Lab4 Deliverables: Requirements Analysis

for

Skyscape

Version 1.0 approved

Prepared by

LI YAOHAO,
LIM ZHI YONG,
JIN YING,
KIERAN LIOW,
KIERSTEN YEO SHU XIAN,
YOON SANG WON

NANYANG TECHNOLOGICAL UNIVERSITY TEAMWON

11 April 2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
LI YAOHAO	11/04/2024	First Draft	1.0.0

1. Black Box Testing

1.1. Login Functionalities

Test ID	Test Case Description	Input: Email	Input: Password	Expected Output	Actual Output	Pass/Fail
EC1	Verify successful login with valid email and password	sunsetlover72@gmail.com	SUNSETlover72	User is logged in successfully	User is logged in successfully	Pass
EC2	Verify error message when email is invalid	sunsetlover72	SUNSETlover72	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC3	Verify error message when password is invalid	sunsetlover72@gmail.com	sunsetlover72	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC4	Verify error message when email is unregistered	sunsetlover73@gmail.com	SUNSETlover73	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC5	Verify error message when email is empty		SUNSETlover72	Error message “Enter an email”	Error message “Enter an email”	Pass
EC6	Verify error message when password is empty	sunsetlover72@gmail.com		Error message “Enter a valid password”	Error message “Enter a valid password”	Pass

1.2. Register/Sign up Functionalities

Test ID	Test Case Description	Input: Username	Input: Email	Input: Password	Input: Confirm password	Expected Output	Actual Output	Pass/ Fail
EC1	Verify successful signup with valid information	sunsetlover74	sunsetlover74@gmail.com	SUNSETlover74	SUNSETlover74	User is signed up successfully and logged in	User is signed up successfully and logged in	Pass
EC2	Verify error message when username is already registered, while email is unregistered	sunsetlover72	sunsetlover75@gmail.com	SUNSETlover75	SUNSETlover75	Error message “Username is already taken. Please choose a different username”	Error message “Username is already taken. Please choose a different username”	Pass
EC3	Verify error message when email is already registered, while username is unregistered	sunsetlover75	sunsetlover74@gmail.com	SUNSETlover74	SUNSETlover74	Error message “Please supply a valid email”	Error message “Please supply a valid email”	Pass
EC4	Verify error message when password and confirm password do not match	sunsetlover75	sunsetlover75@gmail.com	SUNSETlover75	sunsetlover75	Error message “Passwords do not match”	Error message “Passwords do not match”	Pass
EC5	Verify error message when email is invalid	sunsetlover75	sunsetlover75	SUNSETlover75	SUNSETlover75	Error message “Please supply a	Error message “Please supply a	Pass

						valid email”	valid email”	
EC6	Verify error message when password does not meet requirements	sunsetlover75	sunsetlover75@gmail.com	sunset	sunset	Error message “Password must be at least 8 characters long and include at least one capital letter and one number”	Error message “Password must be at least 8 characters long and include at least one capital letter and one number”	Pass

1.3. Add location Functionalities

Test ID	Test Case Description	Input	Expected Output	Actual Output	Pass/Fail
EC1	Verify successful searching of location	Toa Payoh	Toa Payoh	Toa Payoh	Pass
EC2	Verify successful adding of location	Button press “+” next to location name (e.g. Toa Payoh)	Pop up message “Toa Payoh added to favourites”	Pop up message “Toa Payoh added to favourites”	Pass
EC3	Verify successful removal of location	Swipe left from saved location (e.g. Toa Payoh)	Pop up message “Toa Payoh removed”	Pop up message “Toa Payoh removed”	Pass
EC4	Verify error message when adding duplicate location	Button press “+” next to location that is already added (e.g. Toa Payoh)	Pop up message “Toa Payoh is already saved”	Pop up message “Toa Payoh is already saved”	Pass

EC5	Verify error message adding location not in database	Johor Bahru			Pass

1.4. Upload Feed Functionalities

Test ID	Test Case Description	Input	Expected Output	Actual Output	Pass/Fail
EC1	Verify successful uploading of picture (camera)	Button press “Take Picture”, picture taken from camera, button press “Upload”	Pop up message “Image uploaded successfully”	Pop up message “Image uploaded successfully”	Pass
EC2	Verify successful uploading of picture (choose from gallery)	Button press “Choose from gallery”, picture taken from files, button press “Upload”	Pop up message “Image uploaded successfully”	Pop up message “Image uploaded successfully”	Pass

1.5. Follow Users Functionalities

Test ID	Test Case Description	Input Username	Expected Output	Actual Output	Pass/Fail
EC1	Search for username that exists in the database and is currently not followed	“jy”	Searched user is shown and current user is able to follow that user	User is shown and a button exists to follow the user	Pass
EC2	Search for username that exists in the database and is currently followed	“sw”	Searched user is shown and current user is able to unfollow that user	User is shown and a button exists to unfollow the user	Pass
EC3	Search for username that does not exist in the database	“name”	Message shown to show that the user is not found	Message is displayed saying ‘No user found’	Pass
EC4	Search for username that exists in the database in a different case (upper, lower cases) and the changed username does not exist in the database	“SW”	Message show to show that the user is not found	Message is displayed saying ‘No user found’	Pass
EC5	Search with no inputs	“”	Show a list of followed users and allow the current user to unfollow them	List of users who are followed are shown and buttons exist for each user to unfollow them	Pass

1.6. Edit Profile Functionalities

Test ID	Test Case Description	Input Information	Expected Output	Actual Output	Pass/Fail
EC1	Change username to a username that does not exist in the database	Username = “new123”	Current username changes to the selected username	Username changes to the selected username and a message shows saying ‘Profile Updated’	Pass
EC2	Change username to a username that already exists in the database	Username = “sw”	Error message shows saying that username cannot be changed and username does not change	Error message is displayed saying that ‘Username already taken’ and username is not changed	Pass
EC3	Change password to a different password that meets the condition of having a length of more than 8, contains an uppercase letter, and has a number	Password = “Password123”	Current password changes to the selected password	Password changes to the selected password and a message shows saying ‘Profile Updated’	Pass
EC4	Change password to a different password that is shorter than 8	Password = “Pass1”	Error message shows saying that password cannot be	Error message is displayed saying that ‘Password is invalid’ and	Pass

	characters but has an uppercase letter and a number		changed and password does not change	password is not changed	
EC5	Change password to a different password that does not contain an uppercase letter but is longer than 8 characters and has a number	Password = “password1”	Error message shows saying that password cannot be changed and password does not change	Error message is displayed saying that ‘Password is invalid’ and password is not changed	Pass
EC6	Change password to a different password that does not contain a number but is longer than 8 characters and has an uppercase letter	Password = “Passwords”	Error message shows saying that password cannot be changed and password does not change	Error message is displayed saying that ‘Password is invalid’ and password is not changed	Pass
EC7	Key in different passwords into the New Password and Confirm Password tab	New password = “Pas sword1” and Confirm password = “Password2”	Error message shows saying that the passwords do not match and password does not change	Error is shown on the tab saying that ‘Passwords do not match’ and password is not changed	Pass
EC8	Change profile picture to a picture in the Photos Gallery and press Save Changes button		Profile picture changes to the selected photo	Profile picture is changed to the selected photo from the Photo Gallery	Pass

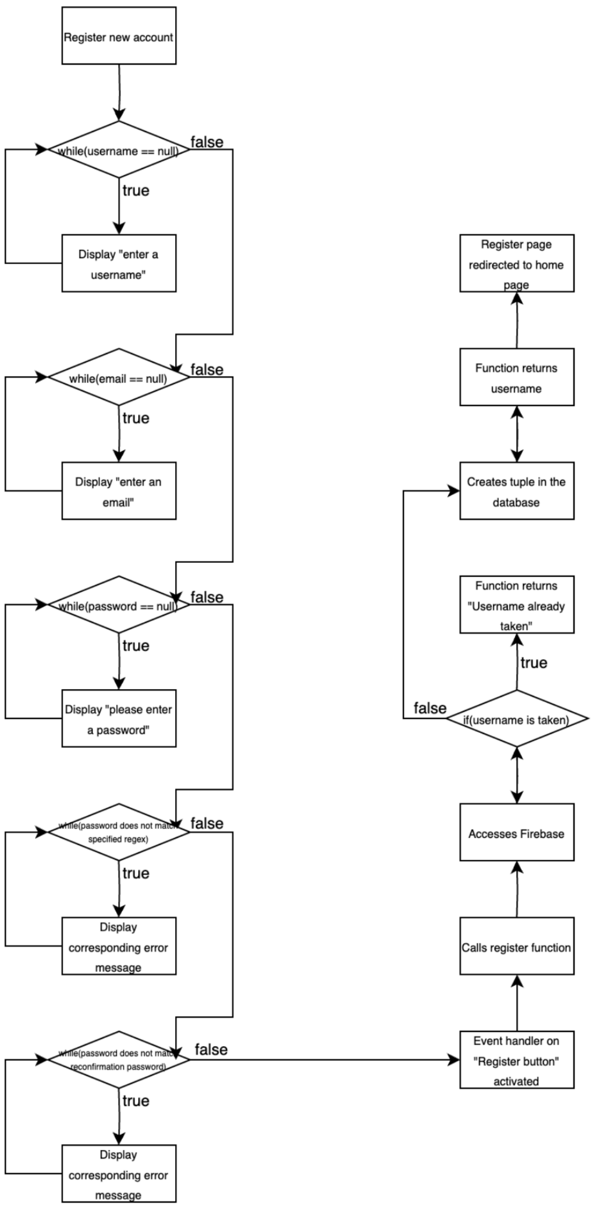
EC9	Change profile picture to a picture in the Photos Gallery and does not press Save Changes button		Profile picture reverts back to the old profile photo	Profile picture does not change to the selected photo and it reverts back to the original profile photo	Pass

1.7. Report Bugs Functionalities

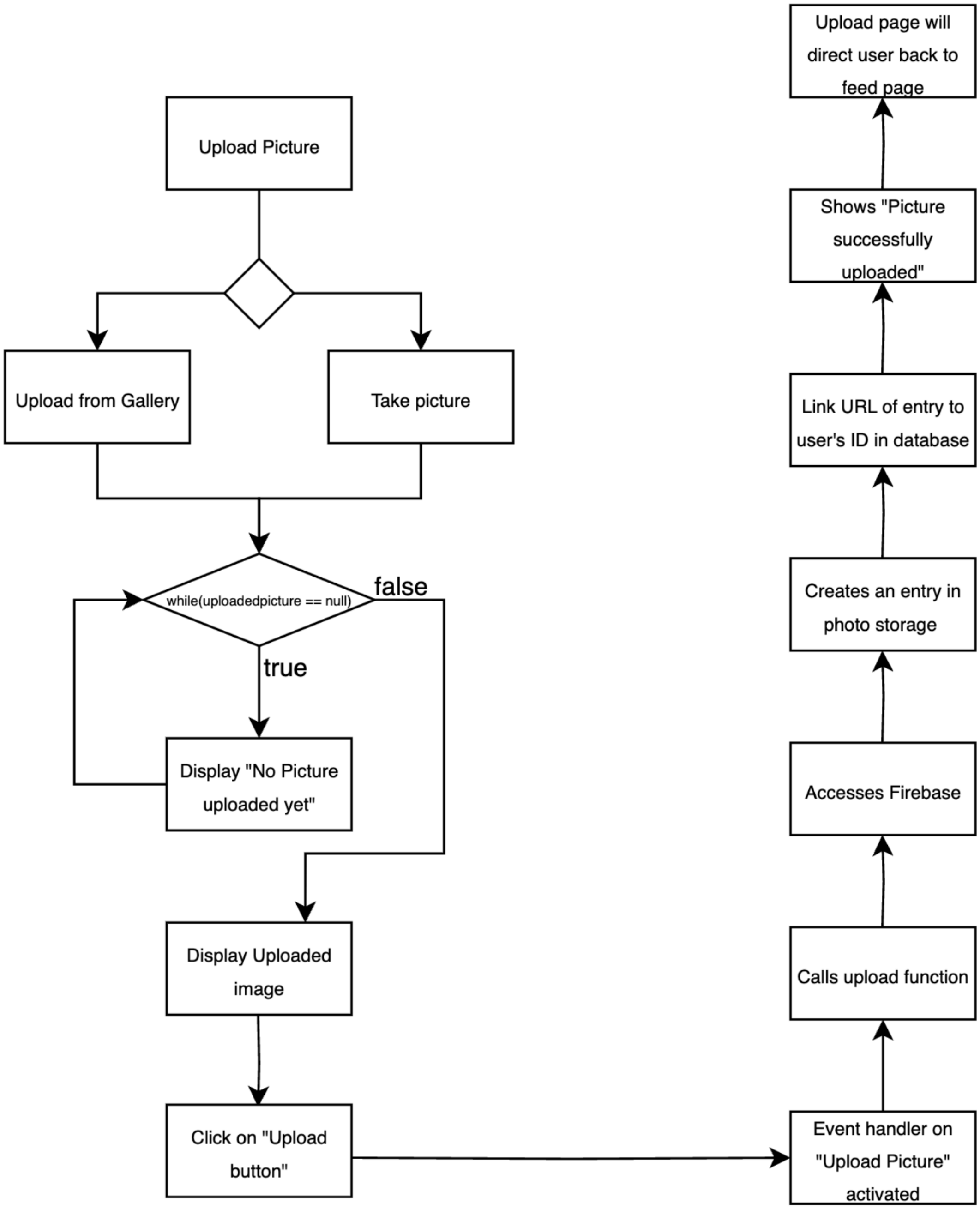
Test ID	Test Case Description	Input Information	Expected Output	Actual Output	Pass/Fail
EC1	File a bug report with a filled title and description	Title = “Hello” Description = “World”	Message showing that the report has been successfully made	Message is displayed saying that ‘Your bug report has been submitted successfully’	Pass
EC2	File a bug report with an empty title and filled description	Title = “” Description = “World”	Error message showing that title needs to be filled	Error message is displayed saying ‘Please fill in the title’	Pass
EC3	File a bug report with a filled title and empty description	Title = “Hello” Description = “”	Error message showing that description needs to be filled	Error message is displayed saying ‘Please fill in the description’	Pass
EC4	Write a description that is over 100 characters long		System prevents further input of characters past 100 characters	Counter exists that shows the current character count and when it hits 100/100, no further inputs can be made into the description textbox	Pass

2. White Box Testing

2.1. Register New Account


Register new account
while(username == null)
false
true
Display "enter a username"
while(email == null)
false
true
Display "enter an email"
while(password == null)
false
true
Display "please enter a password"
while(password does not match specified regex)
false
true
Display corresponding error message
while(password does not match reconfirmation password)
false
true
Display corresponding error message
Event handler on "Register button" activated
Calls register function
Accesses Firebase
if(username is taken)
false
true
Function returns "Username already taken"
Creates tuple in the database
Function returns username
Register page redirected to home page

2.2. Upload Pictures


Upload Picture
Upload from Gallery
Take picture
while(uploadedpicture == null)
false
true
Display "No Picture uploaded yet"
Display Uploaded image
Click on "Upload button"
Event handler on "Upload Picture" activated
Calls upload function
Accesses Firebase
Creates an entry in photo storage
Link URL of entry to user's ID in database
Shows "Picture successfully uploaded"
Upload page will direct user back to feed page

3. Demo Script

INTRO:
Good afternoon, we are TeamWon. I am Kieran and together with me I have Yaohao, Zhi Yong, Jin Ying, Sang Won and Kiersten. Today we will be presenting our mobile application: Skyscape. (next slide)

Here is our table of contents for today (wait 3 seconds) (next slide)

Firstly, let me introduce our app. (next slide)

The purpose of our app is to predict the quality of sunsets on any given day and to bring everyone closer to nature's spectacle. We aim to target the app not only for casual users who enjoy sunsets, but also for nature lovers and photography enthusiasts, so that they may never miss the golden hour again. (next slide)

Skyscape is modelled after social media applications whereby users are able to interact with other users. It draws inspiration from activity-tracking apps like Strava that places strong emphasis on community involvement. This concept of bringing people together virtually, is instrumental in retaining users. (next slide)

We decided to use Flutter framework in the making of our application due to flutter's extensive widget library and reactive framework, which greatly streamlined our UI development. Flutter also has a hot reload feature which allows for much faster development without us running the application everytime we make a change. Lastly, Flutter also uses declarative UI approach,which simplifies building user interfaces by describing what should be displayed rather than how it is displayed, making it very beginner friendly. (next)

Secondly, let me run through the functionalities of our app (next slide)

Some features of our app include showing the quality of sunsets of any given day, uploading your sunset pictures for memories and to follow other users as a social platform as well. (next slide)

Here is our complete use case diagram, our main functionalities include: register & login, viewing homepage and in depth location details, uploading pictures, following other users, viewing their profiles and reporting bugs. (next slide)

And here are some of our sequence diagrams (scroll through all 3, stop at the 3rd one). To briefly explain one, let’s look at the view detail sequence diagram. We have the user who interacts with the homepage ui by selecting a location. Homepage ui then goes to the homepage manager, which passes the location details to the viewdetails manager, which does our API call, and once viewdetails manager receives the data it will compute the sunset quality, before showing it onto the location details ui page. (next slide)

I will now pass the time to Sang Won, who will share more about the good software engineering practices we adhered to.

GOOD SE PRACTICES:
Thank you Kieran, now I will continue with the good SE practices.

(next)

Let's briefly go through the good software engineering practices we have taken into account during our software development life cycle.
There are 4 main practices that we have kept in mind, and we will go through them one by one with a little more detail.

(next)

Firstly, the 3 layered architecture, this established approach structures applications into three distinct tiers: presentation, business logic, and data access. This separation offers significant advantages:

Accelerated Development: With separate tiers, development teams can work on each layer concurrently, expediting the application's time to market.

Improved Scalability: Each tier can be independently scaled based on demand, allowing for more flexible and efficient resource allocation.

Loose Coupling: When modifications are required, changes can be confined to specific tiers without causing ripple effects throughout the entire system, promoting stability and agility.

(next)

Next up, the Don't Repeat Yourself Principle aims at reducing the repetition of code. Through that, we can achieve the following benefits:

Enhanced Code Readability: Code readability is improved as developers can easily comprehend the logic and functionality without going through duplicated blocks of code.

Improved Performance: By reducing redundancies in the code, performance is optimized.
Better Maintainability: When updates are required, developers only need to modify code in one location, minimizing the risk of inconsistencies and errors.

(next)

In this slide, we can see that we build the body of the UI within the fixed template which already has the bottom navigation bar, this removes the need to repeatedly include the implementation for the bottom navigation bar for each individual page.

(next)

There is also Single Responsibility Principle, while the SRP is commonly associated with classes in object-oriented programming, its principles can also be applied to functions as well. When functions adhere to the SRP, debugging becomes simpler due to the well-defined responsibilities of each component. This clarity allows developers to isolate issues within specific functionality, facilitating focused debugging efforts without the need to navigate through unrelated code. Additionally, SRP promotes improved code ownership by clearly assigning responsibility for maintaining each function or module.

(next)

We also carried out black box testing to test the software using equivalence class testing to find bugs.

(next)

By testing out different test cases with a single invalid input, we are able to learn the error handling or behavior for each type of invalid inputs we used.

(next)

GOOD TRACEABILITY:
Now I will move on to the traceability in our project.

(next)

Here is an overview of all the traceability practices that we had made use of in our project.

(next)
We used GitHub/Git for version control in traceability by maintaining a history of changes to code and related artifacts.This enabled our team to track the changes in design, functions and implementations over time
(next)

Following UI mockups during coding helps with design traceability by ensuring that the implemented code accurately reflects the intended user interface and design decisions.
This also facilitates alignment between the design and implementation stages of software development.
Following use case descriptions and functional and non-functional requirements during coding helps with requirement traceability by ensuring that the implemented code directly addresses and fulfils the specified functionalities.
This establishes a clear link between the software's capabilities and its intended purpose.
(next)
Now I will pass on my time to Kiersten and Jin Ying for the demonstration of our application
DEMO
User 1
Registration & login
Presenter: Welcome to our Skyscape platform. I am a new user so let’s start by registering a new account.
Demo: key in duplicated username(jinol), and wrong password(password), try to login
Presenter: Seems like my password did not follow the correct format, let me try again.
Demo: Change password to correct format(Password#1) and try to login
Presenter: Oh, seems like someone has already taken our username, let me change it to something else.
Demo: Change username to (user1)
Presenter: My account is successfully registered and I have logged into my account.
Presenter: After a new account is created, this account will be reflected in our firebase authentication section.
Demo: Show Firebase authentication
Access homepage & add locations
Presenter: Lets now go to our homepage. Here is the homepage. From here, I can access various features. First, I want to add some locations to my list. These locations can be places I stay near to or frequently visit.
Demo: Add some locations, go back to homepage
Presenter: Also, same locations cannot be added twice.
Demo: Add a repeated location, go back to homepage.
Presenter: Right now, I can see that there are some percentages associated with the locations I have added. These percentages represent the golden hour quality which means the quality of the sunset at that particular location.
Calendar function
Presenter: At the top of the screen, there is a calendar. This calendar will let me navigate through the sunset qualities of saved locations across different days which helps me to forecast the sunset qualities.
Demo: Tap on different days on the calendar and show changes of sun set quality of the saved locations.
Location details
Presenter: If I tap on a particular location, I can see more details about the overall weather conditions of that particular day.

Demo: Tap into a particular location from the current date
Presenter: The details include sunset quality, timing till sunset, clouds coverage, air quality and humidity.
Demo: Go back to Homepage
Sort locations
Presenter: On the top left corner of the screen, there is a button which allows me to sort my saved locations via quality or alphabet.
Demo: Click on sort by quality. Show the sorted list of locations Then click on sort by Alphabet. Show the sorted list of locations.
Remove Location
Presenter: If I no longer want a particular location on my list, I can remove it.
Demo: Remove a random location (swipe left)
Feed
Presenter: Now, let’s go to my feed tab.
Demo: Go to feed.
Presenter: Feed is where I can see sunset photos uploaded by other users. I can see the username of the user as well as the date and time when the photo is uploaded.
Demo: Scroll down the feed to see more photos.
Presenter: This can foster a sense of community within the users of the app as they share their own views of the sunset.
Upload pictures
Presenter: So let me upload my own sunset picture and be part of the community. I can do that by tapping on the photo icon on the bottom right.
Demo: Click on the photo icon
Presenter: From here, I can choose to upload my picture from my gallery or straight from the camera. Let’s choose a photo from my gallery.
Demo: Choose a preloaded photo in the gallery.
Presenter: Now that we have chosen the photo we want to upload, let’s go ahead and upload it.
Demo: Click upload.
Presenter: I should now be able see my uploaded photo on Feed.
Demo: Go to Feed and show the updated list of photos.
Demo: We can also take a photo directly with our camera and upload it. Since this is an emulator this is what we have to show.
Demo: try upload photo via camera
Profile
Presenter: Let’s now go to my profile tab.
Demo: Click on the profile tab
Presenter: There are mainly 3 functions in this tab. Personal Feed (where I can see all the photos I have uploaded), Account Details (where I can edit my profile details as well as change my profile picture), and Bug reporting (where I can submit a bug report so that the developers can work on them and make the app better).
View all photos posted
Presenter: So let's now go into my personal feed.
Demo: Click into personal feed.

Presenter: From here I can see all the photos I have uploaded to the platform.
Demo: Show the photos uploaded.
Account Settings
Presenter: Now let's make some changes to my account. First, let's change my profile photo from the default photo to something else.
Demo: Click on account settings, then edit profile photo.
Presenter: I will choose this photo from my gallery.
Demo: Choose a photo from the gallery.
Presenter: Now my profile photo is updated.
Presenter: Let's now change my username and password.
Demo: Change username(user11) and password(Password#2)
Presenter: Just to make sure all changes are reflected, let me logout and login again.
Demo: Logout and try to login with my old password (Password#1).
Presenter: The old password no longer works.
Demo: Login with new password (Password#2).
Presenter: Ok, so I am able to login with the new password. The email address remains the same.
Demo: View profile and show changed username.
Presenter: My username has been changed.
Presenter: If we go to our firebase database, we can see updated details associated with our account.
Demo: go to firebase database, users, look for the userID and display all info
Report bug
Presenter: Let's go into the profile tab again and report a bug.
Demo: Click into profile tab then click report bug
Presenter: For demo purposes, I will key in "example" since no bug was found for now.
Demo: Key in "example" and submit.
Presenter: This bug report will also be reflected in our database.
Demo: Show firebase database, bug report.
Presenter: Alright, I think I'm done for now, hopefully people will like the photos that I have uploaded and I can get some followers, let me just logout for now.
Demo: Log out.

User 2 (with preloaded following list)

Access profile of another user
Presenter: Let me now see how my account is viewed by other users. I will log in as another user, User 2.
Demo: Login as User 2.
jinol
Jinying#110
Demo: Go to the Users bar and enter User 1's username.
Demo: Access and view the profile of User 1.

Follow user
Presenter: User 2 decides to follow me after looking at my profile.
Demo: Follow User 1.

View following
Presenter: After following me, User 2 decides to look at his following list.
Demo: View profile of users User 2 is following.

Unfollow user
Presenter: User 2 decides to unfollow User 4.
Demo: Unfollow User 4.

Feed
Presenter: User 2 then decides to look at his feed.
Demo: Access the feed page.

Feed (Following)
Presenter: User 2 wants to look at the feed containing photos only by the people he follows. Since he now follows us and no longer follows User 4, our photos should appear in the feed, and User 4's photos should not appear.
Demo: Click on the "Following" tab and view photos only by those User 2 are following (User 1, User 3).

Presenter: This is the end of our demo, thank you!