various dedicated sports facilities in Singapore via a simple form on the app where they specify the type, time and capacity of the sport activities. Other users are then able to easily indicate their interest in participating in the activities by simply pressing ‘Join event’ on the activity of interest, provided they have no clashing schedules. Furthermore, additional features like chat groups for each activity allows easy planning of sport events.

Building Friendships

The pandemic has made it more difficult to meet new people due to stringent social guidelines. The SportBuds app aims to ease the forging of new friendships through users’ common love for sports. Users of the app will be able to connect with other users from events that they join together or other users whom they know personally through the application’s social capabilities. This allows users to create or join a community where they can invite one another to sport activities they created or joined.

Appreciation for Sports and Improved Health

COVID-19 has resulted in numerous lockdowns which have thus resulted in many Singaporeans staying indoors for long periods of time. Subsequently, Singaporeans have adopted a more sedentary lifestyle in recent years which can lead to a decline in muscle strength and general health. [1] Playing sports can have immense health benefits in improving one’s fitness and physical well-being. Furthermore, sports can contribute to one’s mental health by reducing stress and expanding one’s social circles. The SportBuds app hopes to nurture a greater love of sports among its users and thus, lead more Singaporeans into adopting healthier lifestyles.

1.3 Document content

This report is a complete compilation of all lab deliverables in the CZ2006 course. It includes the following documentation:

1. Functional, Non-functional requirements
2. UI Mock-up
3. System Architecture
4. Use Cases (with sequence diagrams)
5. Class Diagram
6. Dialog Map
7. Testing
8. Meeting Minutes
9. Data Dictionary

2. Overall description

2.1 Product description

SportBuds is an app developed using the Flutter framework that uniquely combines the facilitation of sports events with social media elements to promote healthy activities in Singapore. Users are able to create sports events at over 400 facilities and parks, through an easy and intuitive interface involving a scrollable and searchable map, indicative facility markers and informative event calendars. Other users can then join and leave these events. Users are able to chat within events with fellow SportBuds users who have joined the same event to foster bonds and cultivate a warm and welcoming environment for all users.

Additionally, users can add other users as friends if they desire, emphasising the friendly and social nature of our app, which encourages users to meet more like-minded users and participate in more events with these users.

After participating in events, users can leave a review on the facility site describing their experience. These reviews let users describe their lovely time during the event, rate their overall experience and even upload a commemorative photo, which helps future users decide if a facility is right for them when creating events.

Users are also allowed to complete events they have participated in to gain points and redeem attractive sports-themed rewards using these points. Upon completing an event, users will be recommended other existing similar events based on a clustering algorithm. This convenient feature makes it easy for users to plan their next event and improves user experience.

We also place high emphasis on user safety and overall user experience. We recognise that the competitive nature of sports can sometimes create heated environments, but we do not condone extreme acts that spoil others' experience and

undermine the spirit of SportBuds. Users are thus allowed to report anyone who they feel violates this spirit. Stern warnings and restrictions are then issued to deter such behaviour.

2.2 Dataset

The SportBuds app is built using the SportSG sports facilities data provided by https://data.gov.sg/dataset/sportsg-sport-facilities. The name, type, address and location on the Singapore map (obtained through coordinates) are extracted and provided to users on the app. The app also made use of the parks data provided by https://data.gov.sg/dataset/parks, if users wish to organise their sport activities at public parks instead of dedicated sports facilities.

2.3 Operating Environment

SportBuds is an android app with a minimum sdk of 16. Internet connection is required for users to log in and use all basic functionality. Location services are needed for users using the map function for completing events.

2.4 Design and Implementation Constraints

The application requires Google Firebase, Firestore API, Huawei Cloud and Mapbox API to be functional. The app is written on Android Studio in Flutter(dart) targeted at Android users. The SportsFacility data needs to be updated by SportBuds developers if there are changes in the future.

3. Functional Requirements

3.1. Registration

3.1.1. The user must be able to access the registration page from the login page(default page).

3.1.2. System must prompt the user for his/her credentials during registration.

3.1.2.1. The system must prompt the user to enter his/her user name.

3.1.2.2. The system must prompt the user to enter his/her email address

3.1.2.3. The system must prompt the user to enter a password.

3.1.3. The system must be able to validate that all fields have been filled up.

3.1.3.1. If validation fails, the corresponding error must be displayed for the missing field.

3.1.4. The system must be able to check on the validity of the fields entered by the user

3.1.4.1. The system must validate that the password entered is at least 6 characters long.

3.1.4.2. The system must be able to validate that the email entered is not linked to an existing account.

3.1.4.3. The system must be able to validate that the email entered is formatted properly.

3.1.4.4. If validation fails, an error message must be displayed for each error.

3.1.5. The system must notify the user upon successful account creation.

3.1.6. The system must display a message on the application to inform the user of a successful account registration.

3.1.7. The system will prompt users if they want to proceed to the onboarding process after successful registration.

3.1.7.1. The users are allowed to skip the onboarding process at any point.

3.1.8. The users must be directed to their homepage.

3.2. Login

3.2.1. The system must provide a login page for users.

3.2.2. The system must provide an email address field for the user to enter his email.

3.2.3. The system must provide a password field for the user to enter his password.

3.2.4. The system must be able to validate the credentials entered by the users

3.2.4.1. The system must validate that the email address field is filled.

3.2.4.2. The system must validate that the password field is filled.

3.2.4.3. If any of the fields are not filled, a corresponding error message will be displayed for the empty field.

3.2.5. The system must validate if the email address entered is a registered email address.
    3.2.5.1. If the email entered is not valid, an error message will be displayed.
    3.2.5.2. The system must validate if the password entered matches the email entered.
    3.2.5.3. If the email entered is not valid, an error message will be displayed.
3.2.6. The system must direct users to the homepage upon successful login.

3.3. Map Display

3.3.1. The system must be able to retrieve the location of the user's device.
3.3.2. The system must be able to display the user's location on the FacilitiesMap.
3.3.3. The system must be able to fetch coordinates of the SportsFacilities from a GeoJSON file.
3.3.4. The system must be able to display the locations of SportsFacilities retrieved as a tappable marker.
3.3.5. The system must allow users to search for a specific SportsFacility by name.
    3.3.5.1. The system must return a list of SportsFacilities matching the search query.
    3.3.5.2. The system must allow the user to select a SportsFacility from the returned list of facilities.
    3.3.5.3. The system must display the selected SportsFacility on the FacilitiesMap upon selection by the user.
3.3.6. The system must be able to display different marker icons corresponding to the SportsFacility type.
3.3.7. The user must be able to zoom in and out of the FacilitiesMap.
3.3.8. The system must be able to fetch the current week's events occurring at the selected SportsFacility from the database.
3.3.9. The system must be able to display specific details about the selected SportsFacility upon the user clicking on a marker.
    3.3.9.1. The system must be able to display the SportsFacility's name.
    3.3.9.2. The system must be able to display the SportsFacility's type.
    3.3.9.3. The system must be able to display the SportsFacility's address.
    3.3.9.4. The system must allow the user to view the event calendar for the selected SportsFacility.
    3.3.9.5. The system must be able to display the list of events occurring on a selected day.
        3.3.9.5.1. The system must be able to display the events in a chronological order.

3.3.9.5.2. The system must be able to display the event's name.
3.3.9.5.3. The system must be able to display the event's current capacity.
3.3.9.5.4. The system must be able to display the event's maximum capacity.
3.3.9.5.5. The system must be able to display more details upon clicking on an event.

3.4. View SportEvent

3.4.1. The system must display the SportEvent name.
3.4.2. The system must display the SportEvent start time.
3.4.3. The system must display the SportEvent end time.
3.4.4. The system must display the location of the SportEvent.
3.4.5. The system must display the SportEvent maximum group size.
3.4.6. The system must display the current number of users who have joined the SportEvent.
3.4.7. The system must display the SportEvent description.
3.4.8. The system must display the option to join the SportEvent if the following conditions are satisfied:
  3.4.8.1. The SportEvent must have at least one vacancy.
  3.4.8.2. The user must not be in the SportEvent already.
  3.4.8.3. The user must not be in any other SportEvents at the same time.
3.4.9. The system must be able to check if the SportEvent has reached its maximum capacity.
  3.4.9.1. If the SportEvent has reached its maximum capacity, the system must indicate that the SportEvent is full.
  3.4.9.2. If the SportEvent has reached its maximum capacity, the system must not allow the user to join the SportEvent.
3.4.10. The system must be able to check if the user has already joined the SportEvent.
  3.4.10.1. If the user has joined the SportEvent, the system must provide the user with the option to leave the SportEvent.
  3.4.10.2. If the user has joined the SportEvent, the system must not allow the user to join the SportEvent again.
3.4.11. The system must be able to check if the user has joined other SportEvents which have overlapping timings.
  3.4.11.1. If the user has overlapping SportEvents, the system must indicate that there is a clash in the timings.
  3.4.11.2. If the user has overlapping SportEvents, the system must not allow the user to join the SportEvent.

3.5. Create SportEvent

3.5.1. The user must be able to set the SportEvent name.
3.5.2. The user must be able to set the SportEvent start time.
3.5.3. The user must be able to set the SportEvent end time.
3.5.4. The user must be able to set the maximum group size for the SportEvent.
3.5.5. The user must be able to select the SportEvent type.
3.5.6. The system must validate that all fields have been filled up.
3.5.7. If validation fails, the corresponding error must be displayed for the missing field.
3.5.8. The system must validate all fields entered by the user.
    3.5.8.1. The SportEvent name must have at least 4 characters.
    3.5.8.2. The SportEvent name must have at most 50 characters.
    3.5.8.3. The SportEvent end time must be after the SportEvent start time.
    3.5.8.4. The maximum group size for the SportEvent must be more than 0.
    3.5.8.5. The maximum group size for the SportEvent must be at most 30.
3.5.9. The system must be able to check if the user has other Active Events which have overlapping timings.
    3.5.9.1. If the user has clashing schedules, the system must not allow the user to create the SportEvent.
    3.5.9.2. The system must display an error message indicating that there is a clash in timings.
3.5.10. If validation fails, the corresponding error must be displayed for each error.
3.5.11. The system must display a message to indicate that the SportEvent has been successfully created.

3.6. Join SportEvent

3.6.1. The system must provide a confirmation prompt to the user. The user must confirm to join the SportEvent.
3.6.2. The system must be able to check if the user has other Active Events which have overlapping timings.
    3.6.2.1. If the user has clashing schedules, the system must not allow the user to join the SportEvent.
3.6.3. The system must increase the current capacity of the SportEvent by 1 in the database.
3.6.4. The system must add the SportEvent to the user’s list of Active Events in the database.

3.6.5. The system must display a message to indicate that the user has successfully joined the SportEvent.
3.6.6. The system must return the user to the FacilitiesMap upon user's acknowledgement that they have joined the SportEvent.

3.7. Leave SportEvent

3.7.1. The system must provide a confirmation prompt to the user. The user must confirm to leave the SportEvent.
3.7.2. The system must update that the current number of users has decreased by 1 in the database.
3.7.3. The system must remove the SportEvent from the user's list of Active Events in the database.
3.7.4. The system must check if the updated current capacity is more than 0.
3.7.5. If the updated number of users is 0, the system must delete the SportEvent.
3.7.6. The system must display a message to indicate that the user has successfully left the SportEvent.
3.7.7. The system must return the user to the FacilitiesMap upon user's acknowledgement that they have left the SportEvent.

3.8. Complete SportEvent

3.8.1. The system must check if the user is in a 100 metre radius from the location of the SportEvent.
3.8.2. The system must check if the current time is before the end time of the event to be completed
3.8.3. The system must check if the current time is after the start time of the event to be completed
3.8.4. The system must provide a confirmation prompt to the user. The user must confirm to complete the SportEvent.
3.8.5. The system must remove the SportEvent from the user's list of Active Events in the database.
3.8.6. The system must add the SportEvent to the user's list of Completed Events in the database
3.8.7. The system must display a message to indicate that the user has successfully completed the SportEvent.
3.8.8. The system must add the number of points earned by the User for completing the event to the User's point count.
3.8.9. The system must inform the user of the number of points acquired from completing the SportEvent.

3.8.10. The system must show recommended events to the user upon completion of events.

3.8.11. The system must return the user to the FacilitiesMap upon user's acknowledgement that they have completed the SportEvent.

3.9. View Recommendations

3.9.1. The system must display recommended events to the user upon completion of a SportEvent.

3.9.1.1. There may be zero or one or many SportEvents recommended to the user based on the recommendation algorithm.

3.9.1.2. The recommended SportEvents must be happening in the future

3.9.1.3. The recommended SportEvents must not be SportEvents that the user has already joined.

3.9.2. The system must display the name, date, start time, end time, location, current capacity and maximum capacity of the recommended events.

3.9.3. The system must return the user to the complete events acknowledgement page upon pressing the exit button on the recommendations page.

3.10. View Reviews

3.10.1. The system must allow the user to view the reviews for a selected SportsFacility from the FacilitiesMap.

3.10.2. The system must display a bar chart illustrating counts of reviews for each rating (1 star, 2 star, 3 star, 4 star, 5 star).

3.10.3. The system must display an average rating for that SportsFacility based on the total number of reviews and ratings .

3.10.4. The system must display usernames of the users who posted each review.

3.10.5. The system must display profile pictures of the users who posted each review.

3.10.6. The system must display a "Refresh" button.

3.10.6.1. The "Refresh" button must reload the review page on click.

3.10.7. The system must display a "Write Review" button.

3.11. Post Reviews

3.11.1. The system must allow the user to create a review for a SportsFacility and post it by pressing on the "Write Review" button found on the review page of the SportsFacility.

3.11.1.1. The “Write Review” button must generate a form for users to write a review.
3.11.1.2. The form must allow the user to enter a review title and review description.
3.11.1.3. The form must allow the user to select a SportsFacility rating of at least 1 star and at most 5 stars.
3.11.1.4. The form must allow the user to upload a picture.
3.11.1.5. The form must prompt users that the title must be at least 4 characters and at most 50 characters.
3.11.1.6. The form must prompt users that the description must be at most 300 characters.
3.11.1.7. The user must be informed that only one picture may be uploaded.
3.11.1.8. The user must be informed that uploading pictures is optional.
3.11.1.9. The system must display a success image upon uploading an image.
3.11.1.10. The system must display a “Post” button.
3.11.1.10.1. The “Post” button must display an error message upon click if the review title, review description or SportsFacility rating has not been inputted.
3.11.1.10.2. The “Post” button must display an error if title or description does not match character limits.
3.11.1.10.3. The “Post” button must add a fully inputted review to the database on click.
3.11.1.10.4. The “Post” button must add the image to the database, if one was uploaded.
3.11.1.10.5. The “Post” button must return the User to the FacilitiesMap upon successful review creation.

3.12. View User Profile

3.12.1. The system must display the user’s profile picture.
3.12.2. The system must display the user’s most updated About.
3.12.2.1. If the About section is empty, the system will display an empty text field.
3.12.3. The system must display the number of points obtained by the user.
3.12.4. The system must display the user’s username.
3.12.5. The system must display the user’s email.
3.12.6. The user must be able to access the edit profile page from the user profile
3.12.7. The system must display the number of the user’s friends
3.12.8. The system must display the number of friend requests the user have
3.12.9. The system must display the number of points the user currently have
3.12.10. The system must display the number of past events the user has attended.

3.13. Edit User Profile

3.13.1. The user must be able to access the edit user profile from the user profile page.

3.13.2. The user must be able to edit his username in the textfield.

3.13.3. The user must be able to edit his About in the textfield.

3.13.4. The user must be able to change his profile picture.

3.13.5. The system must validate fields entered by the user.

3.13.5.1. The system must validate the the new username entered is less than 20 characters

3.13.5.2. The system must validate the the new About entered is less than 140 characters

3.13.5.3. If validation fails, the system must display the appropriate error message.

3.13.6. The system must save the updated profile information changes when the user clicks on the ‘save’ button.

3.13.7. The system must display the most updated user information concurrently as the user edit information.

3.13.8. The system must inform the user when profile information is updated.

3.13.9. The user must be able to return to the user profile page after editing their profile information.

3.14. View Other User’s Profile

3.14.1. The system must display the selected user’s profile information when searched in the friends list.

3.14.1.1. The system must display the user’s number of user points accumulated.

3.14.1.2. The system must display the user’s username.

3.14.1.3. The system must display the user’s user About section.

3.14.2. The system must check if the current user is currently friends with the selected user

3.14.2.1. If the user and current user are friends, the current user will not be allowed to add this user as friend

3.14.2.2. If the user and current user are not friends, the current user will have the option to send this user a friend request

3.14.2.2.1. The system must display a pop-up when a successful friend request is sent.

3.14.3. The system will allow the user to report this selected user.

3.14.3.1. The system must display a pop-up when a successful report is made.

3.15. View Completed Event History

3.15.1. The user must be able to view event history of all past completed events from the user profile page

3.15.2. The system must display the SportEvents the user has completed in chronological order.

3.15.3. The system must check that all events displayed have already taken place and have been completed.

3.15.4. The user must be able to view these events in detail.

3.15.4.1. The system must display the SportEvent name.

3.15.4.2. The system must display the SportEvent start time.

3.15.4.3. The system must display the SportEvent end time.

3.15.4.4. The system must display the location of the SportEvent.

3.15.4.5. The system must display the SportEvent maximum group size.

3.15.4.6. The system must display the updated current number of users who have joined the SportEvent.

3.15.5. The user must be able to go back to the user profile after viewing event history.

3.16. View Active Events

3.16.1. The user must be able to view all upcoming joined events from the current event page.

3.16.2. The system must display all active SportEvents the user has currently joined in chronological order.

3.16.3. The system must check that all events displayed are taking place currently or are going to take place and that they have yet to be completed.

3.16.4. The user must be able to view these events in detail.

3.16.4.1. The system must display the SportEvent name.

3.16.4.2. The system must display the SportEvent start time.

3.16.4.3. The system must display the SportEvent end time.

3.16.4.4. The system must display the location of the SportEvent.

3.16.4.5. The system must display the SportEvent maximum group size.

3.16.4.6. The system must display the updated current number of users who have joined the SportEvent.

3.16.4.7. The system must display the SportEvent description.

3.16.5. The user must be able to invite their friends to the Active Events

3.16.6. The user must be able to chat with other users who are also in the same Active Events

3.16.6.1. The system will display chat messages in chronological order sent by other users and the user himself

3.16.6.2. The system will display a message that includes the sender, message content, time of message sent.

3.16.7. The user must be able to go back to the user profile after using the chat function

4. Non-functional Requirements

4.1. Performance

4.1.1. The system must not crash during the app's lifetime.

4.1.2. The system must load the Mapbox API and SportsFacility markers within 5s after the user logs in.

4.1.3. The system must display requested screens and information within 500ms after user request.

4.1.4. The system database must respond to live updates within 500ms of user request.

4.1.5. The cloud clustering algorithm must update recommendations into the system database within 3s.

4.2. Usability

4.2.1. The system must reduce short-term memory load.

4.2.2. The user interface must cater to all audiences.

4.2.2.1. Text font must be at least size 12 for easy readability

4.2.2.2. First-time users must know how to use the app and its functions after the onboarding process.

4.2.3. The theme of the user interface must remain consistent throughout the app.

4.2.3.1. The color palette utilized must remain consistent throughout the app.

4.2.3.2. Buttons with similar functions must have similar designs throughout the app.

4.2.3.3. Users should be able to intuitively tell the purposes of widgets with inductive reasoning.

4.2.4. Navigation around the app must follow a clear and consistent sequence of actions.

4.2.4.1. There must be no unexpected results when a user performs an action.

4.2.5. The system must allow reversal of actions.

4.2.5.1. The user must be able to revert back to the home page, or the previous page of the app easily.

4.2.6. The system must respond with informative feedback to users actions.

4.2.6.1. Success and error messages should be displayed when a user performs an action.

4.3. Compliance

4.3.1. All application data must be stored in compliance with the Personal Data Protection Act (PDPA) regulations.
4.3.2. All events must adhere to Singapore laws for maximum gathering size.

4.4. Security

4.4.1. The system must authenticate a user via a user login ID and password.
4.4.2. The system must censor the password field with dots during login for privacy reasons.
4.4.3. The user's personal information must be protected against data leakages.
4.4.3.1. User email, password, and location data should be protected against malware.
4.4.4. Users must feel safe using the application as it deals with strangers.
4.4.4.1. Users are subjected to bans after reaching a certain threshold of reports.

4.5. Supportability

4.5.1. The system must be able to run on all android devices.

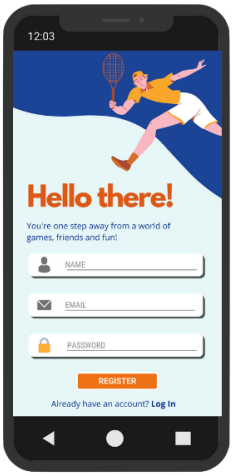
4.6. Reliability

4.6.1. The application must be fully functional at all times.
4.6.1.1. The system database must be online 24/7.
4.6.1.2. The system database must be updated in real time.
4.6.1.3. The system database must have no anomalies.

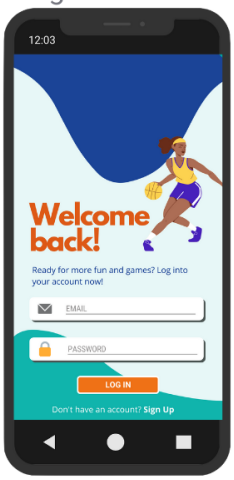
5. UI Mock-up

Welcome Page

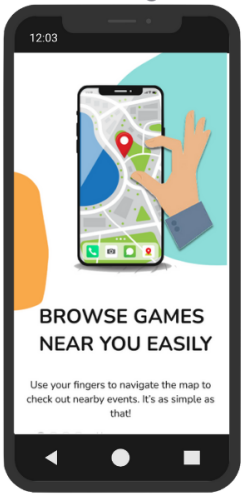
Registration



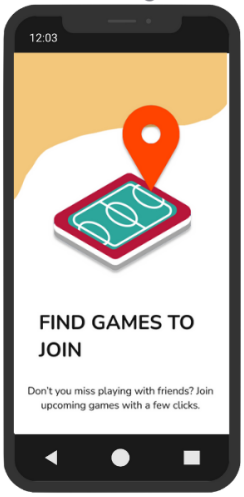
Log in



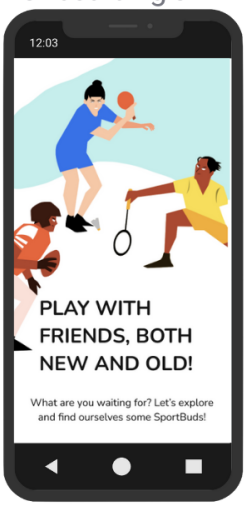
Onboarding 1



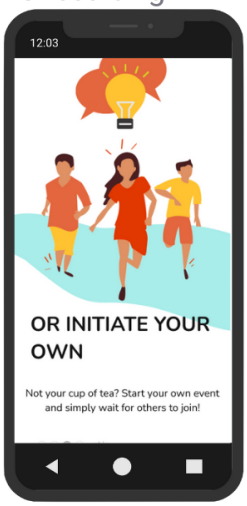
Onboarding 2



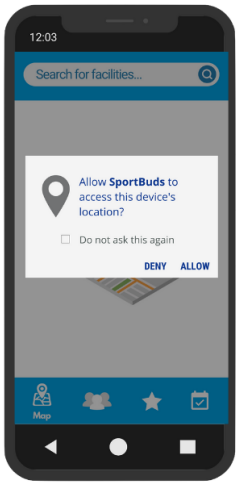
Onboarding 3



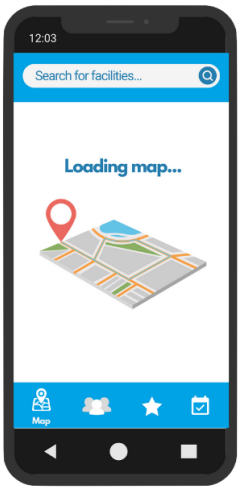
Onboarding 4



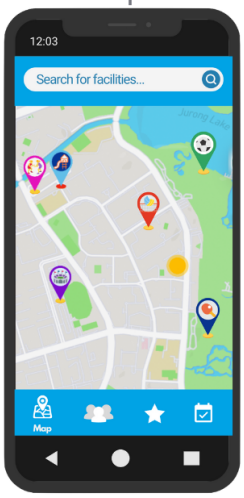
Location Permission



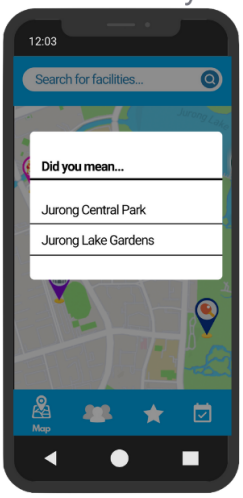
Loading



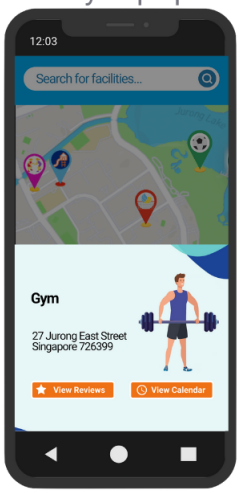
View Map



Search Facility

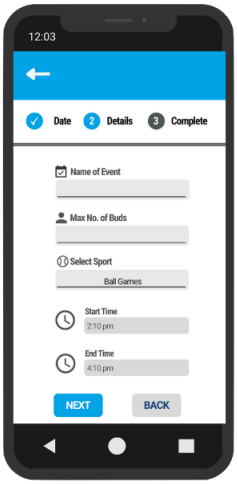


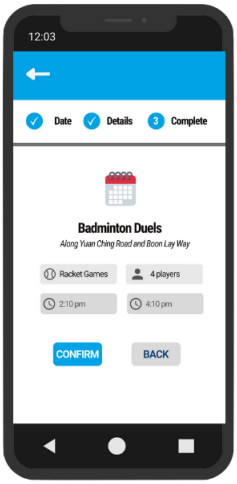
Facility Pop-up



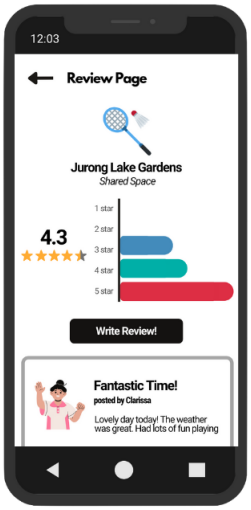
View Calendar

Create Event

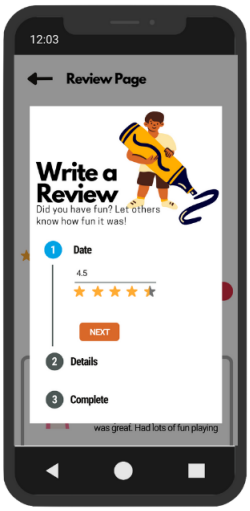


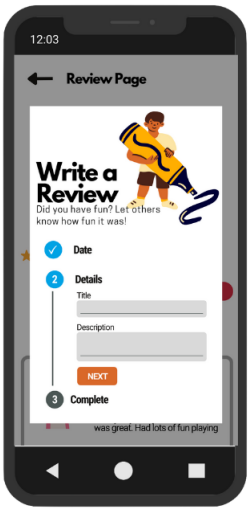


View Reviews

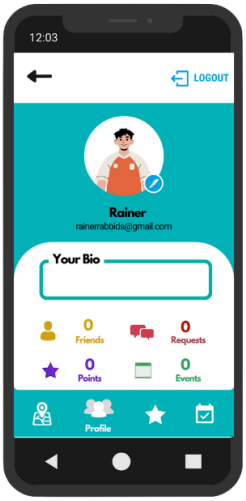


Post Review

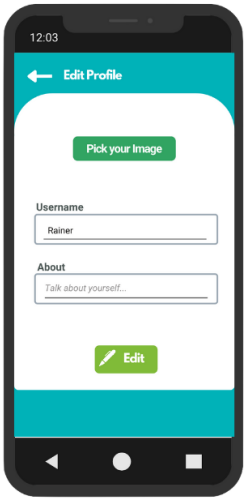




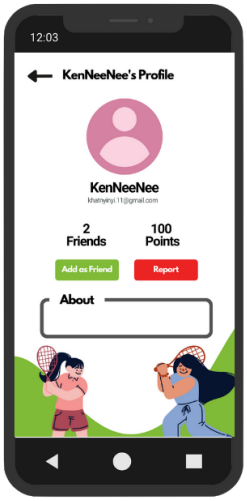
Profile



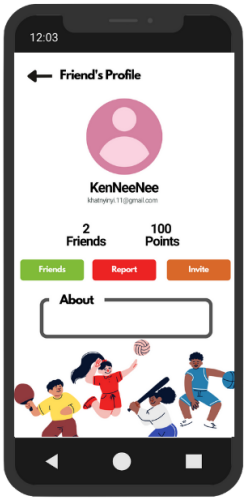
Edit Profile



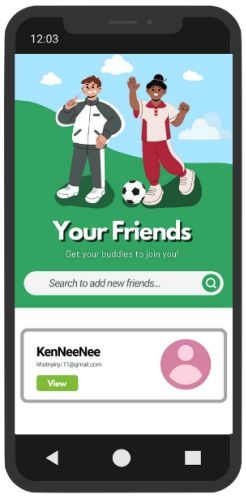
Other's Profile



Friend's Profile

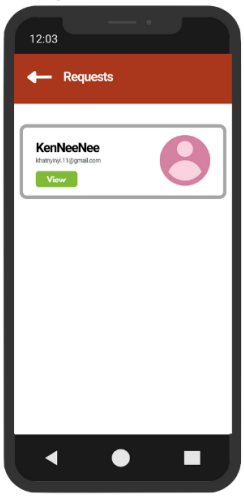


Friends List



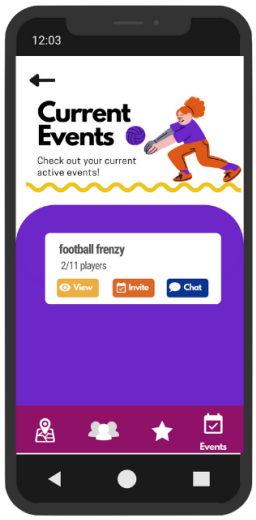
Friend Request

Request List



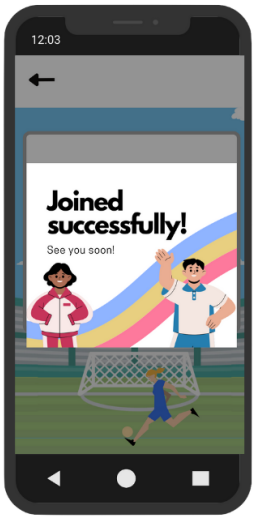
Report

View Active Events

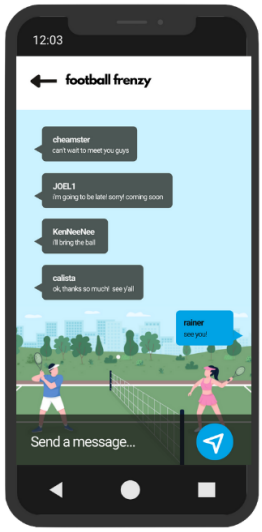


View Event (that user has not joined)

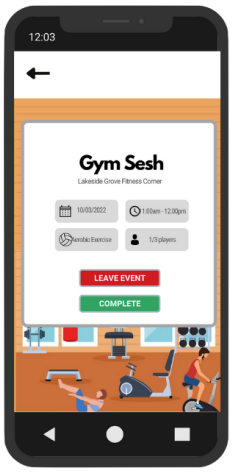
Joined Event



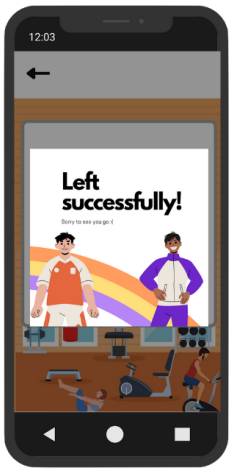
Event Chat



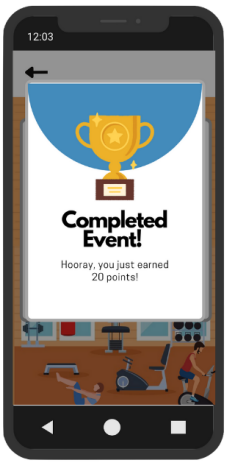
View Event (that user has joined)



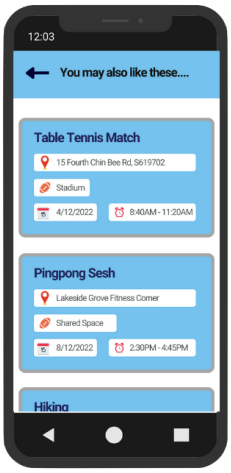
Leave Event



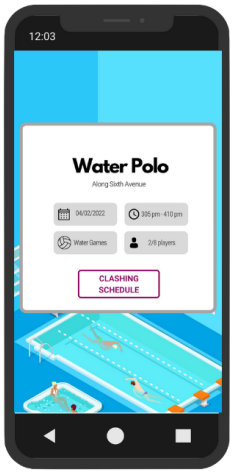
Complete Event

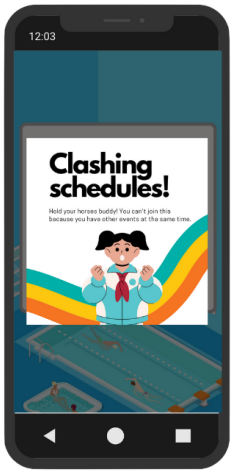


View Recommended

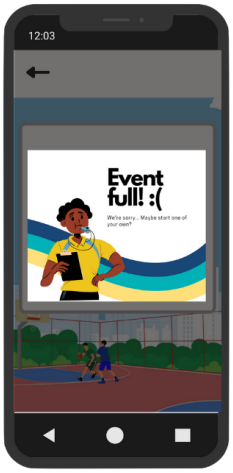


View Event (clash in schedules)

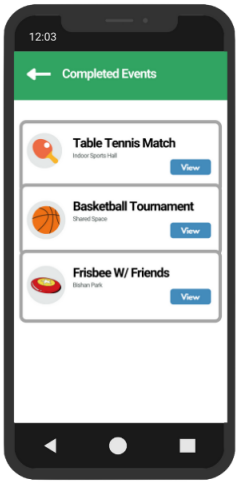




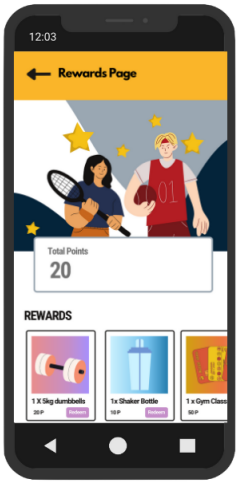
View Event (event full)



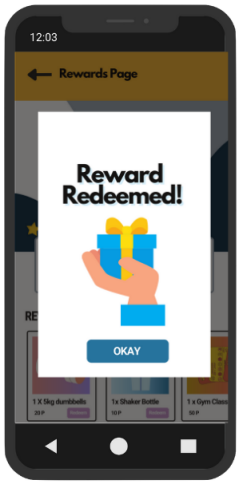
View Completed Event History



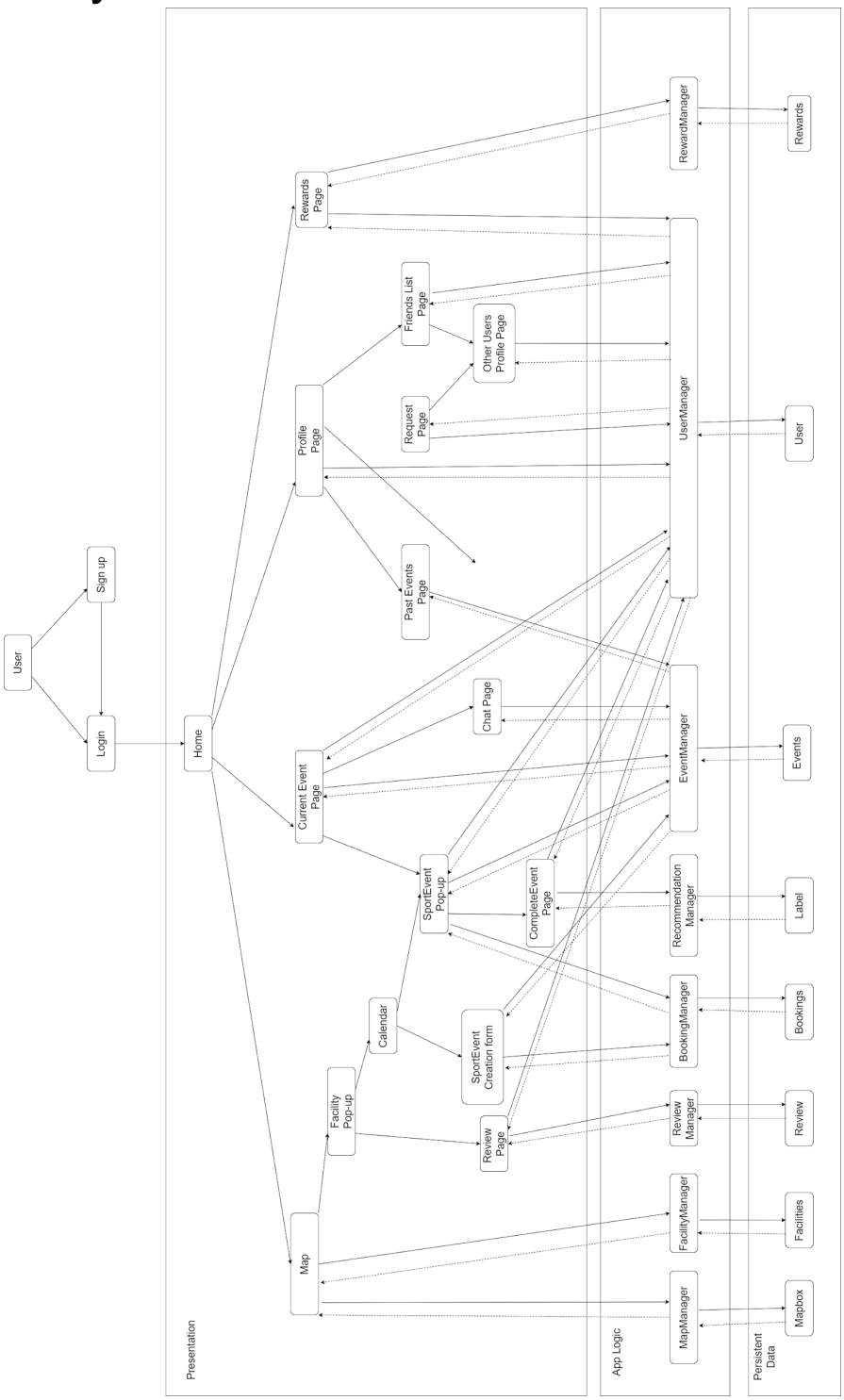
View Rewards



Claim Reward

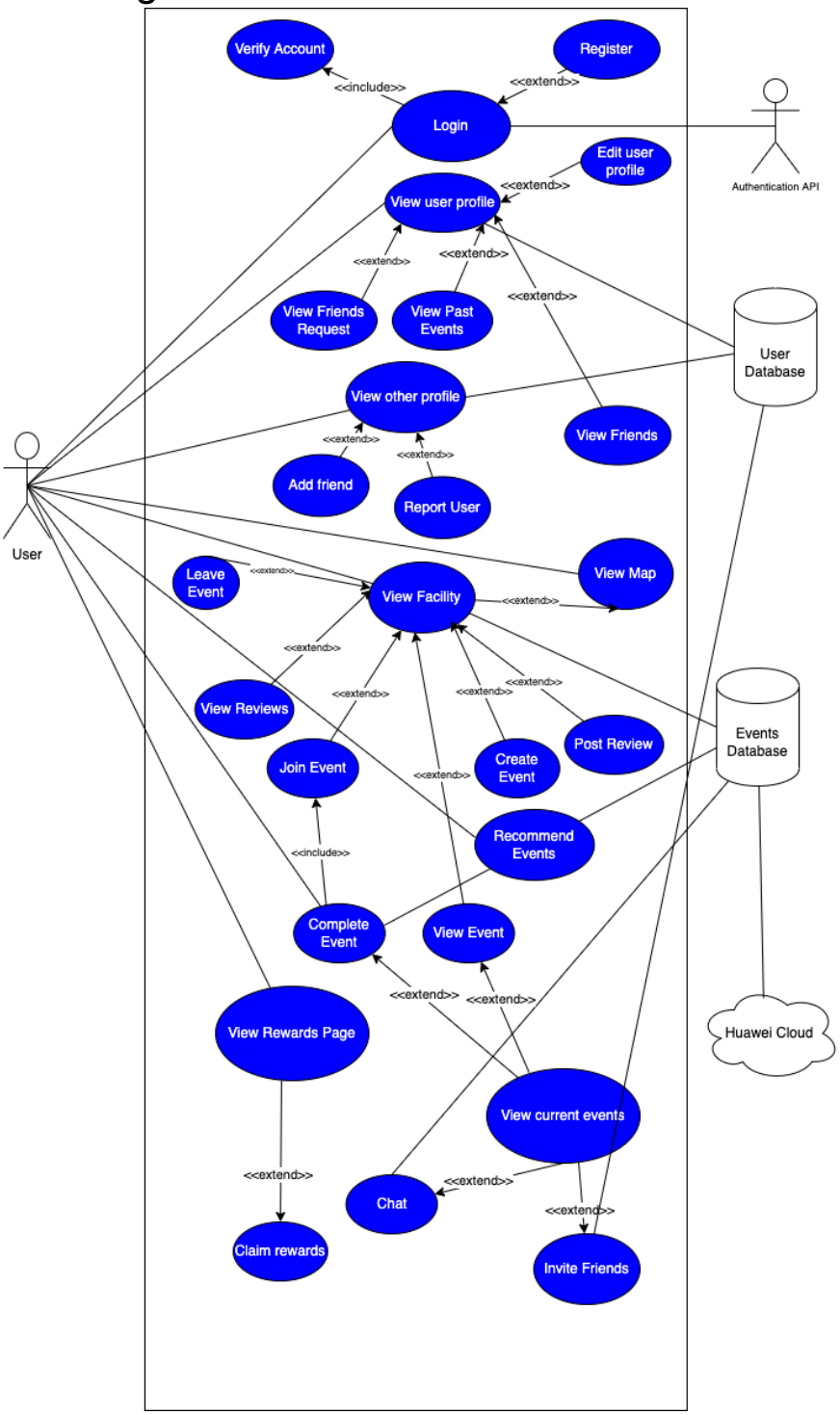


6. System Architecture



7. Use cases

7.1 Use Case Diagram



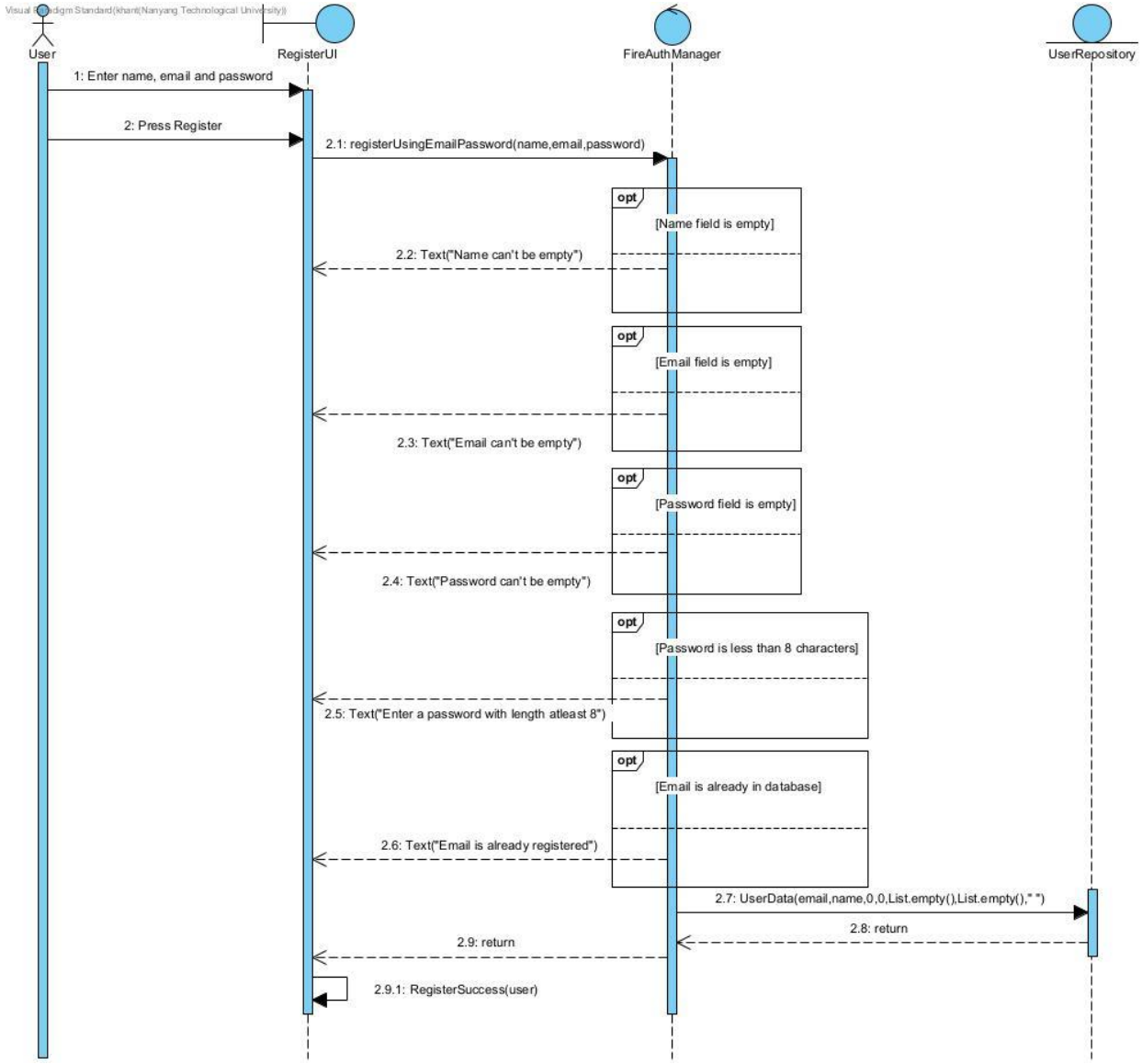
7.2.1 Use Case 1: Register for a new account

7.2.1.1 Use Case Description:

Use Case ID:	1
Use Case Name:	Account registration.
Created By:	Khant **On (Date):** 30 Jan 2022
Last Updated By:	Clarissa **On (Date):** 13 April 2022
Actor:	User, Firebase.
Description:	Registration for a new user account.
Preconditions:	1. The user has not logged onto the system. 2. Active internet connection.
Postconditions:	1. User's credentials will be registered into the firestore. 2. Once a user is registered, the user will be able to log in to the application.
Priority:	High.
Frequency of Use:	1 time per account.
Flow of Events:	1. This use case starts if the user is not logged in and clicks on the 'Create Account' button on the app's starting screen. 2. Alternatively, this use case starts if the user is not logged in and clicks on the 'Sign Up' text button on the app's login screen. 3. The system displays a blank registration form. 4. The user enters their email address, username and password. 5. The system validates the entered email address, username and password. 6. A new account is created for the user in the Firestore database. 7. The system displays a message indicating that the registration was successful. The user may opt for a tutorial. a. If the user chooses to skip the tutorial, the user is brought to their respective homepage. b. Otherwise, the system will bring the user to the onboarding screen.

Alternative Flow:	AF1-S4: System detects empty field / invalid input 1. The system displays an error message corresponding to the missing information ('Email is required.', 'Password is required.' ,'Passwords must be at least 6 characters.', 'Email is already registered') 2. The use case resumes at step 3.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User’s phone is capable of connecting to the internet
Notes and Issues:	

7.2.1.2 Sequence Diagram:



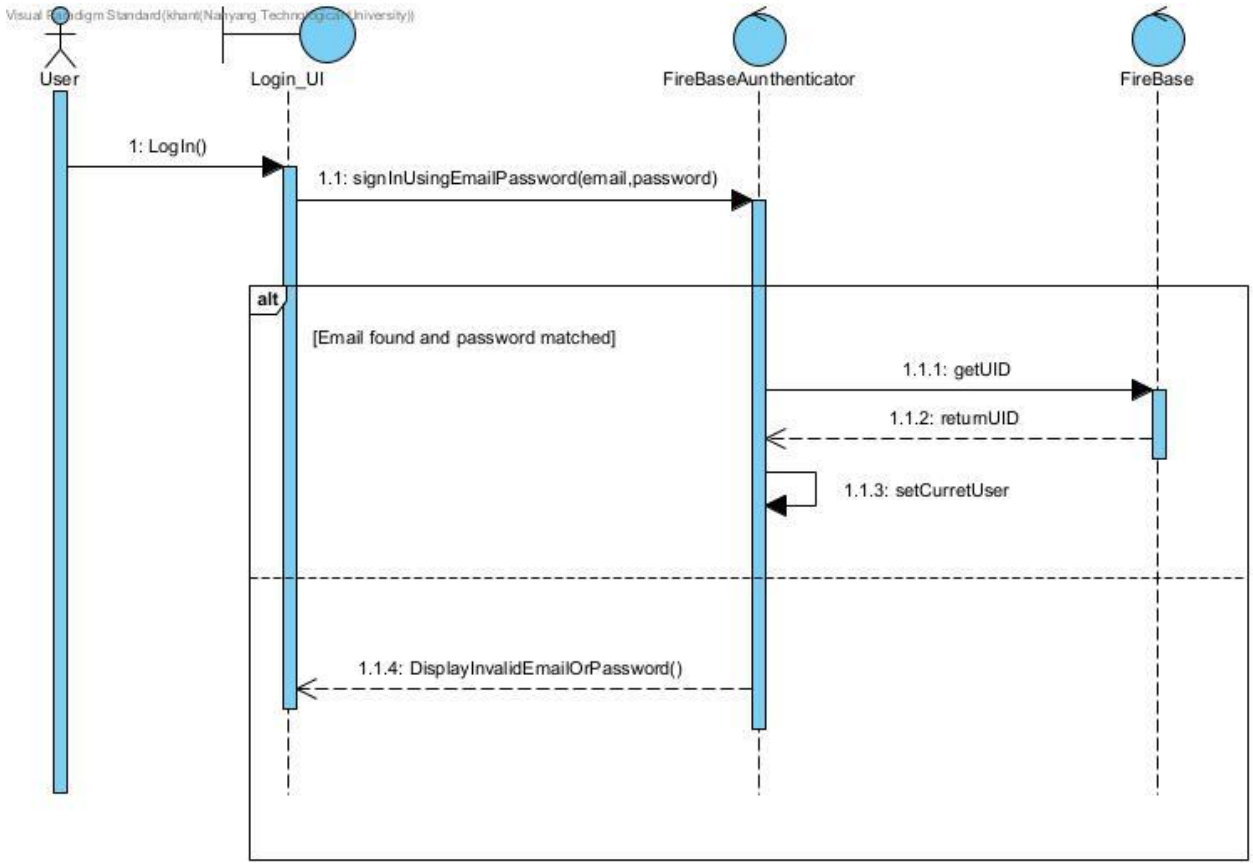
7.2.2 Use Case 2: Login

7.2.2.1 Use Case Description

Use Case ID:	2
Use Case Name:	Login
Created By:	Joel **On (Date):** 30 Jan 2022
Last Updated By:	Clarissa **On (Date):** 20 Feb 2022
Actor:	User, Firebase
Description:	Authenticate the user to use the application.
Preconditions:	1. The user has not logged onto the system. 2. Active internet connection.
Postconditions:	1. The user is logged into the system.
Priority:	High.
Frequency of Use:	High.
Flow of Events:	1. This use case starts if the user is not logged in and clicks on the 'Log In' button on the app's starting screen. 2. Alternatively, this use case starts if the user is not logged in and clicks on the 'Log In' text button on the app's registration screen. 3. The user enters their email address and password. 4. The system validates the entered username and password. 5. The user is logged in and will be brought to the homepage.
Alternative Flow:	AF1-S3: System detects empty field 1. The system displays an error message corresponding to the missing information ('Email is required.', 'Password is required.' ,'Passwords must be at least 6 characters.', 'Email is already registered') 2. The use case resumes at step 3. AF2-S3: Mismatch of account details 1. The system displays an error message indicating the mismatch in email and password ('Email/Password entered is invalid.') 2. The use case resumes at step 3.

Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to the internet
Notes and Issues:	May need to prevent malicious login attempts.

7.2.2.2 Sequence Diagram:



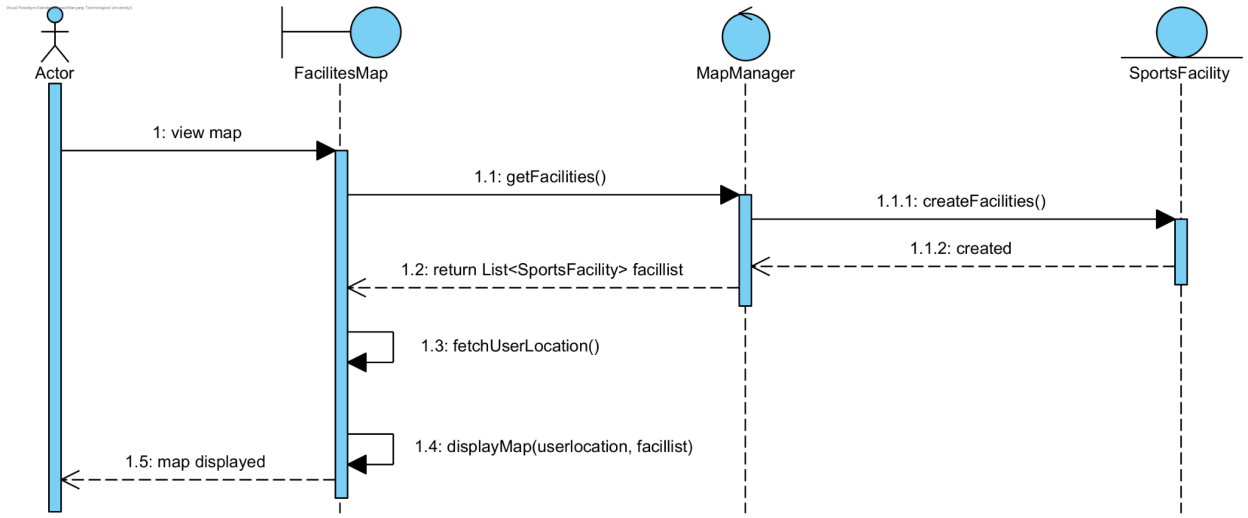
7.2.3 Use Case 3: View Map

7.2.3.1 Use case description

Use Case ID:	3
Use Case Name:	View Map
Created By:	Rainer **On (Date):** 30 Jan 2022
Last Updated By:	Rainer **On (Date):** 13 April 2022
Actor:	User
Description:	Displays user location and facility locations on the FacilitiesMap
Preconditions:	1. User must be logged in
Postconditions:	1. System shows user location on the FacilitiesMap 2. System shows facilities as markers on the FacilitiesMap 3. System shows specific details about a SportsFacility
Priority:	High.
Frequency of Use:	High.
Flow of Events:	1. The user is brought to this page after they have logged into the system. 2. The system will fetch location data on SportsFacilities and display them as markers on the FacilitiesMap. 3. The system will fetch the user's location. 4. The system will display the FacilitiesMap centered on the user's location. 5. The user may zoom in and out of the FacilitiesMap by pinching the screen with two fingers. 6. The user may search for a SportsFacility using the search bar. 7. The user clicks on a marker to view more details about a SportsFacility. 8. The system triggers a pop-up displaying the selected SportsFacility's name, address, facility type and calendar. a. The system fetches the schedule at this SportsFacility for the current week from the database. 9. A user clicks on a date to view the list of events for that day.
Alternative Flow:	AF1-S3 : 1. If location services is disabled, the system will set the user's location on the FacilitiesMap as the center of Singapore (1.354, 103.941).

	AF2-S9: 1. The user taps anywhere outside of the pop-up. 2. The user is returned to the FacilitiesMap.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet and location services
Notes and Issues:	

7.2.3.2 Sequence Diagram:

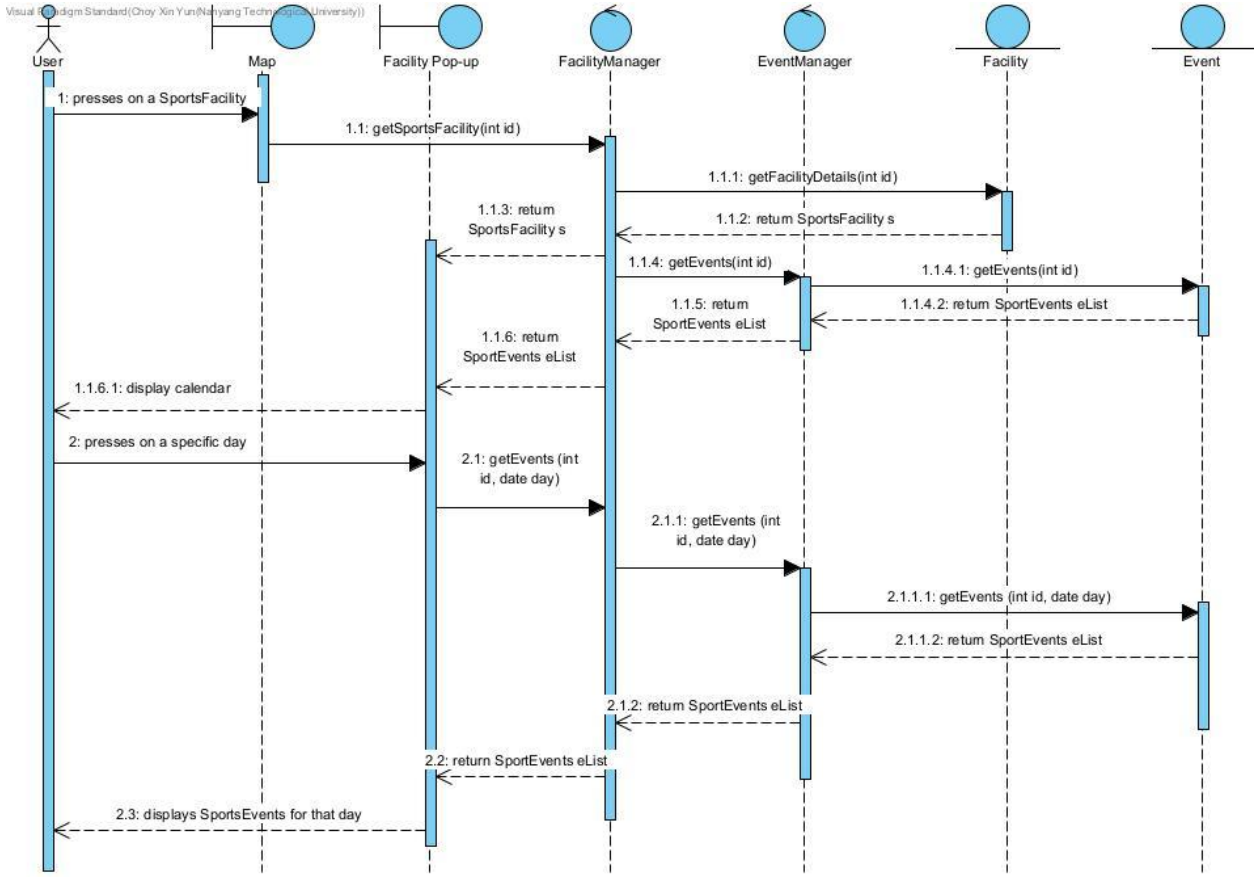


7.2.4 Use Case 4: View Facility

7.2.4.1 Use case description

Use Case ID:	4
Use Case Name:	View Facility
Created By:	Khant **On (Date):** 10 Feb 2022
Last Updated By:	Calista **On (Date):** 14 Apr 2022
Actor:	User
Description:	Views the overall schedule of SportEvents for the week for a particular facility
Preconditions:	1. User must be logged in 2. User's phone must be connected to the internet
Postconditions:	
Priority:	High.
Frequency of Use:	High.
Flow of Events:	1. User taps the facility to view 2. User taps on the day for which he/she would like to view the SportEvent schedule 3. System displays all SportEvents on that day
Alternative Flow:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.4.2 Sequence Diagram:



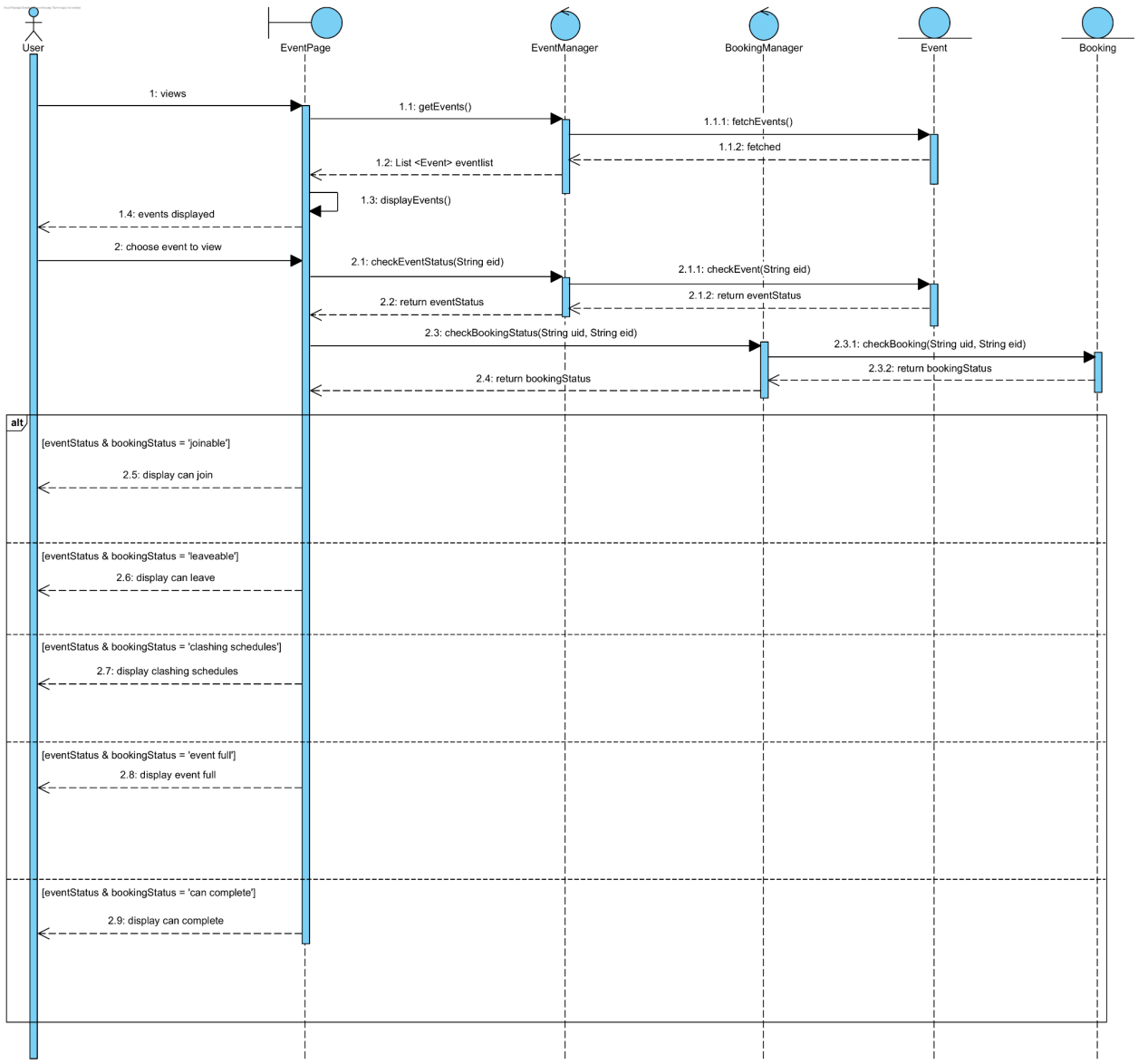
7.2.5 Use Case 5: View Event Details

7.2.5.1 Use case description:

Use Case ID:	5
Use Case Name:	View Event
Created By:	Khant **On (Date):** 07 Feb 2022
Last Updated By:	Joel **On (Date):** 13 April 2022
Actor:	User, Firebase
Description:	Users view a SportEvent from the FacilitiesMap
Preconditions:	1. User must be logged in. 2. User must have selected a specific SportEvent.
Postconditions:	1. System displays selected event details.
Priority:	High.
Frequency of Use:	High.
F7.2.1.2 Sequence Diagram:**low of Events:**	1. This use case will begin when the user clicks on a SportEvent from viewing a SportsFacility from the FacilitiesMap. 2. The system will display the SportEvent name, start time, end time, type of SportEvent, maximum capacity and current capacity. 3. If this SportEvent has reached its maximum capacity, an 'Event Full' button is displayed. 4. If the user has yet to join this SportEvent and has no other SportEvent at this time, the system will display a 'Join' button. 5. If the user has already joined another SportEvent at this time frame, the system will display a 'Clashing Schedules' dialog. 6. If the user has already joined this SportEvent, the system will display a 'Leave' button. a. If the current time is within the timeframe of the selected SportEvent, the system must also display a 'Complete Event' button. b. If the current time is before the start time of the SportEvent, a message telling the user to wait is displayed instead of the 'Complete Event' button. c. If the current time is after the end time of the SportEvent, a message telling the user that the event has expired is displayed instead of the ‘Complete Event’ button. 7. The user clicks anywhere outside of the pop up dialog to return to the FacilitiesMap. The use case ends here.

Alternative Flow:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User’s phone is capable of connecting to internet
Notes and Issues:	

7.2.5.2 Sequence Diagram:



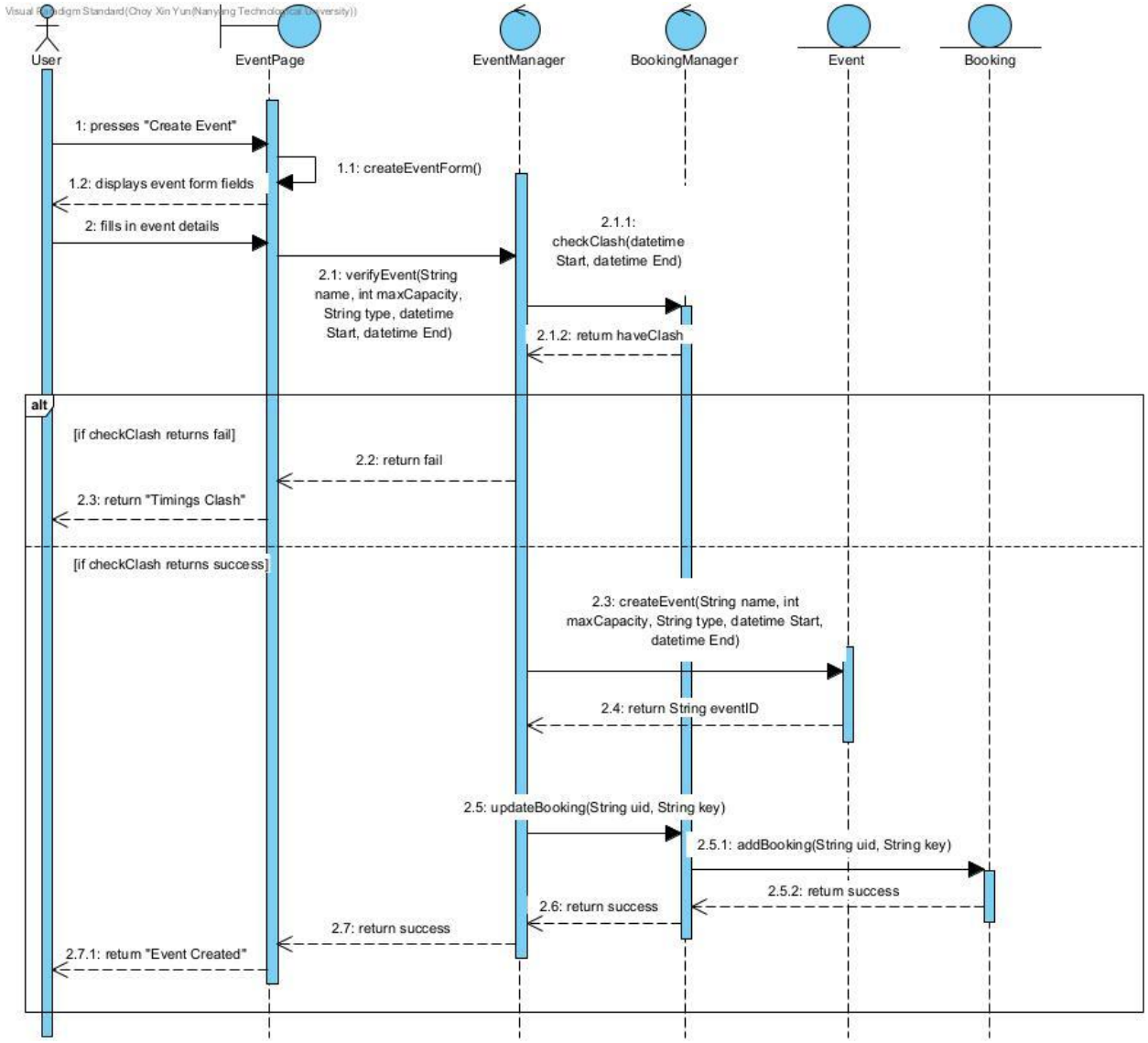
7.2.6 Use Case 6: Create Event

7.2.6.1 Use case description:

Use Case ID:	6
Use Case Name:	Create Event
Created By:	Khant **On (Date):** 07 Feb 2022
Last Updated By:	Rainer **On (Date):** 13 April 2022
Actor:	User, Firebase
Description:	User creates a SportsEvent at a SportsFacility
Preconditions:	1. User must be logged in. 2. User's phone must be connected to the internet. 3. User has selected a specific SportsFacility by clicking on the corresponding marker on the FacilitiesMap. 4. The User has selected a specific date on the Sports Facility's calendar.
Postconditions:	1. SportEvent is created at the facility at specified date and time. 2. The user who created SportEvent will be considered as having joined the event. SportEvent will be created with a current capacity of 1.
Priority:	High.
Frequency of Use:	High.
Flow of Events:	1. The system displays the form pop up dialog. This is triggered when the user clicks on the 'Create Event' button from viewing a SportsFacility. 2. The user enters the name for the SportEvent. 3. The user enters the maximum capacity for the SportEvent. 4. The user selects the type of SportEvent to be done. 5. The start time of the SportEvent is automatically set as the current time. a. The user may pick a different time for the SportEvent's start time. 6. The user selects the SportEvent's end time. 7. The user clicks on the 'Submit' button. 8. If the SportEvent has been created successfully, the system will display a success message. 9. The system will automatically add the user as a participant for this SportEvent. 10. Upon acknowledgement of the event creation status (success/failure), the dialogs will close and the user will be returned to the FacilitiesMap. The use case ends here.

Alternative Flow:	AF1-S6: 1. Selected timebox for the SportEvent to be created overlaps with User's Active Events timeboxes. 2. System displays a 'Clashing Schedules' message. 3. User is returned to the FacilitiesMap. AF2-S6: 1. SportEvent fields to be created do not have valid inputs. 2. System prompts the user to enter valid inputs. 3. Return to step 2.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.6.2 Sequence Diagram:



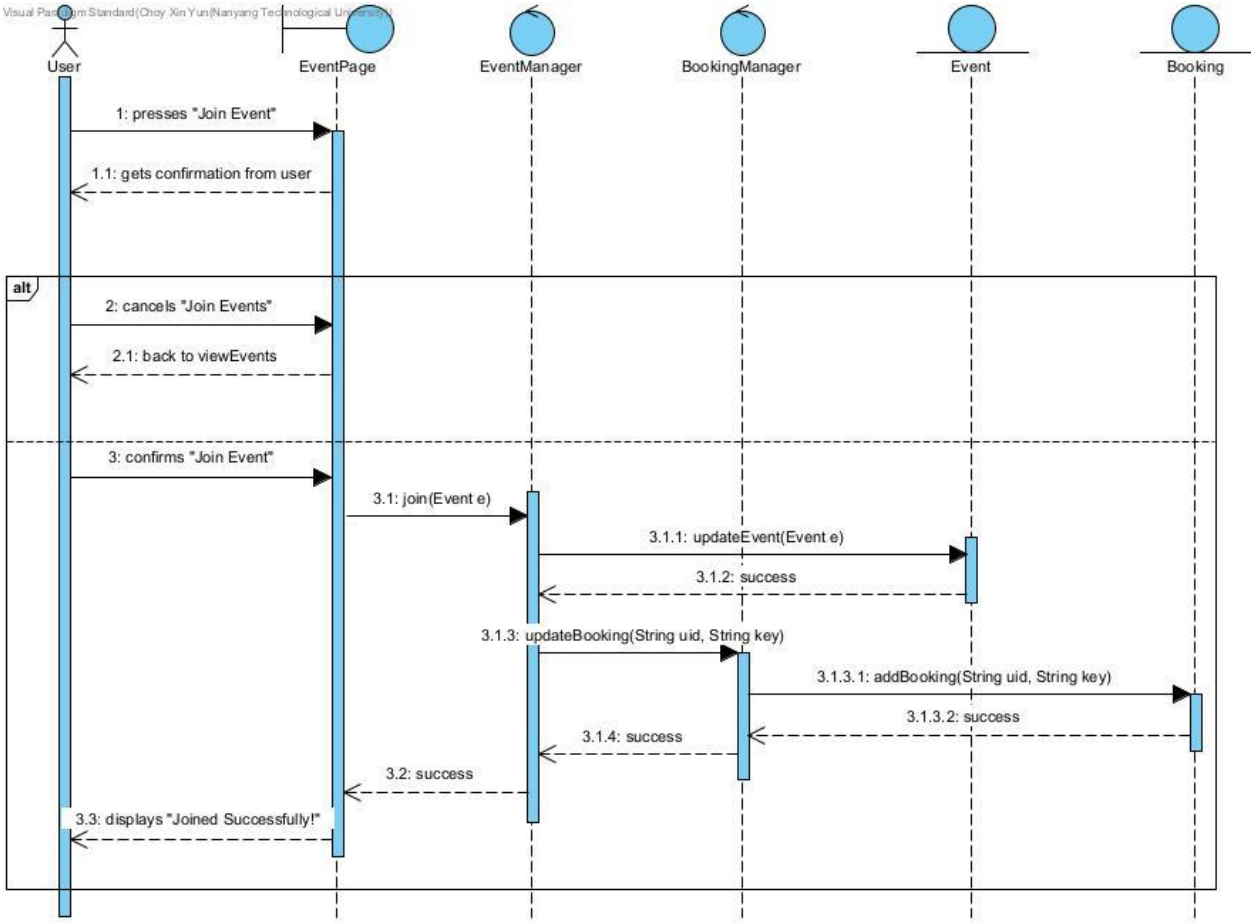
7.2.7 Use Case 7: Join Event

7.2.7.1 Use case description:

Use Case ID:	7
Use Case Name:	Join event
Created By:	Khant **On (Date):** 07 Feb 2022
Last Updated By:	Rainer **On (Date):** 13 April 2022
Actor:	User, Firebase
Description:	Users join a created event from the FacilitiesMap
Preconditions:	1. User must be logged in. 2. The selected SportEvent has at least one vacancy. 3. User must not have Active Events with clashing schedules with the selected SportEvent.
Postconditions:	1. Event is added to User's Active Events. 2. User is added to the buds list of the event.
Priority:	High.
Frequency of Use:	High.
Flow of Events:	1. This use case begins when the user clicks on the 'Join' button from the SportEvent page. 2. The user clicks on this and the system sends a confirmation prompt ('Yes' / 'Cancel'). 3. The user clicks on 'Yes' to confirm that they wish to join the SportEvent. 4. The system will add this user to this SportEvent. 5. The current capacity displayed will be increased by one.
Alternative Flow:	AF1-S3: 1. User tap 'Cancel' to decline action 2. Event is not added to User's Active Events
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet

Notes and Issues:	

7.2.7.2 Sequence Diagram:



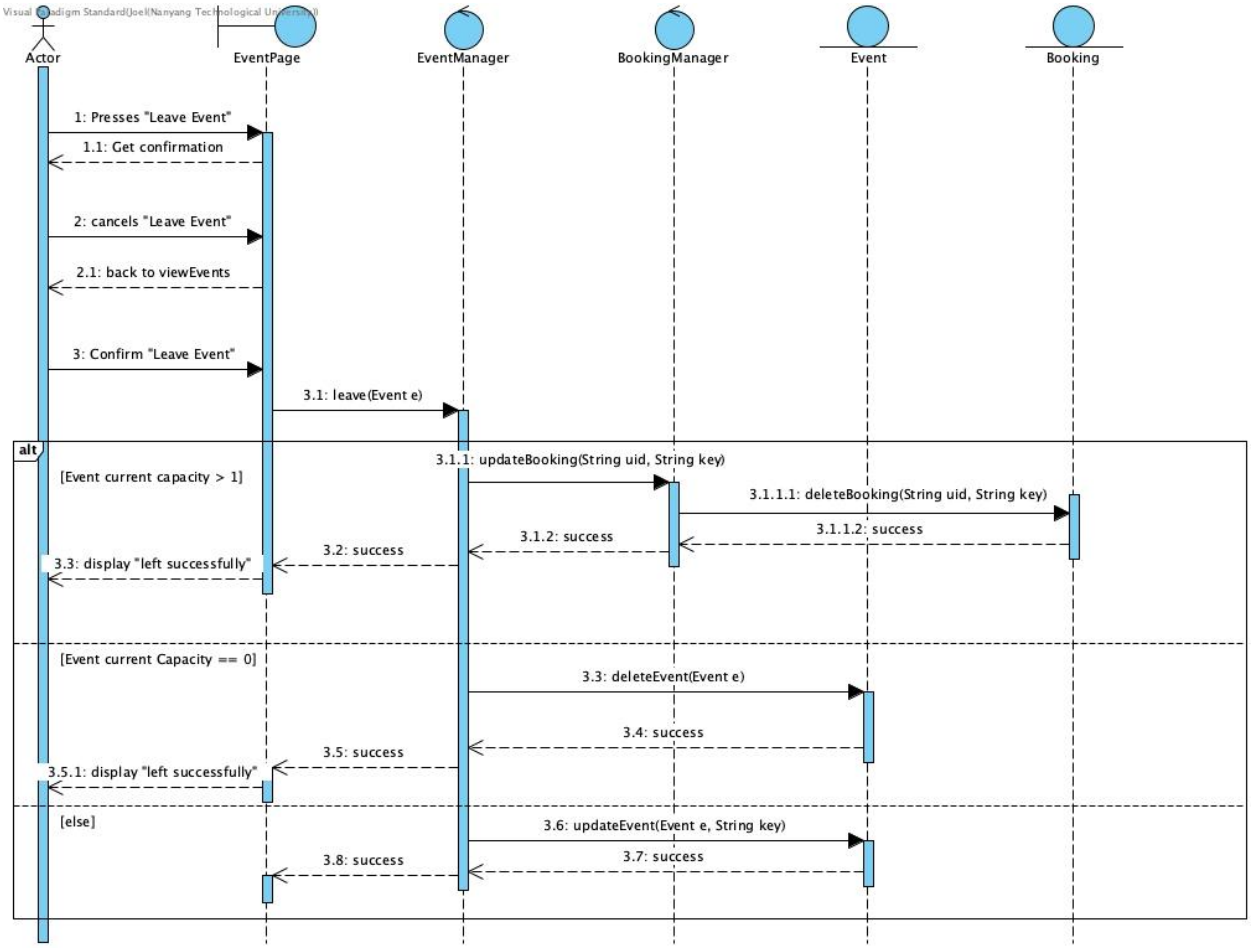
7.2.8 Use Case 8: Leave Event

7.2.8.1 Use case description:

Use Case ID:	8
Use Case Name:	Leave event
Created By:	Khant **On (Date):** 07 Feb 2022
Last Updated By:	Rainer **On (Date):** 20 Feb 2022
Actor:	User, Firebase
Description:	Users leave an Active Event.
Preconditions:	1. User must be logged in. 2. User's phone must be connected to the internet. 3. Selected events must be in User's Active Events.
Postconditions:	1. SportEvent is removed from User's Active Events. 2. SportEvent's current capacity is decreased by 1.
Priority:	High.
Frequency of Use:	Medium
Flow of Events:	1. This use case begins when the user clicks on the 'Leave' button from the SportEvent page. 2. The user clicks on this and the system sends a confirmation prompt ('Yes' / 'Cancel'). 3. The user clicks on 'Yes' to confirm that they wish to leave the SportEvent. 4. The system will remove the selected SportEvent from the User's Active Events. 5. The current capacity for the SportEvent will be reduced by one. If the SportEvent has zero participants remaining, it will be deleted.
Alternative Flow:	AF1-S3: 1. User taps 'Cancel' to decline action 2. Event is not removed from User's Active Events
Exceptions:	
Includes:	

Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.8.2 Sequence Diagram:



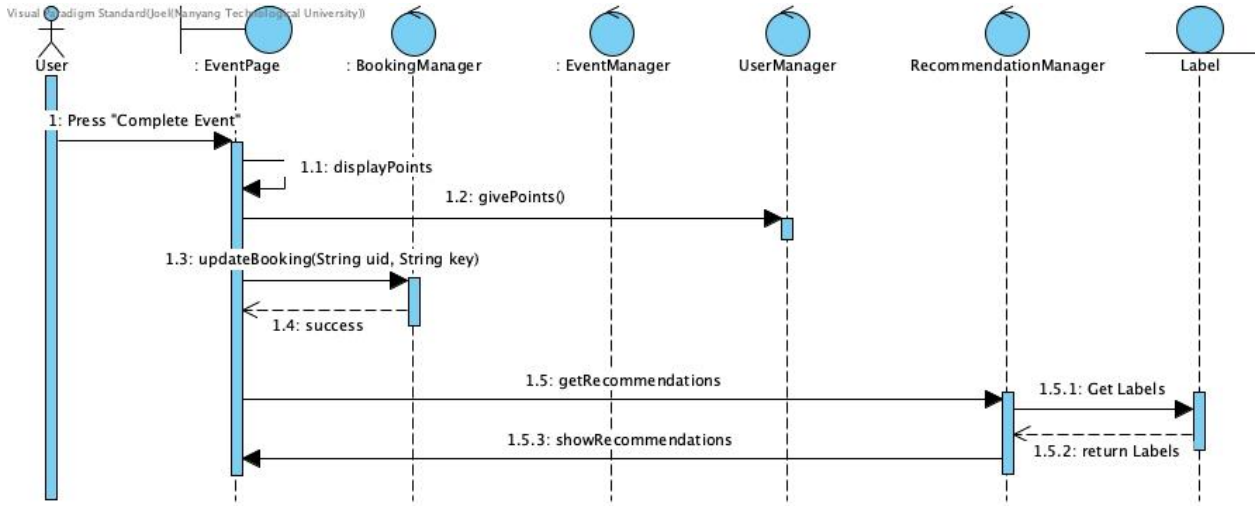
7.2.9 Use Case 9: Complete Event

7.2.9.1 Use case description

Use Case ID:	9
Use Case Name:	Complete Event
Created By:	Joel **On (Date):** 28 Mar 2022
Last Updated By:	Calista **On (Date):** 13 April 2022
Actor:	User, Firebase
Description:	User completes an Active Event.
Preconditions:	1. User must be logged in. 2. User's phone must be connected to the internet. 3. User have joined an event and the current time is after the start time and before the end time. 4. User's phone must have location services enabled. 5. User must be within 100m of the location of the event.
Postconditions:	1. The SportEvent is completed for User and becomes a past SportEvent. 2. Points are added for the User upon completion of the SportEvent. 3. Future SportEvents are recommended to the user based on a clustering algorithm.
Priority:	High.
Frequency of Use:	Medium.
Flow of Events:	1. This use case begins when the user clicks on the 'Complete' button from the View SportEvent page. 2. The user clicks on this and the system sends a confirmation prompt ('Yes' / 'Cancel'). 3. The user clicks on 'Yes' to confirm that they wish to complete the SportEvent. 4. The system will set the SportEvent to be completed for this user. 5. The SportEvent is removed from the user's Active Events, and added to the user's Completed Events. 6. The system displays a dialog informing the User that the SportEvent has been completed. 7. User acknowledges the dialog by clicking the 'OK' button. 8. The system displays a list of recommended SportEvents to the user.

Alternative Flow:	AF1-S2: 1. The user clicks 'Cancel'. 2. User is returned to the FacilitiesMap.
Exceptions:	If the user's current time is before the event time, the button is greyed out and informs the user that the event has yet to start. If the user's current time is after the event time, the event will no longer be an Active Event and can no longer be completed.
Includes:	Join Event
Special Requirements:	
Assumptions:	User's phone is capable of connecting to the internet. User's phone is capable of location services.
Notes and Issues:	

7.2.9.2 Sequence Diagram:

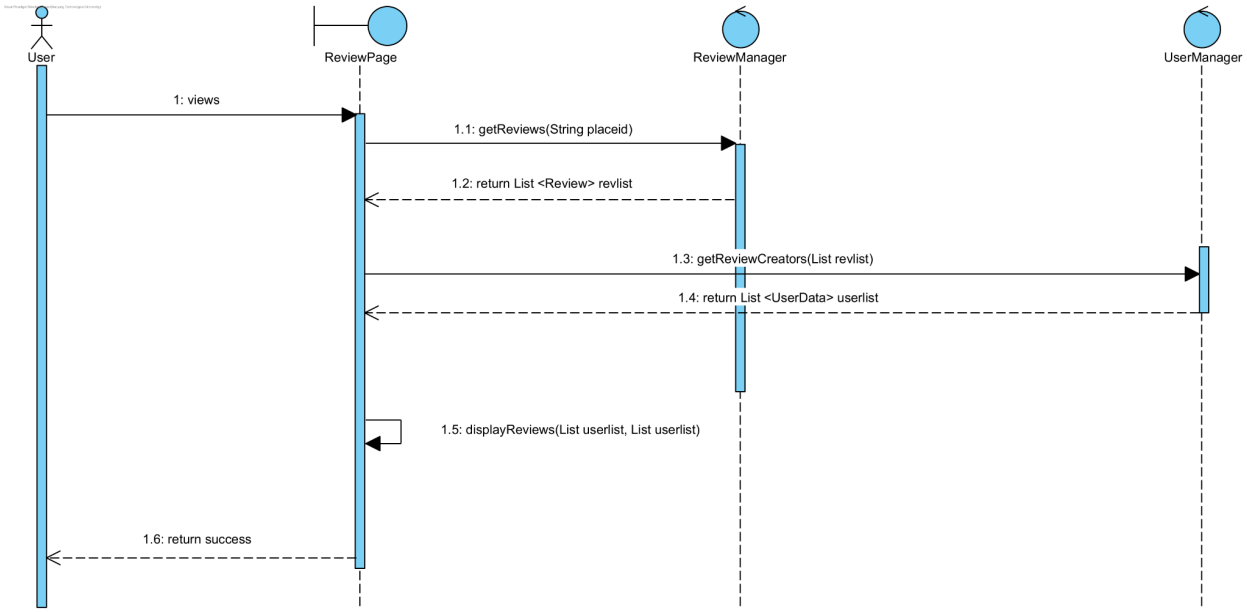


7.2.10 Use Case 10: View Reviews

7.2.10.1 Use case description

Use Case ID:	10
Use Case Name:	View Reviews
Created By:	Rainer **On (Date):** 26 March 2022
Last Updated By:	Rainer **On (Date):** 13 April 2022
Actor:	User, Firebase
Description:	User views Reviews for a selected SportsFacility.
Preconditions:	1. User must be logged in. 2. User's phone must be connected to the internet. 3. User must have selected a SportsFacility from the FacilitiesMap.
Postconditions:	1. System displays all reviews tagged to the selected facility. 2. System displays overall rating for the selected facility.
Priority:	High.
Frequency of Use:	Medium.
Flow of Events:	1. This use case begins when the user taps on the 'View Reviews' button for a selected facility. 2. The review page for the selected facility is displayed.
Alternative Flow:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to the internet.
Notes and Issues:	

7.2.10.2 Sequence Diagram:



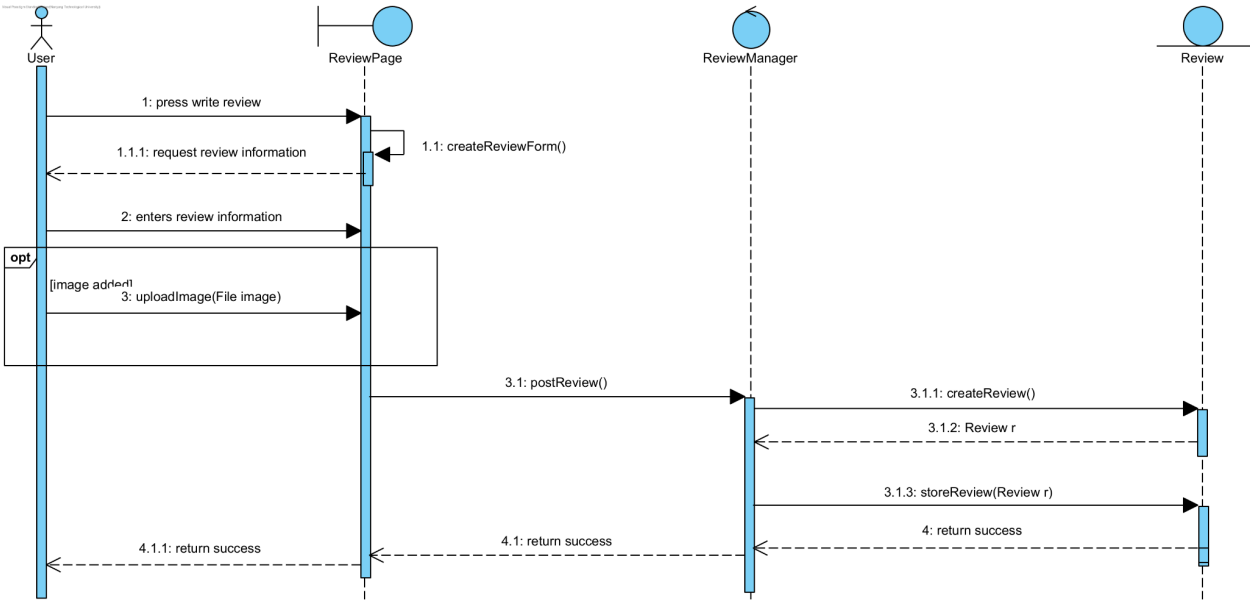
7.2.11 Use case 11: Post Review

7.2.11.1 Use case description

Use Case ID:	11
Use Case Name:	View Reviews
Created By:	Rainer **On (Date):** 26 March 2022
Last Updated By:	Rainer **On (Date):** 13 April 2022
Actor:	User, Firebase
Description:	User posts a Review for a selected SportsFacility.
Preconditions:	1. User must be logged in. 2. User's phone must be connected to the internet. 3. User must have entered View Reviews for a selected SportsFacility.
Postconditions:	1. A new Review is created for the selected SportsFacility.
Priority:	High.
Frequency of Use:	Medium.

Flow of Events:	1. This use case begins when the 'Write Review' button is pressed from the review page. 2. The user enters the Review title and review description. 3. The user selects a rating of at least 1 star and at most 5 stars. 4. The user taps on the 'Select photo' button. 5. The user selects a photo from their photo gallery to be added 6. User clicks on the 'Post' button. 7. The Review is tagged to the selected facility and inserted into the database.
Alternative Flow:	AF1-S4: 1. User chooses not to tap on 'Select photo'. 2. No photo is added to the Review. 3. Go to step 6. AF2-S6: 1. Review information does not have valid inputs. 2. System informs the user of invalid inputs. 3. Return to step 2.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.11.2 Sequence Diagram:



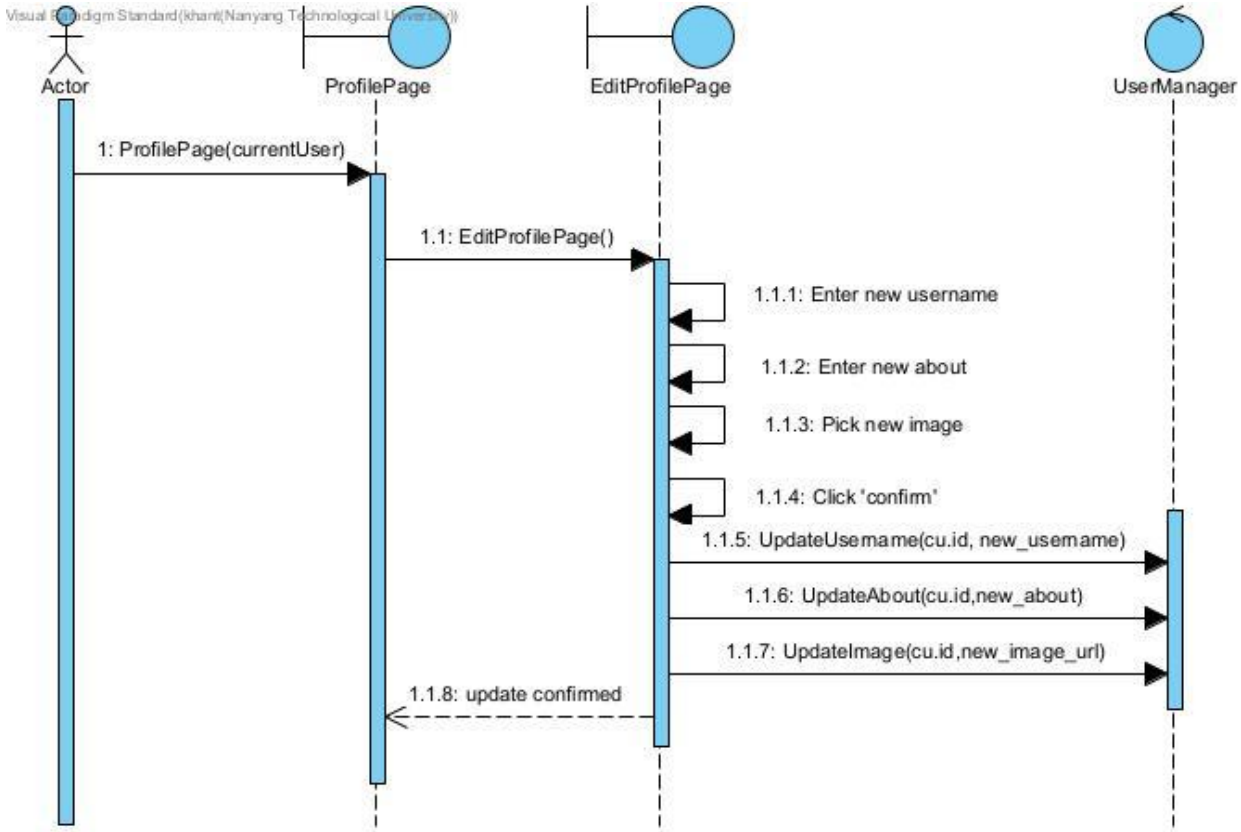
7.2.12 Use Case 12: View Profile

7.2.12.1 Use case description

Use Case ID:	12
Use Case Name:	View Profile
Created By:	Khant **On (Date):** 10 Feb 2022
Last Updated By:	Caleb **On (Date):** 13 April 2022
Actor:	User
Description:	User view their own profile page and optionally customize their profile page
Preconditions:	1. User must be logged in 2. User's phone must be connected to the internet 3. Users must be on their own profile page
Postconditions:	1. System displays User's username, email, profile picture and biographic description 2. System displays User's number of friends,friend requests,points and completed events
Priority:	High.
Frequency of Use:	Medium.

Flow of Events:	1. User taps on My Profile icon to view user profile 2. User may click on friend's button to display a list of their friends a. System displays a list of card for each of the User's Friends b. Users can click on 'Visit' on a Friend's card to visit the Friend's Profile. 3. User may click on friend request to display a list of their friend request 4. User may click on the edit profile icon to customize profile information a. Users can click on current username and change their username b. Users can click on current About and change their new About c. Users can pick a new profile picture from their gallery d. Users then confirm their changes by pressing the 'confirm' button 5. Users exit back to User profile page when done
Alternative Flow:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.12.2 Sequence Diagram:



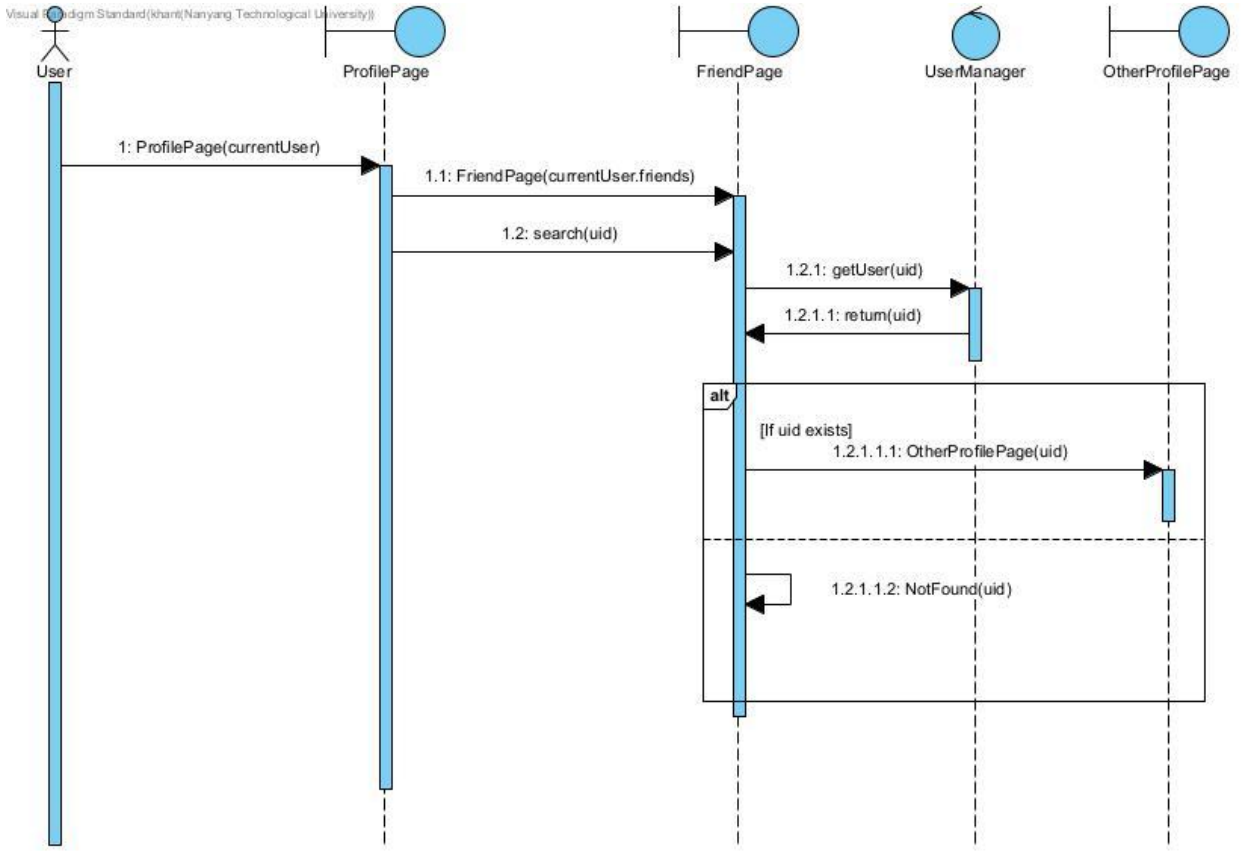
7.2.13 Use Case 13: View Other Users' Profile

7.2.13.1 Use case description

Use Case ID:	13
Use Case Name:	View other user's Profile
Created By:	Khant **On (Date):** 10 Feb 2022
Last Updated By:	Caleb **On (Date):** 24 Mar 2022
Actor:	User
Description:	User view other users' profile
Preconditions:	1. User must be logged in 2. User's phone must be connected to the internet
Postconditions:	1. Other user's profile is displayed 2. Current user is able to report this Other User 3. Current user is able to add this Other User as friend if Other User is not in the Current user's friends yet
Priority:	High.
Frequency of Use:	Medium.
Flow of Events:	1. User taps on My profile page 2. User taps on Friends 3. User types another user's email in the search bar 4. User taps on Search button 5. The system will now display that user's profile which includes their points attained, profile picture and About 6. The system will check if the current user is already friends with the other user they are viewing a. If the User is already friends with the other user, the button would display as 'Friends' and is not clickable b. If the User is not friends with the other user, the button will display 'Add Friend'. Users can click on 'Add Friend' button to send a friend requests' to the user they are viewing 7. Users can file a report to the Bud profile they are viewing by clicking on the 'Report' Button
Alternative Flow:	AF1-S3: 1. User taps on View on one of their friends' card 2. Return to step 5
Exceptions:	

Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.13.2 Sequence Diagram:

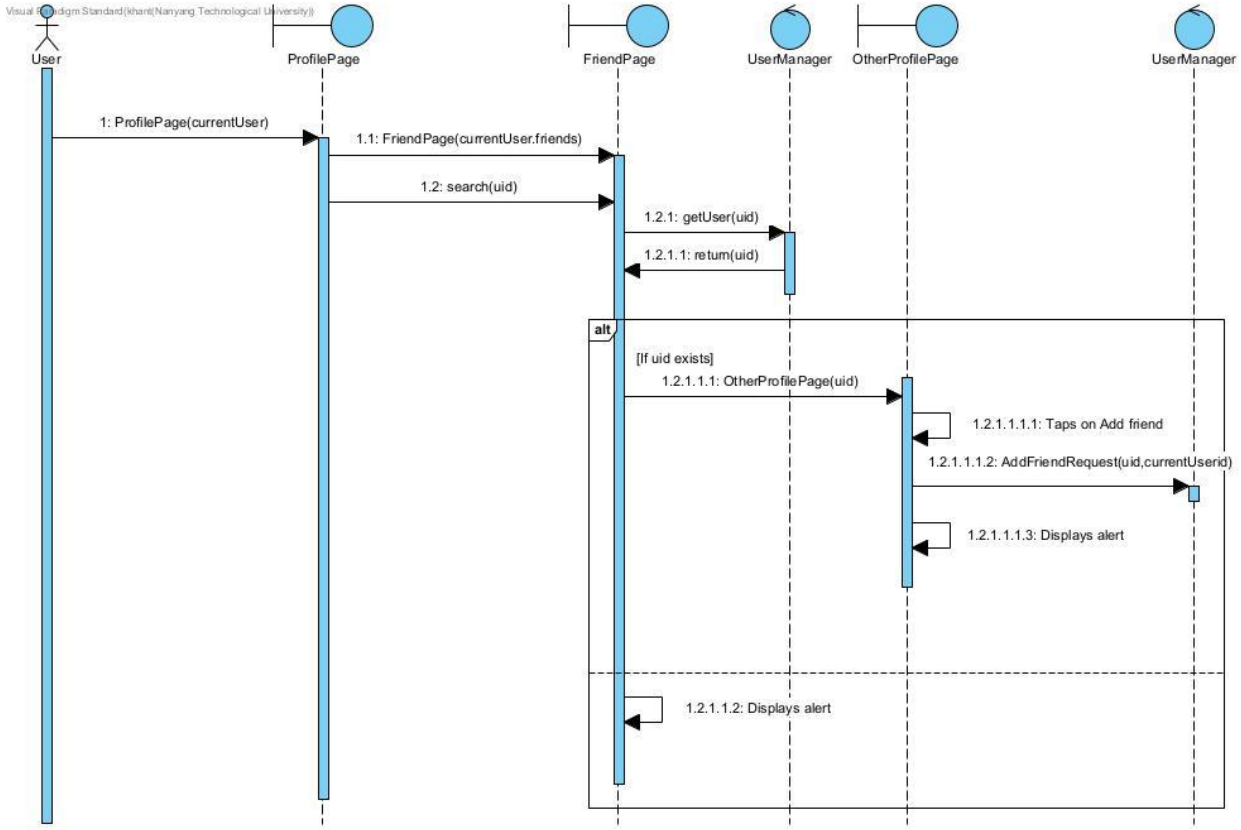


7.2.14 Use Case 14: Send Friend Request

7.2.14.1 Use case description

Use Case ID:	14
Use Case Name:	Send Friend Request
Created By:	Khant **On (Date):** 10 Feb 2022
Last Updated By:	Caleb **On (Date):** 13 April 2022
Actor:	User
Description:	User add other user as friend
Preconditions:	1. User must be logged in 2. User's phone must be connected to the internet 3. User is at other user's profile page 4. Other user must not be user's friend yet
Postconditions:	1. A friend request is sent to other user
Priority:	Medium.
Frequency of Use:	Medium.
Flow of Events:	1. User taps on 'Add friend' on other user's profile page 2. System displays a 'Friend Request sent' message if a friend request is successfully created. a. If a friend request has already been created, the system displays a 'A friend request already exists' message.
Alternative Flow:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.14.2 Sequence Diagram:

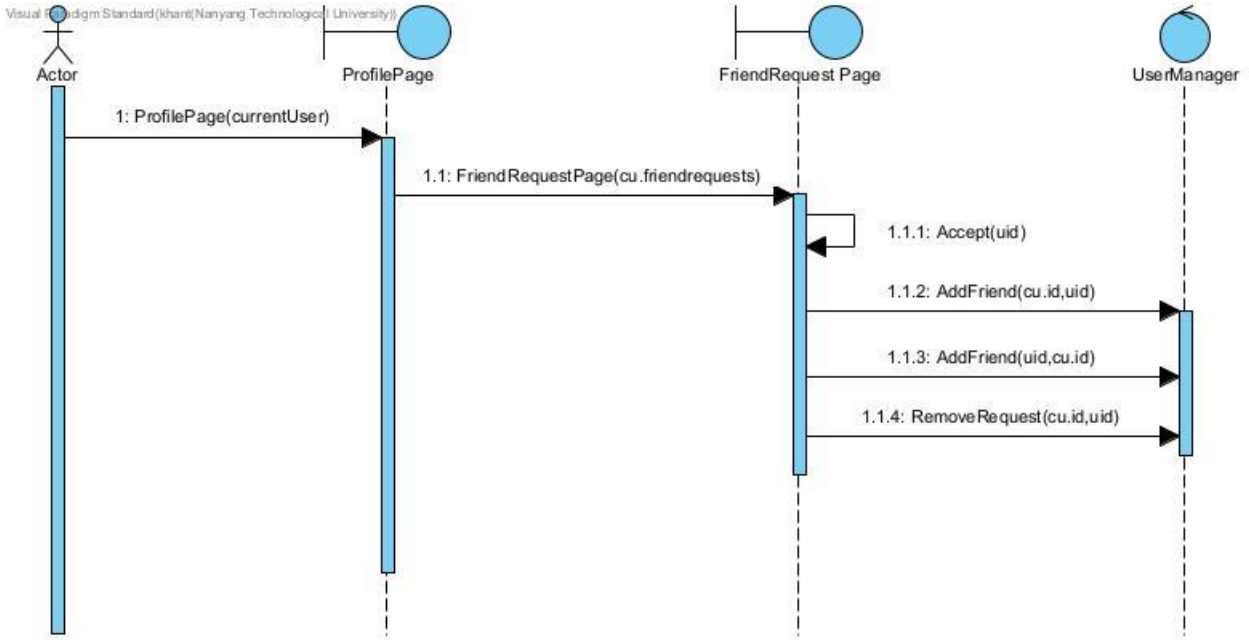


7.2.15 Use Case 15: Accept Friend Request

7.2.15.1 Use case description

Use Case ID:	15
Use Case Name:	Accept Friend Request
Created By:	Khant **On (Date):** 10 Feb 2022
Last Updated By:	Caleb **On (Date):** 24 Mar 2022
Actor:	User
Description:	User accept other user's friend request
Preconditions:	1. User must be logged in 2. User's phone must be connected to the internet 3. User has a friend request from other users
Postconditions:	1. Other user is added as user's friends
Priority:	Medium.
Frequency of Use:	Medium.
Flow of Events:	1. User taps on Friend Requests button on their profile page 2. User taps 'Accept' on a friend request card sent by other users
Alternative Flow:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.15.2 Sequence Diagram:

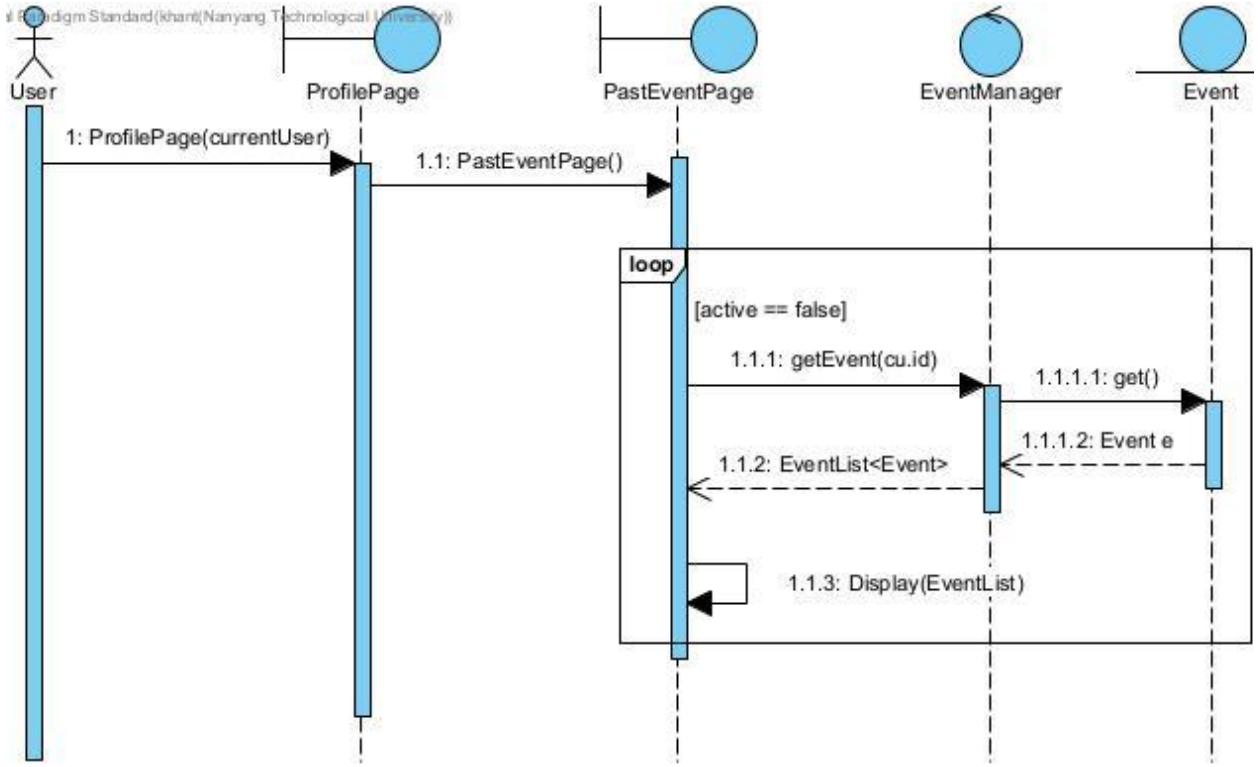


7.2.16 Use Case 16: View Past Event History

7.2.16.1 Use case description

Use Case ID:	16
Use Case Name:	View History of Events
Created By:	Khant **On (Date):** 10 Feb 2022
Last Updated By:	Joel **On (Date):** 13 Apr 2022
Actor:	User
Description:	Users view the entire record of events they have attended
Preconditions:	1. User must be logged in 2. User's phone must be connected to the internet
Postconditions:	1. System displays history of previous events joined by the user
Priority:	High.
Frequency of Use:	Medium.
Flow of Events:	1. User taps on My Profile Icon 2. User taps on My History 3. The system will display the User's event history
Alternative Flow:	
Exceptions:	If user has not attended any events, display 'No records found'
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.16.2 Sequence Diagram:

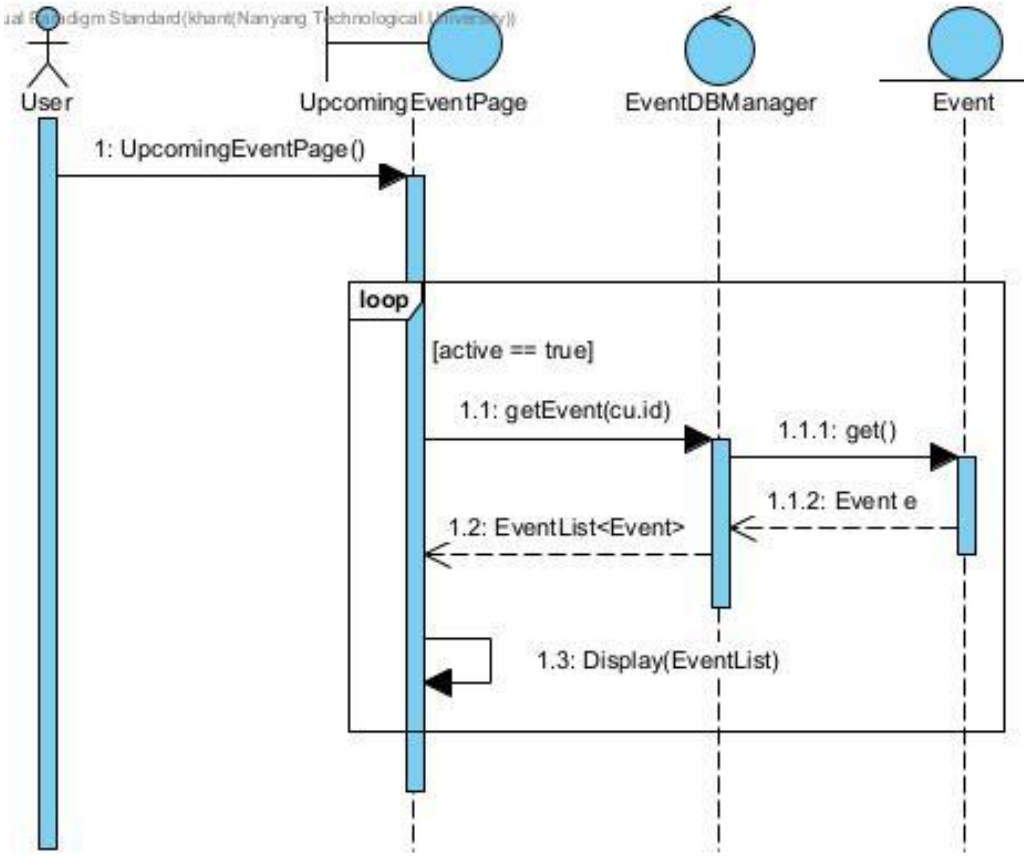


7.2.17 Use Case 17: View Current Events

7.2.17.1 Use case description:

Use Case ID:	17
Use Case Name:	View Current Events
Created By:	Khant **On (Date):** 10 Feb 2022
Last Updated By:	Caleb **On (Date):** 13 April 2022
Actor:	User
Description:	Users view the active events they have currently joined or created.
Preconditions:	1. User must be logged in 2. User's phone must be connected to the internet 3. User have joined or created at least one event
Postconditions:	1. System displays a list of active events of the user
Priority:	High.
Frequency of Use:	Medium.
Flow of Events:	1. User tap on the My Profile 2. User tap on View Events icon
Alternative Flow:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.17.2 Sequence Diagram:

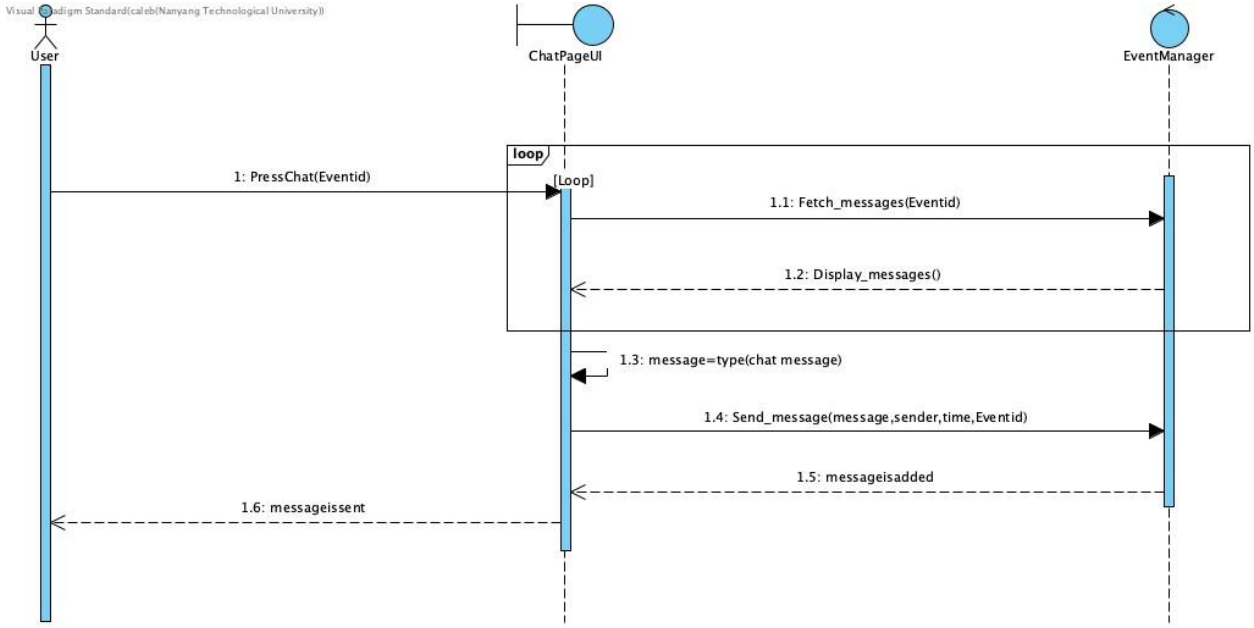


7.2.18 Use Case 18: Event Chat Room

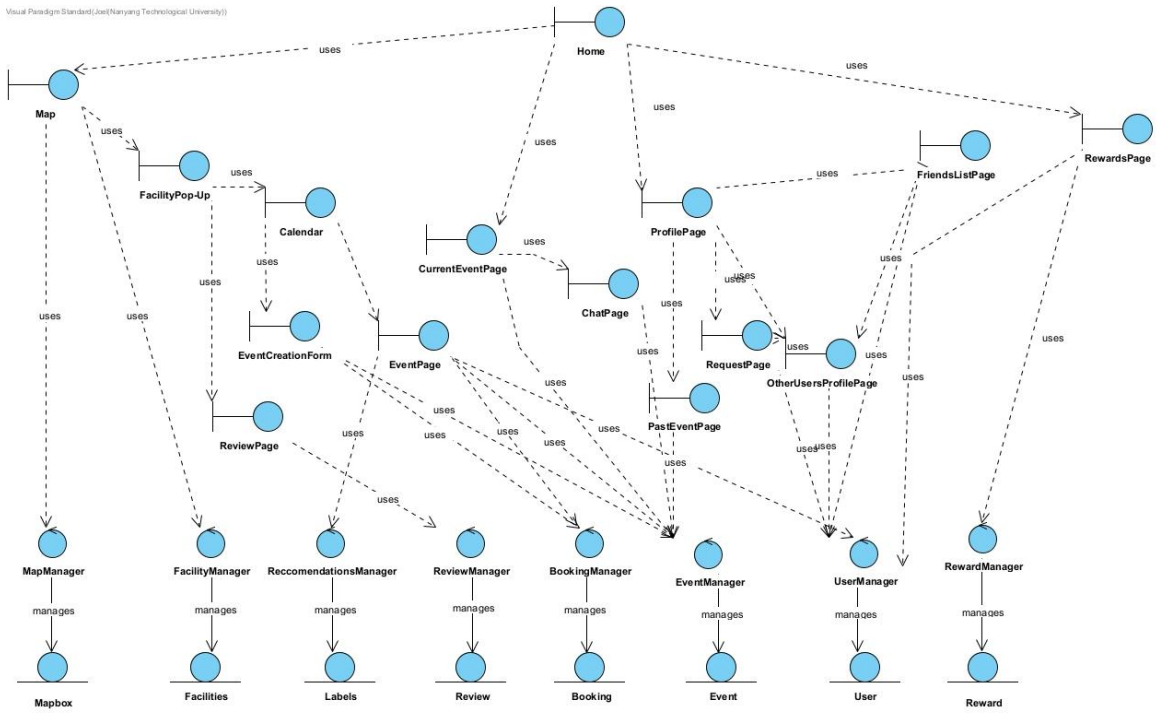
7.2.18.1 Use case description

Use Case ID:	18
Use Case Name:	Event Chat Room
Created By:	Khant **On (Date):** 10 Feb 2022
Last Updated By:	Caleb **On (Date):** 4 Apr 2022
Actor:	User
Description:	Chat room for users in the same event to communicate
Preconditions:	3. User must be logged in 4. User's phone must be connected to the internet 5. User must be in an active event
Postconditions:	
Priority:	Low.
Frequency of Use:	High.
Flow of Events:	4. User taps on Current Event Page 5. User taps on chat button of an event they have joined 6. User can view past chat history 7. User can send messages in the chat
Alternative Flow:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	User's phone is capable of connecting to internet
Notes and Issues:	

7.2.18.2 Sequence Diagram:

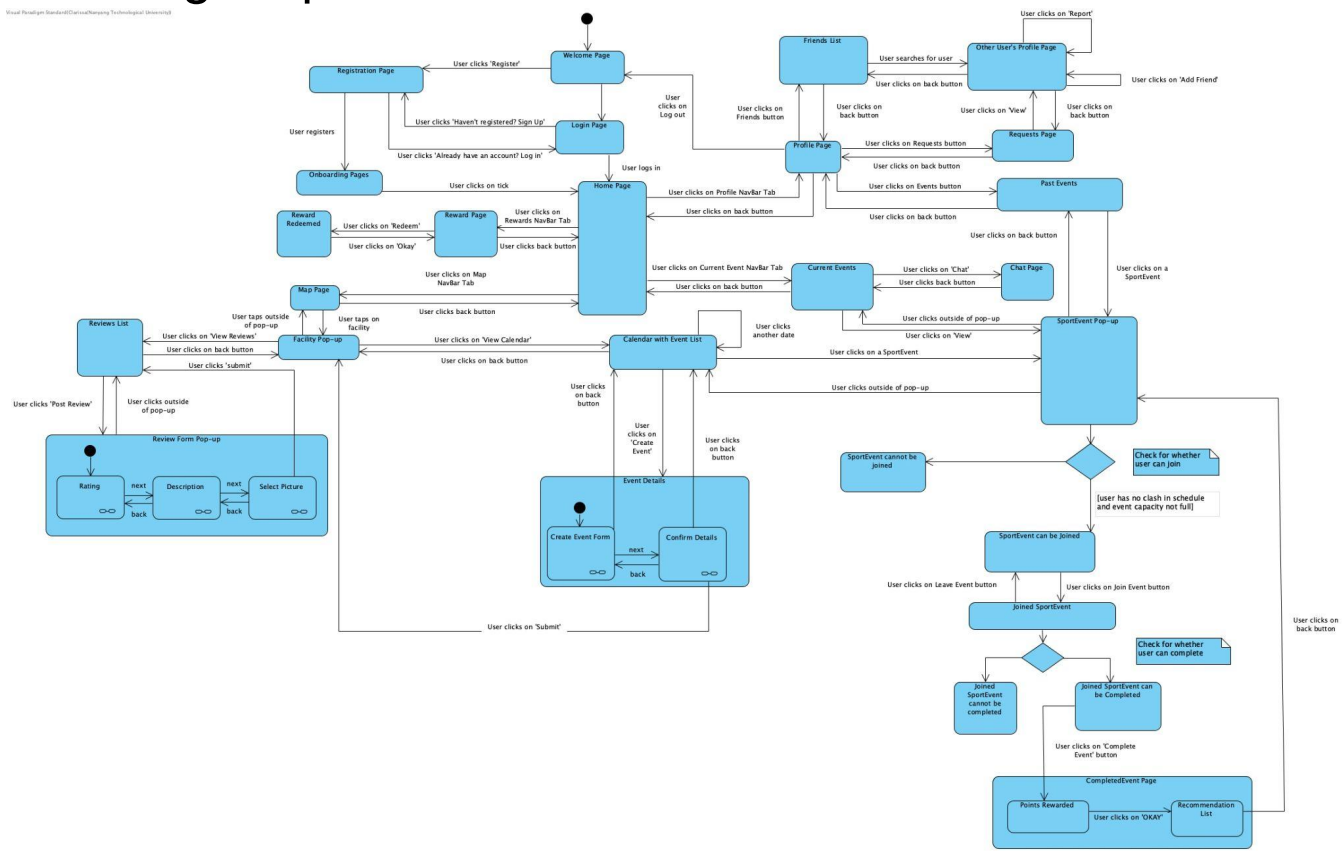


8. Class Diagram



0

9. Dialog Map



10. Good Software Engineering Practices

10.1 Standardised naming conventions

It is not uncommon for old code to be revisited and refactored as part of system maintenance and code optimization. Thus, it is important to ensure that code is readable and traceable through consistent and understandable naming conventions. Reading code that is straightforward and self-explanatory makes debugging and code maintainability significantly easier. We employed standard naming practises across components for this purpose. The view classes have '_page' appended to their names to indicate that they are the classes that users interact with. The utility classes have '_widget' appended to their names to indicate that they can be used in other classes to build elements more readily.

10.2 Adhering to design principles

It is crucial to employ object-oriented design principles to ensure easy maintainability and flexibility. As developers, we frequently deal with changing customer requirements and demands, which would be expensive and difficult to accommodate with “smelly” code. By adhering to the SOLID design principles, we can prevent such situations and make implementing changes easier. In our project, we made good use of the Single Responsibility Principle. We did this by efficiently separating our classes using the Model-View-Controller concept ensuring that each class has a single specific role to play as models, views and controllers with no overlapping responsibilities.

10.3 Agile methodologies

We adapted agile development methodologies similar to Scrum throughout the project in order to improve our team’s efficiency, thereby reducing the time taken to develop the app. Our project’s timeline was broken down into sprints which lasted one week. We held Scrum meetings on Mondays, where we determined the sprint backlog. Stand-up meetings were set on Thursdays, where members update on their progress on their tasks and address challenges faced. We also reviewed past sprints to estimate our team’s velocity and thus predict the amount of tasks to be completed in future sprints.

Our team also employed moderate pair programming, another agile development technique which originates from Extreme programming (XP). In this project, the team was further split into smaller teams :

1. Two members worked together on use cases relating to Users
2. Another three members worked on use cases relating to Events
3. The remaining member worked on the User Interface and integration of components

By adopting pair programming, there was continuous review of the code being developed, thereby resulting in fewer bugs in the program. Hence, this technique improves the quality of the code while facilitating team cohesion as members learn to communicate and work together.

10.4 Version Control

Our team employed a Git version control branch management strategy to manage and track changes to our source code throughout the project. A branch was created for every member to allow each member to isolate development work without affecting the main stable master branch and other member’s branches. This ensured that all changes pushed to the master branch were through pull requests instead of direct commits. Our team also followed git best practices which includes making small and frequent commits with concise and informative commit messages. This facilitates easier management of merge conflicts and bugs.

11. Testing

11.1 Black Box Testing

11.1.1 Register Function

With reference to 1.1 of our functional requirements, the application must accept a valid email, username and password when a user registers. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject any registration with badly formatted email, short password, or any missing field of user's information.

Equivalence classes:

Input data	Valid	Invalid
Username	Any string input	Empty string
email	Any valid email	Empty or badly formatted email
password	Any string with length of more than 6 characters	Any string with length of 6 characters or less

Test cases:

Inputs	Expected results	Actual results
"KhantNyiNyi" "khantnyinyi.11@gmail.com" "1234567"	Registration successful	Registration successful
"" "khantnyinyi.11@gmail.com" "1234567"	Registration unsuccessful	Registration unsuccessful
"KhantNyiNyi" "khantnyinyi.11.gmail.com" "1234567"	Registration unsuccessful	Registration unsuccessful
"KhantNyiNyi" "khantnyinyi.11@gmail.com" "123456"	Registration unsuccessful	Registration unsuccessful

11.1.2 Search User Function

With reference to 1.14 of our functional requirements, the search bar must accept a valid user email when a user searches for other users. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject any searches with invalid user emails.

Equivalence classes:

Input data	Valid	Invalid
email	Existing user email, which is not email of current user	Any string input that is not a user's email, Current user's email

Test cases:

Inputs	Expected results	Actual results
"calebcheam@gmail.com"	Search successful	Search successful
"calebcheam"	Search unsuccessful	Search unsuccessful

11.1.3 Edit profile information function

With reference to 1.13 of our functional requirements, the application must accept a valid username and description when users edit their profile information. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject any update of information with long description or username

Equivalence classes:

Input data	Valid	Invalid
About Description	Any string input less than or equal to 140 characters words	String input with more than 140 characters
Username	String input less than or equal to 20 characters	String input with more than 20 characters

Test cases:

Inputs	Expected results	Actual results
“Hi my name caleb and i like to play football” “caleb”	Edit description Successful	Edit description Successful
“Hi, my name is Caleb and I like to play football. In order for this description to be an invalid input, this will need to be more than 140 characters long” “caleb”	Edit description unsuccessful	Edit description unsuccessful
“Hi my name caleb and i like to play football” “Caleb cheam is too long”	Edit description unsuccessful	Edit description unsuccessful

11.1.4 Post review function

With reference to 1.8 of our functional requirements, the application must accept a valid review title, review description and facility rating when a user creates a review. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject any review with badly formatted title, description, rating or any missing field of review information. The application must accept reviews with and without uploaded pictures.

Equivalence classes:

Input data	Valid	Invalid
Title	Any input string with at least 4 characters and at most 50 characters	String input with less than 4 characters or more than 50 characters
Description	Any input string with at most 300 characters	String input with more than 300 characters
Rating	Any number of stars selected from the rating bar	Empty selection
Picture	Any pictures selected from gallery / no selection	Wrong file type

Test cases:

Inputs	Expected results	Actual results
“This is a valid title” “This is a description with less than 300 characters” 5 stars selected Image uploaded	Post Review Successful	Post Review Successful
“This is a valid title” “This is a description with less than 300 characters” 3 stars selected No image uploaded	Post Review Successful	Post Review Successful
“Valid title” “Valid description” No rating selected Image uploaded	Post Review unsuccessful	Post Review unsuccessful
“Hi, my name is Caleb and I	Post Review unsuccessful	Post Review unsuccessful

like to play football. In order for this title to be an invalid input, this will need to be more than 50 characters long” “Caleb” 1 star selected No image uploaded		
“Long description” “aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaa” 3 stars selected No image uploaded	Post Review unsuccessful	Post Review unsuccessful

11.1.5 Create Event

With reference to 1.5 of our functional requirements, the application must accept a valid event title, maximum capacity, start time, end time, event type when a user creates an event. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject any event with badly formatted title or invalid maximum capacity, start time, end time or any missing field of event information. The application must also reject events if the user creating the event has other Active Events which have overlapping timings with the newly created event.

Equivalence classes:

Input data	Valid	Invalid
Title	Any input string with at least 4 characters and at most 50 characters	String input with less than 4 characters or more than 50 characters
Maximum Capacity	Any integer input greater than 0 and at most 30	Integer input less than or equal to 0 or more than 30
Start time	Any DateTime selected from clock widget	Empty selection
End time	Any DateTime selected from clock that is after start time	Empty selection or time selected is before start time
Event type	Any string selected from drop-down menu	Empty selection

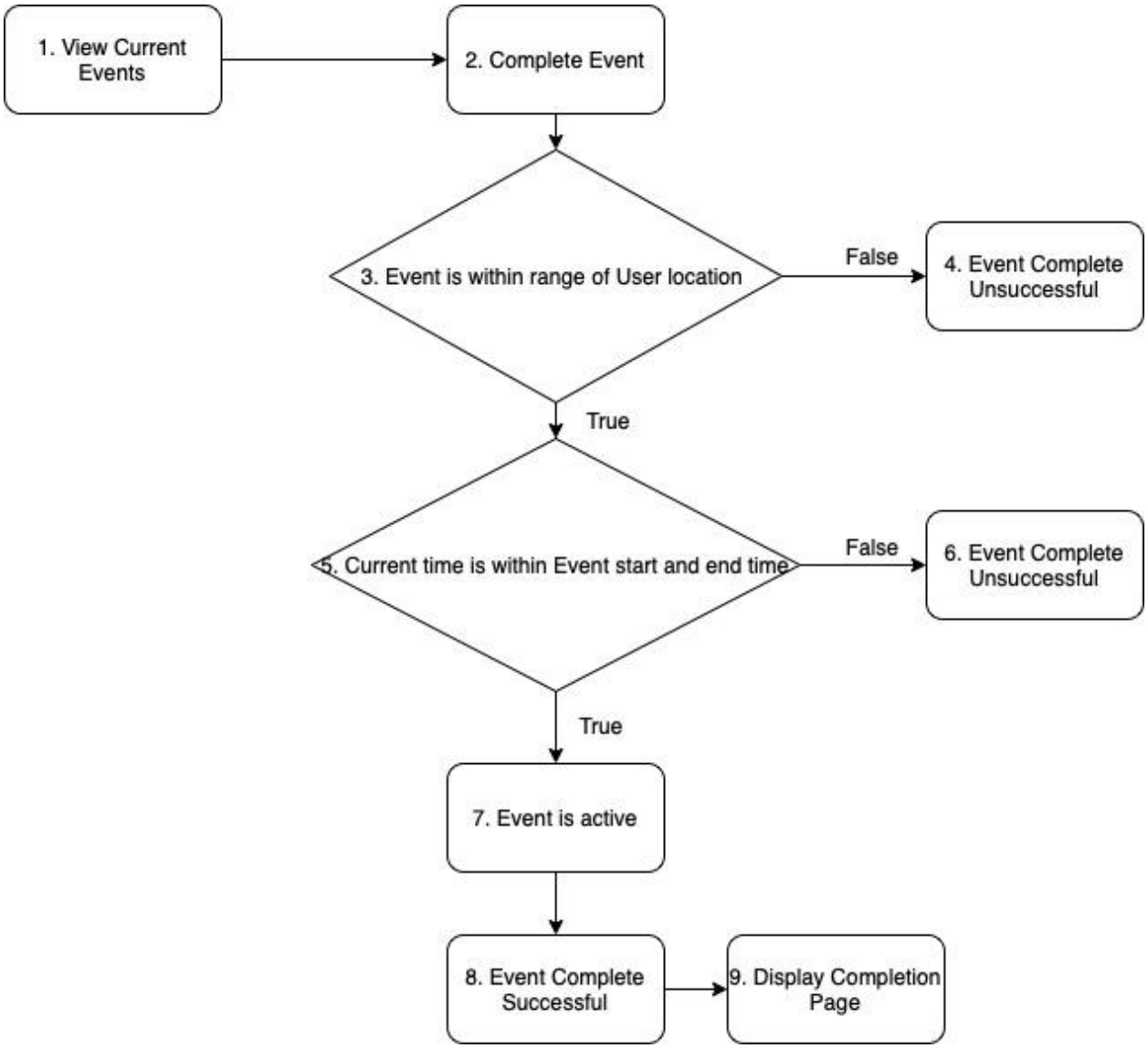
Test cases:

Inputs	Expected results	Actual results
“Soccer” 6 10:42 11:43 “Ball Game”	Event Creation Successful	Event Creation Successful
“Soccer” 35 10:42 10:43 “Ball Game”	Event Creation Unsuccessful	Event Creation Unsuccessful
“Soccer”	Event Creation Unsuccessful	Event Creation Unsuccessful

10 10:45 10:43 “Ball Game”		

11.2 White Box Testing

11.2.1 Control Flow Test: Complete Event



11.2.1.1 Basis Paths:

{1,2,3,5,7,8,9}
{1,2,3,4}
{1,2,3,5,6}

Test Cases:
Basis path (a):
This would be the path if the user successfully completes the active event and is brought to the completion page. This happens when the user is within proximity of the selected joined event and the current time is within the start and end time of the event(enables the complete button).

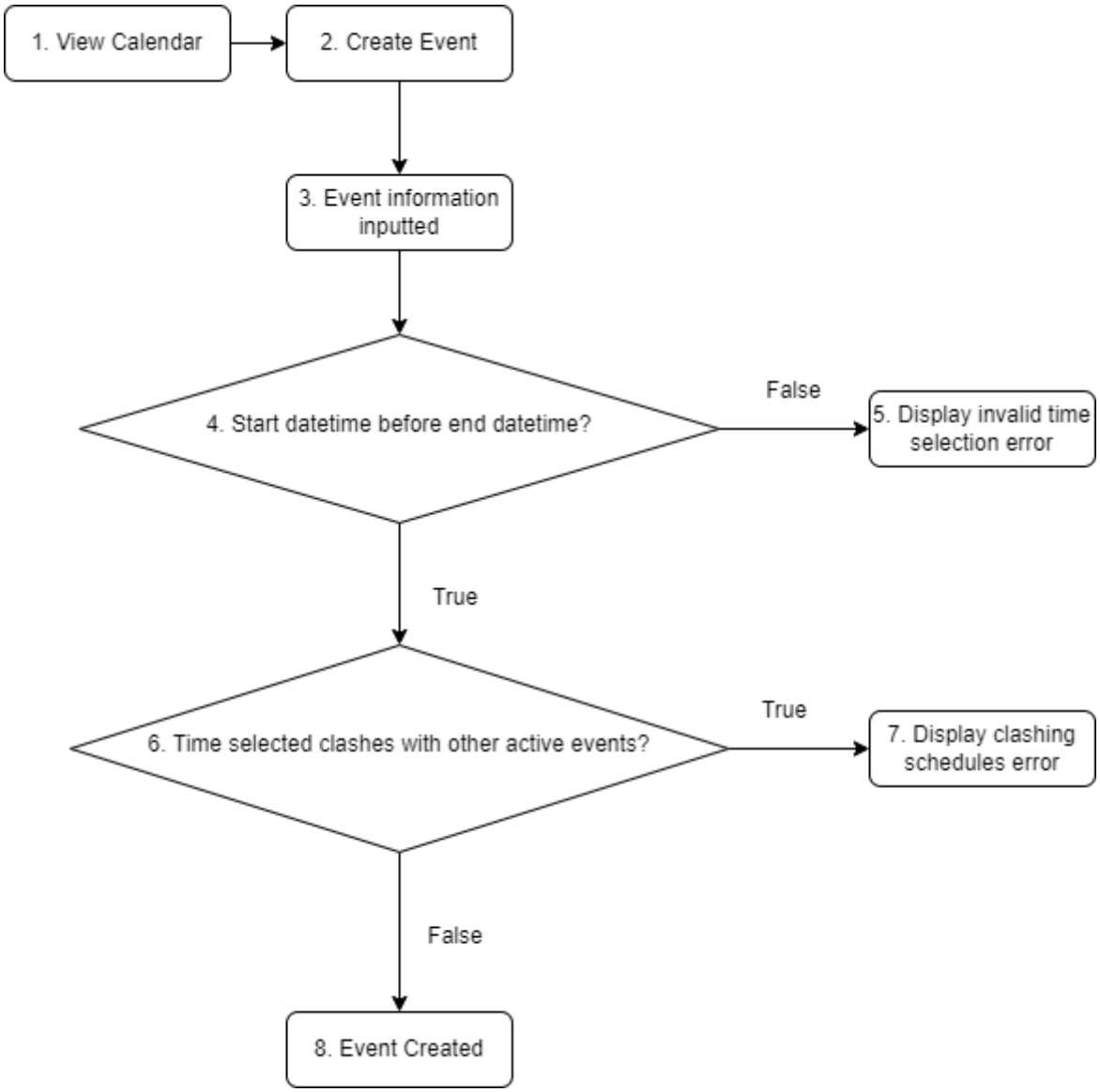
Basis path (b):
This scenario occurs when the user is not within proximity of the selected joined event.

Basis path (c):
This scenario occurs when the current time is after the end time or before the start time of the selected joined event.

11.2.1.2 Cyclomatic Complexity:

Since all decision points are binary, CC =|2| + 1 = 3

11.2.2 Control Flow Test: Create Event



11.2.2.1 Basis paths:

a) {1,2,3,4,6,8}
b) {1,2,3,4,5}
c) {1,2,3,4,6,7}

Basis path (a):
This scenario occurs as the main flow of a successful event creation. This happens when the user selects a valid time frame where end date-time is after start date-time, and does not have any existing Active Events which have clashing time frames with the event to be created.

Basis path (b):
This scenario occurs when the user attempts to create an event, and selects an end date-time that is later than the start date-time.

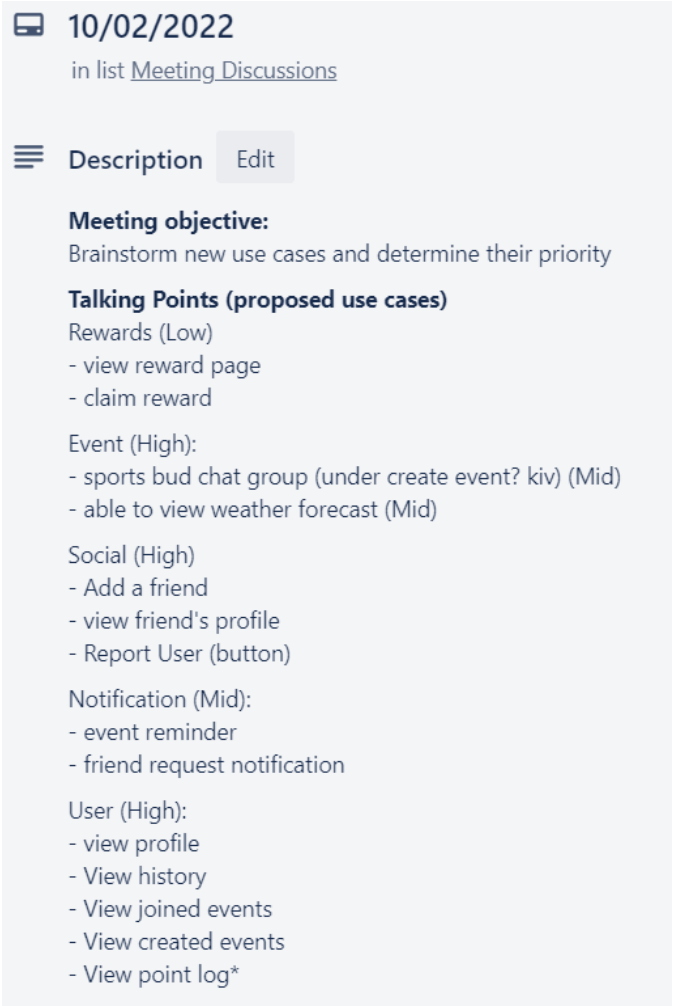
Basis path (c):
This scenario occurs when the user attempts to create an event with a valid time selection where end date-time is after start date-time, but has other existing Active Events with time frames that clash with the time frame of the event to be created.

11.2.2.2 Cyclomatic complexity:

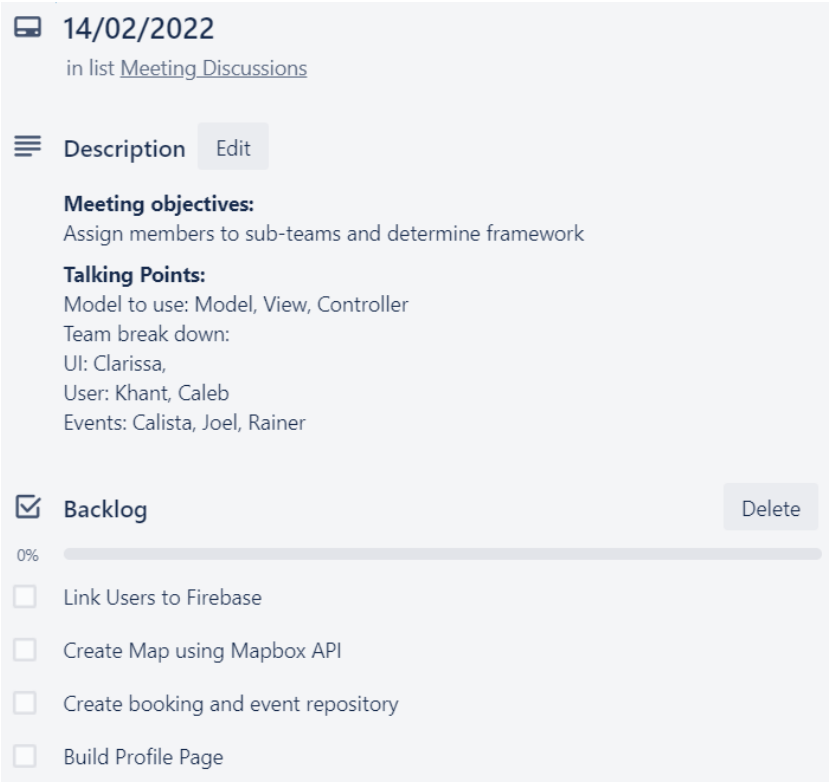
Since all decision points are binary, CC =|2| + 1 = 3

12. Meeting Minutes

The following section contains examples of minutes of our weekly meetings. The minutes are written on Trello, a dashboarding web application for various project meetings.



Minutes of meeting conducted on 10th of February 2022



Minutes of meeting conducted on 14th February 2022

13. Appendix

13.1.1 Data Dictionary

Term	Definition
SportsFacility	Refers to a place in which a SportEvent is held. A SportsFacility may hold more than one SportEvent. No two SportEvents can occur at the same time in one SportsFacility. There are 8 types of SportsFacilities : Basketball Court, Playground, Park, Gym, Stadium, Swimming Complex, Field, Tennis Court.
SportEvent	A type of sports activity occurring at a certain Sports Facility at a specific date and time. There are 5 types of SportEvents : Gym, Ball Games, Aerobic Exercise, Racket Games and Water Sports. The number of participating users cannot exceed the maximum capacity indicated by the user who created the SportEvent. When a user creates a SportEvent, they will automatically be added to the list of participating users. A SportEvent will cease to exist if there are 0 participating users. A SportEvent is active if it has yet to take place. After which, it will expire and users are not allowed to interact with the SportEvent. A user may join an active SportEvent if: 1. The user does not have any existing Active Events which have overlapping timings with the event they are trying to join. 2. The maximum capacity of the SportEvent has not been reached. A user may leave an active SportEvent that they have joined. A user may create an active SportEvent if the user does not have any existing Active Events which have overlapping timings with the event they are trying to create. The user automatically joins the event created. A user may complete an active SportEvent they have joined if the following conditions are satisfied: 1. The user is within 100 metres of the SportsFacility where the SportEvent is taking place. 2. The SportEvent has begun. 3. The user has not already completed this SportEvent. 4. The SportEvent has not ended. A user may complete a SportEvent that they have joined at any point within the event time frame. Upon completion of the SportEvent, the

	user is awarded with points.
Active Events	SportEvents that a user has joined where the current time has not passed the end time of the event.
Completed Events	SportEvents that a user has already completed.
Review	A document containing user inputted information describing their experience at a selected SportsFacility. Information includes a title, description, and rating. Ratings are between 1 star and 5 stars inclusive. Information may also include a photo uploaded by the user. The Reviews for each SportsFacility are displayed in that SportsFacility's Review Page.
About	A short description that the User can write to describe themselves in their profile page.
Friend	Another user that has accepted the current user's friend request invitation or vice versa. Friends are allowed to send invites to join specific SportEvents to one another.
FacilitesMap	A scrollable, searchable and zoomable map that contains Location Markers for all SportsFacilities.
Location Marker	An icon on the FacilitiesMap that indicates the location of SportsFacilites.
Event Calendar	A calendar that shows all the existing SportEvents in the Current Week.
Current Week	Current week consists of the current day and the next 7 days.
Points	Reward points that can be accumulated by completing SportEvents. The amount of Points earned depends on the duration and type of SportEvent attended. Points can be exchanged for rewards via the Rewards Page.
Profile	The Users' individual profile page. This page displays information about users such as biography, username, user email, number of friends, friend request, current points and past events. Users can customise their username, biography and profile picture.
Recommendation	An upcoming event with a certain label suggested to a User upon that user's completion of an event with the same label.
Label	An integer that is assigned to each event upon creation and used as a similarity measure. The labels of each event are determined by a k-Means clustering algorithm.

13.2 References

1. *Unhealthy covid-19 lifestyle habits you need to kick: See all health articles*. Raffles Health. (n.d.). Retrieved April 17, 2022, from https://www.rafflesheath.com/health-articles/unhealthy-covid-19-lifestyle-habits-you-need-to-kick/