NUS_FREE_THINGS

ONE MAN'S TRASH IS ANOTHER MAN'S TREASURE

NUS Orbital 2024

Milestone 3

Brandon Kang

Sean Ter

Table of Contents

1 | Foreword

This report was specially crafted to showcase our product, NUS_Free_Things, a mobile app that we have built for our summer project, Orbital 2024. This segment serves as a guide on how to navigate this report, as well as background information about us and our app.

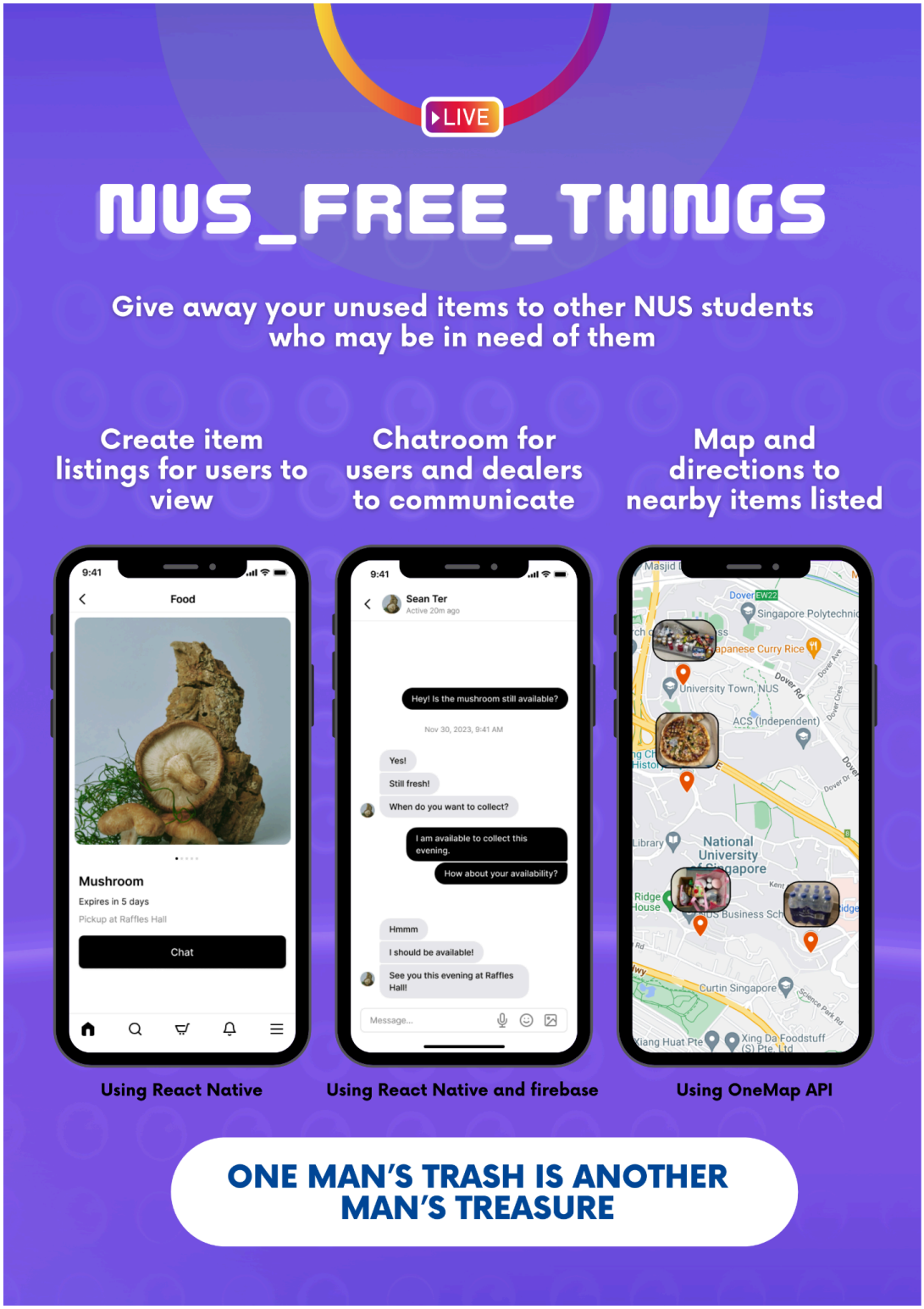
The poster in **Section 2** was our liftoff poster, which shows what we initially envisioned our app to be like. More posters will be added after each of the milestones to form a gallery which showcases the progress of building our app. We also shared our **motivation** for choosing this project, as well as what we **aim** to achieve by building our app. **Multiple user stories** are provided to allow you to better understand who are the target audience of our app and why they would want to use our app. For each of the features, we included details about the **implementation**, the **challenges faced**, and even **diagrams** to help you visualize the details better.

Building an app from scratch exposed us to new areas and tools, and also allowed us to appreciate the importance of proper documentation. This app gave us a taste of what it is like being developers of a project in the future. It has also exposed us to the challenges faced when developing software, and the numerous hours of bug fixing and implementing new features. Thus, we have tried to explain as much as possible for each section, so that it will be informative so that you can have a better reading experience as a reader.

Brandon and Sean

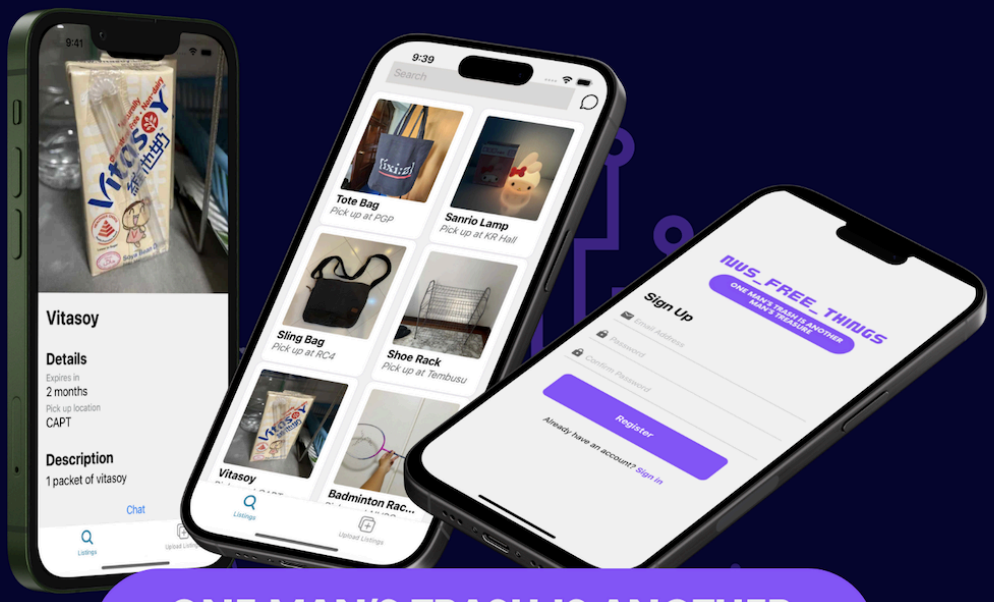
NUS Computer Engineering Year 1 Students

2 | Posters



Liftoff Poster

NUS_FREE_THINGS



ONE MAN'S TRASH IS ANOTHER MAN'S TREASURE

FEATURES

- **User Account Authentication**
  - Users are able to create an account and register using their email and password
- **Listings Page**
  - The main page where uploaded listings will appear.
- **Upload Listings Page**
  - Users enter details about the item that they want to give away and upload them here

Powered by:

Milestone 1 Poster

NUS_FREE_THINGS

ONE MAN'S TRASH IS ANOTHER MAN'S TREASURE

FEATURES IMPLEMENTED

Chatrooms

- Users are able to chat with owners of listings to arrange details and meet up

Listing Map

- Users can also view the listings from a map for convenience

Powered by:

Milestone 2 Poster

ABOUT

NUS Free Things is an app for users to give away unwanted items to others who have more use for it than them. We aim to help reduce wastage in NUS hostels this way.

FEATURES

- User Account Authentication
- View listings uploaded by users
- Upload your own listings
- Search for listings using a map
- Chatroom to chat with other users
- Profile page to see all your listings

TECH STACK

NUS_FREE_THINGS

ONE MAN'S TRASH IS ANOTHER MAN'S TREASURE

LISTINGS MAP

Users can use the Listings Map to scroll through the map of NUS to see all available listings based on their location. Locations with available listings will have a pin and clicking on that pin will display a list of all the listings at that location.

VIEW LISTINGS

The main page of the app is the Listings page where all the listings uploaded by other users are displayed. Users can scroll through the listings to find listings that they are interested in. There is also a search bar and filter to aid the users in finding specific listings.

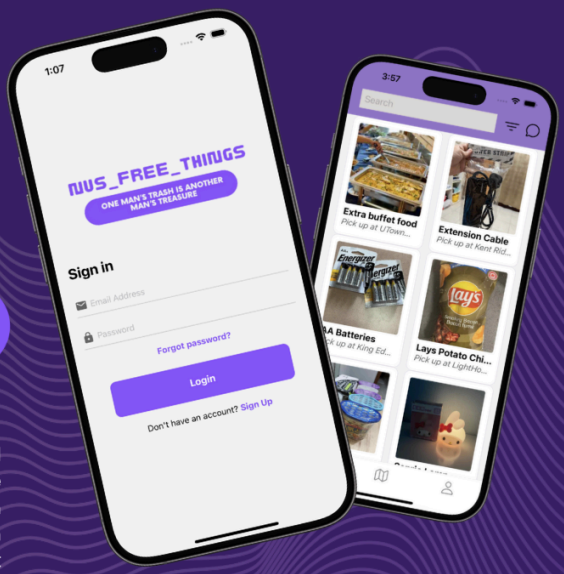
UPLOAD LISTINGS

Users can upload their own listing by selecting an image of the item and entering the details about the item that they want other users to see such as name, pick up location, expiry date (if any), and description. After uploading the listing, it will be visible for other users on their listings page, while it will be visible for the uploader on his/her profile page.

SWE PRACTICES

We use Git for version control and for managing our code. Pull requests are used when a new feature is implemented and wants to be merged with the main code.

We also conducted software testing on our app for quality control. We conducted unit testing and integration testing for functional testing, and user testing for non-functional testing.



USER ACCOUNT AUTHENTICATION

Users would have to sign up with an account using their email address. They can then sign in with a registered email account to access the app.

PROFILE

Users can view and manage all their active listings in his/her profile page. Users can upload new listings, delete current listings and sign out from the app here.

CHATROOM

Users can start a chat with dealers of listings that they are interested in. In the chatroom, 2 users can send messages to each other to discuss about details about the listing, meet up location and timing. There also is a page which displays all active chatrooms that the user is currently in.

Milestone 3 Poster

3 | Proposed Level of Achievement

Apollo 11

4 | Aim

We aim to reduce wastage in NUS by providing a platform for users to give away items which they do not want to those who have more use for it than them.

5 | Motivation

Both of us are students staying on campus in Tembusu College, and in Tembusu we have a Telegram Channel called “Tembu Free Things” where residents can give away things that they no longer want instead of throwing them away. We noticed how this channel is frequently used and how many items would potentially be thrown away if they were not given away. The amount of items being given away reaches its peak near the end of the semester as seniors or exchangers who are no longer staying in Tembusu next semester would give away more items to get rid of them.

Thus, we were inspired by this and would like to expand this concept to the whole of NUS. Not all unwanted items that are being disposed of are completely unusable. Therefore, these items that are still usable can be given to other students through the help of our app.

6 | Vision

NUS_Free_Things will provide a quick and easy way for students to dispose of their unwanted items, as well as find items that they need without spending any money. This way, we can decrease the amount of waste being generated in hostels in NUS, and students would also benefit from getting free items as well.

7 | User Stories

1. As an NUS student who stays on campus and wants to get rid of an old mini desk lamp, I want to be able to give it away to other students who may want it rather than throwing it away.
2. As an event organizer in NUS who has excess food from a buffet for an event, I would like to be able to get rid of the food by giving it to others without throwing them away.
3. As an engineering student in NUS who would need materials such as batteries for my engineering related projects, I would like to scroll through a platform to see if anybody is giving away these types of materials so that I can save money.
4. As an NUS student who stays on campus and has many unwanted snacks in my room that I am unable to finish, I would like to give them away to other students who may want them instead of leaving them in my room to expire.
5. As an NUS student who wants to save money, I would like to have a platform to see if there are any free food options that are available nearby.
6. As an NUS student who is moving out or no longer staying on campus, I would like to give away my mini shoe rack that I used when I was staying, so that it does not go to waste.

8 | Features

8.1 User Account Authentication

Each user would need to be authenticated using his/her unique account to access the app. The sign-in method used for NUS_Free_Things is through email and password.

Utilizing the Firebase JS SDK Authentication, users can:

- Sign up to create an account if the user does not have an existing account
- Log in via email and password for existing users
- Remain signed in from previous sessions if they have not signed out
- Reset their password if they have forgotten their password

Users signing up would have to verify their email address by clicking the verification link that was sent to that email address. Users would only be allowed to sign in after their email has been verified.

Only when the user has successfully logged in will the user be routed to the main listings page of the app. All accounts that have been registered will be saved and reflected in FIrebase Authentication.

Users can click on the “Forgot Password” button at the bottom of the sign in page if he/she has forgotten his/her password. An email will be sent to the user’s email address where he/she can reset her password. After the password has been successfully reset, the user can now sign in with the new password.

8.1.1 Implementation

We used an observer that constantly checks the authentication status of the user. When the app is first opened by the user, the observer finds out if the user is signed in or not and directs them to the appropriate page.

If the user is signed in, he will be directed to the main listings page.

If the user is not signed in, he will be directed to the sign in page where the user has to sign in before proceeding. If the user does not have an account, there is an option for the user to click to create an account.

This ensures that the app registers any new users and stores their user information.

We have also implemented error handling which will present a popup on screen to inform the user that an error has occurred and show the relevant error message.

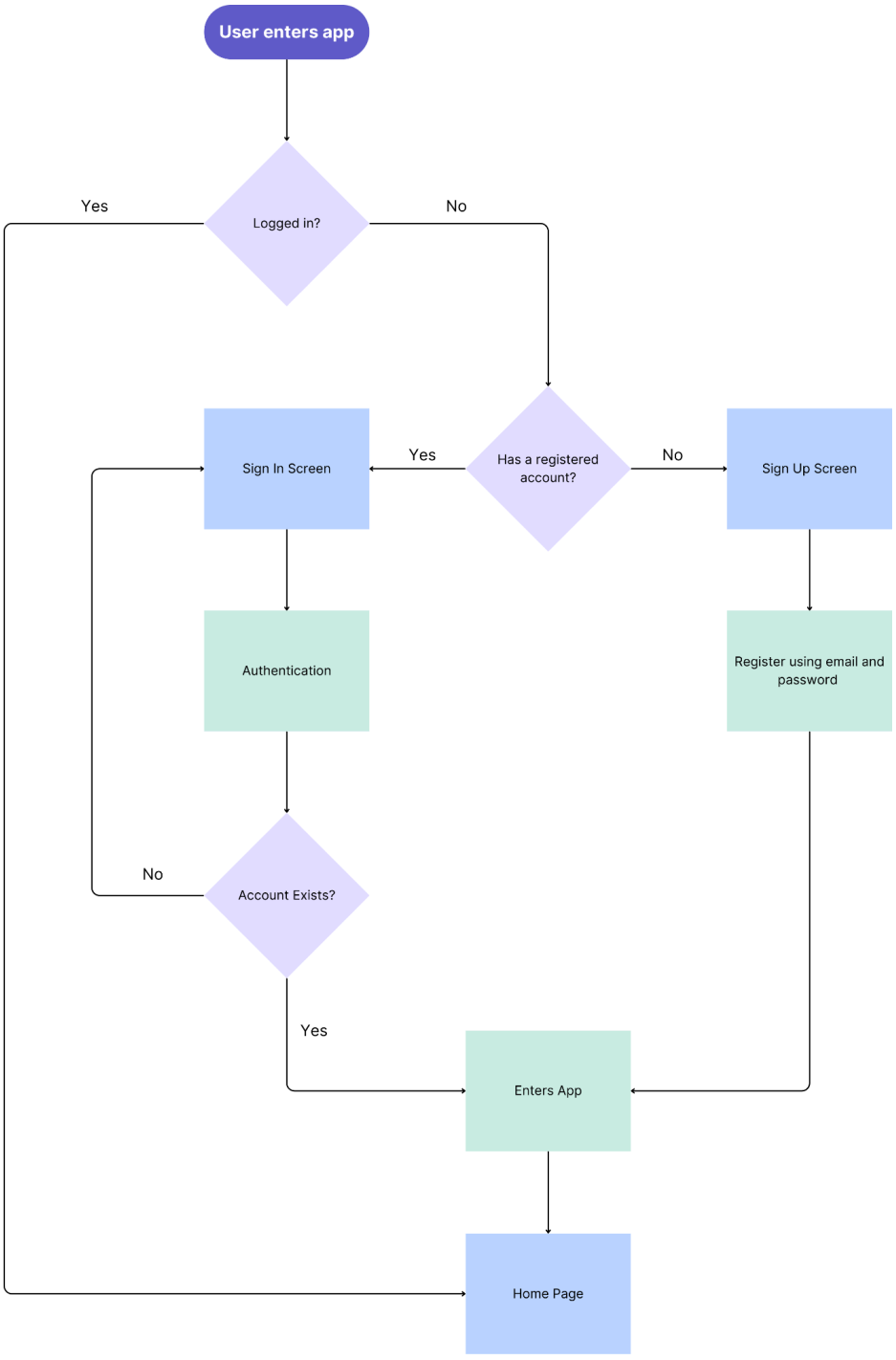
If the main component with the observer is unmounted, such as when the app is closed, it will unsubscribe from the observer to prevent any memory leaks.

The app also remembers the authentication status of the user from previous sessions, so if the user did not sign out after using the app, the user does not need to sign in again the next time the user opens the app again. This makes it more convenient and less tedious for the user when using our app.

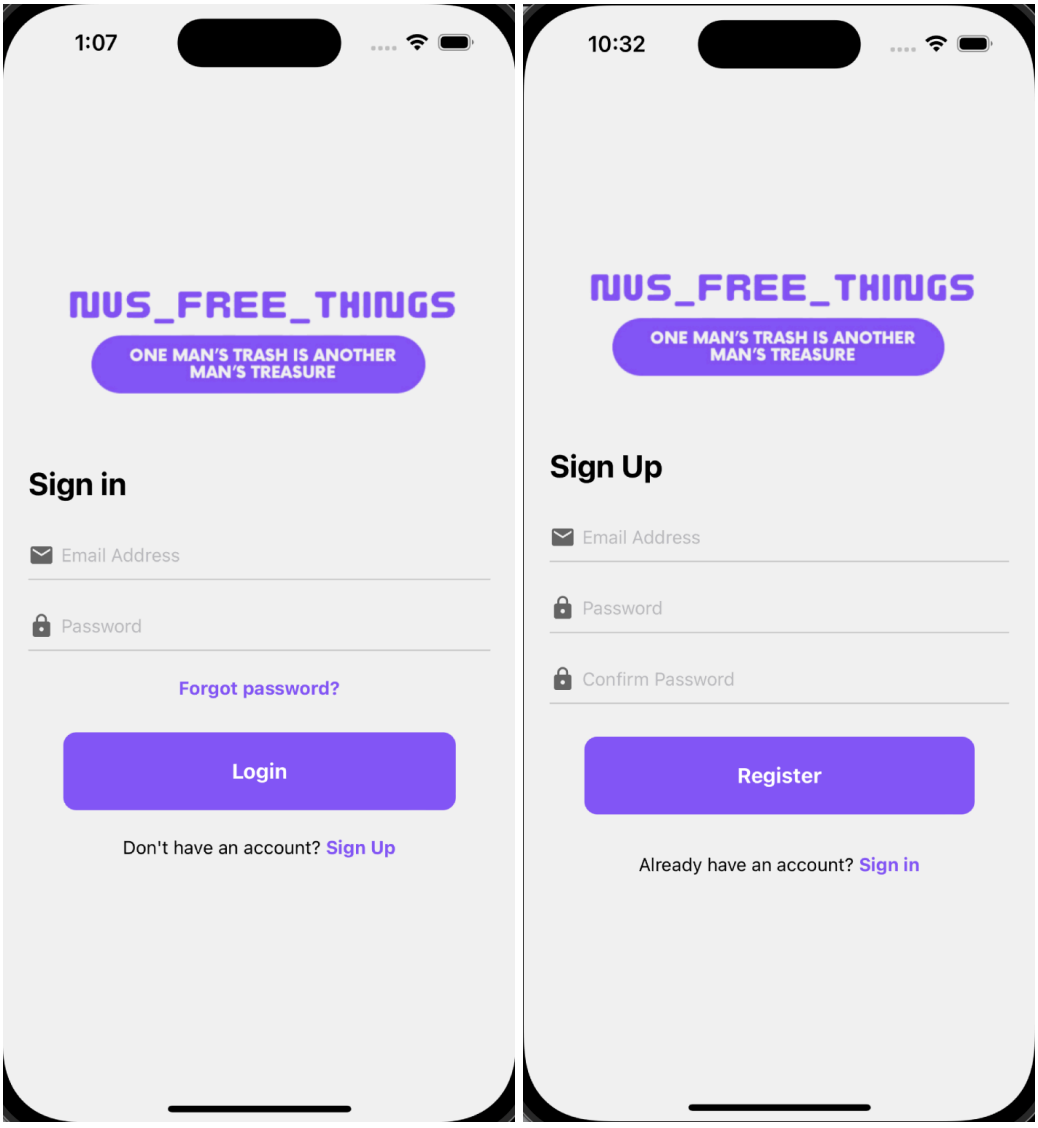
8.1.2 Challenges Faced

The main issue faced was handling the routing of users to the relevant pages after they have signed in or signed out. Initially we did not use observers and manually checked if the user is signed in once after the main component of the app has been rendered. Similarly, the users are manually signed out once they have clicked the sign out button. However, this may pose problems as users may access parts of the app that they are not supposed to if they are not signed in, or might be denied access to parts that they should be able to reach. We thus opted to use observers to track the authentication status and handle real-time changes to the authentication state of the user. This way, handling authentication is more straightforward and efficient.

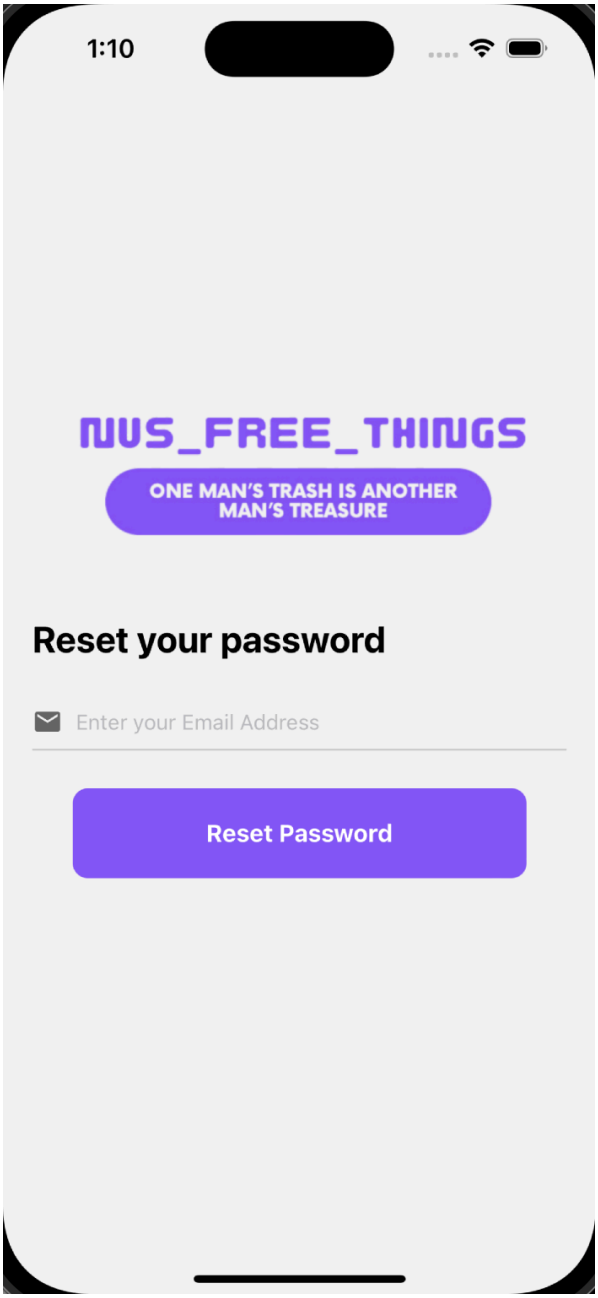
8.1.3 Diagrams



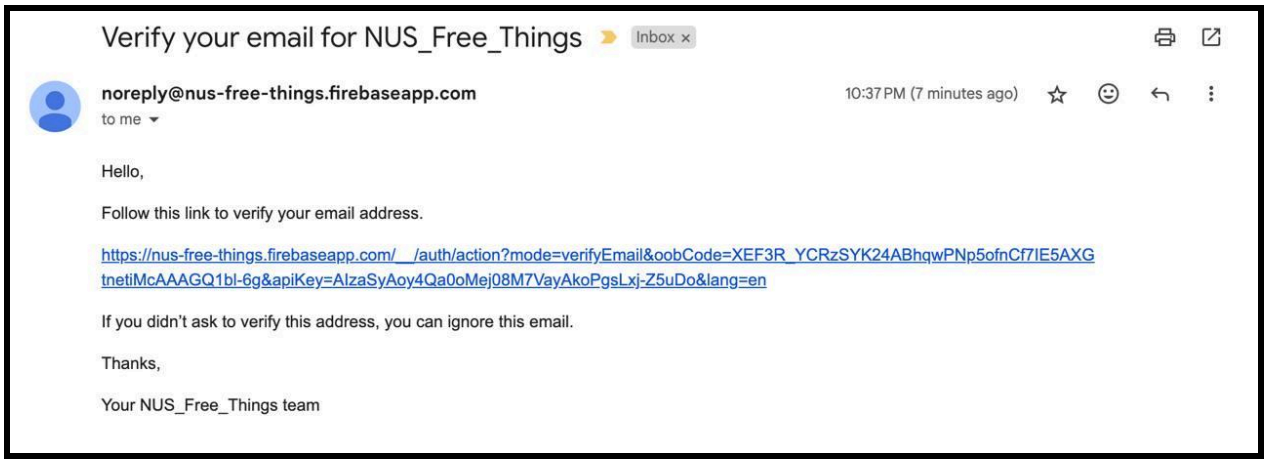
Authentication Flow



Sign In (left) and Sign Up (right) screen



Reset password page



Verify your email for NUS_Free_Things Inbox ×

noreply@nus-free-things.firebaseapp.com 10:37 PM (7 minutes ago)
to me

Hello,

Follow this link to verify your email address.

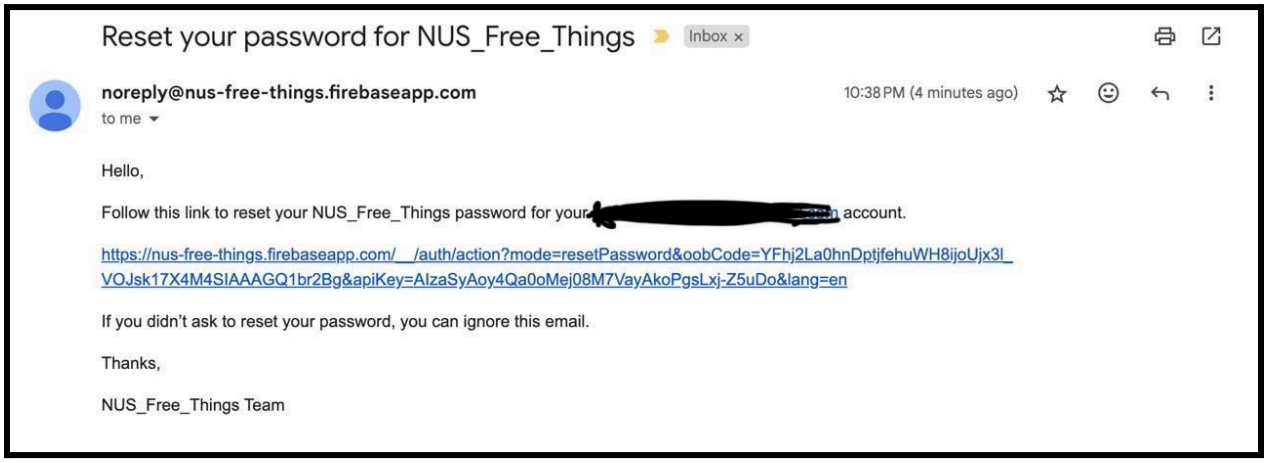
https://nus-free-things.firebaseapp.com/__/auth/action?mode=verifyEmail&oobCode=XEF3R_YCRzSYK24ABhqwPNp5ofnCf7IE5AXGtnetiMcAAAGQ1bI-6g&apiKey=AIzaSyAoy4Qa0oMej08M7VayAkoPgsLxj-Z5uDo&lang=en

If you didn't ask to verify this address, you can ignore this email.

Thanks,

Your NUS_Free_Things team

Sample verification email



Reset your password for NUS_Free_Things Inbox ×

noreply@nus-free-things.firebaseapp.com 10:38 PM (4 minutes ago)
to me

Hello,

Follow this link to reset your NUS_Free_Things password for your [redacted] account.

https://nus-free-things.firebaseapp.com/__/auth/action?mode=resetPassword&oobCode=YFhj2La0hnDptjfehuWH8ijoUjx3l_VOJsk17X4M4SIAAAGQ1br2Bg&apiKey=AIzaSyAoy4Qa0oMej08M7VayAkoPgsLxj-Z5uDo&lang=en

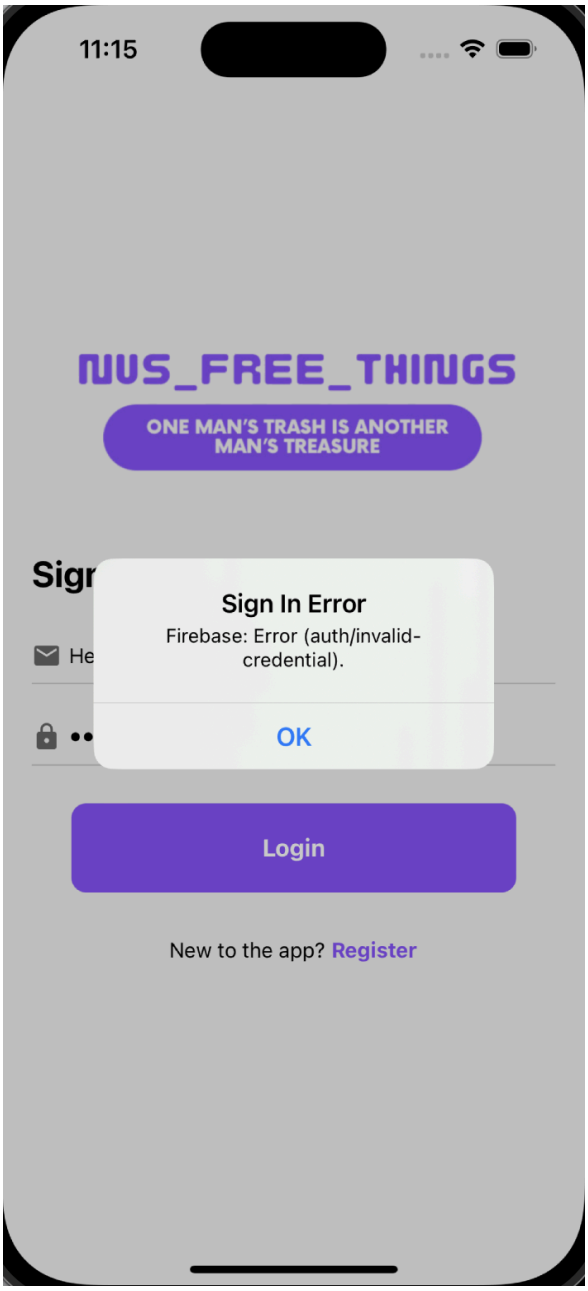
If you didn't ask to reset your password, you can ignore this email.

Thanks,

NUS_Free_Things Team

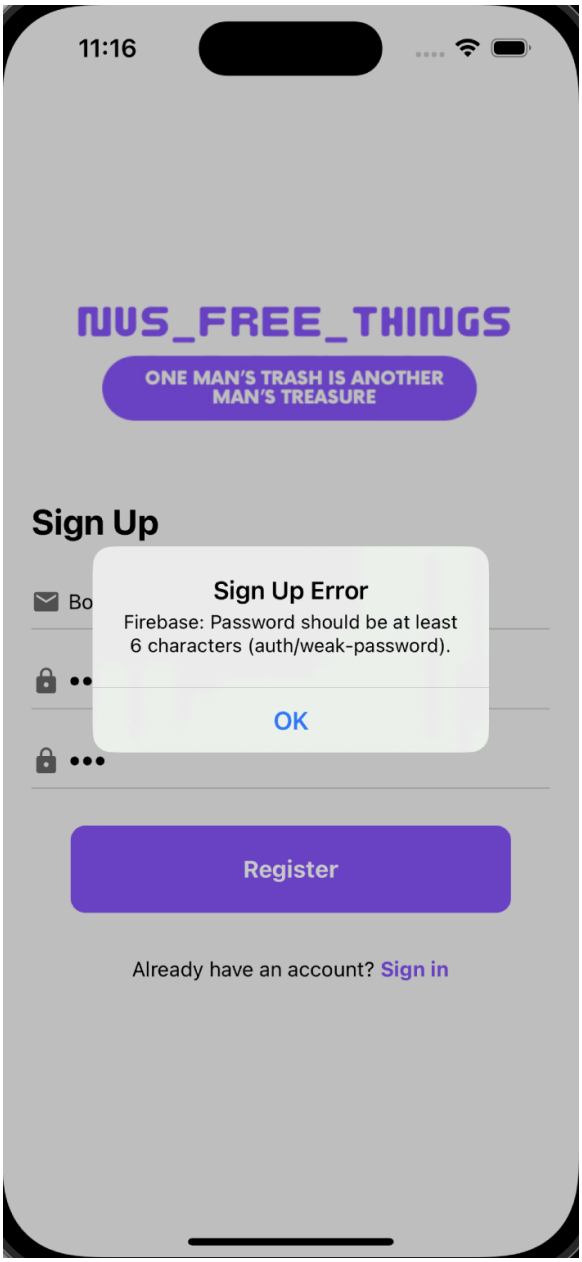
Sample email for resetting password

Errors when signing in

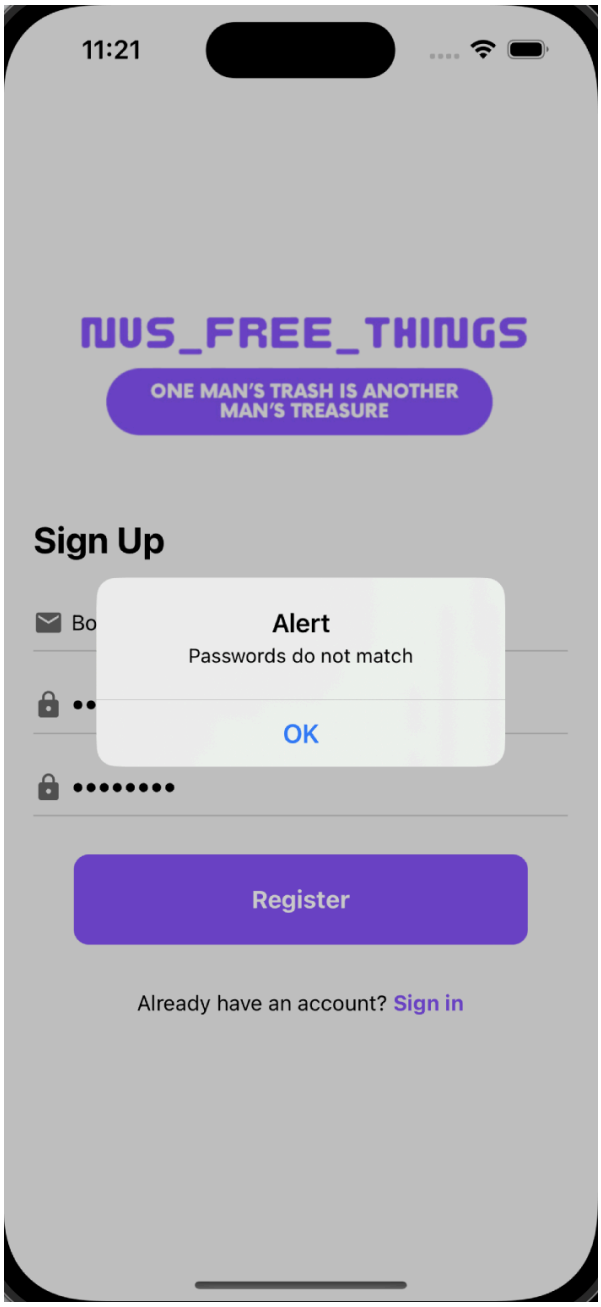


Error popup if account has not been registered

Errors when signing up



Error popup if password entered is too short



Error popup when passwords entered do not match

8.2 Listings

Once logged in, users will be redirected to the main screen consisting of 2 tabs, listings and upload listings. The listings page will contain all of the listings in the database.

Users will be redirected to different screens depending on what they tap on the screen. They will be redirected to:

- Their chat history with other users by clicking on the chat icon on the top right corner of the screen
- A more detailed description of the listing when a listing is tapped on
- The upload listings page after tapping on the upload icon at the bottom of the screen

General information regarding the listing such as pick up location and name of the item will only be shown at the listing page. However, additional information such as expiry date and description of the item will be shown if the listing is tapped on as shown on the diagram below.

Users can filter the listings by categories by clicking on the filter icon at the top right of the listings page, next to the chat icon. A pop up will appear for the user to select which category he/she would like to filter the listings by. Only listings that fit the categories selected will be displayed on the listings page. Click “Confirm” to confirm the categories that you wish to be filtered by.

8.2.1 Implementation

The tabs navigator of the home screen is nested under a stack navigator which includes the sign in screen and sign up screen. Under the tabs navigator, there are 2 screens which include the listing page and the upload listings page.

We used the useNavigation function from expo-router to direct the user to the different screens on our application. The function is used together with

react native elements such as TouchableOpacity and Button to redirect users to the respective page whenever a button or icon is pressed.

We used TouchableOpacity if clicking on an icon or a text was more intuitive for the user and used the button for logging the user in and out of our application.

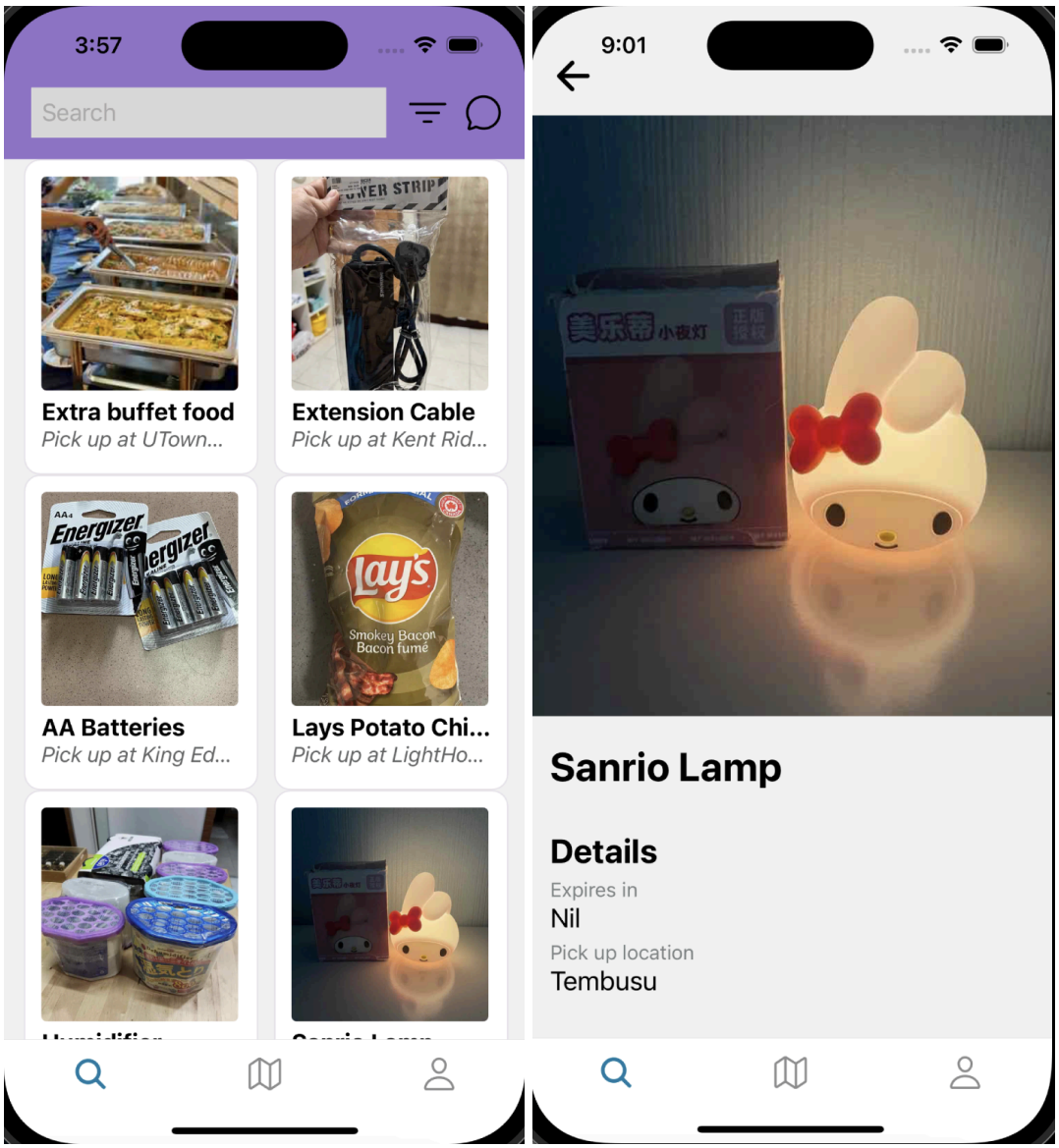
Listings will be read from the Firebase Database under the “listings” collection and displayed on this page.

The filter checks the “listings” collection for all the listings under that category based on their documents, and displays the filtered listings on the listing page.

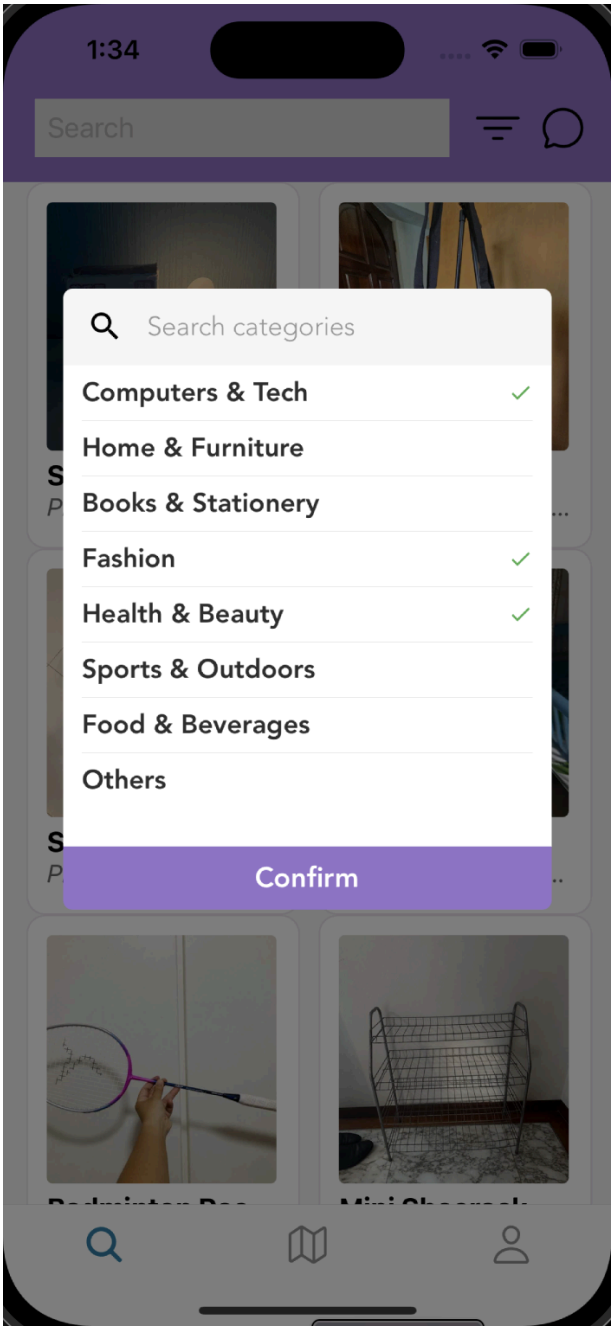
8.2.2 Challenges Faced

As react native is compatible on both IOS and Android, the layout shown on IOS slightly differs from that of Android. As such, we had to experiment multiple times by slowly changing the styling of the listings in order to make it as similar as possible. Additionally, many phones have different widths and length which might cause the layout to be different from each phone, hence we have to take that into consideration as well when styling in page.

8.2.3 Diagrams



Listing screen (left) and Details of listing (right)



Pop up to filter by category

8.3 Uploading of listings

The upload listing page on our application allows users to input information such as name, expiry, pick up location, description and category which they would like to see reflected on the listings page for all users to see.

Users are also able to preview the image they selected before uploading it to our database to see how it would be shown on the listing page.

The users will not be able to click on the “Upload Listing” button if they have not fully provided all the necessary details in the form. The missing portions not filled in will be indicated in red under the “Upload Listing” button. The button will also be grayed out to show that it is disabled if the form is not fully completed. Once all listing details have been filled in correctly, the button will be enabled and will turn purple to indicate that the listing is ready to be uploaded.

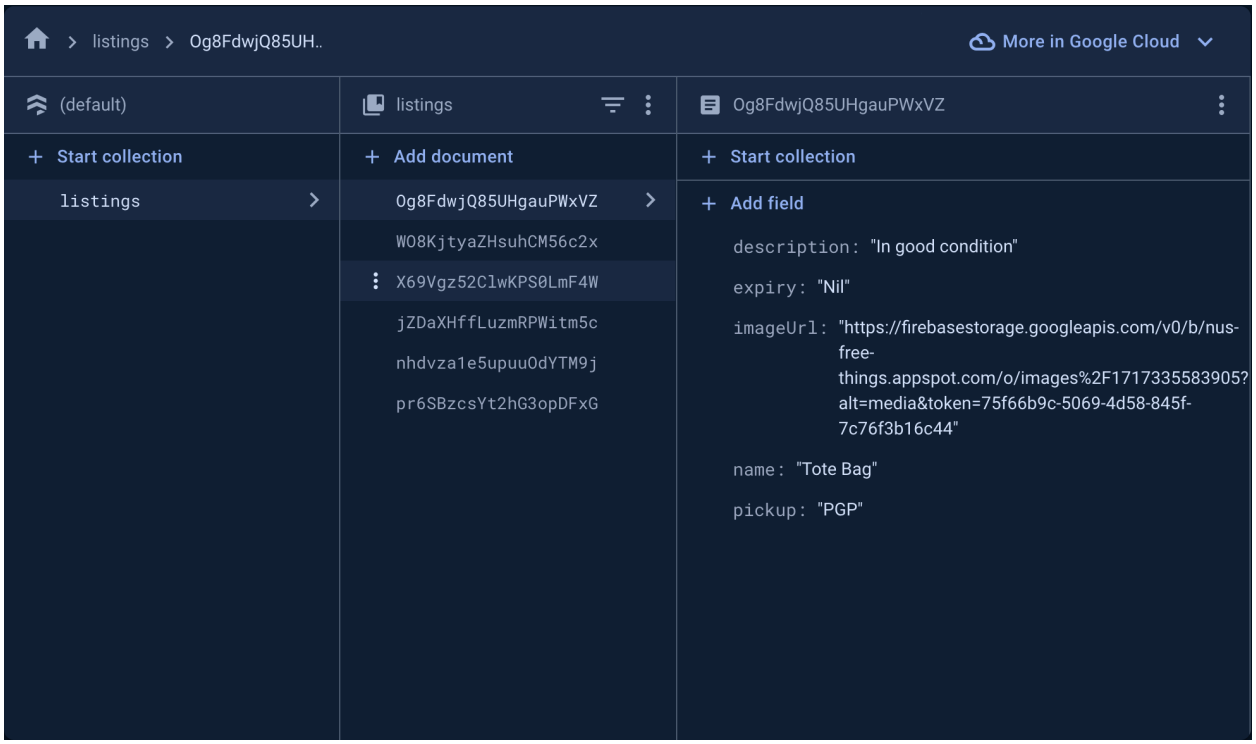
Once the “Upload Listing” button is pressed, a screen which says “Listing uploaded successfully” will be shown if the upload was successful.

8.3.1 Implementation

The page makes use of the ImagePicker library from expo-image-picker which allows the application to pick an image from the device’s media library.

The application first requests the user for permission to access their media library and once approved, users are to pick an image from their media library which will be stored on a variable named file through the useState function. We set a conditional such that if the file variable is not empty, the image will be displayed on the screen for users to see.

Information such as name, expiry, location and description will be stored in the Firebase Database. Images will be stored separately under Firebase Storage and the URL of the image in the storage will be used to display the images on the listings page.

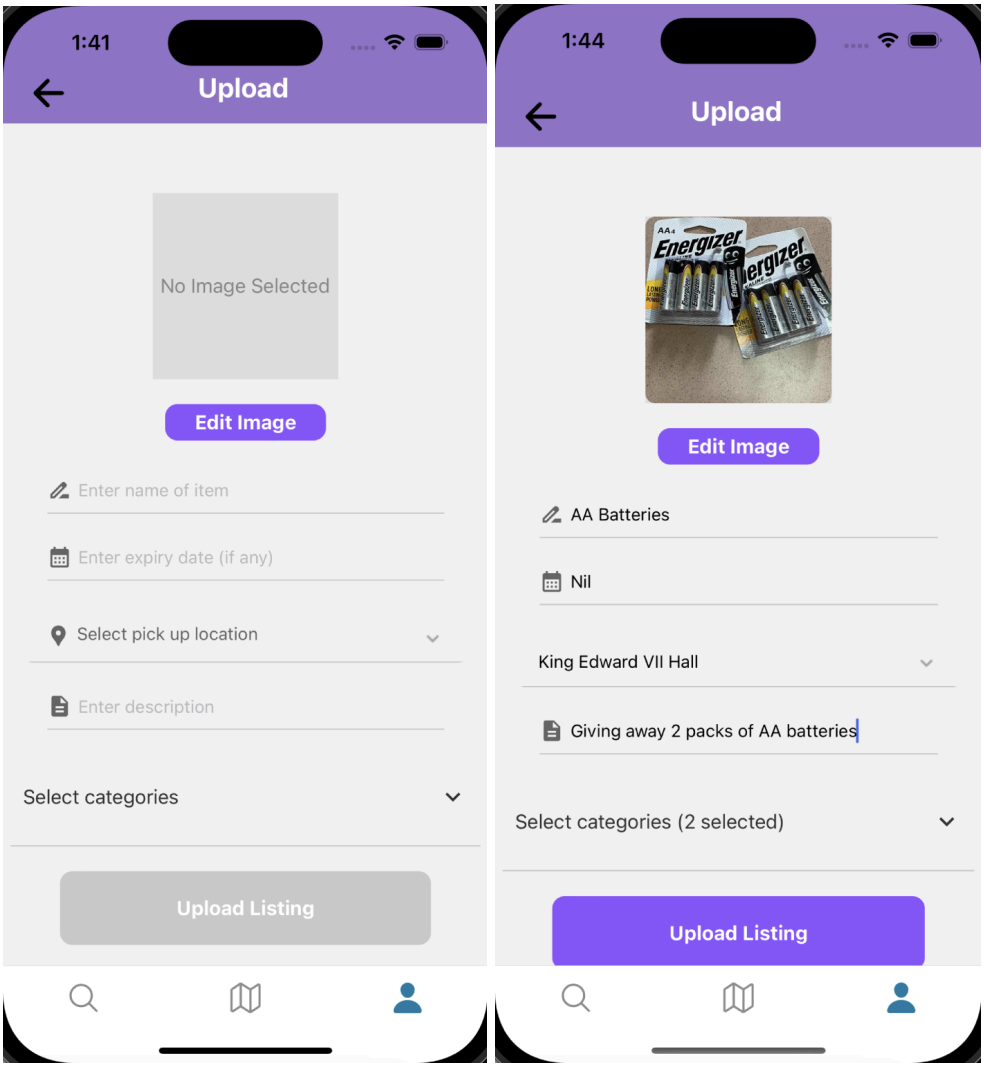


Firebase Database entries

8.3.2 Challenges Faced

One of the challenges faced was that the image uploaded was not reflected on the listing page. We later found that when using firestore, the uri of the image stored is the local uri of the image uploaded by the user. As such, we had to use firebase storage instead to ensure that the image uploaded would be seen by all users.

8.3.3 Diagrams



Uploading of listing page

8.4 Chatroom

The chatroom allows users to communicate with the listing owners. Users will be able to arrange when to pick up the item as well as enabling the dealers to provide additional information if needed.

Users can visit their chat history to see all existing chatrooms that they have created. The chatroom shows the listing image and name at the header of the chatroom, along with the email of the listing owner. Refer to Section 8.4.3 to view the sample images.

8.4.1 Implementation

We created a collection “chatrooms” on Firestore Database which will store individual chatrooms as documents that contain the following fields:

- buyer: Contains the user ID of the user interested in the listing
- owner: Contains the user ID of the listing owner
- listingId: Contains the listing ID
- listingName: Contains the name of the listing
- messages: A subcollection which contains documents that represent individual messages in that chatroom. Each document contains the following fields:
    - sender: Contains the email address of the user that sent the message
    - text: Contains the text message that the user has typed
    - timestamp: Contains the date and time of the message that was sent using serverTimestamp();

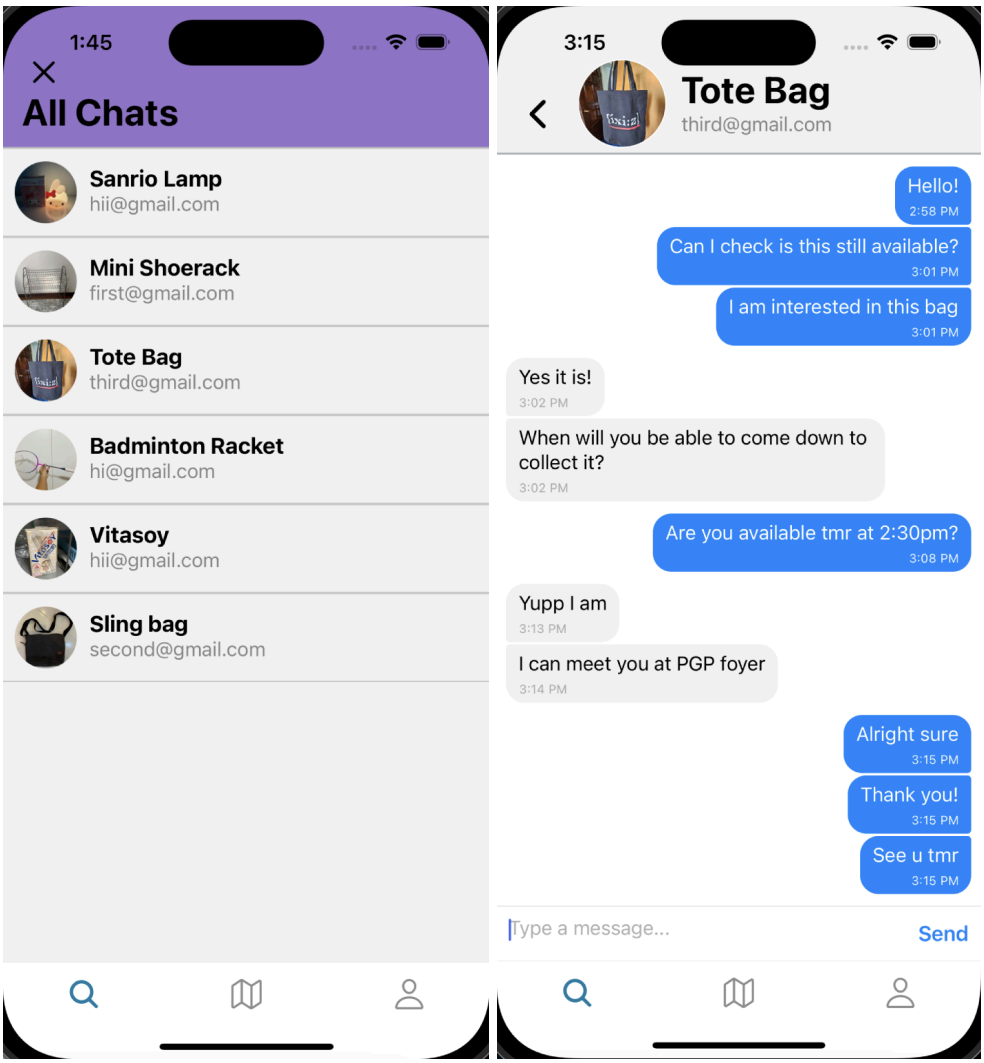
Using this database structure, we can store all necessary information of all the chatrooms that exist on our platform. In order to start a chat with the owner of the listing, a user can click on the “Chat” button at the bottom of the specific listing where a chatroom will be created if it does not already exist. The app will search through the database to find a chatroom with the listing ID of that specific listing and check if a chat exists between the listing owner and the user. Under the chat history, the app will query the database

to find chatrooms where the buyer or the owner field contains the same user ID as the user and display all individual chats in a list. Thus the chat history will be able to display all chatrooms that the user is involved in on a single page. The user can then click on the individual chats to bring them into that chatroom.

8.4.2 Challenges Faced

One challenge that was faced was coming up with the database structure for implementing the chatrooms. We realized that we need to store a lot of data just for one chatroom, and furthermore we also needed to store data for all the messages and message details inside each of the chatrooms. Thus we have decided to come up with our current database structure which is able to cover and store all of the data that we need in an easy to read manner.

8.4.3 Diagrams



Chat history page (left) and Chatroom page (right)

8.5 Listing Map

The listing map will be an additional tab on the homescreen which primarily functions as a map of listings to allow users to see listings near the user. All the listings on our platform will be tagged to specific pick up locations and will be pinned on this map for ease of viewing and for the user's convenience. The user can scroll around to view all the available listings and their locations. When a specific pin is clicked, it will display a list of all the listings that are tagged to that location. Users can click on the listing to bring them to the detailed listing page.

8.5.1 Implementation

Since our app is meant for NUS only, the map is also implemented with boundaries so that the user cannot scroll too far away from the NUS campus as rendering locations that are not in NUS is redundant. This is done by setting a minimum and maximum latitude and longitude, and ensuring that when the user navigates around the map, the latitude and longitude do not exceed these limits.

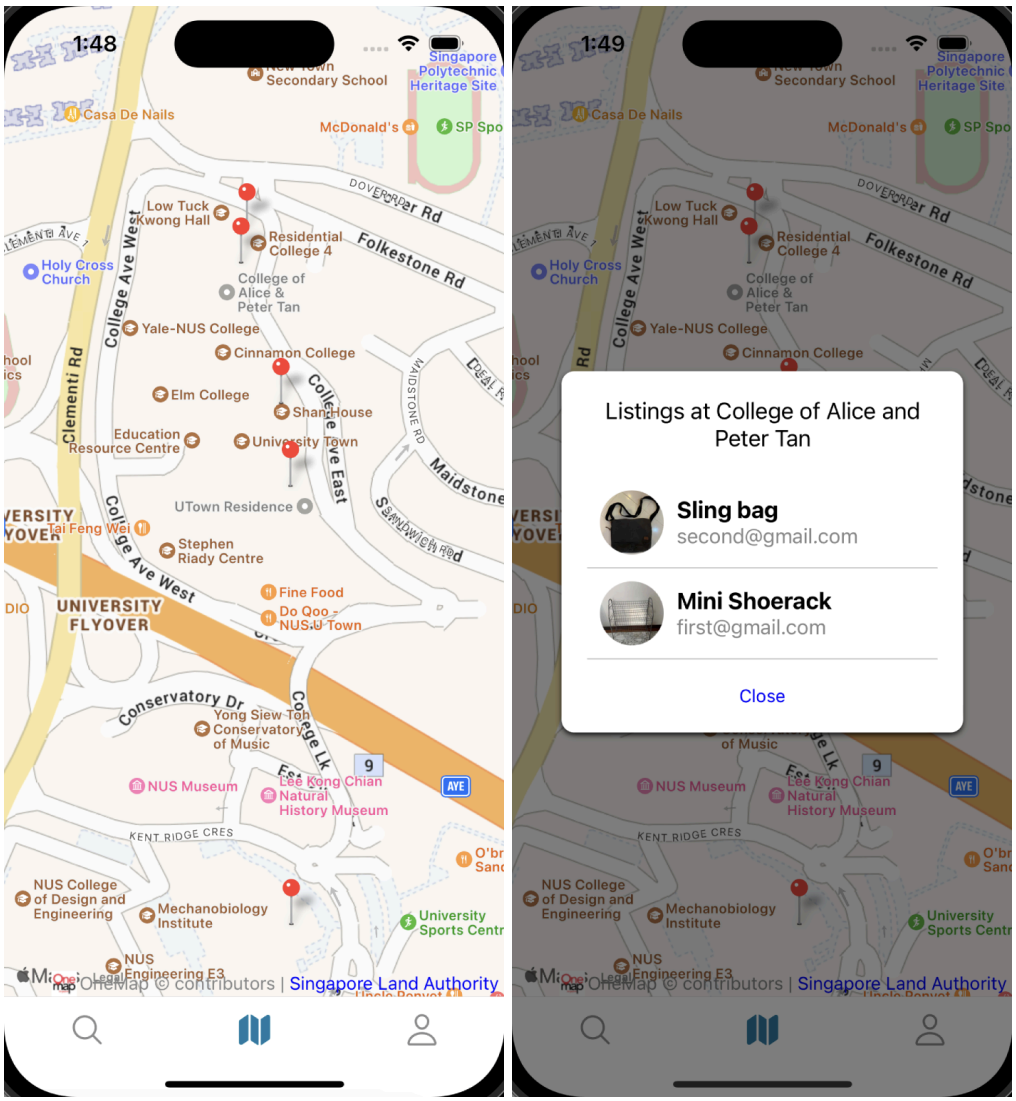
The app queries the database for all the listings and checks their pick up location. It then checks the "locations" collection in Firestore Database for the latitude and longitude of that location, which is then used to place a marker at that specific coordinates on the map. All listings that have the same location will be under the same marker. Once the marker is clicked, a list will be displayed to show all the listings that are tagged under that location.

8.5.2 Challenges Faced

It was difficult to find a way to plot the listings on the map as the map required latitude and longitude values. However, the user does not key in latitude and longitude values when they upload their listings, thus these values cannot be obtained from the listing data in the database. Thus we created a new collection "locations" to store all the latitude and longitude values of all the specific locations in NUS. The user will then select their pick up location from a dropdown list when they upload their listing, where

each location on the dropdown has their latitude and longitude values already stored in the “locations” collection. This way the app is able to obtain the coordinates of the locations of all the listings to plot the markers on the map.

8.5.3 Diagrams



Listing Map

8.6 Profile

The profile page serves as a page to store and display all of the users information such as their email and the listings they have uploaded. It allows users to view the items they have listed onto the platform as well as delete the listings once the item has been given away or expired. The profile page allows users to navigate to the upload listings page and also allow them to log out once the respective icons have been clicked.

8.6.1 Implementation

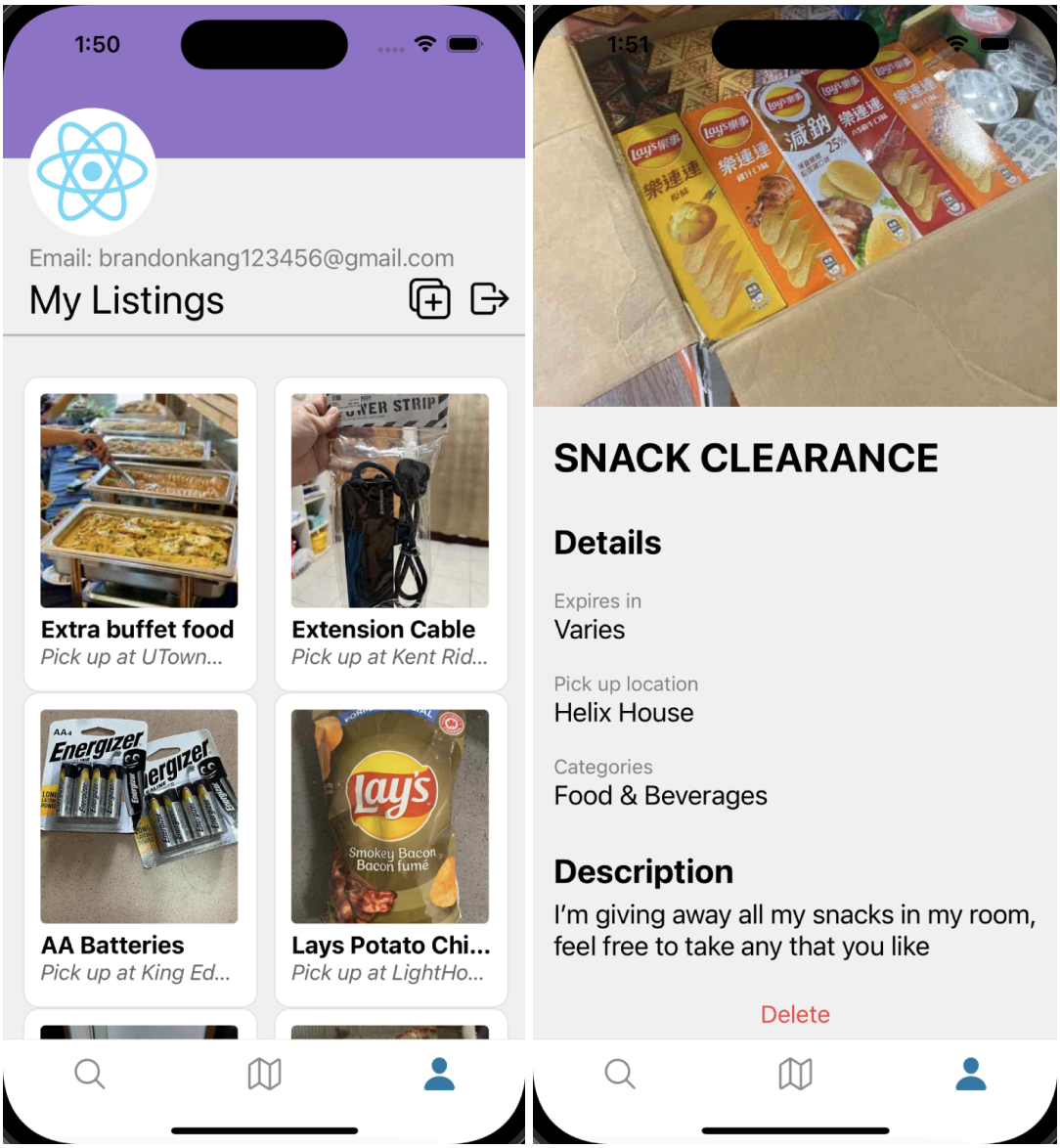
Firstly, the profile page makes use of react native navigation to allow users to navigate to the upload listing page once the upload icon has been clicked. We also used firebase's sign out function to handle users signing out of our application. Once the user has been signed out, onAuthStateChanged will handle the navigation and navigate the user to the sign in screen.

As each listing is tagged to the email of the user which uploaded the item, we are able to easily display all of the users listings by filtering the listings we currently have in our database with the user's email address. If the listing email is not the current user's email, we do not display the listing in the profile page. When a listing is clicked, the application will navigate the user to the listings information screen where information such as pickup location, description and expiry will be displayed. Additionally, if the listing is uploaded by the user, they are able to delete the listing by clicking onto the red delete button at the bottom of the screen. The delete feature is implemented through the use of firebase's deleteDoc function.

8.6.2 Challenges Faced

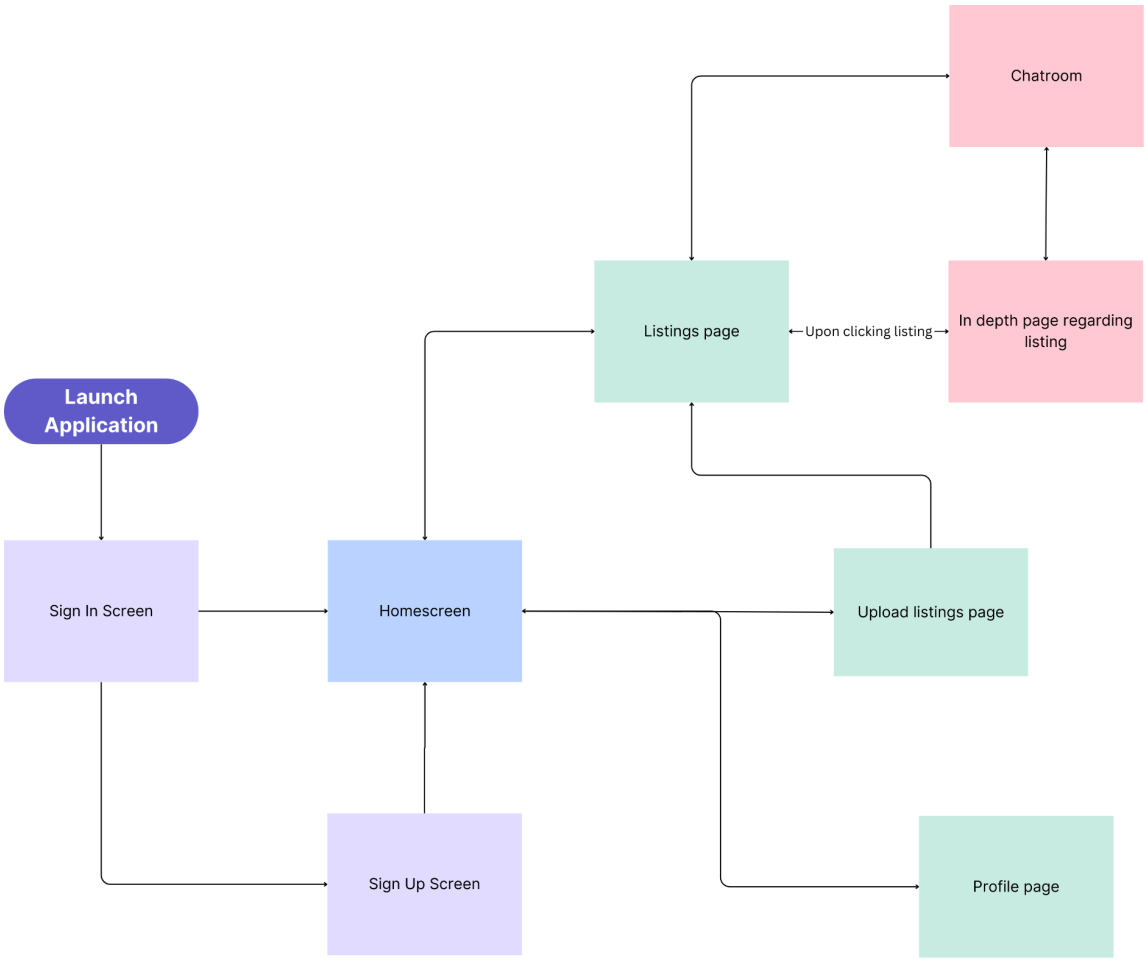
One of the challenges faced when implementing this feature was the inability to change the users profile picture to whatever they want. As doing such would require us to link the image to each user, it was very tedious and gave us a lot of bugs. As such, we have chosen to leave this feature out for milestone 2. In milestone 3, we will experiment with ways to implement this feature.

8.6.3 Diagrams

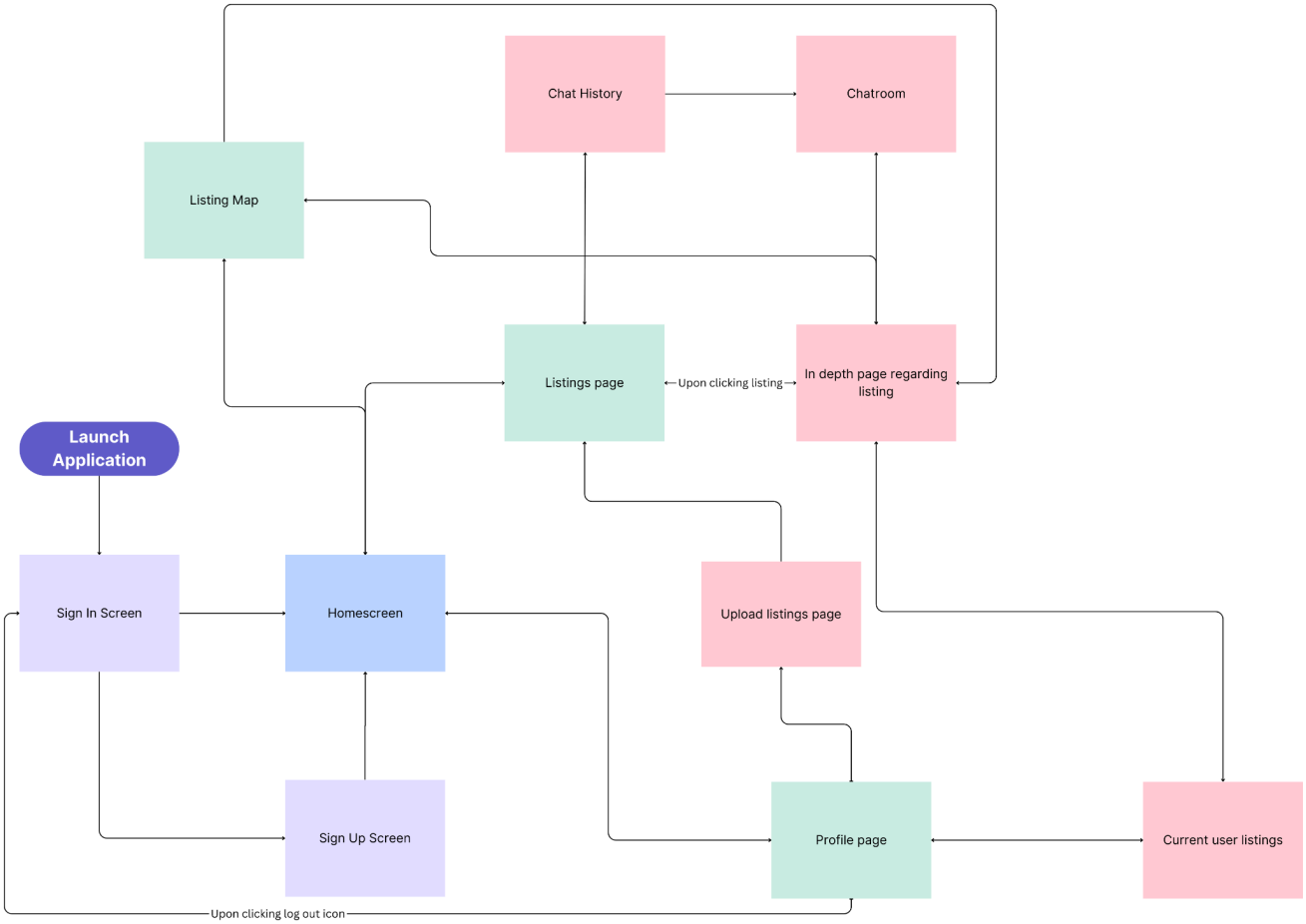


Profile page (left) and profile details of listings(right)

9 | Overall Navigation Flow



Milestone 1 navigation flow



Milestone 2 Navigation Flow

10 | Timeline and Development Plan

S/N	Tasks	Description	In-Charge	Date
1	Finalse Ideas	Drawing up of mockups and design for implementation of our frontend Testing out different ideas of design using figma	Brandon Sean	13 May - 18 May
2	User account authentication	Linking the project with Firebase Authentication	Brandon	18 May - 24 May
		Do up sign in/sign up screen	Sean	
3	Listings	Linking up the listings information with firebase firestore	Brandon	24 May - 2 June
		Do up the listings main page as well the the uploading of listings page	Sean	
Evaluation Milestone 1				3 June

4	Profile Page	Link up the profile page to Firebase Database	Brandon	3 June - 7 June
		Do up the profile page	Sean	
5	Chat Room	Link up the chat rooms with firebase's firestore database	Brandon	7 June - 18 June
		Do up the chat log screen as well individual chat room screen	Sean	
6	Listing Map	Use OneMap API to link out the map with our project	Brandon	18 June - 28 June
		Do up the listing map tab page	Sean	
Milestone 2				1 July
7	Quality testing	Create test cases, automated and manual for our application and run them	Brandon Sean	Milestone 3
Milestone 3				29 July

11 | Software Engineering Practices

11.1 Version Control

We use Git for version control and for managing our code. Pull requests are used when a new feature is implemented and wants to be merged with the main code. The developer that is working on a new feature creates a pull request to the partner who will be the code reviewer to look through his/her code to verify before merging it with the main branch. This process ensures communication and helps prevent incorrect resolution of merge conflicts that could introduce bugs. The pull request is only completed successfully after the code is reviewed, approved, and all merge conflicts are resolved.

12 | Quality Control

12.1 Functional Testing

Unit Testing

Unit testing is a software testing method where individual units or components of a software application are tested in isolation from the rest of the system. For our app, the following are the results of our unit testing.

Unit Test					Results	
Test ID	User Story	Testing Objective	Steps Taken	Expected Results	Pass/ Fail	Date Tested
1	As a new user, I want to create a new account	Test that the email address enter is a valid email	1. Saves the current email input into a react state 2. Pass the value into firebase's createrUserWithEmailAndPassword function	createrUserWithEmailAndPassword function does not throw an error	Pass	25/08/2024
		Test the error message for an invalid email	1. Saves the current email input into a react state 2. Pass the value	Expects an error message of "Sign Up Error Firebase: Error	Pass	25/08/2024

			into firebase's createrUserWithEmailAndPassword function	(auth/invalid-email)"		
		Test the error message for an weak password (less than 6 characters)	1. Saves the current password input into a react state 2. Pass the value into firebase's createrUserWithEmailAndPassword function	Expects an error message of "Sign Up Error Firebase: Password should be at least 6 characters (auth/weak-password)	Pass	25/08/2024
		Test that the password entered is a strong password (at least 6 characters)	1. Saves the current password input into a react state 2. Pass the value into firebase's createrUserWithEmailAndPassword function	createrUserWithEmailAndPassword function does not throw an error	Pass	25/08/2024
		Test that the verification link is sent to the email address inputted	1. User information such as email is saved under a user object 2. Pass the value of user into Firebase's sendEmailVerification function	Expects an email with the verification link sent to the Email address used during sign up	Pass	25/08/2024
		Test that the alert message is shown to users after clicking the sign up button	1. User information such as email is saved under a user object 2. Pass the value of user into Firebase's sendEmailVerification function 3. Waits for the verification Email to be sent out	Expects an alert message of "Verification Email Sent. A verification email has been sent to your email address. Please verify your email before logging in."	Pass	25/08/2024

			4. An alert is sent			
		Test that the verification link is working	1. User.emailVerified is set to false 2. Listener awaits the link to be clicked	Expects User.emailVerified to be set to true after the verification link is clicked	Pass	25/08/2024
2	As an existing user, I want to log in with my email and password	Test the error message for an unverified email	1. User information such as email and password is saved under a user object 2. Checks the user.emailVerified field if it is false	Expects an error message of “Email Not Verified. Please verify your email address before signing in.” to be thrown	Pass	25/08/2024
		Test that the Email and password inputted are valid	1. User information such as email and password is saved under a user object 2. Pass the user object into Firebase’s signInWithEmailAndPassword function 3. Return a true if email and password is valid, false otherwise	Expects that the screen will be navigated to the main listings screen	Pass	25/08/2024
		Test the error message for an invalid email or password	1. User information such as email and password is saved under a user object 2. Pass the user object into Firebase’s signInWithEmailAndPassword function 3. Return a true if email is valid,	Expects an error message of “Sign In Error. Firebase: Error (auth/invalid-credential)”	Pass	25/08/2024

			false otherwise			
3	As an existing User, i want to reset my password if I have forgotten the password for my account	Test the error message for an invalid Email	1. User information such as email is saved under a user object 2. Firebase database will be queried with the inputted Email 3. Return a true if email is in Firebase's database, false otherwise	Expects an error message of "Error. This email is not registered."	Pass	25/08/2024
		Test that the password reset link is sent to the email	1. User information such as email is saved under a user object 2. Sends a password reset link using Firebase's sendPasswordResetEmail function	Expects a password reset link is sent to the inputted Email	Pass	25/08/2024
		Test that the password is reset after submitting the password reset form	1. User information such as password is saved under a user object	Expects the new password is updated in Firebase's Firebase and the user is now able to log in using the new password	Pass	25/08/2024
4	As a user who wants to look for a specific item, I want to search for items being listed on the application	Test that search bar correctly filters the listings	1. The value typed in the search bar is stored in a variable named Search 2. The listings is filter with whether the listing name contains the value in Search	Expects the listings to be filtered accordingly to the value in the search bar	Pass	25/08/2024

		Test that the listing is able to be filtered according to categories	1. The categories selected in the filter is saved to a variable named selected categories 2. The listings is filter with whether the listing filters contain the categories selected in the filter	Expects the listings to be filtered accordingly to the categories selected	Pass	25/08/2024
5	As a user who likes a specific item, I want to be able to find out more information about the item	Test that the listing is navigated to the listing information page when it is clicked	1. Each listing is a touchable opacity object and can be clicked	Expects the screen navigate to the listing information screen when the listing is clicked	Pass	25/08/2024
6	As a user who likes a specific item, I want to be able to be able to chat with the user which listed the item	Test that the messages are able to be sent in the chatroom	1. Firebase's database is queried if the chatroomID is valid 2. If it is not valid it will create a chatroom with that specific chatroomID 3. Message sent to the chatroom are saved to a state variable 4. Messages are saved to Firebase's database	Expect messages to be displayed in the chatroom whenever a message is sent	Pass	25/08/2024
		Test that the messages are stored in the chatroom and be viewed at anytime	1. Firebase's database is queried if the chatroomID is valid 2. Firebase's database is queries to get the chatroom's messages	Expects that the messages will remain in the chatroom even after leaving the screen or the application	Pass	25/08/2024

			3. The messages are saved to a variable			
7	As a user who does not need an item anymore, I would like to list an item on the application	Test that the listing cannot be uploaded if all the required fields are not filled	1. Each listing field have a state which stores the input in that field 2. If any of the field is empty, it will set a error state which contains all the errors	Expects that the errors messages will appear on the application with each error message indicating which field is empty	Pass	25/08/2024
		Test that the listing image can be uploaded and have an upload view	1. Permission is request to the application to access the images in the phone's camera roll through the requestMediaLibraryPermissionsAsync function 2. Application waits for an image to be picked 3. If an image is picked, the image state is set to the image uri	Expects that the preview image is shown on the screen	Pass	25/08/2024
		Test that the listing has been successfully uploaded	1. Values inputted in the respective fields are saved to the respective variables 2. All the values are then saved to Firebase's database	Expects a screen to be shown saying "Listing uploaded successfully" after all the data has been uploaded to Firebase's database	Pass	25/08/2024
8	As a user who have listed a lot of items in the application, I would like to view all the listings i have	Test that the uploaded listings are reflected on the profile page	1. All the listings are queried from Firebase's database 2. Listings are	Expects the profile page to only display the listings uploaded by the user	Pass	25/08/2024

	uploaded		filtered by the listings user's email			
		Test that the uploaded listings are able to be deleted	1. The application will store the current user email into a variable 2. The user which uploaded the listing is compared to the current user 3. The delete button is shown if the user matches 4. A delete request is sent to firebase	Expects all the associated chatrooms with the listings and the listings to be deleted	Pass	25/08/2024
9	As a user who wants to know which items are available for free in my area, I would like to map to see all the current available item	Test that the map can be moved and that user are not able to move out of the map boundary	1. Map is rendered using react native maps 2. Map boundaries are preset with coordinates 3. Current coordinates are saved whenever the map is moved 4. Current boundaries is compared with the map boundaries	Expects that the map is moved back to the map boundaries if the current boundaries exceeds the map boundaries	Pass	25/08/2024
		Test that there are pins on the location which the item's pick up location are at	1. Each pick up location is tagged with a geo coordinate 2. When a listing is uploaded onto the application, a pin will be placed	Expects that a red pin is placed on the pick up location respective location	Pass	25/08/2024
		Test that the items can be viewed on the map	1. Each listing is tagged to a pick up location 2. A clickable pin a placed onto a	Expects all the listings listed at the pick up location is displayed if the	Pass	25/08/2024

			map where is listing pick up location is at	pin is clicked		
10	As a user, I would like to be able to log out of the application	Test that the logout button is working	1. A variable is declared whether the current user is signed in. If the variable is true it will display the listings page 2. Variable will be set to false if the log out button is clicked	Expects the screen will be navigated to the login screen once the logout button is clicked	Pass	25/08/2024

Integration Testing

Integration testing is used to simulate user interactions in the app. We have identified the following interactions to conduct integration testing on:

1. Sign up
2. Sign in
3. View listings (from listings page)
4. View listings (from listings map)
5. Upload listings
6. Delete listings
7. Send messages in the chatroom
8. Reset password

Integration Test					Results	
Test ID	User Story	Testing Objective	Steps Taken	Expected Results	Pass/ Fail	Date Tested
1	As a new user, I want to create a new account	Test the ability to create a new user account.	1. Launch the app 2. Click on the “Sign Up” button 3. Enter an email address to sign up with in the email field	Users are able to create a new account	Pass	25/08/2024

			4. Enter a password in the password field 5. Click the “Register” button 6. Log in to the email address entered in step 3 7. Click on the email sent by the app. 8. Click on the verification link in the email. 9. Go back to the app to sign in			
2	As a user with an existing account, I want to log in to the app	Test the ability to sign in to the app.	1. Launch the app 2. Enter the email address of the account in the email field. 3. Enter password of the account in the password field. 4. Click the “Sign in” button	Users are able to sign in to the app	Pass	25/08/2024
3	As a user, I want to create a listing to upload on the app	Test the ability to upload a listing on the app for other users to view.	1. Launch the app and sign in 2. Click on the profile icon on the navigation bar 3. Click on the upload icon 4. Click on the “Edit Image” button and select the image of the item to be uploaded as a listing 5. Enter the name, expiry date, pick up location and description of the item in their respective fields 6. Select the category of the item from the “Select Categories” dropdown. 7. Click on the	Users are able to upload listings.	Pass	25/08/2024

			“Upload Listing” button			
4	As a user, I want to view available listings on the app	Test the ability to browse through other user’s listings	1. Launch the app and sign in 2. Click on the search icon on the navigation bar 3. Click on the filter icon 4. Select the categories to be filtered by.and click “Confirm” 5. Enter on the search bar to search for listings by the listing name 6. Scroll through the page to view other user’s listings according to the filter 7. Click on the desired listing	Users are able to view other user’s listings	Pass	25/08/2024
5	As a user, I want to view available listings that are near me using a map	Test the ability to view other user’s listings from a map	1. Launch the app and sign in 2. Click on the map icon on the navigation bar 3. Scroll through the map to find the desired location 4. Click on the pin on that location 5. Scroll through the list of listings 6. Click on the desired listing	Users are able to view and search for other user’s listings using a map	Pass	25/08/2024
6	As a user, I want to delete my listing that I have already given away	Test the ability to delete a current listing	1. Launch the app and sign in 2. Click on the profile icon on the navigation bar 3. Scroll and click the desired listing to delete 4. Scroll down and click the “Delete”	Users are able to delete their listings	Pass	25/08/2024

			button 5. Click “Delete” when prompted to confirm deletion			
7	As a user, I want to communicate with a dealer on a listing that I am interested in	Test the ability to send messages in the chatroom with another user.	1. Launch the app and sign in 2. Click on the search icon on the navigation bar 3. Scroll through listings page to find the desired listing 4. Click on the listing 5. Scroll down and click the ”Chat” button 6. Send a message to the owner of the listing 7. Click on the search icon on the navigation bar again 8. Click on the chat icon on the top right 9. Select the desired chat to continue chatting	Users are able to use the chatroom to send messages to other users.	Pass	25/08/2024
8	As a user who forgot his password, I want to be able to reset my password so that I can log in again	Test the ability to reset the password for an existing user’s account.	1. Launch the app 2. Click on the “Forgot Password?” button 3. Enter the email address of the account with the forgotten password 4. Click on the “Reset Password” button 5. Log in to the email address entered in step 3 6. Click on the email sent by the app. 7. Click on the link in the email. 8. Enter a new password and save 9. Go back to the	Users are able to reset their password	Pass	25/08/2024

			app sign in screen 10. Sign in using the new password			

12.2 User Testing

User testing is a technique used to evaluate a software application by testing it with real users. The main goal is to assess how easy and intuitive the software is to use, and to identify any usability problems, collect qualitative and quantitative data, and determine the user's satisfaction with the product. This process helps ensure that the final product is user-friendly and meets the needs of its intended audience.

We employed Donald A. Norman's seven fundamental design principles to test and understand user interactions with the app. Briefly, the principles are:

- **Discoverability**: Helps users understand where to perform actions.
- **Conceptual Models**: Provide simple explanations of how something works.
- **Affordance**: Refers to the perceived properties of an object.
- **Mappings**: Define the relationships between controls and their effects.
- **Signifiers**: Indicate exactly where users should perform an action.
- **Constraints**: Limit the interactions that can occur.
- **Feedback**: Communicates the response to a user's action.

Testers were given specific tasks and started with a cognitive walkthrough of the Home Page. Their actions and thought processes were recorded to provide valuable insights for improving the app and enhancing the overall user experience.

User Story	Task	Discovery	Conceptual Models, Affordance & Mappings	Signifiers and Constraints	Feedback
As a new user, I would want to	Guide the user to enter	1. Enter an email	Conceptual model: Tester understood that	Signifiers: Tester is able to identify which	After verification, the user is

experience a simple and easy to understand onboarding flow.	the necessary information in order to create an account	address and password 2. Log in to the email address provided and click on the email verification link that was sent to the tester to verify their email address.	he had to provide his email address and password in order to create an account. Good. Affordance: Tester expects the relevant fields for entering the information required from him. Good. Mapping: The onboarding flow is one-directional and sequential, so no important steps will be missed out. Good.	text box requires what information from the tester. Can identify that he is supposed to enter his email address and click on the "Register" button to create an account. The alert that pops up on the screen clearly indicates to the tester to check his email for the verification link. The email contains the instruction to click on the link to verify the email before successfully creating the account. Good. Constraints: The tester will be prompted with an alert if he tries to sign in without verifying his email. Good.	provided with instructions to return to the app to sign in with their new verified account. Users are redirected to the home page after they have successfully signed in.
As a user, I want to be able to recover my password if I forget my password.	Guide the user on the procedures to reset his password.	1. Click on the "Forget Password" button 2. Enter the email address of the account with the lost password	Conceptual Model: Tester understood that he had to provide his email address in order to reset his password. Good. Affordance: Tester expects a text box where he can input the email address of his account. Good. Mapping: The flow for resetting password is one-directional and sequential, so no important steps will be missed out. Good.	Signifiers: Tester can identify that the text input requires him to input his email address. Can identify that he is supposed to click the "Reset Password" button. The alert that pops up on the screen clearly indicates to the tester to check his email for the link to reset his password. The link redirects the tester to a page which allows the user to enter his new password. Good.	After entering the new password, the user is informed that the password has been updated and may now sign in using the new password. Users are redirected to the home page after they have successfully signed in.

				Constraints: Tester is prompted with an alert if he tries to click the “Reset Button” without enter his email. Good.	
As a student, I want to be able to give away items that I no longer use.	Create a listing	1. Navigate to the profile page where the upload listing icon is visible. 2. Click on the upload listing button. 3. Fill in the relevant information about the item that the user wishes to upload as a listing. 4. Click the button that says “Upload Listing”	Conceptual Model: Tester knew that he had to navigate to the profile page to upload a new listing. Good. Affordance: Tester expected a form to fill up the listing details to upload the listing. Good. Mapping: The icons in the navigation bar indeed bring the tester to their respective pages. The icon with a plus is mapped to creating and uploading a new listing. The form to fill up contains drop down menus which will present options for the tester to choose from. Good.	Signifiers: The upload listing icon does not provide enough perceived affordance. Tester did not know that the button was for uploading listings. Lacking. Constraints: “Upload Listing” button is grayed out and disabled if any of the fields are not filled in. Good.	Buttons were responsive and the user was informed which areas needed to be filled in before allowing the user to upload the listing. Users are redirected to a success page when the listing has been uploaded successfully.
As a dealer, I want to be able to delete my listing after I have given away the item.	Delete an existing listing	1. Navigate to the profile page where all of the user’s listings are visible. 2. Click on the listing to be deleted. 3. Click the “Delete” button at the bottom.	Conceptual Model: Tester knew that he had to navigate to the profile page to select a listing to delete. Good. Affordance: Tester expects a button after clicking on the listing that lets him delete the listing. Good. Mapping: Clicking on the listing indeed brings them to a more detailed page of the listing.	Signifiers: The tester can identify that the “Delete” button is responsible for deleting the listing. The tester is also aware that he has to click on the listing to find the “Delete” button. Good. Constraints: “Delete” button only found on listings that belong to the tester. Tester is only able to delete	User is alerted with a warning to confirm that they want to delete the listing and all associated chatrooms with that listing will be deleted. The listing cannot be found on the listing page or on the user’s profile page after the

		4. Click confirm to delete the listing.	Clicking the “Delete” button will indeed remove the listing from the app. Good.	his own listings and not listings by other users. Good.	listing has been deleted. Chatrooms associated to the listing will also no longer be found.
As a student, I want to be able to save money by finding items that others no longer want instead of buying new ones.	Search for listings using the listings page	1. Navigate to the listings page where all uploaded listings are visible. 2. Scroll through the listing page 3. Enter in the search bar the name of an item the user is interested in. 4. Click on the filter icon. 5. Select the relevant category to filter by.	Conceptual Model: Tester knew that they had to navigate to the listings page to browse through listings. Good. Affordance: Tester expected a page to scroll through listings uploaded by other users. Tester also expected a way to search for specific listings. Good. Mapping: The main page scrolls according to how the tester scrolls. The search bar lets the tester enter names of listings. The filter icon brings up a pop up that allows the tester to select the categories to filter by. Good.	Signifiers: The filter icon does not provide enough perceived affordance. Tester did not know that the button was for filtering listings by their respective categories. Lacking. Constraints: Tester cannot view his own listing and deleted listings as both will not be displayed on the listing page. Good.	Searching or filtering will remove the listings on the listings page that do not meet the requirements specified by the user. When the user clicks on the listing that he is interested in, it will bring him to a detailed page about the listing.
As a student, I want to be able to browse whether there are free things that I might want in areas nearby.	Search for listings using map	1. Navigate to the map page where listings will be pinned on a map based on their location 2. Scroll through the map 3. Select a location that is pinned.	Conceptual Model: Tester knew that they had to navigate to the map page to use the map to look for listings. Good. Affordance: Tester expected a map where they could scroll around to look for listings in different areas. Good. Mapping: The map scrolls according to	Signifiers: Tester can identify that the pins can be clicked to show the listings in that area. Can identify that he is required to scroll around on the map to navigate. Good. Constraints: Tester is unable to scroll far away from the main NUS campus. Good.	Scrolling and the pins on the map are both responsive. Clicking on the pins will show all the available listings in that area. Clicking on a listing from that list will bring the user to the detailed page of that listing.

		4. Browse through available listings	how the user scrolls, and zooms in and out depending on how the tester drags. Pins are responsive in displaying the listings in that location when clicked by the tester. Good.		
As a student looking for items, I want to be able to discuss with the dealer certain details about the item or details about meetup timing and location.	Using the chatroom	1. Navigate to the listings page. 2. Search for a listing. 3. Click on the listing. 4. Click on the “Chat” button at the bottom of the page. 5. Send a message to the dealer. 6. Navigate to the listings page again. 7. Click on the chat icon to view all chatrooms that the user is involved in.	Conceptual Model: Tester knew that they had to navigate to the listings page to find the chatroom icon. Tester knew that he could send and receive messages from the dealer of a listing. Good. Affordance: Tester expected a button after clicking on a listing to start a chat with the dealer. Tester also expected a way to view all the chats with different dealers that he is involved in. Good. Mapping: The “Chat” button indeed brings the tester to the chatroom with the dealer. The chat icon also brings the tester to a page with the list of all active chatrooms that the tester is currently in. Messages sent by the tester indeed get sent to the dealer. Good.	Signifiers: Tester can identify that the “Chat” button on a listing will start a chatroom with the dealer of that listing. Tester can identify that the chat icon brings them to the chatrooms. Good. Constraints: No way to edit or delete the messages that are already sent. Unable to check for profanities or inappropriate language. Lacking.	Buttons were responsive. Messages sent by the user and the dealer will be displayed in the chat.

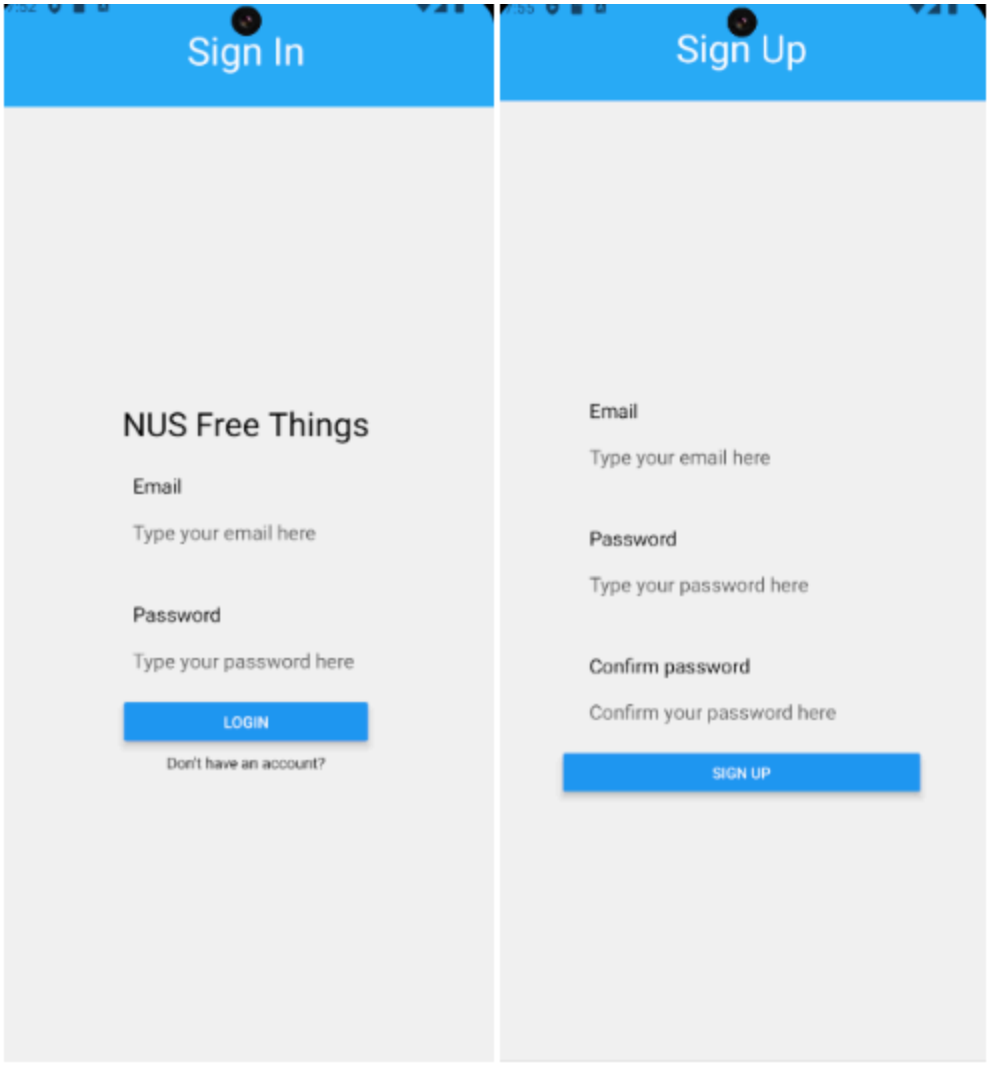
13 | Limitations

Quality control and safety is one of the limitations of our application. There is the potential for safety hazards, such as expired food items, damaged electronics, or faulty appliances, which could pose health risks or cause accidents. As the dealing of items is out of our control, it is difficult to check on whether or not items listed on our platform will pose a safety hazard. Additionally, there is the risk of liability for the university if an exchange results in harm, necessitating clear guidelines and disclaimers to mitigate legal risks. As we are not able to ensure the quality of all of the items listed on our platform, this is one of the limitations our application faces.

However, while not foolproof, we can introduce a rating system on our platform to allow users to rate other users after a deal has been done. This will discourage users from dealing with users with bad reviews which can help in reducing the chances of users getting bad quality items.

14 | Challenges Faced

While planning for the design of our app, the biggest challenge we faced was trying to come up with a user interface that is both user friendly and aesthetically pleasing. This was particularly challenging for us as neither of us have much experience in design. This was also our first time building a mobile app, which also made it harder since we had to spend many hours learning new concepts and apply them to our project. After a few attempts of trial and error, we settled with our current design for Milestone 1. Further improvements to the user interface may be carried out for future Milestones.



Sample of our initial design of the sign in page

15 | Tech Stack

1. React Native (Cross-platform mobile app development)
2. Firebase (Backend and Database)
3. Git and Github (Version Control)