Deliverable 5, Final Report

University of New South Wales
Requirements & Design Workshop (SENG2021)
by Team BugFreeSince93

2020, Term 1

Andrew Han	**z5204829**
Harikrishna Manogaran	**z5124767**
Niraj Sapkota	**z5259070**
Omar Lotfi	**z5265983**
Timothy Thacker	**z5115699**

1.0 Problem Statements

1.1 The Problem Contextualized

The problem we aim to address is the difficulty of socialising and making friends at University. The problem mostly is observed in students that are transitioning from high school into university. However, this problem is not limited to high school students as international students also face difficulty making friends. They have often left their families and old lives behind to attempt to build a better life but face difficulty adjusting and coping with the new environment that confronts them. International students make up ~38% of the student demographic in UNSW (UNSW Annual Report 2018)[1] so it is a considerable problem. Furthermore, for local students, it is comparatively easier to make friends in high school as students have shared classes and break schedules. This in conjunction with the fact that they are not taught advanced social skills means they may face difficulty in university. This is because in university, making friends requires significantly more effort as everyone works to a different schedule. Thus, as we can extrapolate, this problem is affecting the entire student body of university - both local and international students face difficulty making friends either due to a lack of social ability or being put in an environment drastically different than what they were accustomed to.

1.2 Analysis of Existing Systems

From our research and as far as we are aware, there are no available systems that target the specific issue we have contextualized. Although there are no exact systems, there are social platforms that share traits with what we are trying to achieve. These platforms include Facebook, Tinder, Linkup, Meetup and Timeweave. However, the fundamental target market and aims of each of these platforms are different from the system we propose to implement and consequently, they lack features that are critical to increasing the accessibility of making friends.

Although Facebook has effective mechanisms for tracking events and messaging users, it lacks features that are important to making friends. An example of this, is the inability to find a person or group who is interested in attending a certain event. Ideally, they would also share some interests and hobbies with you. Furthermore, the large scale of facebook makes it hard to browse events easily.

[1] UNSW Annual Report 2018 https://www.unsw.edu.au/sites/default/files/uploads/UNSW%20Annual%20Report%202018%20170120.pdf

Similarly, Tinder has useful features that allow finding, filtering and matching with people that hold similar interests. However, due to the nature of it being a dating platform, it has a totally different target market and thus lacks features that making friends would require. One such feature that Tinder lacks is the option to match in groups and track social events to attend.

Linkup aggregates and categorizes social events taken from Facebook, and displays the information concisely. However, this is the extent of the application's features - it is more of a website that provides data and does not have any other features beside the aggregation and categorisation of data.

Additionally, Meetup is a system that carries the same fundamental essence of making friends doing what you love - however, these groups are very large with thousands of members attending each event, we wish to have a more smaller and personalised experience to create long-lasting friends you can meet day to day at university.

Timeweave is a good system for managing both your own and your friends timetables to know when you are able to meetup. However, the user experience of the application is very unrefined with constant glitches and bugs. Furthermore, we wish to expand further beyond simply being able to see a friend's timetable. This app is only useful for viewing current friends timetables and there is no way to message people to see if they want to meet-up through the app, nor is there a feature that allows you to make new friends.

1.3 Consequences of Leaving the Solution Unsolved

If this problem is left unsolved, many students will continue through University struggling to refine their interpersonal and social skills which are critical for building networks and entering the industry. The consequences extend beyond University and into the workforce as effective communication and team skills are a vital component of success.

In support of this, the academic paper Friendship supported learning - the role of friendships in first-year students' university experiences[2] highlights that friendships are vital to health and wellbeing. It also expresses that good interpersonal, social and communication skills are a major component in achieving academic and career success, and, as a result of increased sense-of-self, increase the prospect of a student completing their degree.

[2] by Catherine Picton, Ella R. Kahu, and Karen Nelson accessible at https://pdfs.semanticscholar.org/c633/84937ea58210b4e33655566f040bebe442e8.pdf

1.4 The Purpose and Uniqueness of our System

We want to create an application that leverages the strengths of the platforms we analysed (Facebook, Tinder, LinkUp, Meetup, Timeweave) to increase the accessibility and ease of making friends at University. We wish to do so using society events as a medium to match people with similar interests so they are able to find someone who they can relate to. This will ensure that they do not feel overwhelmed, which might be the case if they go alone. Having a friend will also provide a greater sense of social confidence which will allow them to socialise and fit in easier. Thus, our proposed system is an integration of the above existing systems - we plan to take the strengths of each platform and utilize them to solve our specific problem by providing a platform where users are able to find new friends easily.

1.5 Summary Statements

To reiterate the problems that must be addressed and provide a clear guide towards the features we should implement, summary statements are listed below:

1. Both domestic and international students struggle to make friends in university.
2. Students find it difficult to approach and talk to new people without a stimulus.
3. It is hard to find people to go to society events with, and it can be intimidating going by oneself.
4. There is no highly integrated system that allows you to make friends at university.
5. These systems should be smart and find people that hold similar interests and are in the same area of study.
6. If this problem of social ineptitude is not addressed it extends beyond university and into adult life.
7. Friendships are vital to health, wellbeing and especially during university which can be one of the most stressful times of a person's life.
8. Without help, students continue to struggle to refine their interpersonal and social skills. Skills, which are critical for succeeding in industry.

2.0 Features

2.1 Account

- Register for an account with an email and password
- Verify sign-up through verification email
- After verification, users select their age, birthday, faculty and classes they are taking for that year
- After verification, users can create an avatar for their profile
- After verification, users select their interests from a list of interests
- Login with a registered email and corresponding password

2.2 Profile

- View profile details by clicking on the dropdown below the avatar icon
- Edit personal details while viewing your profile
- Change interests while viewing your profile
- Delete account while viewing your profile

2.3 Group Chat

- View a list of all group chats by clicking on the ‘Groups’ tab
- Each group chat contains the first names of the other users
- Each group chat contains the event that everyone is going to
- Each group chat contains the last message sent, who sent the message and the time it was sent
- Selecting a specific group chat opens up the chat.
- Opening up a chat allows a user to message the other users in the chat and view the chat history
- Text messages can be sent to the users in the group chat

2.4 Matching Users

- Match users going to similar events
- Match users based on similar interests, age, degrees and classes
- 3 users going to similar events are put into a chat

2.5 Events (available)

- Browse from a list of society and uni events by clicking on the ‘Explore’ tab
- Filter events based on tags
- Tags are based on interest type chosen
- Filter events based on a date range
- When viewing an event, users can join the event by pressing on the “+” icon
- When viewing an event, users can leave the event by pressing on the ‘tick’ icon
- Selecting an event brings up the details of the event such as a description of the event and the location of the event

2.6 Events (attending)

- View a list of all events that you are attending by clicking on the ‘Going’ tab
- The list of all events contains past and upcoming events.
- Users can unjoin the event by pressing on the “tick” icon

2.7 Achievements

- View a list of badges and activity graph by clicking on the ‘Achievements’ tab
- Gain badges by reaching milestones
- Badges not achieved yet are ‘locked’
- Activity graph provide information on a user’s activity. Each week provides details about the number of badges a user has earned, as well as the number of groups they have joined.

2.8 Milestones and Stats

- View how many groups you’ve joined
- View how many badges you have earned
- View the score you’ve earned as a progression graph

3.0 User stories, Sequence Diagrams and Screenshots

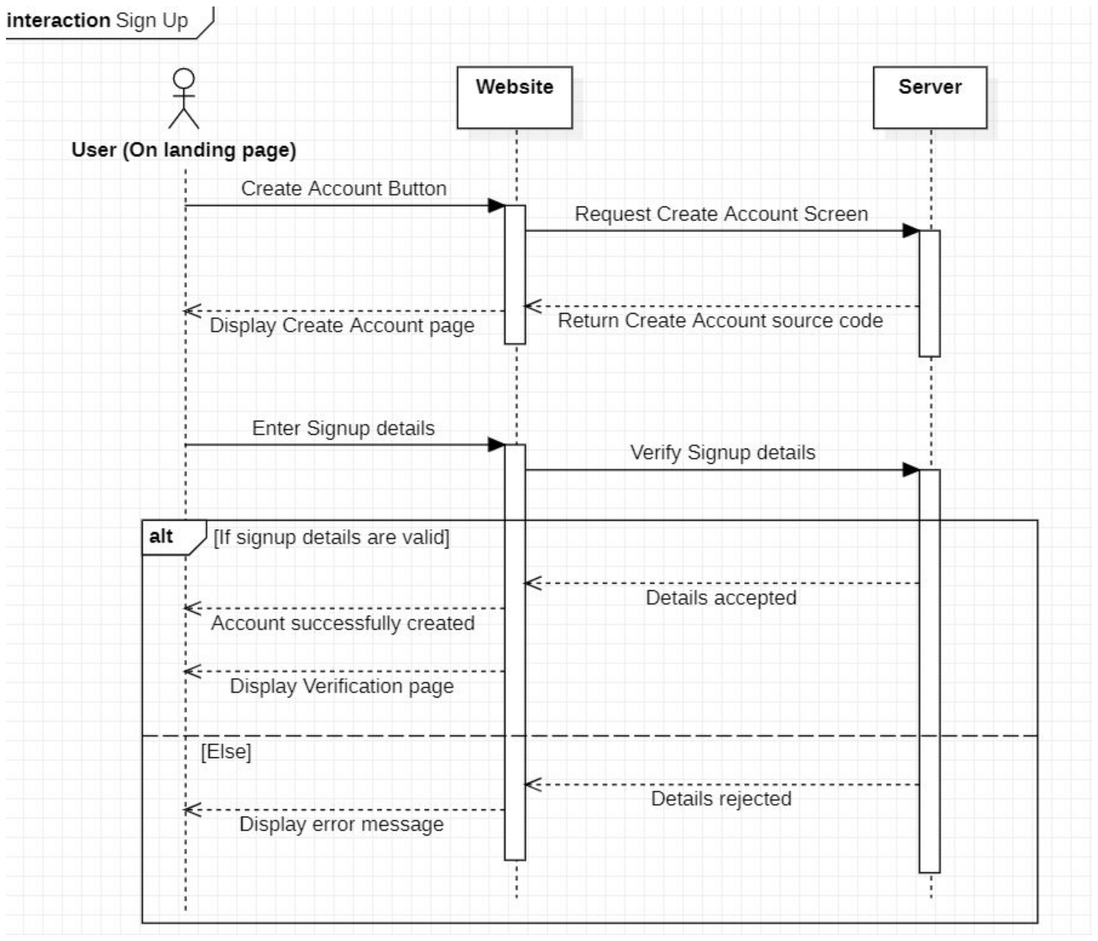
3.1 Sign Up / Sign In / Onboarding

Epic 1: As a student, I want to sign up and log into Holler so that I can start using the app

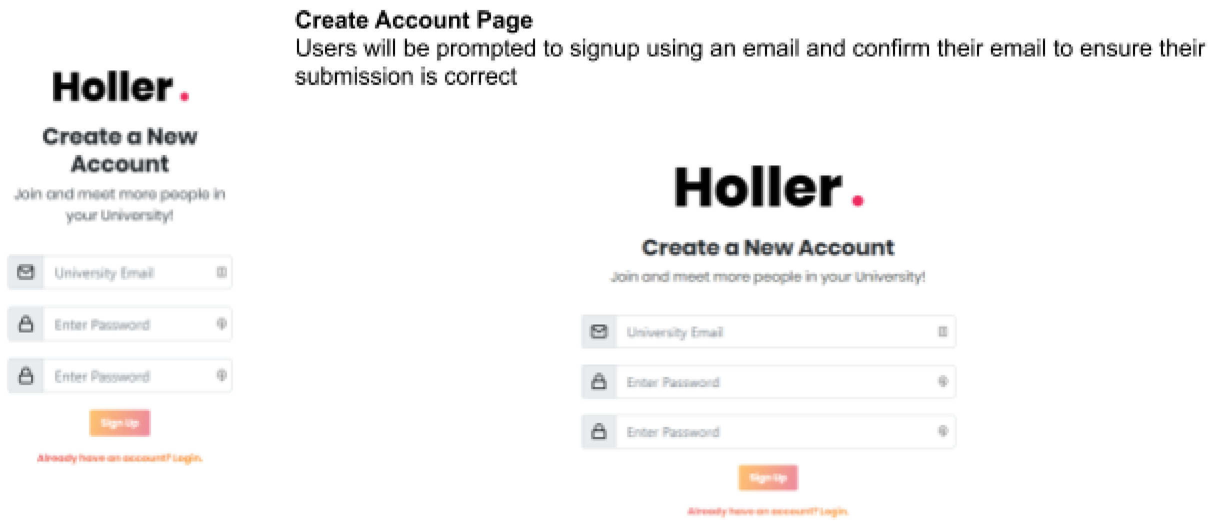
1. As a student, I want to sign up for Holler so that I can start making friends.

- Given a user is in the landing page, a 'Sign Up' button is available
- Given a user is on the sign up page, then a user should be able to input an email, password and confirmation password
- When a user puts in a password and confirmation password, then they must match
- Email must be valid and must not have been used to create a prior account
- When an invalid input is put in, then an error message must be printed
- When all the inputs are filled and are valid, then a confirmation email is sent to the user

Sequence Diagram 1: User Story 1 - Signup



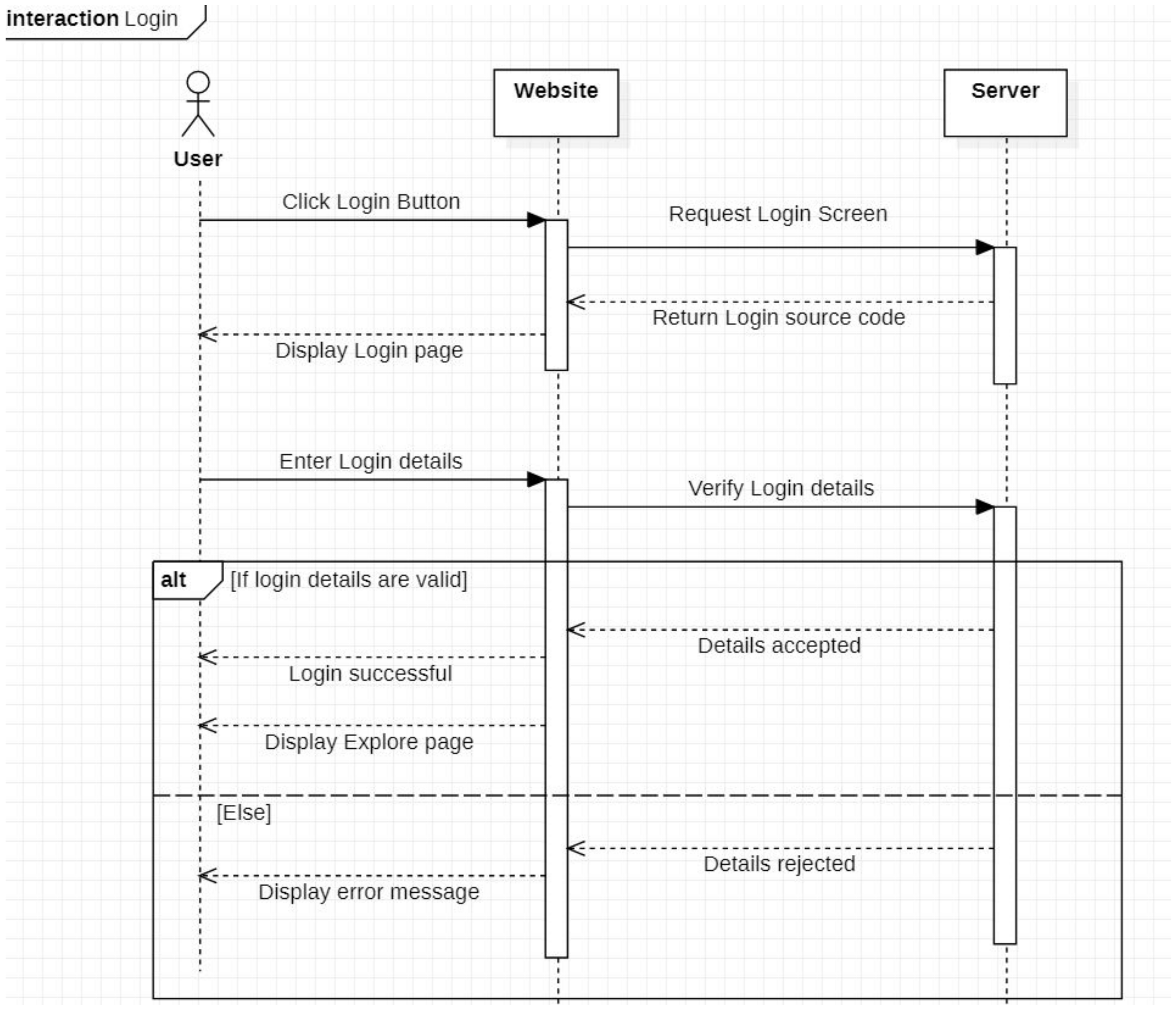
Screenshot 1: Create Account Page


Create Account Page
Users will be prompted to signup using an email and confirm their email to ensure their submission is correct

2. As a student, I want to log into Holler so that I can start using the app.

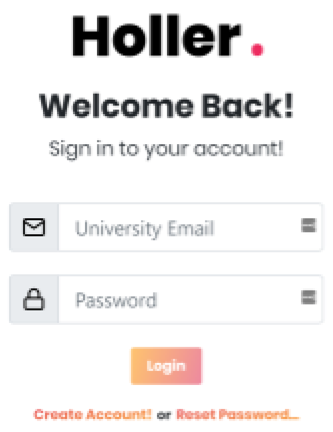
- Given a user is in the landing page, a ‘Login’ button is available
- Given a user is in the login page, then a user can input an email, password and re-enter password
- When a user logs in with an email, the email must be valid and registered
- When a user puts in a password, it should correspond to the email they use to log in with
- When an invalid input is entered, an error message must be printed
- When I login and the signup is not fully completed, then users are returned to onboarding
- When I login and registration is fully completed, then users are sent to the ‘Explore’ tab

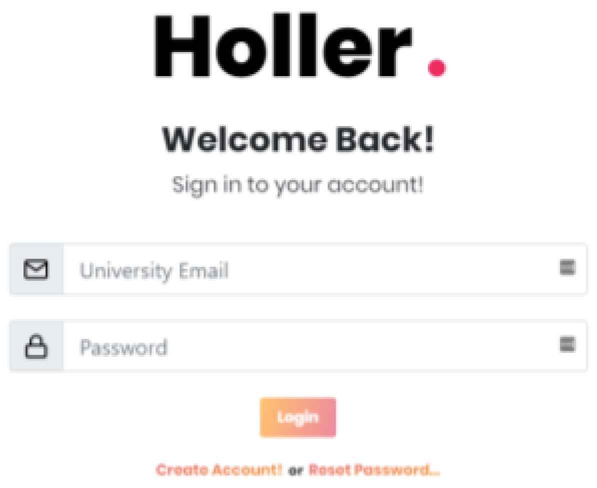
Sequence Diagram 2: User Story 2 - Login



Screenshot 2: Login Page

Login Page
Users must first verify their account before they can login





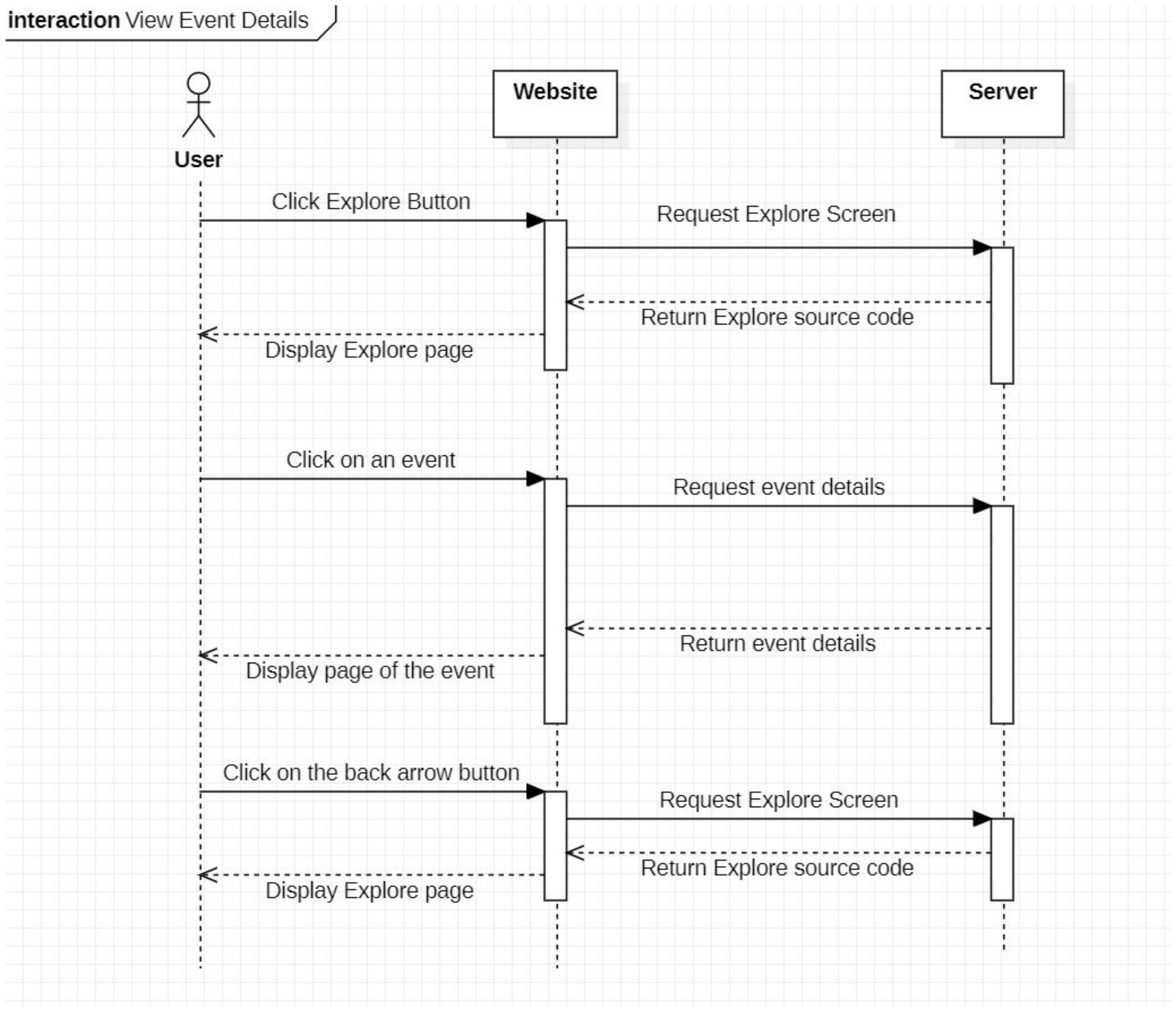
3.2 Events

Epic 2: As a student, I want to search for events so that I can indicate if I am interested in the event or not.

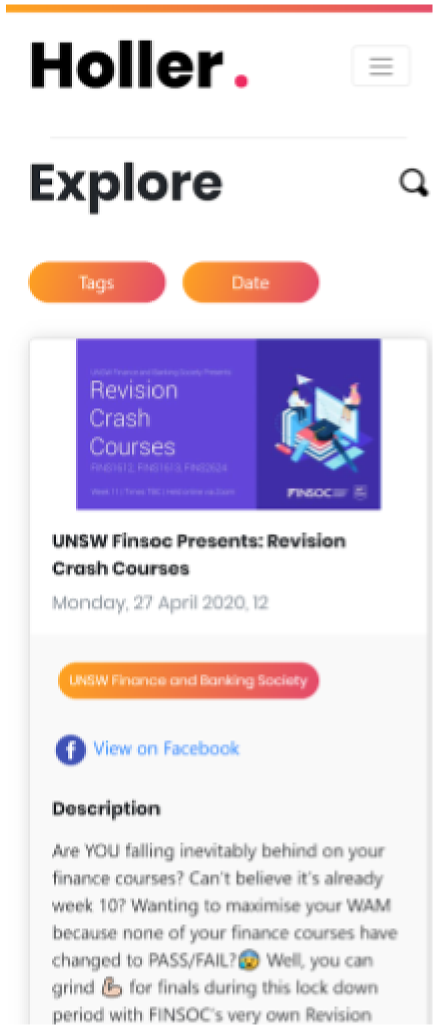
3. As a student, I want to look at an event's details so that I can learn more about the event I am interested in.

- Given a user is anywhere on the dashboard, then an ‘Explore’ tab is available
- Given a user is in the ‘Explore’ tab, then users are automatically presented with society events that match their interests
- Given a user is in the ‘Explore’ tab, then society events are displayed in a list
- Given a user is in the ‘Explore’ tab, users are shown an image, name, date, location and for the event/society.
- Given a user is in the ‘Explore’ tab, then a ‘+’ button is on the bottom right of each event
- Given a user is in the ‘Explore’ tab, when they click on the event, then it brings up a description of the event and all prior details

Sequence Diagram 3: User Story 3 - View Event Details



Screenshot 3: Event Details Page



UNSW Finsoc Presents: Revision Crash Courses

Monday, 27 April 2020, 12

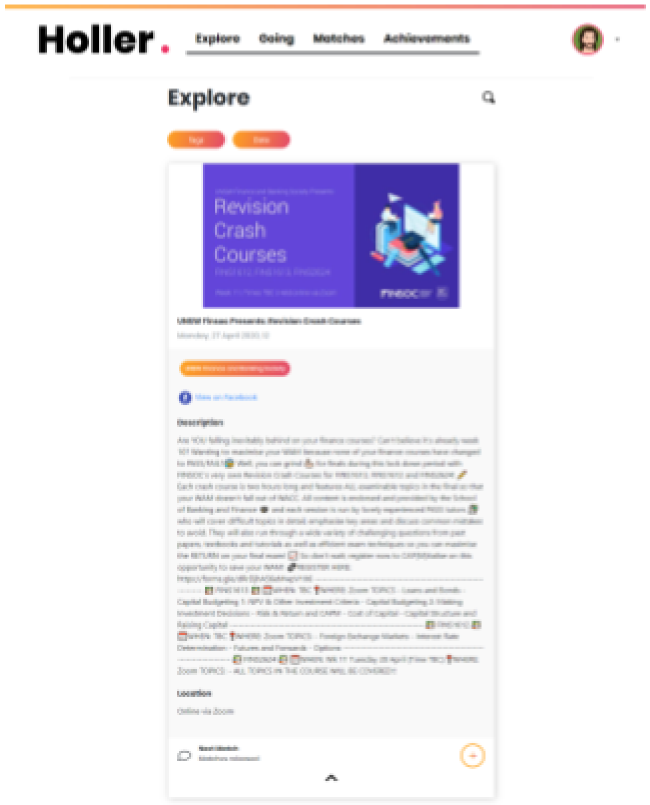
UNSW Finance and Banking Society

View on Facebook

Description

Are YOU falling inevitably behind on your finance courses? Can't believe it's already week 10? Wanting to maximise your WAM because none of your finance courses have changed to PASS/FAIL?😱 Well, you can grind 💪 for finals during this lock down period with FINSOC's very own Revision

Event Details
Users can click on the arrow icon (bottom of right image) to expand and collapse each event. Expanding an event provides more information about that event



UNSW Finsoc Presents: Revision Crash Courses

Monday, 27 April 2020, 12

UNSW Finance and Banking Society

View on Facebook

Description

[illegible]

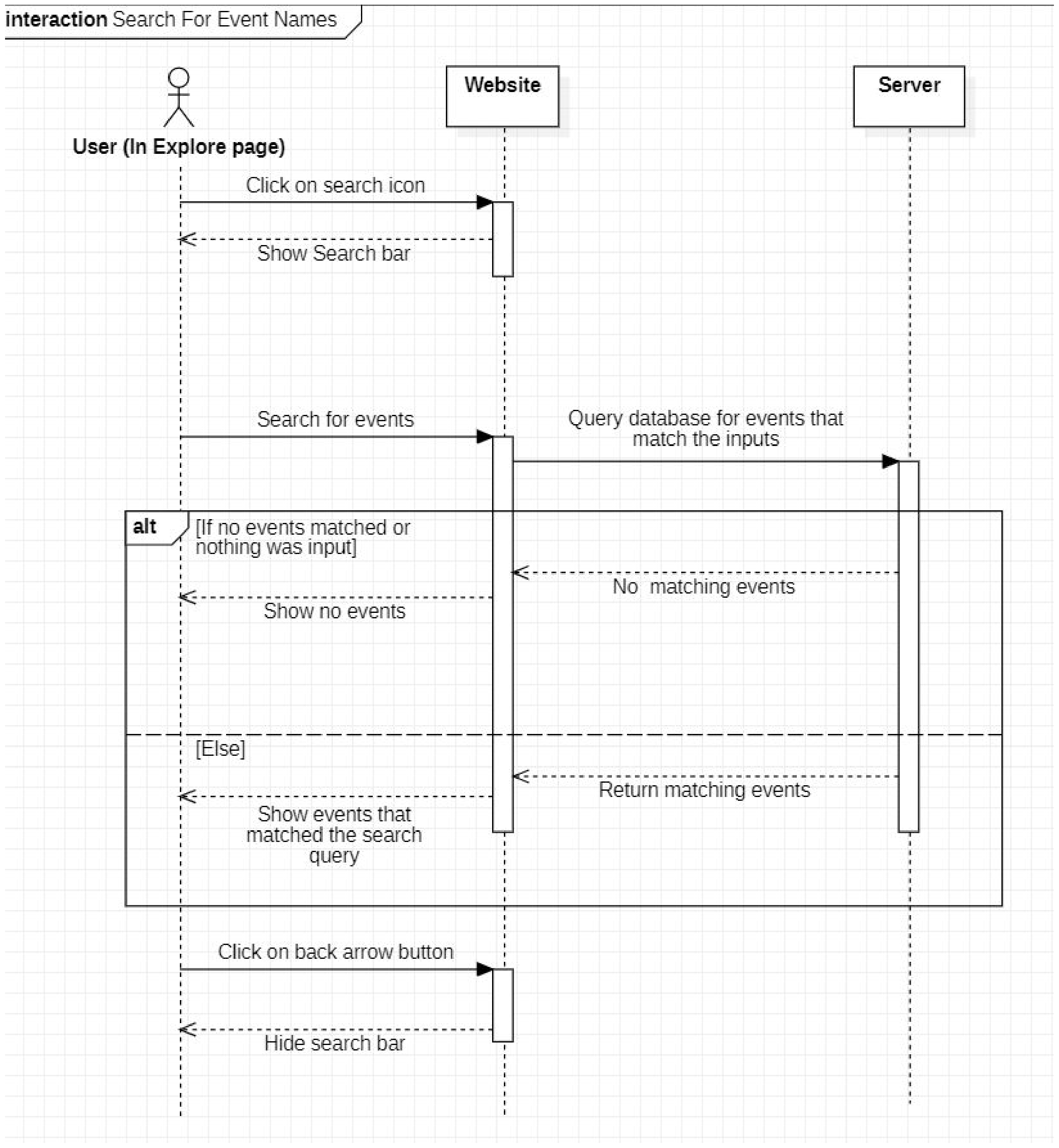
Location

Online via Zoom

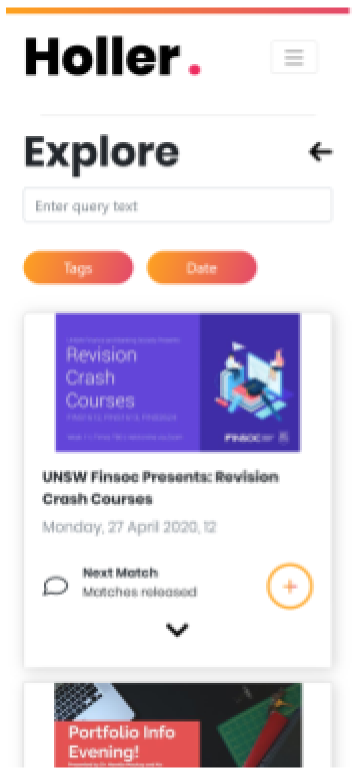
4. As a student, I want to search for events by name so that I can find specific events that I remember being interested in.

- Given a user is in the ‘Explore’ tab, then a searching icon is available near the top of the screen
- Given a user is in the ‘Explore’ tab, when they click on the search icon, then the search bar is displayed and users can type in the search bar for event names.
- Given a user is in the ‘Explore’ tab, when a user is searching for events, then they can click on the back arrow icon to close the search bar.

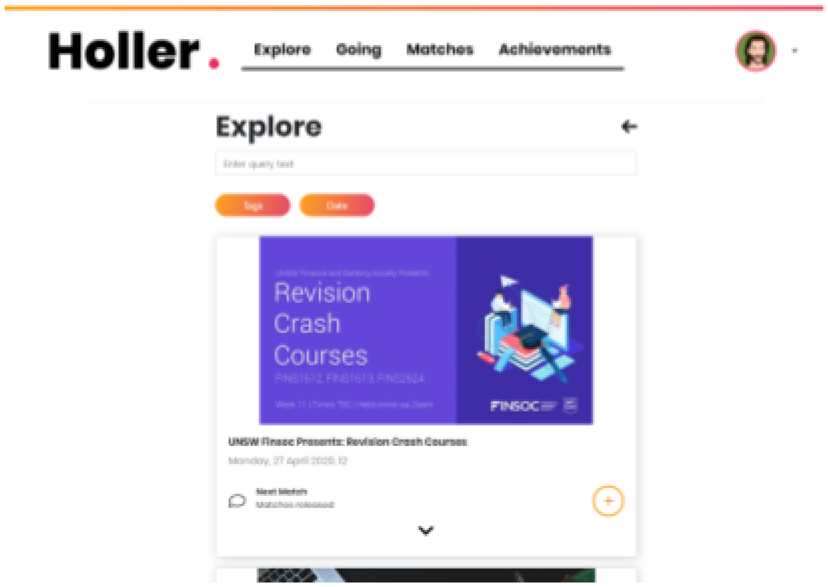
Sequence Diagram 4: User Story 4 - Searching For Event Names


interaction Search For Event Names
User (In Explore page)
Website
Server
Click on search icon
Show Search bar
Search for events
Query database for events that match the inputs
alt
[If no events matched or nothing was input]
No matching events
Show no events
[Else]
Return matching events
Show events that matched the search query
Click on back arrow button
Hide search bar

Screenshot 4: Event Search Feature



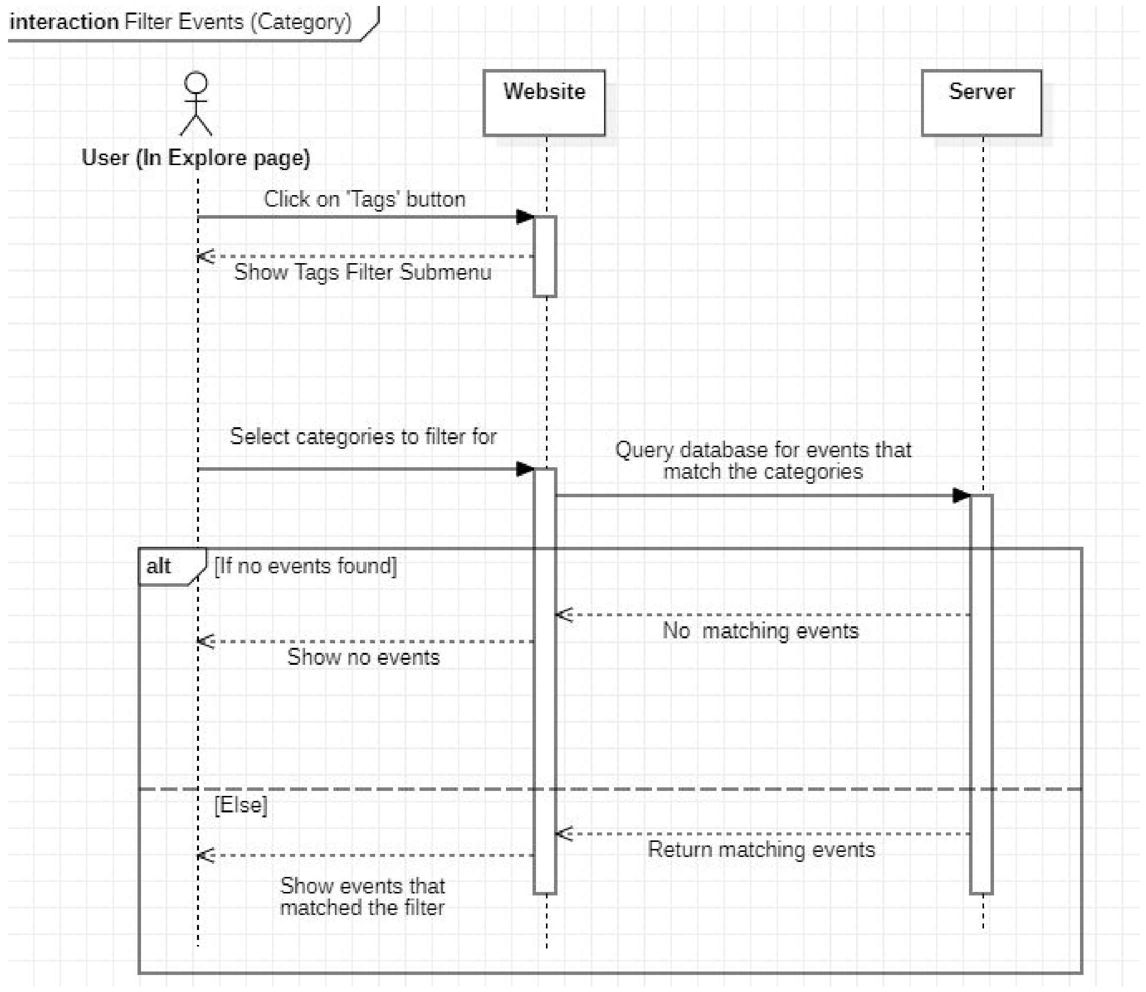
Search
Users can type in text and will search for event names with the query. The search bar can be collapsed by clicking on the arrow above the search bar.



5. As a student, I want to filter events by category so I can find events that match my interests.

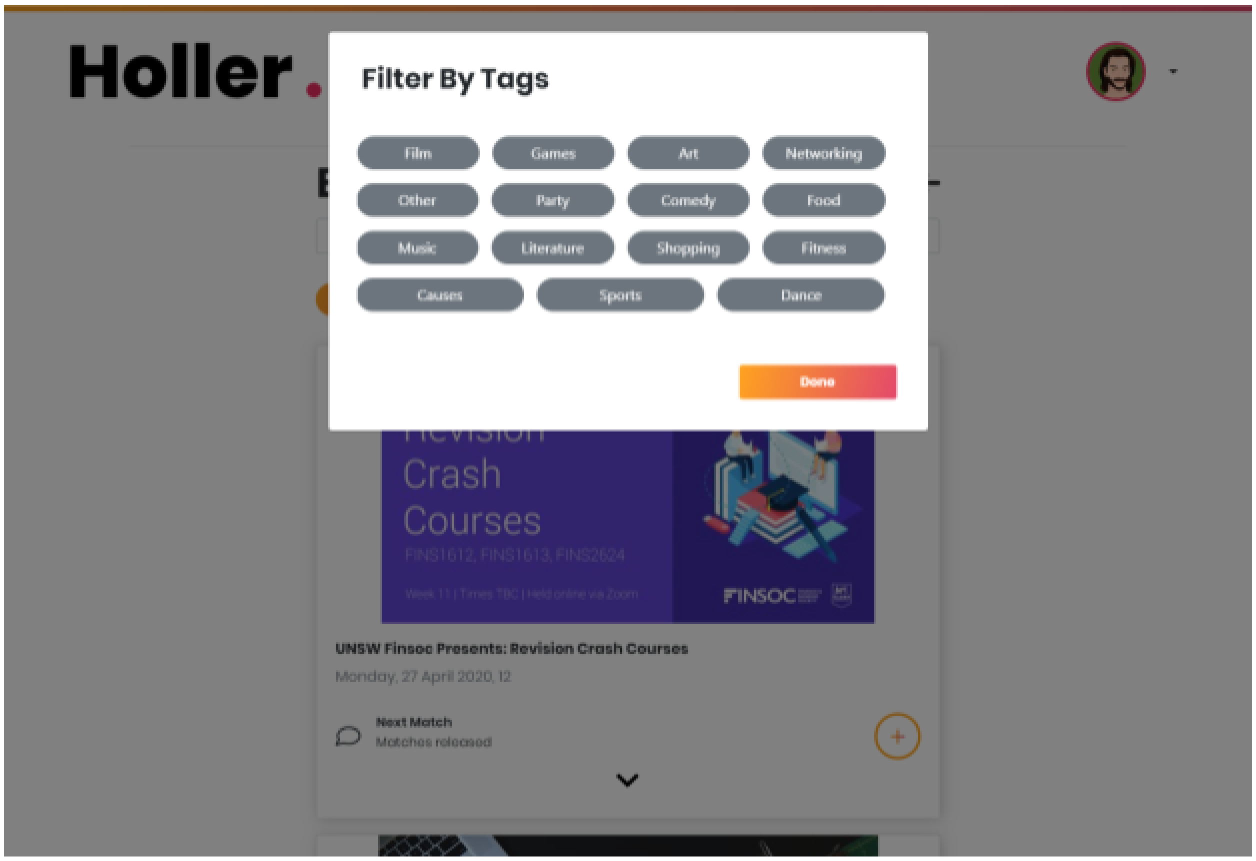
- Given a user is in the ‘Explore’ tab, when users click on ‘tags’ users can filter events by tags
- When a user is filtering by tags, then events can be filtered based on interests (e.g. Language, Music, Dancing, Games)
- Given a user is in the ‘Filter’ menu, when a user selects a category and searches, then events that match those tags will be displayed
- Given a user is in the ‘Filter’ menu, when a user selects no categories and searches, then it will have no effect

Sequence Diagram 5: User Story 5 - Filtering Categories



Screenshot 5: Filtering Event Categories Popup

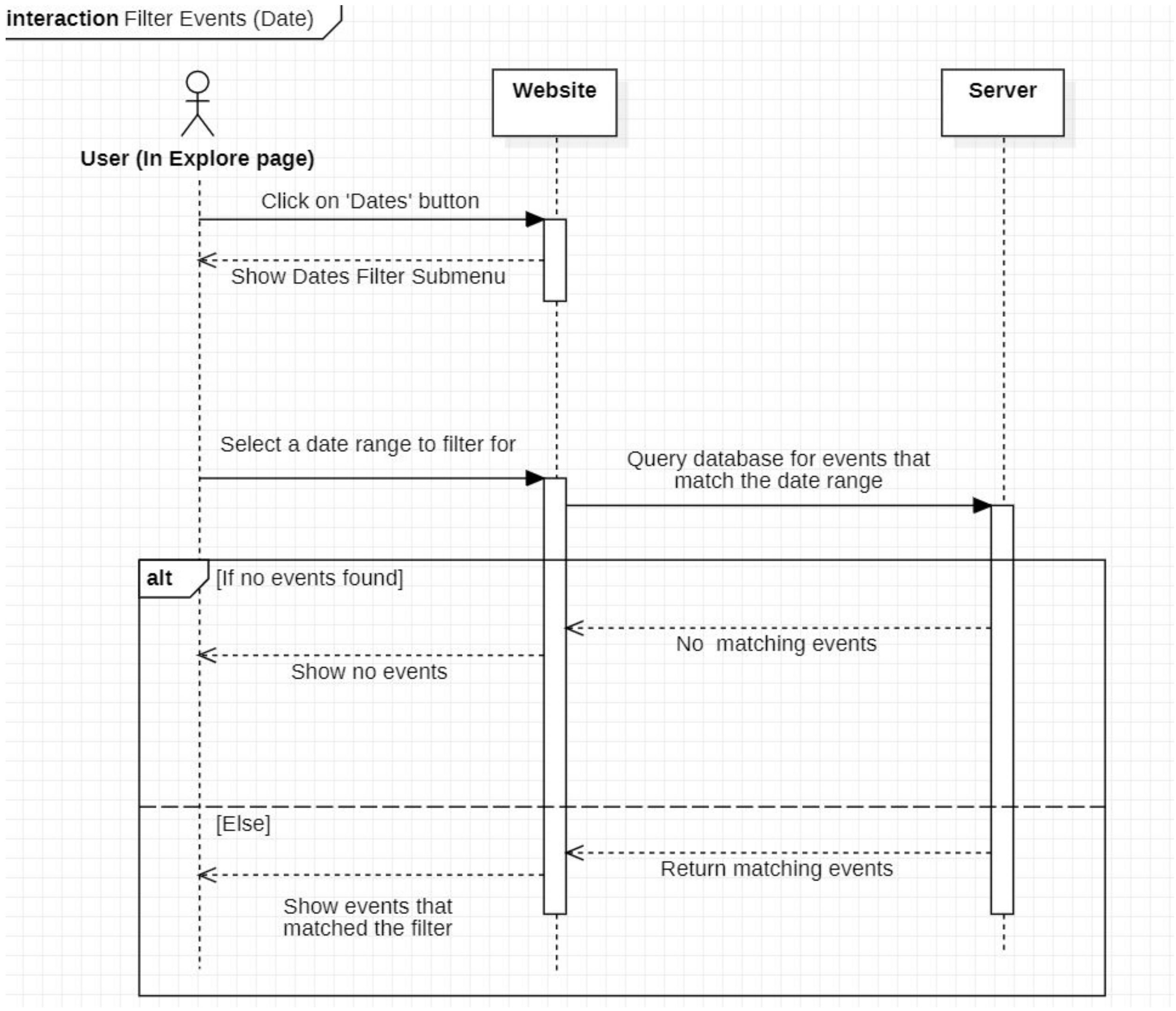
Filtering Events (Tags)
A user can select tags to search for events that match those tags



6. As a student, I want to filter events by time so I can find events that match my schedule.

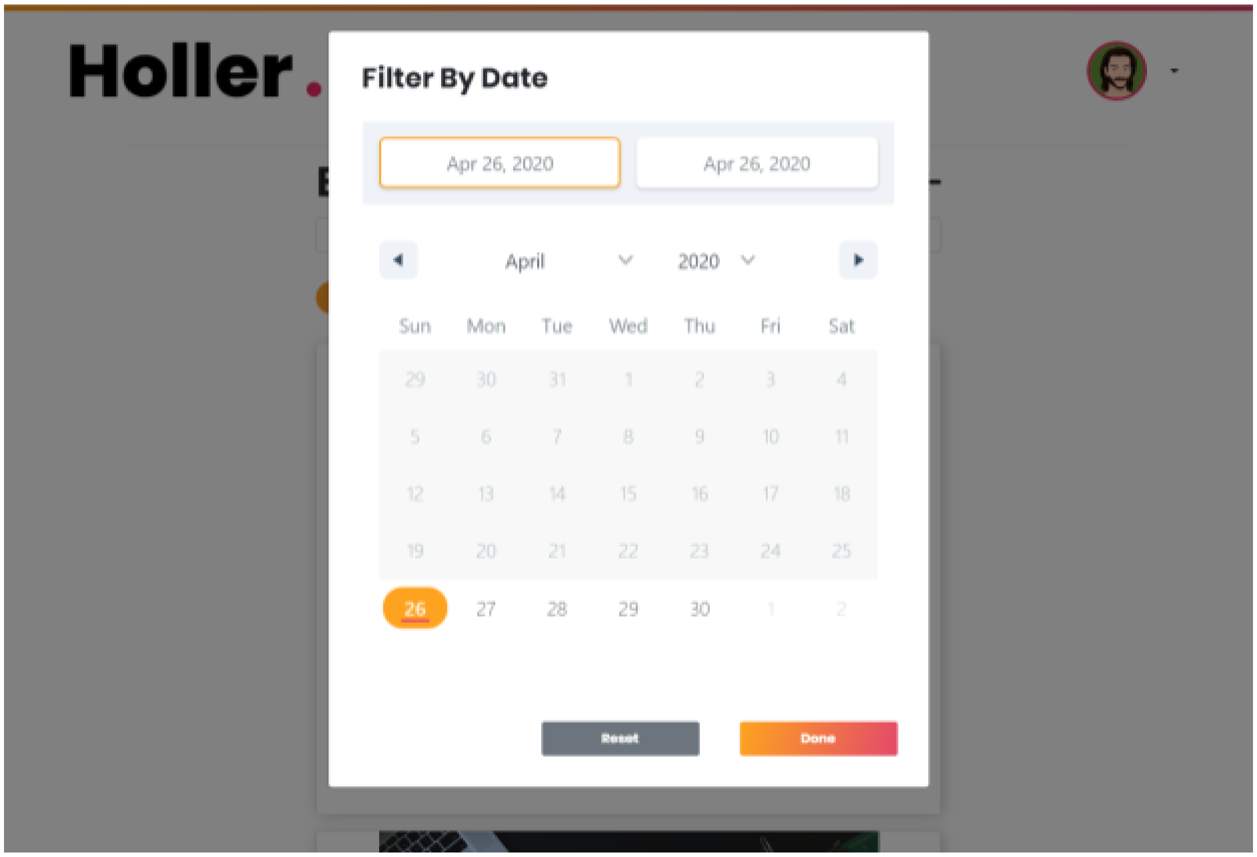
- Given a user is in the ‘Filter’ menu, then a user can select multiple categories at a time
- Given a user is in the ‘Explore’ tab, when users click on ‘date’ users can filter events by a time range
- Given a user is in the ‘date’ menu, when a user selects the start date and end date, then the start date must be less than the end date

Sequence Diagram 6: User Story 6 - Filtering Dates


interaction Filter Events (Date)
User (In Explore page)
Website
Server
Click on 'Dates' button
Show Dates Filter Submenu
Select a date range to filter for
Query database for events that match the date range
alt
[If no events found]
No matching events
Show no events
[Else]
Return matching events
Show events that matched the filter

Screenshot 6: Filtering Event Date Popup

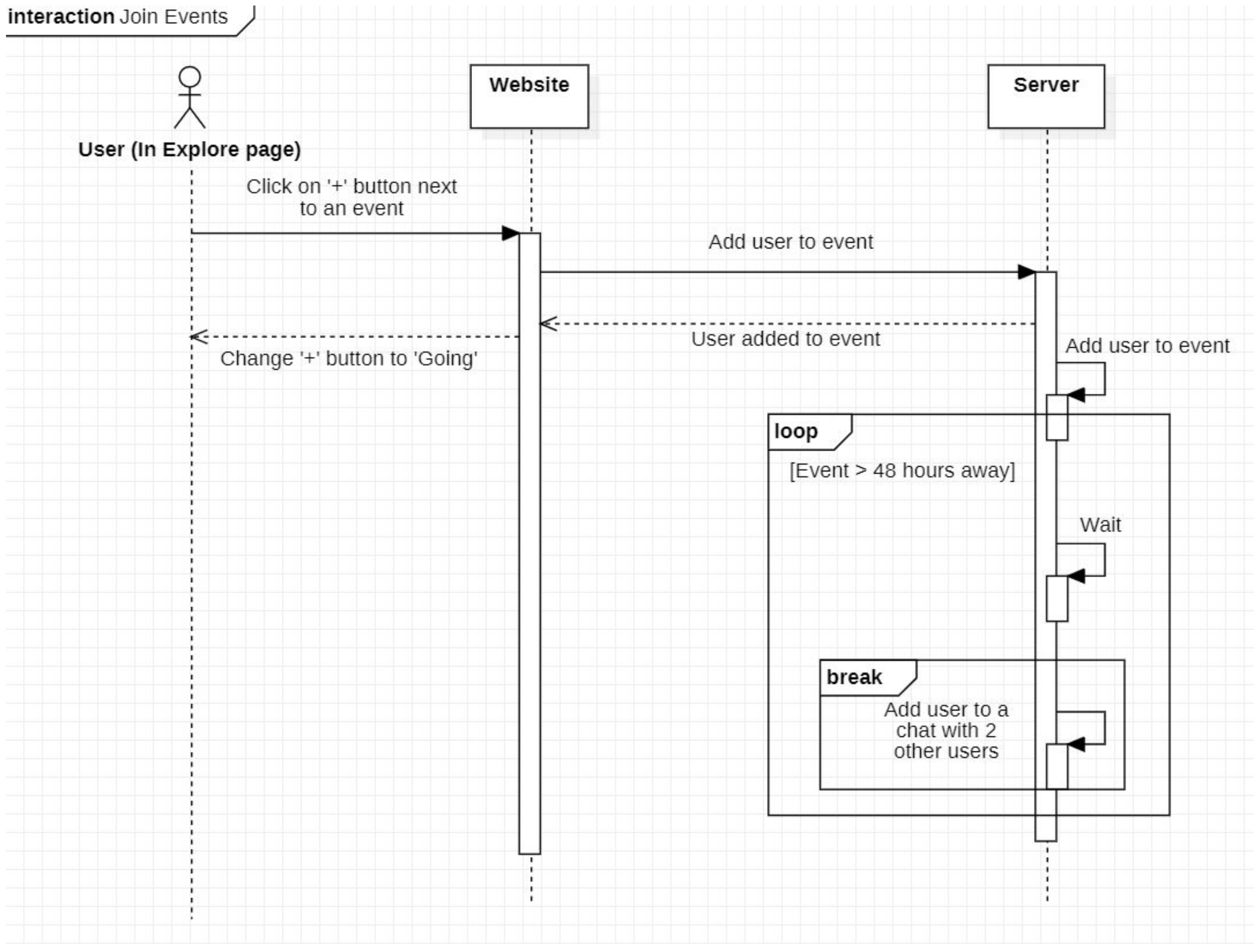
Filtering Events (Date)
A user can filter for events that lie within a specific date range.



7. As a student, I want to join events I am interested in so that I can start talking to people.

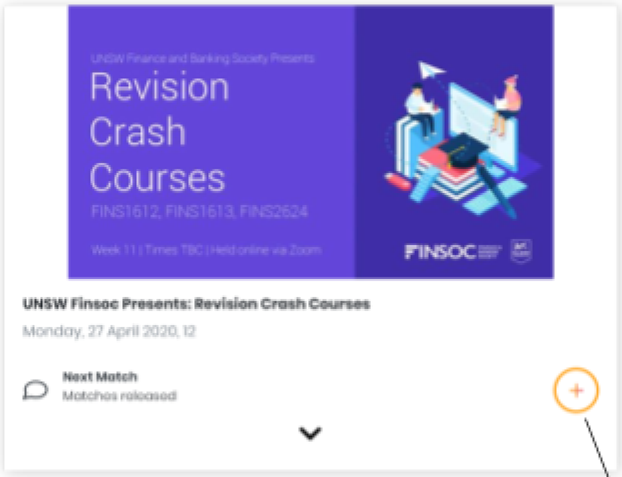
- Given a user is in the ‘Explore’ tab, when a user clicks on the ‘+’ button, then a user will indicate they are interested in that society event.
- Given a user is in the ‘Explore’ tab, when the ‘+’ button is pressed, then it changes to ‘Going’ with a tick.
- When a user is going to a society event, then they are matched with 2 other users who are also going to the same event.

Sequence Diagram 7: User Story 7, 8, 9 - Joining an event and getting matched


interaction Join Events
User (In Explore page)
Website
Server
Click on '+' button next to an event
Add user to event
User added to event
Change '+' button to 'Going'
Add user to event
loop
[Event > 48 hours away]
Wait
break
Add user to a chat with 2 other users

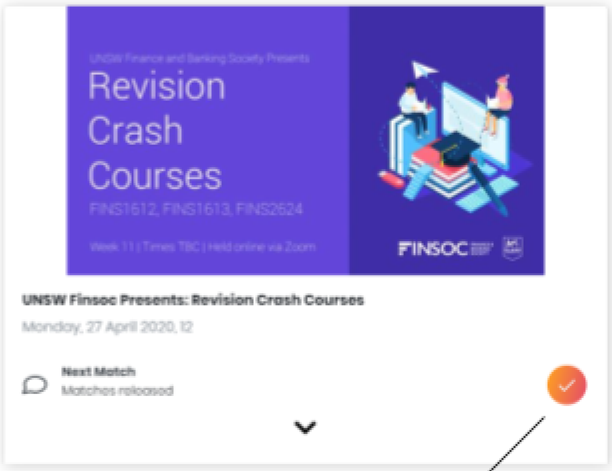
Screenshot 7: Joined vs Uninterested Event Comparison

Joining Events
A user can select on the orange to indicate they are going to that event. They will then be matched with other users.



UNSW Finance and Banking Society Presents
Revision Crash Courses
FINS1612, FINS1613, FINS2624
Week 11 | Times TBC | Held online via Zoom
FINSOC

UNSW Finsoc Presents: Revision Crash Courses
Monday, 27 April 2020, 12
Next Match
Matches released



UNSW Finance and Banking Society Presents
Revision Crash Courses
FINS1612, FINS1613, FINS2624
Week 11 | Times TBC | Held online via Zoom
FINSOC

UNSW Finsoc Presents: Revision Crash Courses
Monday, 27 April 2020, 12
Next Match
Matches released

Click to go to the event

Click to leave the event

3.3 Matches / Chat

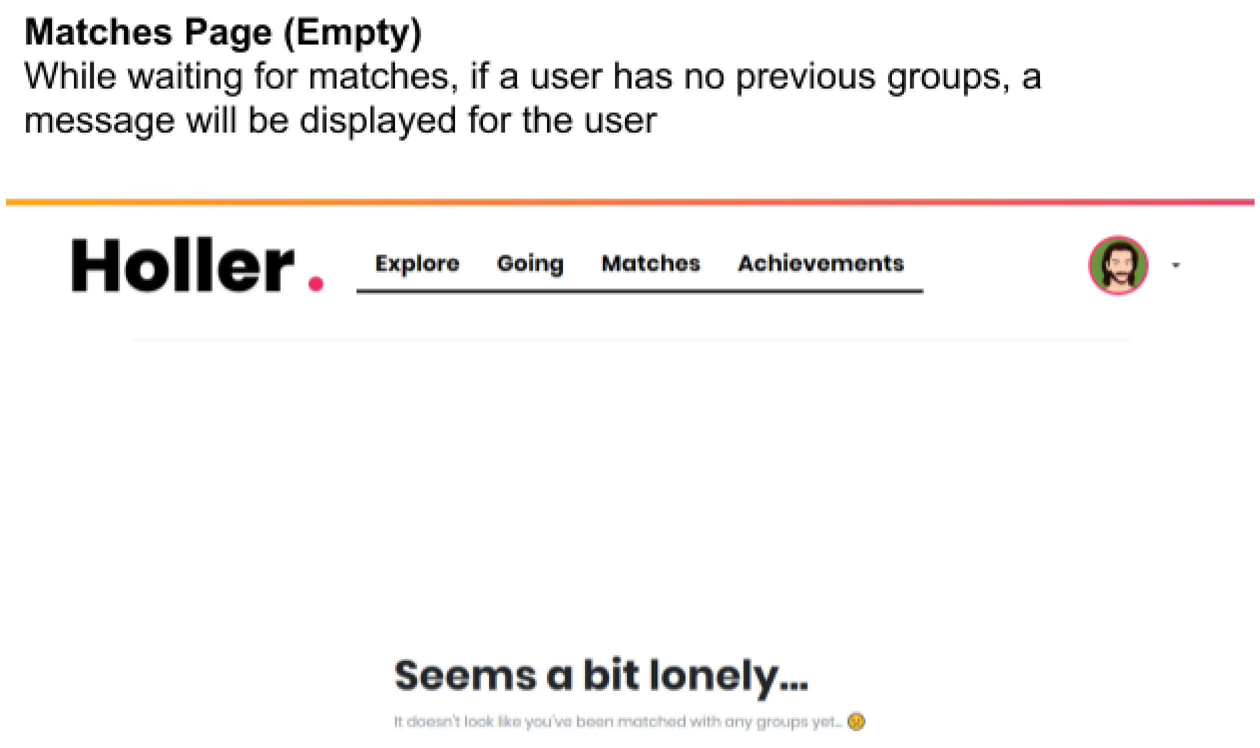
Epic 3: As a student, I want to be matched with users going to similar events and who have similar interests to me, so that I can meet like-minded people.

8. As a student, I want to be matched with users attending similar events so that I can avoid going alone.

- When a user indicates they are going to a society event, then they are matched with 2 other users who are also going to the same event.
- When a user indicates they are going to a society event, then users are matched 24 hours before an event starts
- Given that a user is in the 'Going' page, when they are waiting to be matched, then the event is shown in the matches screen with a note labelled "Matching with other users…"
- Given that a user is in the 'Going' page, when they are waiting to be matched and it is more than 24 hours before the event, then a timer is put next to the event and after it is 24 hours before the event, then they are matched with 2 other users with similar interests.
- When a user is finished being matched, then the matched users are put into a chat

Refer to sequence diagram 7 above.

Screenshot 8: Waiting For Matches



Matches Page (Empty)
While waiting for matches, if a user has no previous groups, a message will be displayed for the user

Seems a bit lonely...

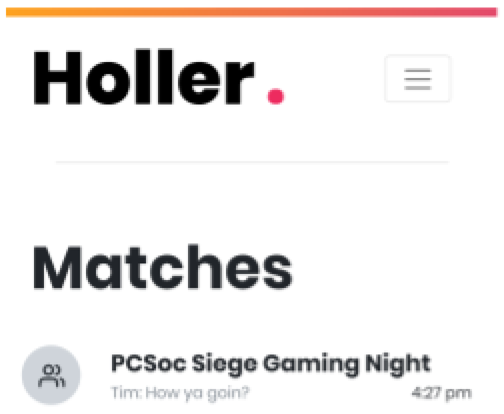
It doesn't look like you've been matched with any groups yet.. 😟

9. As a student, I want to be matched with user with similar interests so that I can meet like-minded people

- When a user is being matched, then users are matched based on factors such as interests, degree and classes
- When a user is being matched, then users are matched with the most similar users
- When a user is being matched, and users have matched before for that event, then they cannot match again.
- When a user joins an event, and the user joins within 24 hours before the event, then they are matched with 2 random people, regardless of matching interests.

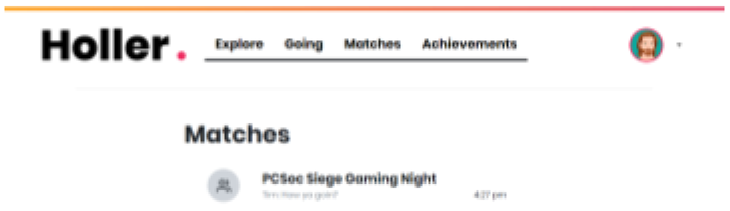
Refer to sequence diagram 7 above.

Screenshot 9: Group Chats Page



PCSoc Siege Gaming Night
Tim: How ya goin? 4:27 pm

Matches Page
Displays all the group chats that were created after matching with two other users



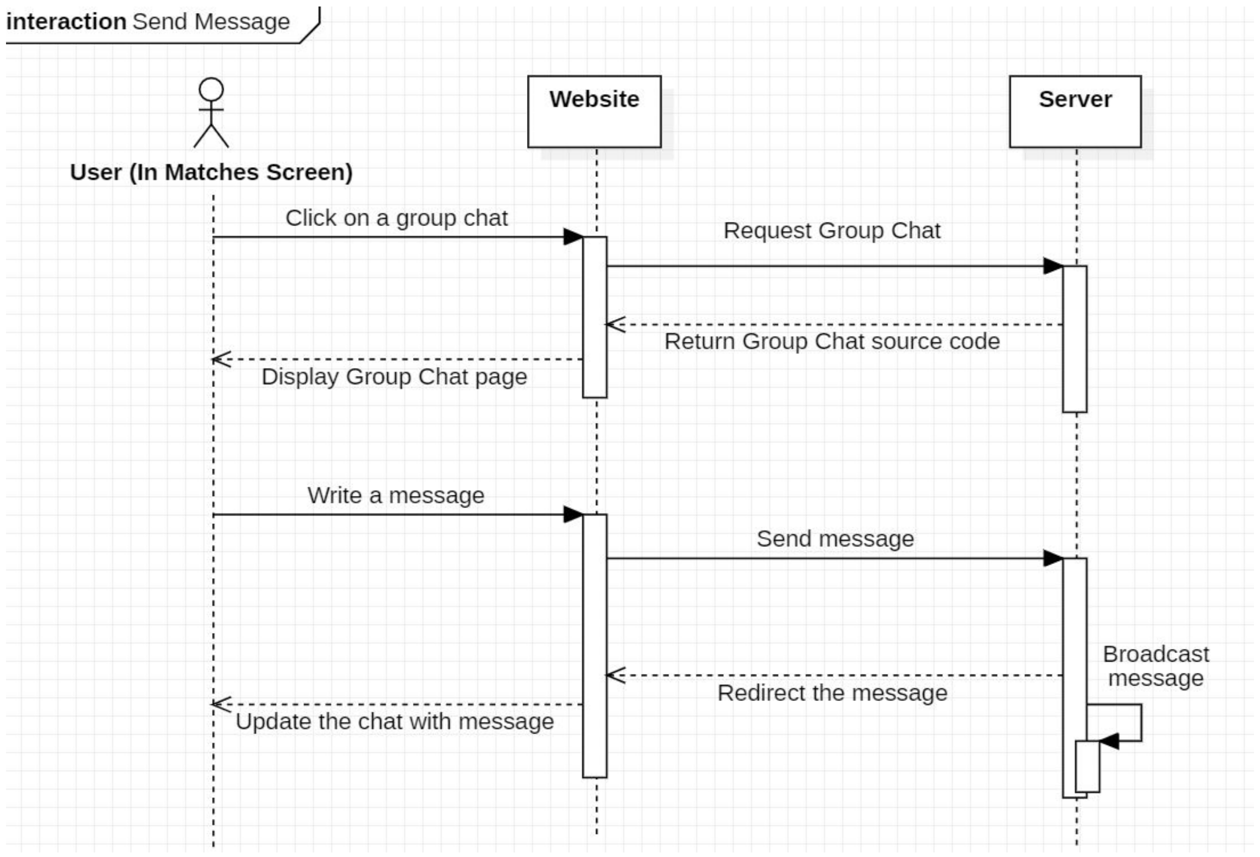
PCSoc Siege Gaming Night
Tim: How ya goin? 4:27 pm

Epic 4: As a student, I want to chat with multiple people so that I can get to know more people.

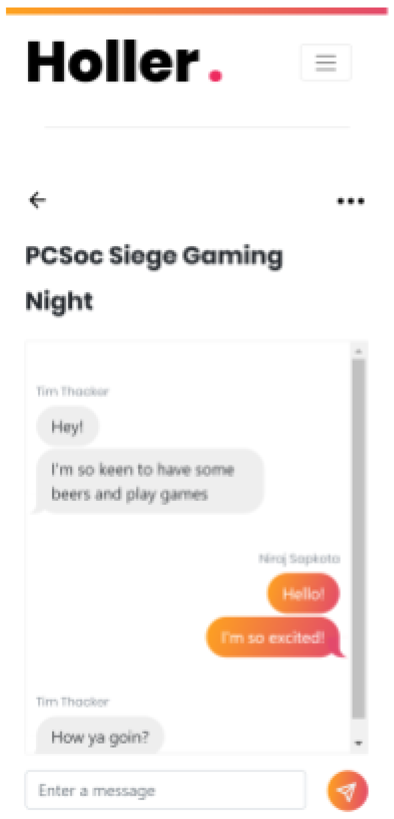
10. As a student, I want to send messages so that I can communicate with other users.

- When a user is matched, then a chat is created with 2 other users
- Given that a user is in the home page and they are logged in, then a 'Groups' tab is available in the navigation menu.
- Given that a user is not currently in the 'Groups' page, then clicking the tab brings up a list of all current chats.
- Given that a user is in the 'Groups' page, then a user should see each group chat with: the event that the users matched for, the most recent message, who sent the most recent message and when it was sent, and which event the group is going to.
- Given that a user is in the 'Groups' page, when a user clicks on the group chat, then it opens up the chat
- Given a user is in an opened group chat, then users can send text messages
- Given a user is in an opened group chat, then past messages are saved and can be viewed later

Sequence Diagram 8: User Story 10 - Sending a Message



Screenshot 10: Group Chat Example



PCSoc Siege Gaming Night

Tim Thacker

Hey!

I'm so keen to have some beers and play games

Niraj Sapkota

Hello!

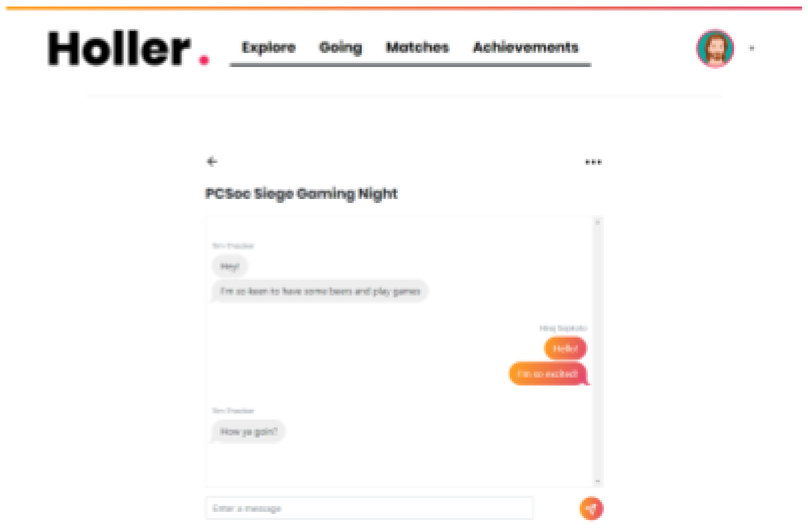
I'm so excited!

Tim Thacker

How ya goin?

Enter a message

Group Chat Page
Selecting a group chat will bring up the chat, as well as the chat history



PCSoc Siege Gaming Night

Tim Thacker

Hey!

I'm so keen to have some beers and play games

Niraj Sapkota

Hello!

I'm so excited!

Tim Thacker

How ya goin?

Enter a message

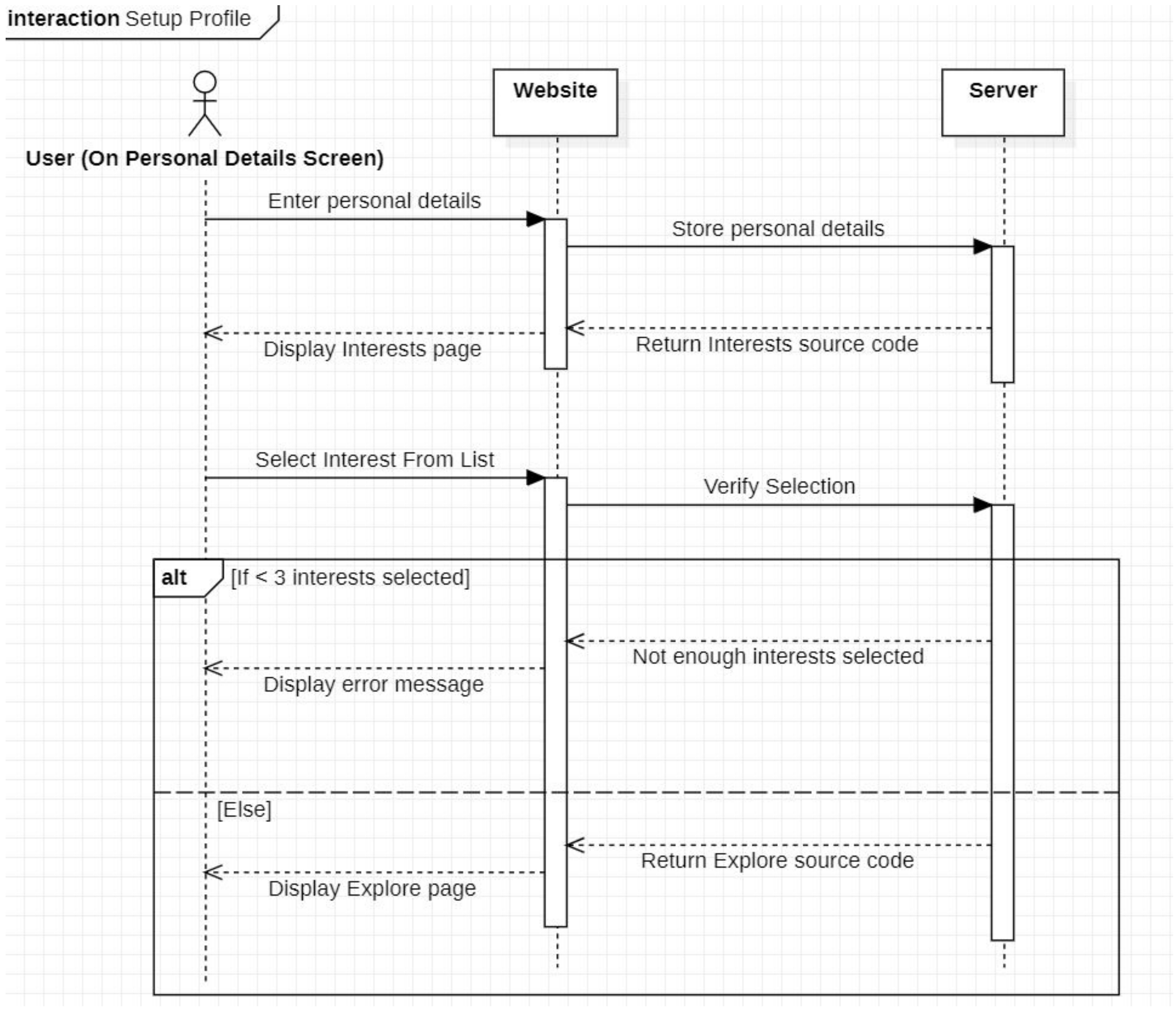
3.4 Profile

Epic 5: As a student, I want to view my profile and change my settings so that all my details are accurate, and I can keep track of my progress.

11. As a student, I want to set my personal details so that I can be matched according to my preference.

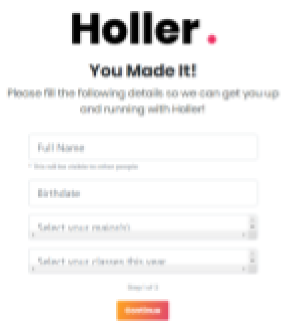
- When a user is signing up and logging in for the first time, then users can set their details
- Given a user is in the onboarding section, then users must put their first name
- Given a user is in the onboarding section, then users must put their date of birth
- Given a user is in the onboarding section, then users must select the classes they are taking that year
- Given a user is in the onboarding section, then users must select their faculty
- Given a user is in the onboarding section, then users must select their university (UNSW by default)
- Given a user is in the interests section of the onboarding process, then users can select from a list of various interests
- Given a user is in the interests section of the onboarding process, then a user must select at least 3 interests
- Given a user is in the avatar creation section of the onboarding process, then a user must create an avatar.

Sequence Diagram 9: User Story 11 - Setup Profile


interaction Setup Profile
Website
Server
User (On Personal Details Screen)
Enter personal details
Store personal details
Return Interests source code
Display Interests page
Select Interest From List
Verify Selection
alt [If < 3 interests selected]
Not enough interests selected
Display error message
[Else]
Return Explore source code
Display Explore page

Screenshot 11: Onboarding Process

The Onboarding Process
Users must fill out all the forms and all the details will be displayed on their profile page


Holler.

You Made It!

Please fill the following details so we can get you up and running with Holler!

Full Name

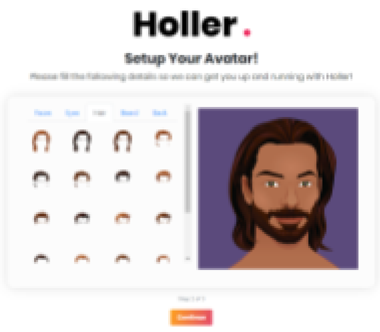
Birthdate

Select your major(s)

Select your classes this year

Step 1 of 3

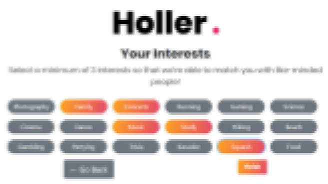
Step 1: Personal details


Holler.

Setup Your Avatar!

Please fill the following details so we can get you up and running with Holler!

Step 2: Create avatar


Holler.

Your Interests

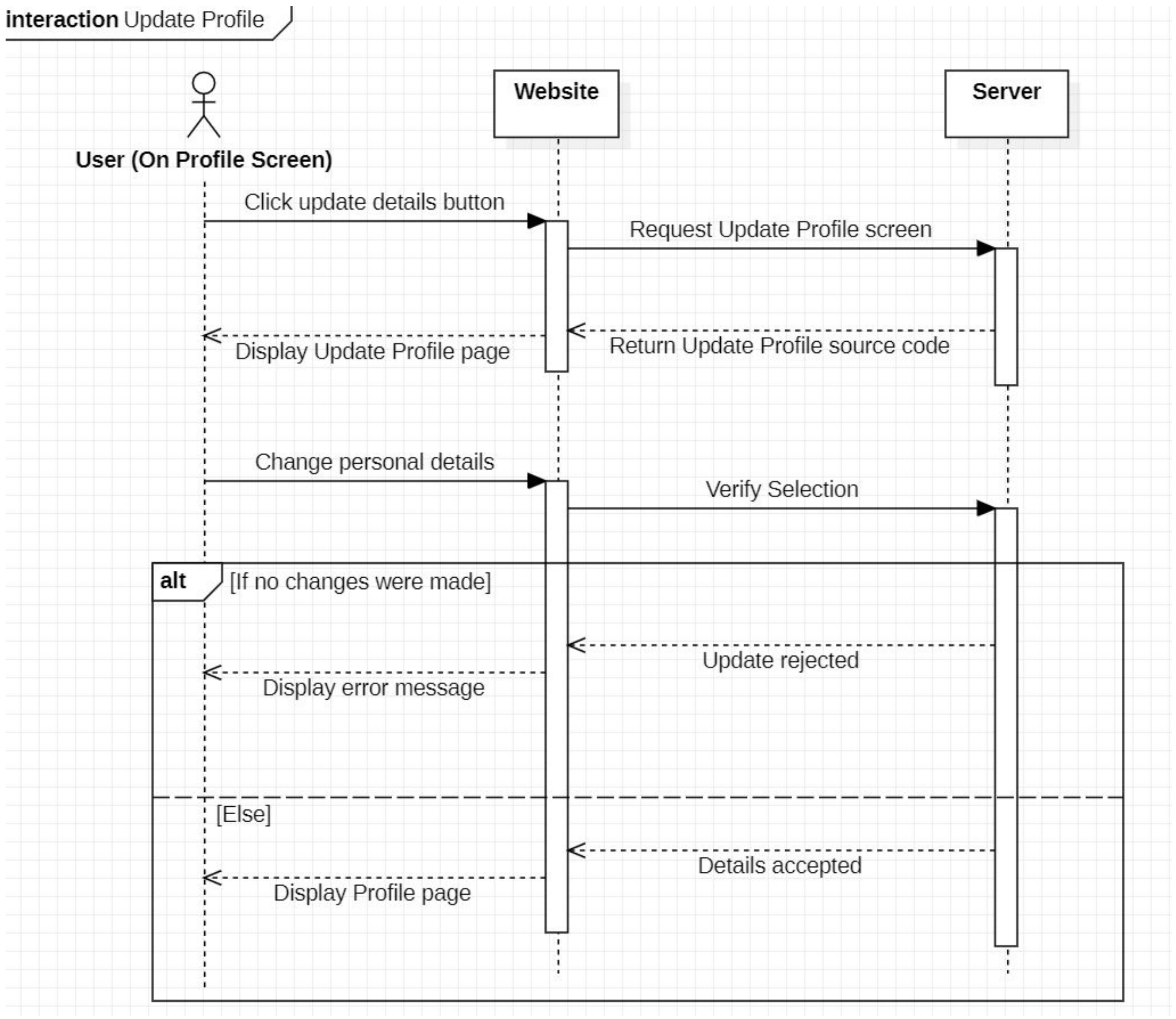
Select a minimum of 3 interests so that we're able to match you with like-minded people!

Step 3: Select personal interests

12. As a student, I want to update my personal details, so that my interests and details are up to date.

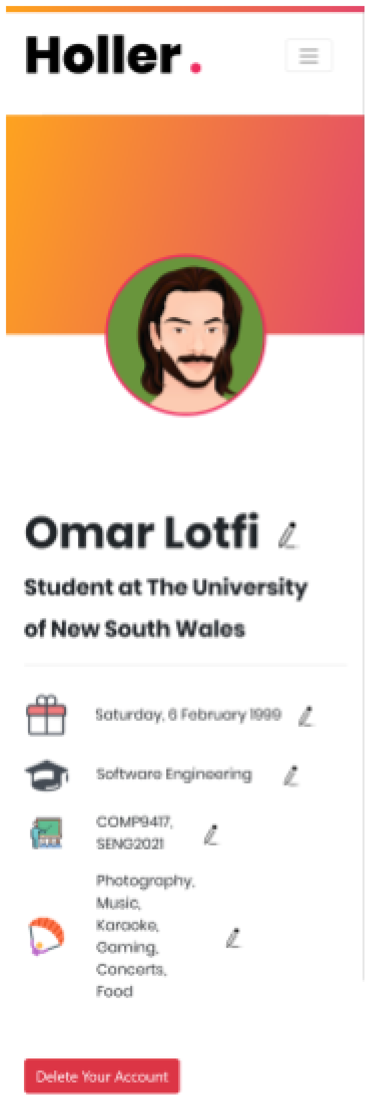
- Given a user is in the ‘Profile’ tab, when they click on edit profile, then they should be able to see their profile editing screen
- Given a user is in the profile editing screen, then users can change their list of interests by checking or unchecking from the list.
- Given a user is in the profile editing screen, then users can change their faculty
- Given a user is in the profile editing screen, then users can change their first name
- Given a user is in the profile editing screen, then users can change their classes

Sequence Diagram 10: User Story 12 - Update Profile



interaction Update Profile
User (On Profile Screen)
Website
Server
Click update details button
Request Update Profile screen
Return Update Profile source code
Display Update Profile page
Change personal details
Verify Selection
alt
[If no changes were made]
Update rejected
Display error message
[Else]
Details accepted
Display Profile page

Screenshot 12: Update Profile Page



Omar Lotfi

Student at The University of New South Wales

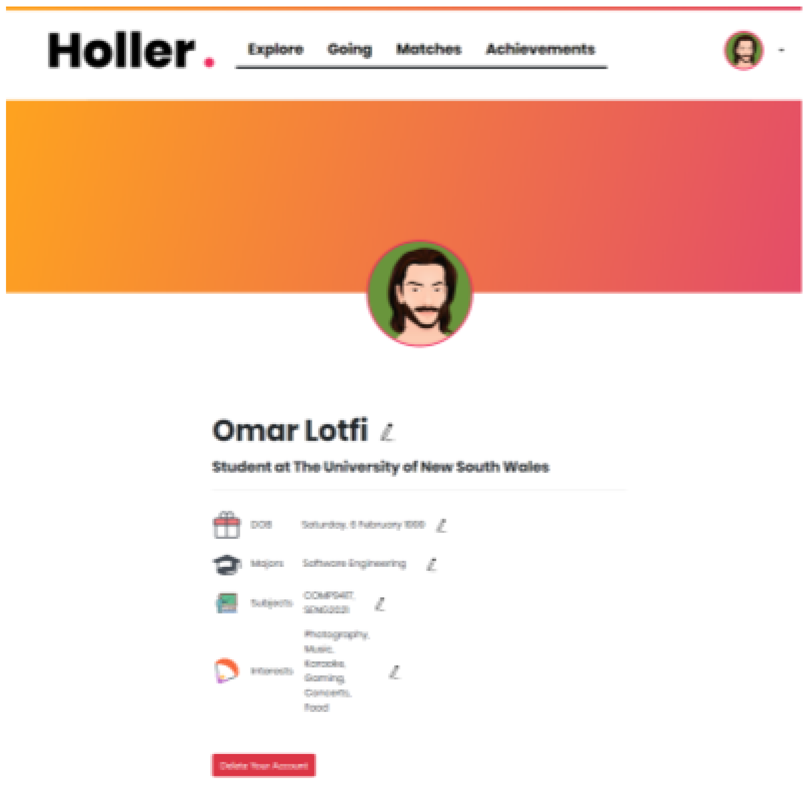
Saturday, 6 February 1999

Software Engineering

COMP9417, SENG2021

Photography, Music, Karaoke, Gaming, Concerts, Food

Profile Page
Users can edit their profile details by selecting the pencil icon next to their personal information



Omar Lotfi

Student at The University of New South Wales

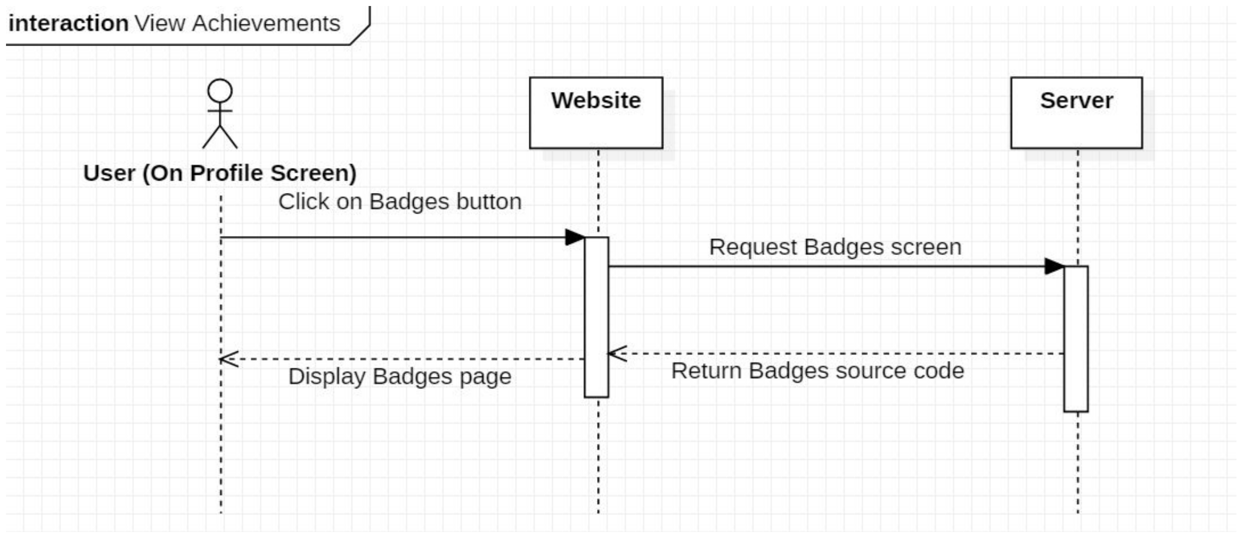
DOB	Saturday, 6 February 1999
Majors	Software Engineering
Subjects	COMP9417, SENG2021
Interests	Photography, Music, Karaoke, Gaming, Concerts, Food

Epic 6: As a student, I want to view my achievements and statistics so I can keep track of how much progress I have made in the app.

13. As a student, I want to keep track of my achievements, so that I can feel a sense of pride and accomplishment.

- When a user completes a certain task for the first time, then users are given a badge from a list of achievements (Examples: Change your avatar, Match with someone for the first time, Match with 10 people)
- Given a user is on the dashboard, when they click on Achievements, then users can view the badges they have earned

Sequence Diagram 11: User Story 13 & 14 - Viewing Badges

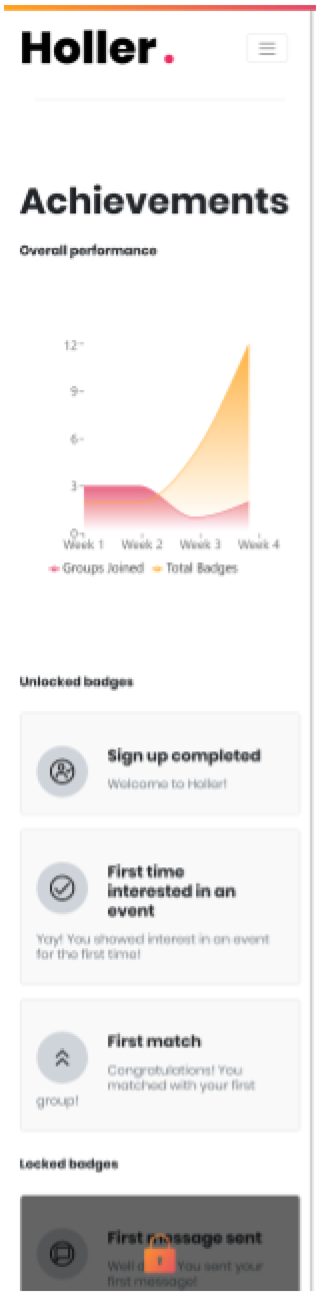


14. As a student, I want to view statistics related to my account, so I can keep track of how much I have used the app.

- Given a user is in their profile, then users can view how many groups they have joined
- Given a user is in their profile, then users can view how many badges they have earned
- Given a user is in their profile, then users are provided with a graph that tracks their score and their percentile rank

Refer to sequence diagram 11 above.

Screenshot 13: Badges Page



Achievements

Overall performance

Unlocked badges

Sign up completed

Welcome to Holler!

First time interested in an event

Yay! You showed interest in an event for the first time!

First match

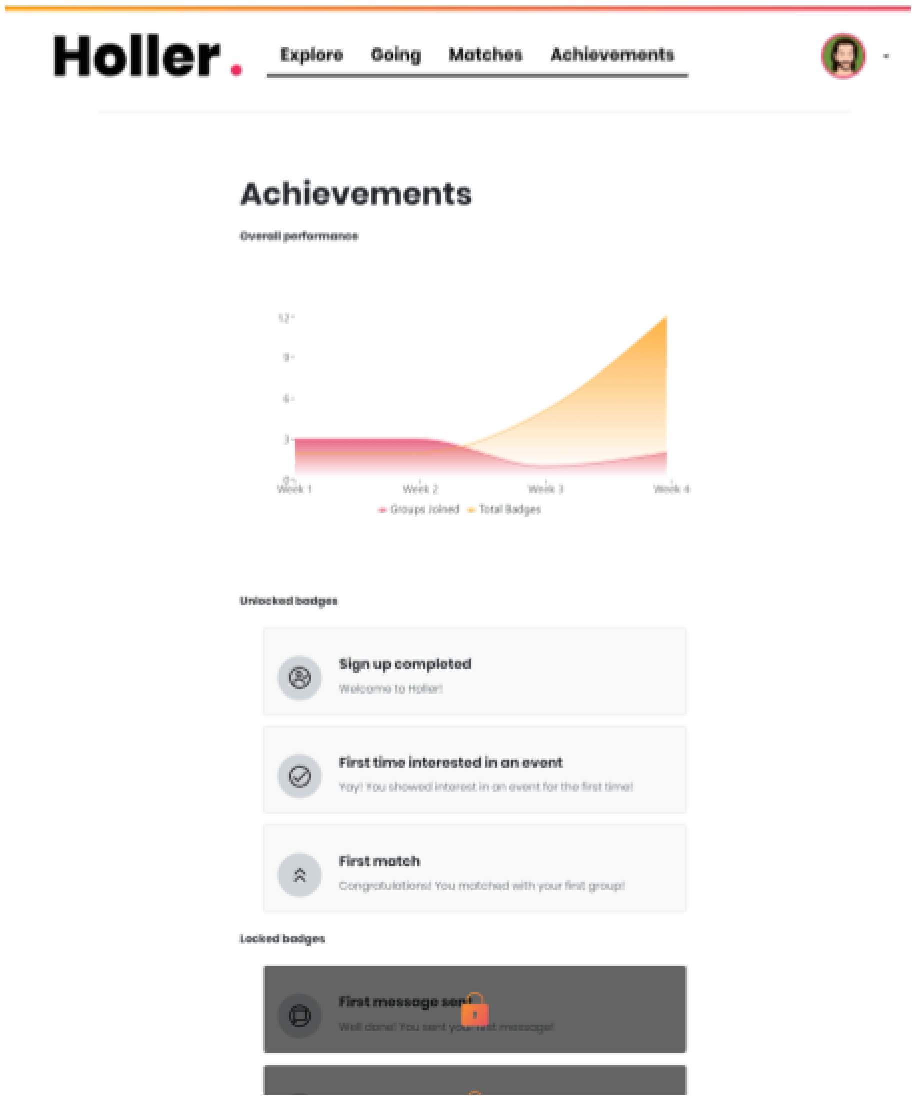
Congratulations! You matched with your first group!

Locked badges

First message sent

Well done! You sent your first message!

Achievements Page
A user can track the progress they have made on their app. They can check the number of groups they have joined, as well as the number of badges they have earned



Achievements

Overall performance

Unlocked badges

Sign up completed

Welcome to Holler!

First time interested in an event

Yay! You showed interest in an event for the first time!

First match

Congratulations! You matched with your first group!

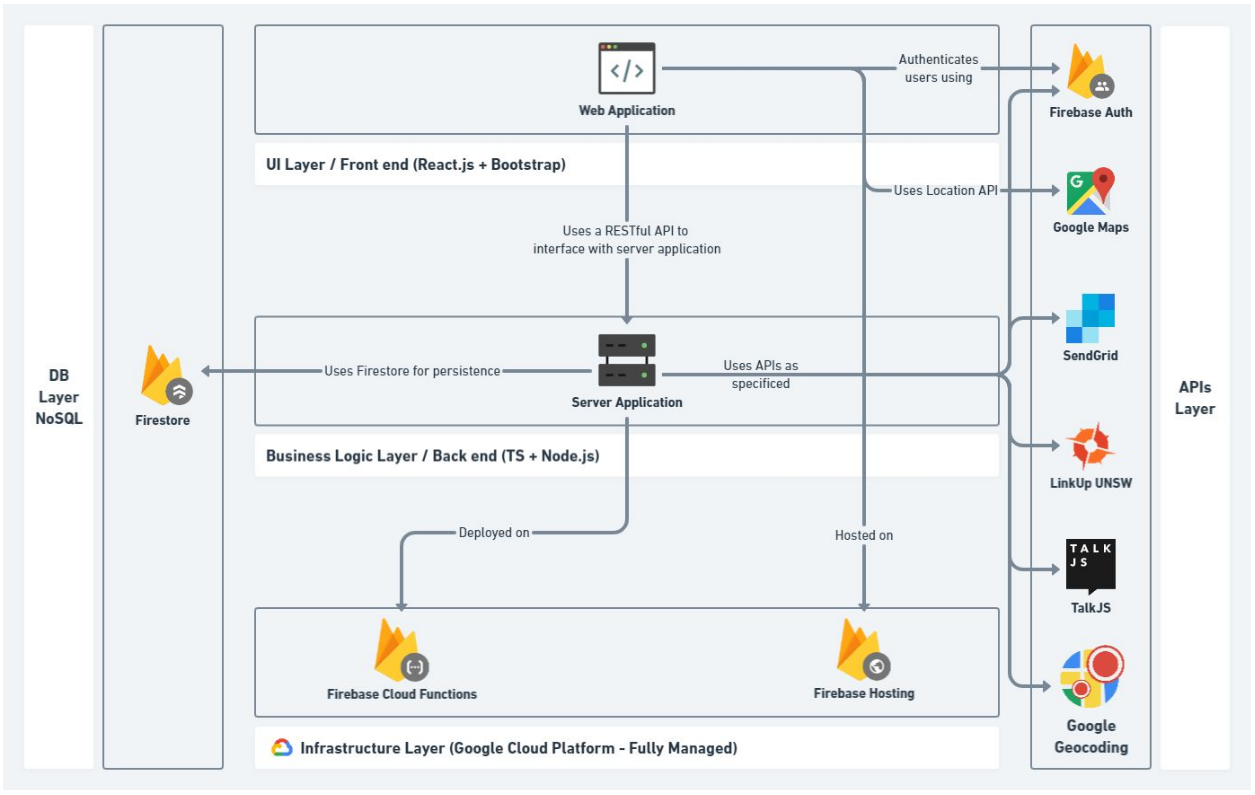
Locked badges

First message sent

Well done! You sent your first message!

4.0 Software Architecture

4.1 Architecture Diagram


DB Layer NoSQL
Firestore
Web Application
UI Layer / Front end (React.js + Bootstrap)
Authenticates users using
Uses Location API
Uses a RESTful API to interface with server application
Uses Firestore for persistence
Server Application
Uses APIs as specificed
Business Logic Layer / Back end (TS + Node.js)
Deployed on
Hosted on
Firebase Cloud Functions
Firebase Hosting
Infrastructure Layer (Google Cloud Platform - Fully Managed)
Firebase Auth
Google Maps
SendGrid
LinkUp UNSW
TalkJS
Google Geocoding
APIs Layer

4.2 Analysis of Software Architecture

Web Architecture

Our web application follows a versatile and modern web design architecture that allows for maximum flexibility and minimum coupling. Firstly, We will be building our server application to expose a RESTful JSON API. Secondly, our client application (web app) will be a Single-Page Application (SPA) that utilises the JSON API. This solution was chosen over others like multi-page server-side rendered applications because of its many advantages and benefits. This architecture allows the separation of the backend and frontend so they can be developed concurrently by different team members. It encourages the decoupling of business logic and the user interface in our application. It provides a fast and smooth experience for the user since the client application has full control over the data it requests over the network.

Infrastructure Layer

Our RESTful JSON API requires a server to run and respond to user requests. For example, a user may input an email and password and the server will respond to that event by validating the inputs. Typically, requests are sent to a server that you manage. However, this usually requires an upfront cost to buy the hardware and has extra costs (electricity, updates) to maintain the server for 24/7 use. It also does not scale well, as extra servers may be needed if the server capacity is full. The alternative is using a serverless architecture, which provides computing resources as they are needed, rather than having a constantly running server. All the server space and infrastructure is maintained and handled by a third party vendor. Also, it is charged based on the services that are used and the number of computations. For a small scale project such as ours, it would be cheaper to use serverless applications as there would not be too many executions and it would be faster to develop, as the infrastructure is handled by the vendor.

For our project, we will be using Google's serverless 'Cloud Functions for Firebase'. "Cloud Functions for Firebase let you automatically run backend code in response to events triggered by Firebase features and HTTPS requests." (Google, 2020)[3]. Although there are other alternative products that work in the same way, such as Amazon's 'AWS Lambda', 'Cloud Functions for Firebase' easily integrates other Firebase features that we will use, such as 'Firebase Authentication' and 'Firebase Firestore'. 'AWS Lambda' can integrate these features, but it will take more effort.

Our web application consists of static HTML, CSS, and JavaScript files that are served over the network to the user. Every time the user navigates to the web app URL, the browser downloads and executes these files. To ensure maximum speed and minimise the loading time for our users, we chose Firebase Hosting to host these static files. Firebase Hosting provides a global Content Delivery Network (CDN) built on top of the Google backbone network which has edge servers all around the world. The user gets these static files from the nearest one to their location to minimise latency. Firebase Hosting also provides automatic SSL certificates for security, easy deployment and rollback. This provider was chosen over hosting these files on our own servers, and other static hosts like Netlify and Github Pages due to it's deep integration with other Firebase services and superior features/performance.

[3] "Cloud Functions for Firebase - Google." 3 Dec. 2019, https://firebase.google.com/docs/functions. Accessed 24 Mar. 2020.

Database Layer

A database is a collection of data or information that is organised in a way to store, manage and retrieve information. An example of stored information could be a user’s personal details. Databases are usually stored on database servers, which follow similar pros and cons in the interface layer above. Third party providers can provide storage for databases or developers can build their own hardware for storing data. For our project, we are using ‘Firebase Firestore’, a scalable database that can sync data across different apps. It can be easily integrated into the previously mentioned ‘Cloud Functions for Firebase’. It is useful to us as our app ‘Holler’ can be used for both desktop and mobile users. If a user uses both devices, the ability to sync data in real-time can be helpful to us and the users. Firestore also automatically scales based on our requirements; pricing is based on how often the data is accessed and the amount of data stored.

There are alternative products which have similar functionalities, such as Amazon’s DynamoDB. However, DynamoDB does not sync devices in real-time like Firestore. Also, DynamoDB’s payment system is based on the amount you request; it does not automatically scale like Firestore. As we are not sure on how much traffic our application will generate, Firestore is suitable for our small application. However, in the future, if more users use the application, it will automatically scale the database for more users automatically.

The database is accessed by a NodeJS client library provided by Firebase only on our server application and is not accessed from the web application directly.. This is to ensure only authenticated and authorised users get access to the right data. While security mechanisms exist in Firebase like ‘Firestore Security Rules’ that could allow for direct access from the client, we chose to limit the database access to the server application to ensure the decoupling of our server/client applications.

API Layer

Application Programming Interfaces (APIs) are programs that offer services to other programs. For example, a developer can use authentication APIs to verify a user's inputs on their website. Our team will be utilising various APIs and other external data sources for our application. For login and signup authentication, our team will use Google's Firebase Authentication API. As another Firebase service, it will also be integrated seamlessly with other Firebase products we have mentioned. It provides everything related to signing in and logging in. It allows users to sign up and log in with email addresses, passwords and even phone numbers, or through Google Log-in or Facebook Login. It also provides two factor authentication to verify that an email belongs to the user that is signing up. Firebase delegates authentication/user management to ensure security and proper implementation. There are many other authentication APIs, such as Amazon Cognito or Auth0. However, we have chosen Firebase Authentication, as it is another Firebase product, which can be easily integrated with other Firebase products mentioned. Amazon Cognito and Auth0 may require more setup and may be more complex to integrate. Firebase Authentication also provides syncing across multiple clients in real time. As events and users are added to the application, our application would update the changes instantly, rather than having to update manually.

As our app displays information related to society events within the university, we will be using LinkUp UNSW API for this purpose. The API allows searching for all events happening in UNSW. There are many other event APIs to be added later so we can have a larger selection of events both inside and outside university, the most prominent of these APIs is Ticketmaster. There are other event APIs like the Eventbrite API which has a large selection of events, but does not support searching for events. And the Songkick API which requires an application for an API key before a developer can search for events.

Our app also allows users to chat with other users, and we will be using the TalkJS API to build a fully featured chat. TalkJS features integrated messaging, which allows developers to add user-to-user chat to a web application or platform. It has been chosen over other popular chat APIs such as SendBird or OpenTok, as our application will not and should not feature voice calls or video chats. SendBird and OpenTok does have these features, but our application is focused on making friends based on interests rather than appearances, thus, a voice or video chat video is unnecessary. TalkJS only features a text chat which is the only form of communication needed on our app, and the lack of extra features makes it easier to use.

If a user decides to allow email notifications, it will be handled through the SendGrid API. SendGrid is a cloud based service for sending emails. It allows email newsletters, shipping notifications and more importantly, sign-up confirmations. Although there are simpler alternatives such as MailChimp, SendGrid features the ability to track and evaluate marketing statistics such as how many emails were opened or sent to spam. This is useful as it can score how active a user is, which would be displayed in a user's profile. It also has one of the most generous free trials, allowing 40,000 emails for the first 30 days.

We will use Google Maps API and the Google Geocoding API to track the locations of each event. It provides Street View imagery and allows users to find routes to a specific location. Mapbox was an alternative that we considered, but it has features that are ultimately unnecessary, such as creating custom online maps. Due to the widespread usage of Google Maps and Google products, people would be most familiar with the Google Maps layout.

Other data sources that we will use is the UNSW Timetable, which features all the subjects in UNSW. This allows users to put in their profile the subjects they are taking for the term. There are unfortunately no better alternatives than from an official UNSW website.

All the above mentioned APIs are accessed on our server application to ensure users' security and proper separation of concerns. We do not access any API from the frontend except for 2 APIs as illustrated in the architecture diagram. The first one is the Firebase Authentication API. Our frontend calls this API to log in users directly without interfacing with our servers since this API abstracts the whole JWT, stateless authentication process so it's required to be done on Firebase's servers. The second API is Google Maps API which is used from the frontend to display the location and directions to the events' venues for the user. It's not used from the server application since it's purely user interface data delivered directly from the Maps API using their JS SDK.

Matching Engine Layer

The matching engine is it’s own microservice which polls the database for new interests in events which are starting within one day. The reason for only matching one day before the event is so that the database of event interests can be fully populated before finding the best matches amongst them.

The following steps define the vector based algorithm for matching:

1. Gets list of all event interests for events occurring within one day
2. Sorts the event interests into lists for each event
3. For each event compute all combinations of 3 users (n choose 3 combinations where n is the list length)
4. For each combination compute a metric for how similar each group member is (based on degree, subjects, age and interests)
5. Sort all combinations by this metric from high to low
6. Traverse sorted combinations and make matches if everyone in the group hasn't been matched before (this guarantees one match per user and maximises the total similarity of all the matches combined)
7. Remove event interests for all matched users for each event so that they aren’t matched again for the same event

Business Logic Layer / Backend

The technology stack used for the backend will be:

- Language: TypeScript
- Runtime: NodeJS
- Web Framework: ExpressJS
- Persistence: Firestore NoSQL Database..

We have chosen Node.JS and React JS as the main backend and frontend languages as it allows for "Full Stack JavaScript". Having JavaScript on both the backend and frontend makes development faster and also allows us to only focus on one overall language. Our team is also familiar with Javascript overall, as all of us have experience using Javascript before. Also, using Javascript allows us to make use of the large ecosystem out there through NPM; thousands of free packages can be downloaded and used.

We have considered using alternative languages. For example, for the backend, we have discussed using Java or Python, as they are also commonly used for backend development. However, Python is not an application built for mobile or desktop application development. There are few libraries compared to Javascript which has numerous libraries tailored towards developing applications. It is possible to build applications using Java, but it is a more general usage language. Compared to Javascript, which again, is tailored towards application development, Javascript is more suitable towards our project. There were also considerations for the database language for Firestore. We went with the default NoSQL, as it is highly scalable, and there are well designed development tools made for faster development. We considered Firebase FireSQL for querying our database, but it does not allow the deletion or updating of data.

User Interface Layer / Frontend

The technology stack used for the frontend will be:

- JavaScript Framework: ReactJS
- CSS Framework: Bootstrap (for better responsive design)

For the frontend, we have considered using Angular or just plain HTML. However, Angular is a Javascript framework compared to React which is just a Javascript library. The learning curve is steeper and could hinder development as concepts such as the Model View Controller (MVC) architectural pattern need to be learned. MVC is only optional in React. Also, using plain HTML is difficult as it requires us to manually read or update the DOM, which is a representation of the page that a user sees. Bootstrap was used alongside React as it is the most popular CSS framework; there is extensive documentation and help online, compared to smaller frameworks such as Foundation or Pure.

A user of our application would access it through their web browser. The 5 most common browsers are Chrome, Firefox, Internet Explorer, Edge and Safari, which takes up ~93% of the browser market share. (NetMarketShare, 2018)[4] Testing will be done to ensure that these browsers are compatible with the 5 main browsers. As the rest of the market is made up of many other browsers, it would be a waste of time and resources to test for all of them.

[4] "Browser market share - NetMarketShare." https://netmarketshare.com/browser-market-share.aspx. Accessed 24 Mar. 2020.

4.3 Summary

- Our client application (web app) is a Single-Page Application (SPA) that utilises the JSON API
- We used Google's serverless 'Cloud Functions for Firebase' to automatically run backend code in response to events triggered by Firebase features and HTTPS requests.
- Our web application consists of static HTML, CSS, and JavaScript files that are served over the network to the user
- We are using 'Firebase Firestore', a scalable database that can sync data across different apps.
- For login and signup authentication, our team will use Google's Firebase Authentication API.
- LinkUp UNSW API allows searching for all events happening in UNSW.
- We are using the TalkJS API to build a fully featured chat.
- We will use Google Maps API and the Google Geocoding API to track the locations of each event.
- Other data sources that we used are the UNSW Timetable, which features all the subjects in UNSW.
- The matching engine is it's own microservice which polls the database for new interests in events which are starting within one day

The technology stack used for the backend will be:

- Language: TypeScript
- Runtime: NodeJS
- Web Framework: ExpressJS
- Persistence: Firestore NoSQL Database..

The technology stack used for the frontend will be:

- JavaScript Framework: ReactJSWe built our server application to expose a RESTful JSON API.
- CSS Framework: Bootstrap (for better responsive design)

5.0 Reflections

5.1 Team Roles

Name	**Role**	**Contributions**
Andrew Han	Assistant Team Leader / Front-end Assistant	- Helped organise teams towards deadlines - Responsible for tracking deadlines and submitting deliverables - Helped in developing the front end
Harikrishna Manogaran	Front-end Assistant	- Helped develop the front end - Designed desktop mock-ups - Provided aid in deliverable reports
Niraj Sapkota	Head of Front end	- Developed the front end and helped in application design - Allocated roles in the development of the front end
Omar Lotfi	Team Leader / Designer / Full-stack Assistant	- Responsible for allocating roles and tracking progress - Designed idea, lo-fi wireframes and functional prototype - Developed chat API library and landing page
Timothy Thacker	Head of Back end	- Designed and developed the RESTful API - Designed and developed the matching algorithm - Allocated roles in the development of the back end

In the beginning of the project, Omar volunteered to take the role as team leader. Together with Andrew, they assigned roles and ensured the project kept on track. Omar was assigned to create the lo-fi and functional prototype of the application and help with back end/front end. Andrew was assigned to work on the frontend as an assistant, however, he was responsible for organising and submitting the deliverables.
Timothy was assigned to lead the back end, as he was the most experienced in developing RESTful APIs.
Niraj was assigned to lead the front end as he was the most experienced in using React, which was chosen early on as the main framework for front-end development.
Hari took the role as general front-end assistant, providing assistance and support for development of the front end as well as the desktop UI prototype. He also provided assistance in developing the deliverables.

5.2 Our Experiences

Our team had a bit of a tough time while undertaking SENG2021. This was due to the high standards we put on ourselves; to achieve high distinction and to make a serious attempt at the Macquarie prize.
The team was formed out of a desire to achieve the Macquarie prize, and our choice of language and idea was developed within the first two meetings.
Our first and second deliverable was received positively and there were only a few issues while making it, primarily of understanding what the marking criteria expected of us.
Around this time, the front end was being developed by copying the styles used in the high fidelity prototypes, and the backend was being populated by the classes in UNSW. This information was scraped from the UNSW courses website.
Our third deliverable was also received well, and around this time was where development truly began and issues really began to arise.
Timothy began creating an endpoints document that allowed the front end to request information from the backend (through POST/GET). The front end faced some issues in the responsiveness, as we tried to tailor our design to even the smallest screen available (iPhone 5). This caused some css styles to spill out or cause issues with the design. Also, after a review with our mentor, several features had to be scrapped and other features added to ensure a smoother user experience.
After a basic front end was made, next was hooking up the front end with the back end. This resulted in bugs being found in the back end, which slowed down overall development. However, once these bugs were fixed, the rest of the development progressed steadily, until the application was finished.

5.3 What We Learned

- Experienced working in a team environment
- Learned ow to develop sequence diagrams
- Learned how to use software to develop professional low fidelity and high fidelity prototypes
- Gained information about the software architecture stack
- Gained familiarity with respective front end/back end languages and frameworks, as well as general Javascript
- Gained an understanding in using APIs and front end packages.

5.4 Positives/Negatives

Pros:

- More experience with using Javascript
- More experience working with team members
- Created a beautiful and responsive website
- Gained some prestige by being chosen as one of the finalists for the Macquarie prize (hopefully we win overall)

Cons:

- Stress due to high standards placed on ourselves
- Due to limited time, more features could not have been added

5.5 Suggestions For Improvement

We believe that there are a few possible areas that this course can improve on. The information provided in the first deliverable was a bit vague. For example, there was nothing that clearly indicated we should analyse other similar systems or to provide a list of features. It would be better if it explicitly said “For part 1, we expect ‘x’, ‘y’, ‘z’” and then ensure the marking criteria reflects these expectations. This deliverable could have also had less documentation tasks and more ones that can directly aid in development of the application later.

Another possible improvement is to push the idea of developing earlier. Our team struggled to put together our application in time, but it felt like there was not much urgency to develop the code until after deliverable 3. There should be a bigger push to start coding quickly after deliverable 2. In addition to that, we think it’s a good idea to emphasise the importance of UI/UX principles early in the course so the teams can have a better shot at designing a professional application.