Ideas (app or website)
-Food promotion(app that collates all offers and promotions offered in restaurants to users. Similar to tele grps)
(reference to: SeeFood App from previous cx2006 project https://github.com/Javelin1991/CZ2006_Software_Engineering)
(UI - categorize by cuisine/ show everything with some filter for tags)
(Data Input - create our own dataset? Entered by admin? web scraping? Web scraping is the process of using bots to extract content and data from a website)
(function to bookmark/save coupon/deal)
(eg coupon and deals website: https://myfave.com/singapore/eat)

https://singpromos.com/dining-restaurants-food/

-Carpark space tracker (closest/cheapest)

-social platform for gym buddies (SGgym locations https://data.gov.sg/dataset/gymssg, directions to said locations, weather forecast, online chat rooms, daily step counter based on GPS location, locate car parks near gym)
https://wger.de/en/software/api

-bin tracker (recycling and e waste bins)(2nd hand goods collection (eg salvation army)))(cash for trash) (locating, give directions) Geolocation api
(https://www.nea.gov.sg/our-services/waste-management/3r-programmes-and-resources/recycling-collection-points)
https://data.gov.sg/dataset/e-waste-recycling
https://data.gov.sg/dataset/cash-for-trash
https://data.gov.sg/dataset/2nd-hand-goods-collection-points
https://data.gov.sg/dataset/recycling-bins?resource_id=895f1883-d3bf-467c-833c-226ad92c6229
https://dollarsandsense.sg/6-ways-make-money-recycling-singapore/

-booking appointment based on how crowded a place is (clinic, restaurant, etc)

-school app (education level, location, requirements, etc)
(school account)

-Vehicle tracker using driver gps(https://nevonprojects.com/vehicle-tracking-using-driver-mobile-gps-tracking/)
(Used by companies to track company owned vehicles)

-College social networking app(https://nevonprojects.com/college-social-network-web-project/)

Schedule:

- Decide on **theme** at the end of **Week 2. (Next lab on 4th Feb)**
- Finish lab 1 stuff by next week (week 3) cos week 4 is CNY

DELIVERABLES

• Documentation of functional and non-functional requirements
Functional requirements describe interactions between the system and the environment, to map program inputs to program outputs. Basically the things that the system must do.
Non-Functional requirements describe the properties the system must have, that is not directly related to the functional behaviour of the system.

• Data dictionary
(Document important terms of your application (e.g., user, device, input, output) in a data dictionary. Explain each term with a brief description. Identify attributes of each term and relationships between terms.)

• Initial Use Case Model, consisting of Use Case diagram and Use Case descriptions
(For each Use Case, start writing the use case description about how the user interacts with the system to carry out the system functionality. As a rule-of-thumb, each Use Case should have a maximum of 6~7 steps in its flow of events. A small Use Case indicates that the functionality has been sliced too finely; a large Use Case can be further broken down.)

• UI Mockups

Please submit the deliverables to your SVN repository (under the folder “lab1”) before Lab#2 starts. Your Lab Supervisor will want to see and discuss the deliverables with you during Lab #2. The initial Use Case model will be refined and elaborated in Lab#2.

*Data.gov.sg provides live data updates, one idea for the app is to periodically request access to the api and then input the data into the app.

BIN TRACKER (TrashTogether (tbc))
Functional Requirements

1. Main menu
    1.1. The system must be able to display the option to find the locations of recycling bins and collection points.
    1.2. The system must allow user to register their account.
    1.3. The system must allow user to login to their account.
2. Registration
    2.1. The user must be able to register for a new account via our system registration.
        2.1.1. Information include:
            2.1.1.1. Name
            2.1.1.2. Password
            2.1.1.3. Confirm Password
            2.1.1.4. Valid Email Address
    2.2. The system must validate all required fields have been filled up.
    2.3. The system must validate the account availability
3. Login
    3.1. The user must be able to login using their email address and password.
    3.2. The system must validate that both fields have been filled up.
    3.3. The system must validate the email address and password are valid.
    3.4. If the login information is invalid, the system will display an error message.
    3.5. The system will redirect the user to the application's main menu.
    3.6. The user must be able to request for a change of password.
    3.7. The system must be able to send email verification code upon the user's request for a change of password.
4. Location
    4.1. The system must be able to display recycling bin locations in ascending order of distance from the user.
    4.2. The user must be able to search for a variety of recycling bins based on types of recyclable material. (e.g. plastic, paper, clothes, e-waste, etc.)
    4.3. The system will redirect the user to the directions to the location of interest.
5. Collection points (cash for trash)
    5.1. The system must be able to display collection points for trading cash for trash in ascending order of distance from the user.
    5.2. The system will redirect the user to the directions to the location of interest.
6. Directions
    6.1. The system must be able to display directions to the user's location of interest.
7. Estimate cash returns
    7.1. The system must be able to display the estimated cash returns of *common trash* traded for cash.
    7.2. The user must be able to record items to be traded for cash.
8. Interface with other systems
    8.1. The system must be able to retrieve location from user's device via Geolocation API.

    8.2. The system must be able to retrieve the locations of recycling bins and collection points from Data.gov.sg.

9. Formats for information to be processed
    9.1. Location
        9.1.1. Distance format must be in “km” and two decimal places of accuracy (e.g. 1.50km) if distance >=1000m.
        9.1.2. Distance format must be in “m” if distance < 1000m.
    9.2. Price
        9.2.1. Price format must be in “S$” and two decimal places of accuracy. (e.g. S$1.50)
    9.3. Weight
        9.3.1. Weight format must be in “kg” and two decimal places of accuracy (e.g. 1.50kg) if weight >=1000g.
        9.3.2. Weight format must be in “g” if distance < 1000g.

Non-functional requirements

1. Security requirements
    1.1. The system will perform hashing on the password created for the account before storing the password into the database for better security of the user’s password.
    1.2. When the user inputs the password for the account, the system will hash the input password and compare it with the hashed password in the database for that user account.
2. Usability requirements
    2.1. The system must allow easy reversal of actions, where the user must be able to change the recorded items for cash for trash.
    2.2. The system must offer informative feedback, and display proper error messages and feedback when user keys in invalid inputs
3. Performance requirements
    3.1. The application should not have any bugs and the user interface must be fast and responsive according to the user’s action.
    3.2. The system should return results in the minimum time possible.
4. Data Integrity requirements
    4.1. The application must only provide accurate data of locations.
    4.2. The system should ensure that the directions to the recycling bins must be correct
5. Scalability requirements
    5.1. The system should perform efficiently when tasked with any number of users.
    5.2. The system should ensure that the application is reliable and minimize downtime as much as possible.
    5.3. The system must be designed with the Model-View-Controller architecture and design patterns to support future enhancements and increase scalability.

Data dictionary *can consider adding another column*

Term	Definition
User	A user is a person who is using the application/website to find recycling bins or collection points to trade trash for cash in Singapore
System	A system refers to the TrashTogether mobile application/website.
Filter	Filter is a feature that allows users to find specific types of recycle bins or collection points that they want to search for, such as plastic, paper, clothes, e-waste, etc.
Registration	Registration is a feature that allows users to sign up for an account.
RB Location	Recycling Bin Location is a feature that allows users to find recycling bins
Collection points	Collection points is a feature that allows users to find collection points
Direction	Direction is a feature that allows users to be guided to a location of interest using GPS.
Model-View-Controller	Model-View-Controller is a software architectural pattern for implementing user interfaces in the application.
GPS	Global Positioning System which will detect a user's current location. Is used interchangeable with WIFI positioning system

Use Case Description
standardize alternative flow

Use Case Model 1 : Creating a new account

Actor:	User
Description:	Creating a new account for a user.
Preconditions:	1. New user account must not exist in the database previously. 2. Users must be connected to WiFi or mobile data during this process.
Postconditions:	1. A verification email will be sent to the email address that the user used to create a new account.
Priority:	Medium
Frequency of Use:	Once in a lifetime

Flow of Events:	1. User enters username,account password, email address and confirmation password. 2. User clicks on the “Create an account” button. 3. System sends a verification email to the email address indicated previously. 4. User inputs the verification code that is found in the verification email. 5. New user account is created.
Alternative Flows:	AF-S1-1: Password does not match confirm password. 1.System will display an error message 2. User will be prompted to re-enter password details. 3. User will retry registration using new details 4. Return to step 1.
Exception:	-
Includes:	Checking whether a user account has previously existed or not.
Special Requirements:	-
Assumptions:	Each user only needs one account and doesn’t create a second account.
Notes and Issues:	-

Use Case Model 2 : Searching for near trash for cash locations

Actor:	User
Description:	Searching for near trash for cash locations
Preconditions:	1. User must be connected to internet 2. User must turn on location/ set location
Postconditions:	1. System displays the closest collection point and offers to direct the user to it.
Priority:	Medium
Frequency of Use:	Once per week
Flow of Events:	1. User searches for cash for trash locations 2. All Locations are displayed in ascending order of distance from the user 3. The user may select one of the locations 4. The system will then use gps to guide the user to the location chosen

Alternative Flows:	AF-S1-1: User is not connected to Internet and has not set a location: 1.System will generate error message and inform the user that they need to connect. AF-S1-2: Cash for trash stations are suspended: 1.System will inform the user that no stations are available now
Exception:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case Model 3 : Validating account availability

Actor:	System
Description:	Validate the account availability
Preconditions:	1. User account must not already exist in the database 2. Mobile must be connected to WIFI/Mobile Data
Postconditions:	1. User account is successfully created and stored in the database. 2. Users are able to login using their account successfully.
Priority:	Medium
Frequency of Use:	Once in a lifetime
Flow of Events:	1. System will check if the account already exists in the database. 2. System will store the account details in the database
Alternative Flows:	AF-S1-1: The account already exists in the database. 1. System will display an error message 2. User will enter new account details 3. User will reattempt to register using the new details 4. Return to step 1

Exception:	
Includes:	1. Validate the account availability
Special Requirements:	-
Assumptions:	Database refers to system's database
Notes and Issues:	-

Use Case Model 4 : Change password

Actor:	User
Description:	Allows User the option to change their account's password
Preconditions:	1. Mobile must be connected to WiFi/data
Postconditions:	1. System successfully updates User's new password. 2. User is able to login with the new password successfully.
Priority:	Medium
Frequency of Use:	1 - 3 times per year
Flow of Events:	1. User type in their email address and tap the Change password button. 2. System will check if the email address is registered. 3. System will send an OTP to the registered email address provided. 4. User will enter the OTP. 5. System will validate the OTP. 6. User will key in the new password and confirm new password, and tap Confirm password changes. 7. System will check if new password matches confirm new password. 8. System will save the new password into the database.

Alternative Flows:	AF-S2-1: The provided email address is not registered and cannot be found in the database: 1.System will generate error message 2. User will be prompted to re-enter their details 3.System will validate the new details. 4.Return to step 1. AF-S5-1: User enters an invalid OTP: 1.System will generate an error message. 2.System will resend the OTP. 3.User will enter the new OTP. 4. Return to step 5 AF-S7-1: New password does not match confirm new password: 1.System will generate an error message. 2.User is to re-enter the password details. 3. System is to validate the new password. 4. Return to step 7
Exception:	
Includes:	
Special Requirements:	-
Assumptions:	
Notes and Issues:	-

Use Case Model 5: Searching for recycling bin locations

Actor:	User
Description:	Searching for recycling bin locations
Preconditions:	1. User must be connected to internet 2. User must turn on location/ set location
Postconditions:	1. System displays the closest recycling bin and offers to direct the user to it.
Priority:	High
Frequency of Use:	Once per week

Flow of Events:	1. User searches for recycling bin locations 2. All locations are displayed in ascending order of distance from the user 3. The user may use the filter feature to find recycling bins for a certain recyclable material. 4. The user may select one of the locations 5. The system will then use gps to guide the user to the location chosen
Alternative Flows:	AF-S1-1: User is not connected to Internet and has not set a location: 1.System will generate error message and inform the user that they need to connect.
Exception:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case Model 6: Estimate Cash Return

Actor:	User
Description:	Estimate cash return from trash for cash
Preconditions:	
Postconditions:	1. System displays the estimated cash return for the amount of trash that the user has entered.
Priority:	High
Frequency of Use:	0-5 times per week
Flow of Events:	1. User selects the type of recyclable item. 2. User enters the weight of the recyclable item.

	3. System will calculate the estimated cash return based on the input. 4. System will display the estimated cash return.
Alternative Flows:	AF-S2-1: User enters an invalid input for weight of recyclable item: 1. System will generate an error message. 2. User will be prompted to re enter the weight of the recyclable item. 3. Return to step 3.
Exception:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case Model 7: Search for bins using filters

Actor:	User
Description:	To filter between recycling bins and cash for trash collection points
Preconditions:	1. Mobile must be connected to WiFi/Mobile Data.
Postconditions:	1. System will display either recycling bins or cash for trash collection points based on user's filter.
Priority:	High
Frequency of Use:	0-5 times per week
Flow of Events:	1. User selects a preferred filter between recycling bins and cash for trash collection points. 2. System displays the map with either recycling bins or cash for trash collection points. 3. User taps on “Directions” button. 4. System will guide user to the nearest

	bin via gps.
Alternative Flows:	-
Exception:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case Model 8 : Logging into user account

Actor:	User
Description:	Logging into user's account
Preconditions:	1. User account must already exist in the database 2. Mobile must be connected to WIFI/Mobile Data
Postconditions:	1. User is logged into the account. 2.
Priority:	High
Frequency of Use:	0-2 times per week
Flow of Events:	1. User keys in username and password. 2. User taps on the login button. 3. System checks if user's account is in the database and if password is correct. 4. System authenticates the user to login. 5. User is brought to the main menu page.
Alternative Flows:	AF-S1-1: The username and password does not match any account in the database. 1. System will display an error message. 2. User will be prompted to re-enter login details. 3. Return to step 2.
Exception:	-
Includes:	-
Special Requirements:	-

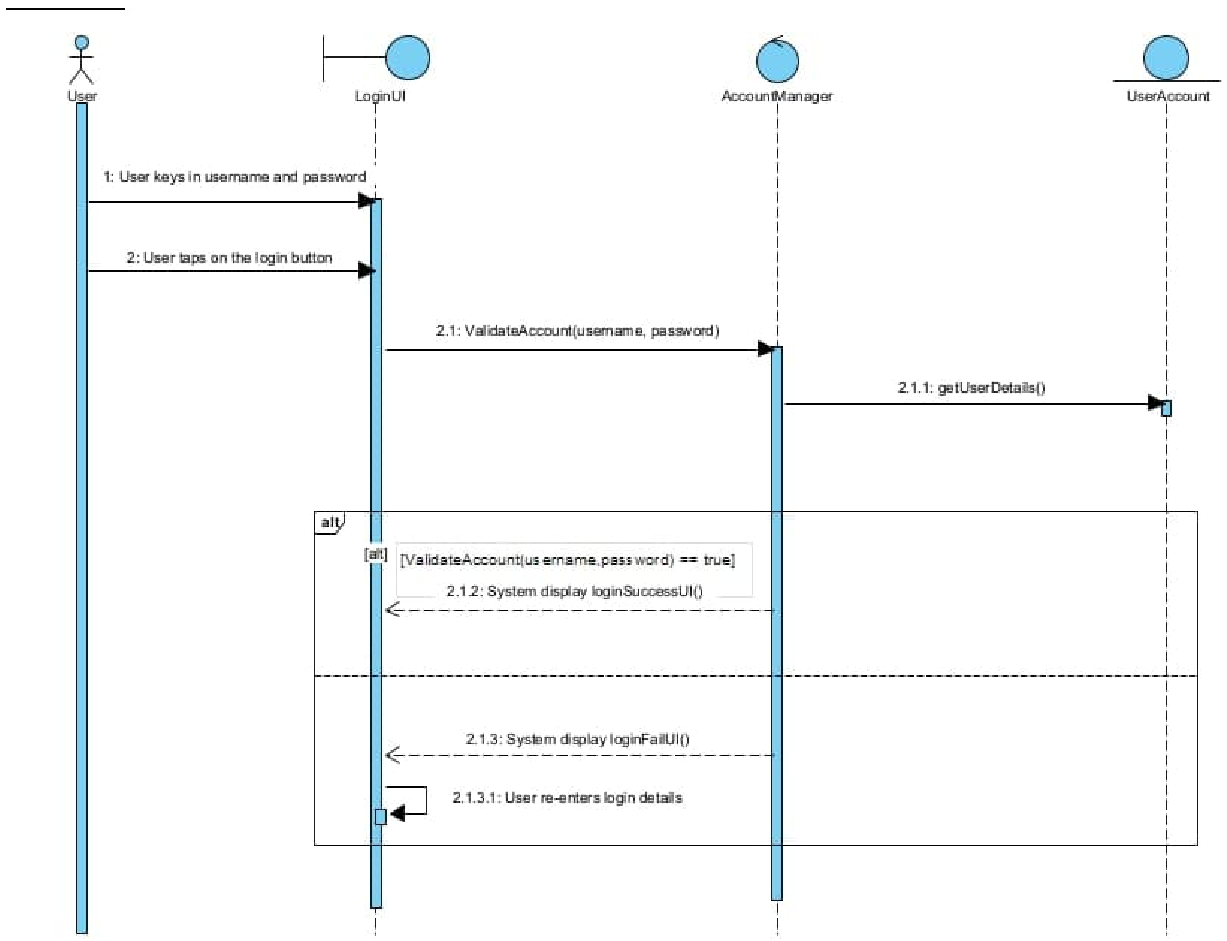
Assumptions:	Database refers to system's database
Notes and Issues:	-

Use Case Model 9 : Logging out of user account

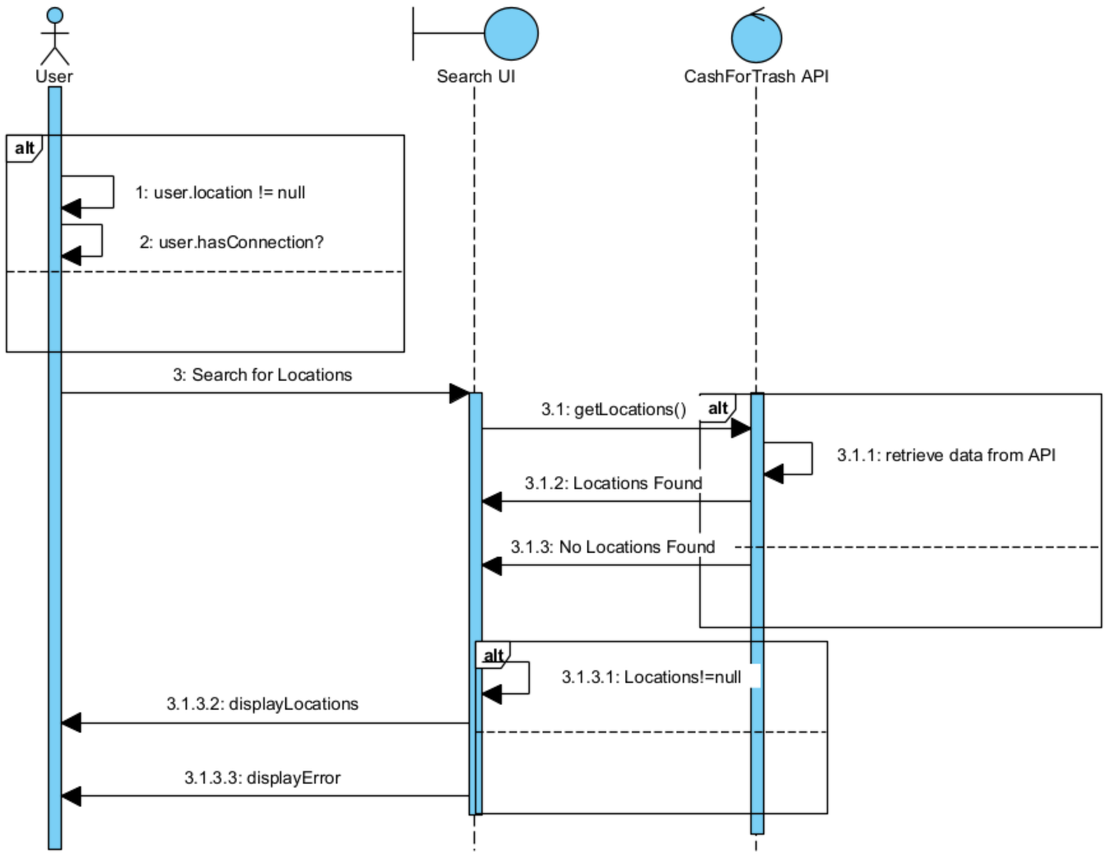
Actor:	User
Description:	Logging out of user's account
Preconditions:	1. Mobile must be connected to WIFI/Mobile Data
Postconditions:	1. User is logged out of the account.
Priority:	Medium
Frequency of Use:	0-2 times per week
Flow of Events:	1. User taps on the logout button in the main menu. 2. System will log user out of the account. 3. User is brought to the login page.
Alternative Flows:	-
Exception:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Sequence Diagrams:

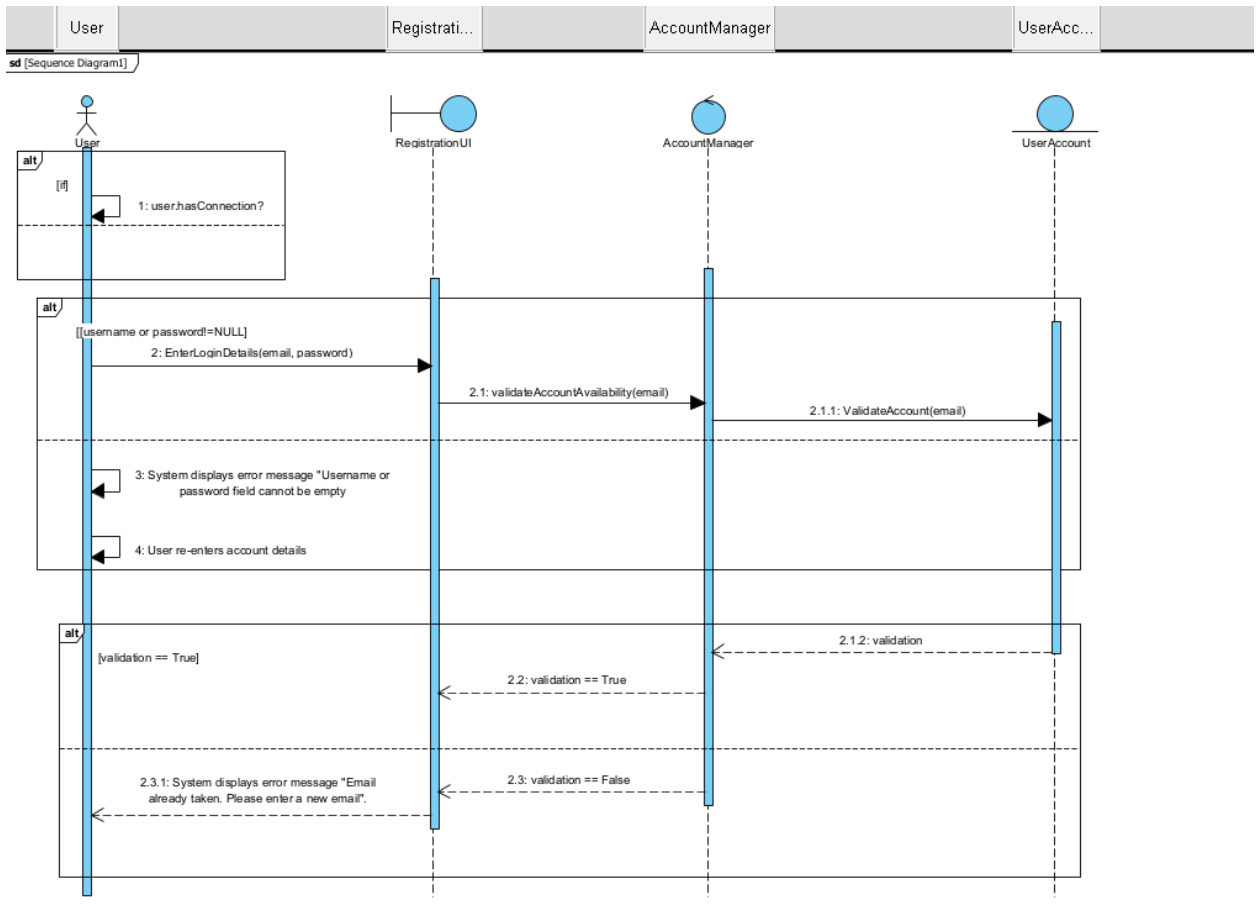
UC1: Register an account


User
LoginUI
AccountManager
UserAccount
1: User keys in username and password
2: User taps on the login button
2.1: ValidateAccount(username, password)
2.1.1: getUserDetails()
alt
[alt]
[ValidateAccount(username,password) == true]
2.1.2: System display loginSuccessUI()
2.1.3: System display loginFailUI()
2.1.3.1: User re-enters login details

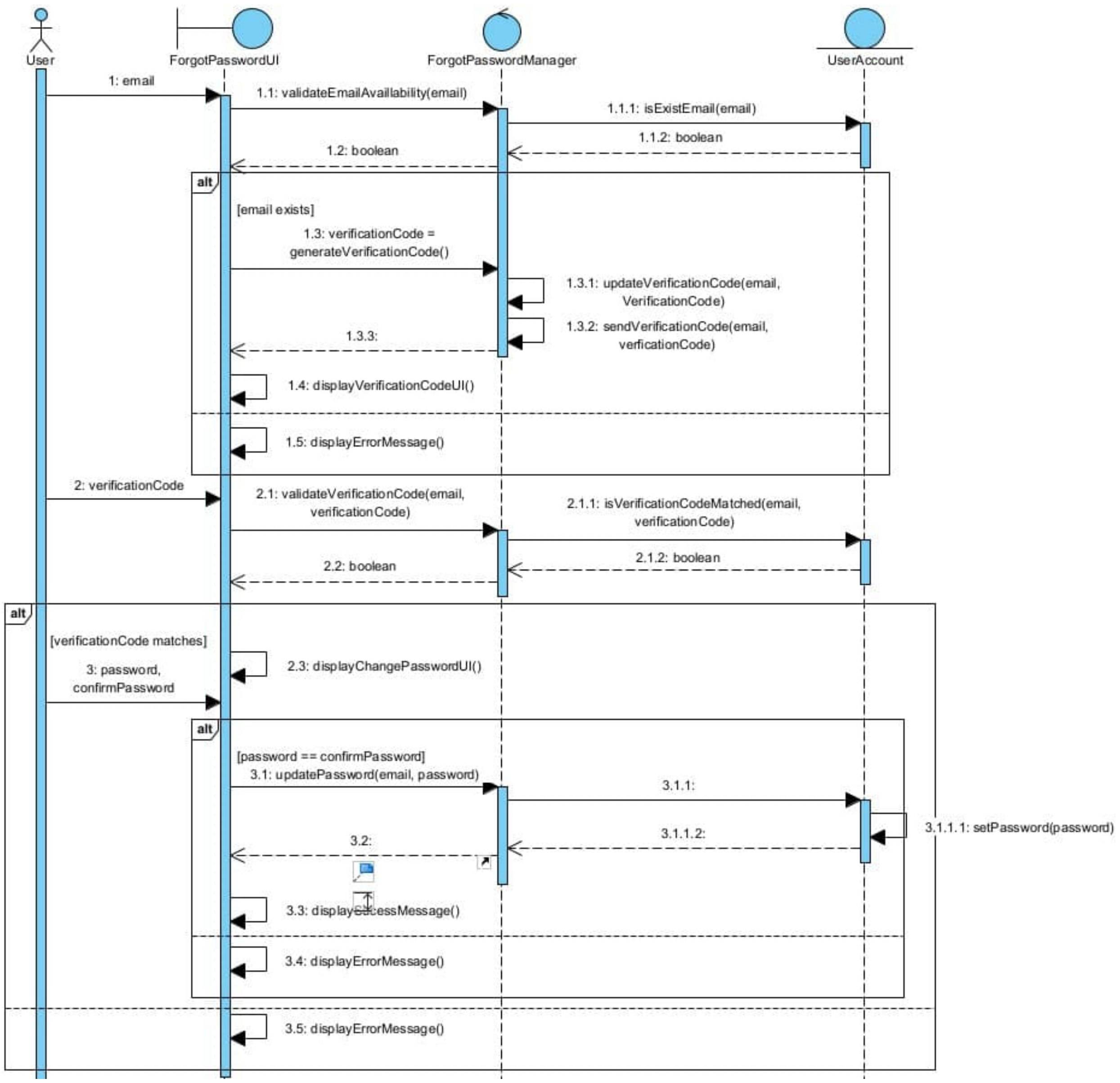
UC2: Searching for Cash For Trash Locations



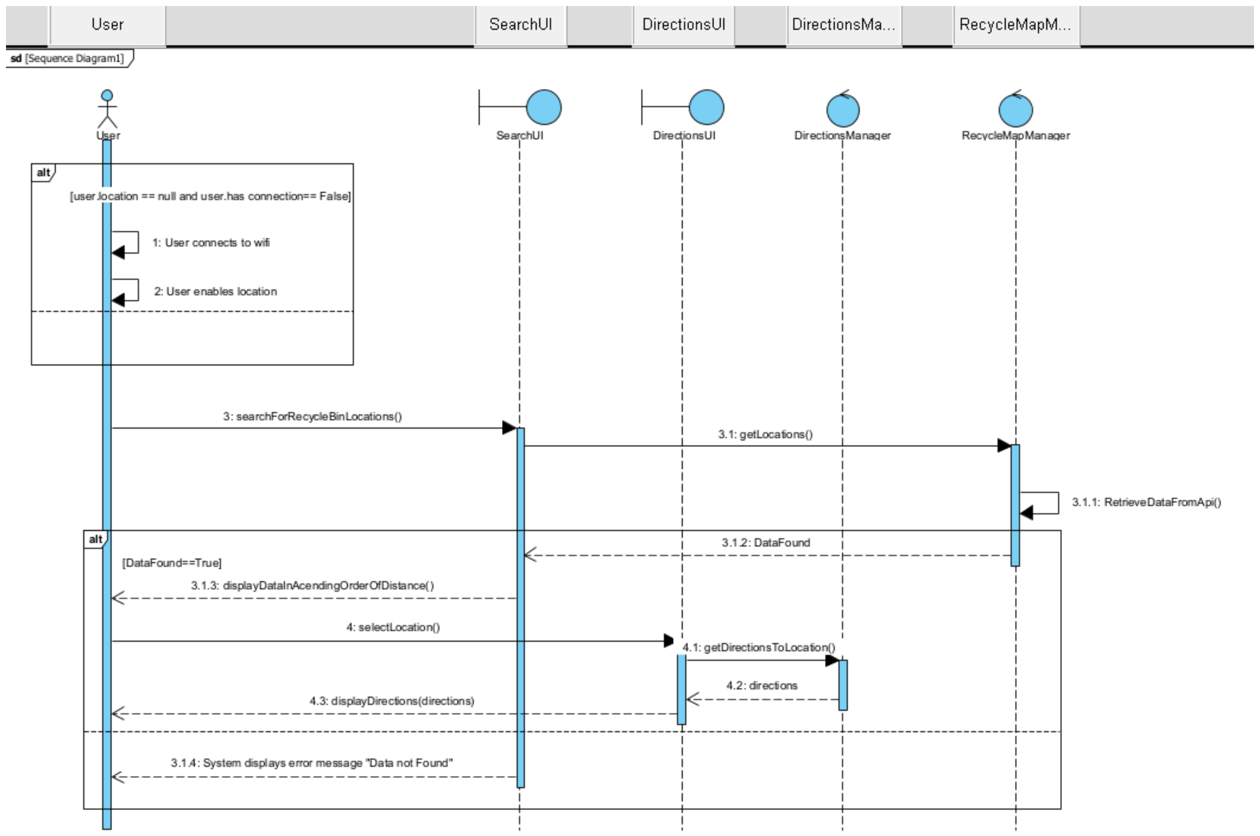
UC 3:Searching for recycling bin locations


User
Registrati...
AccountManager
UserAcc...
sd [Sequence Diagram1]
User
RegistrationUI
AccountManager
UserAccount
alt
[if]
1: user.hasConnection?
alt
[[username or password!=NULL]
2: EnterLoginDetails(email, password)
2.1: validateAccountAvailability(email)
2.1.1: ValidateAccount(email)
3: System displays error message "Username or password field cannot be empty
4: User re-enters account details
alt
[validation == True]
2.1.2: validation
2.2: validation == True
2.3: validation == False
2.3.1: System displays error message "Email already taken. Please enter a new email".

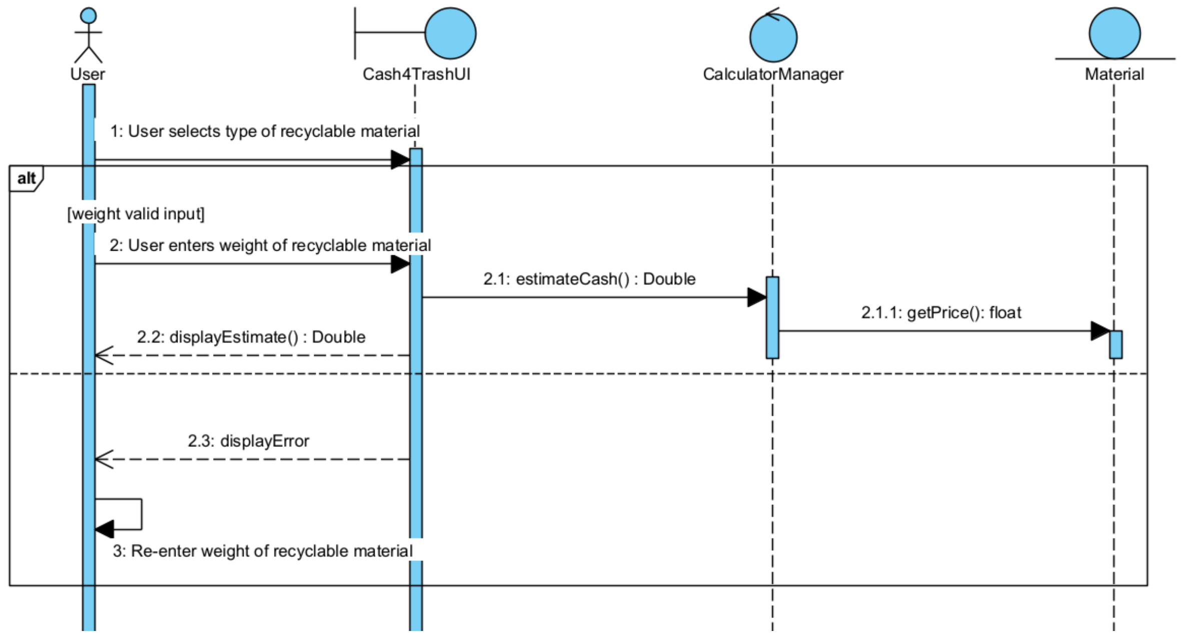
UC4: Change Password


User
ForgotPasswordUI
ForgotPasswordManager
UserAccount
1: email
1.1: validateEmailAvaillability(email)
1.1.1: isExistEmail(email)
1.1.2: boolean
1.2: boolean
alt
[email exists]
1.3: verificationCode = generateVerificationCode()
1.3.1: updateVerificationCode(email, VerificationCode)
1.3.2: sendVerificationCode(email, verficationCode)
1.3.3:
1.4: displayVerificationCodeUI()
1.5: displayErrorMessage()
2: verificationCode
2.1: validateVerificationCode(email, verificationCode)
2.1.1: isVerificationCodeMatched(email, verificationCode)
2.1.2: boolean
2.2: boolean
alt
[verificationCode matches]
2.3: displayChangePasswordUI()
3: password, confirmPassword
alt
[password == confirmPassword]
3.1: updatePassword(email, password)
3.1.1:
3.1.1.1: setPassword(password)
3.1.1.2:
3.2:
3.3: displaySuccessMessage()
3.4: displayErrorMessage()
3.5: displayErrorMessage()

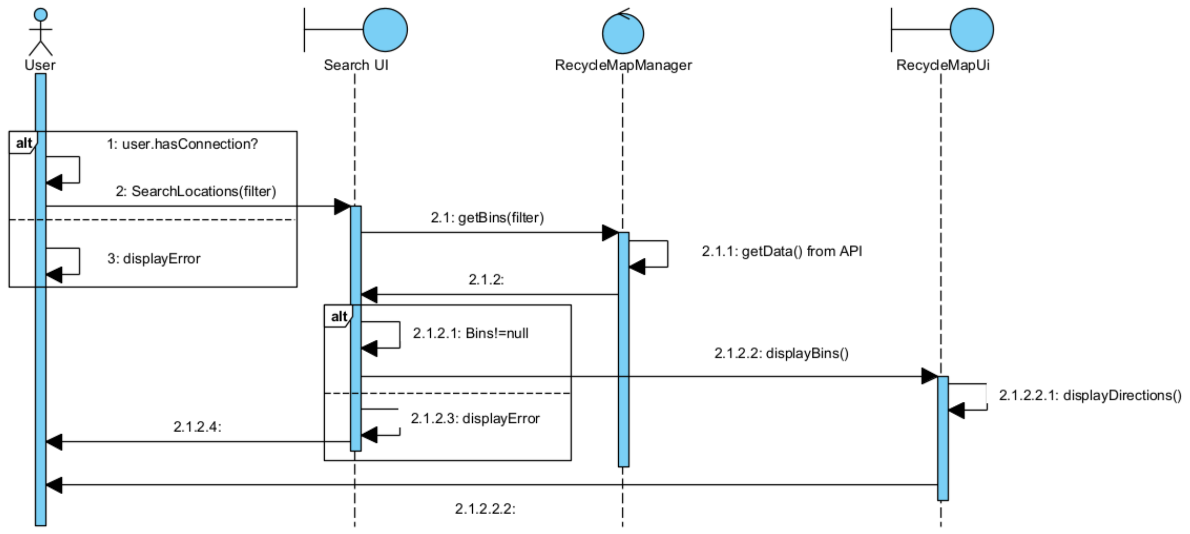
UC 5: Searching for recycling bin locations


User
SearchUI
DirectionsUI
DirectionsMa...
RecycleMapM...
sd [Sequence Diagram1]
User
SearchUI
DirectionsUI
DirectionsManager
RecycleMapManager
alt
[user.location == null and user.has connection== False]
1: User connects to wifi
2: User enables location
3: searchForRecycleBinLocations()
3.1: getLocations()
3.1.1: RetrieveDataFromApi()
alt
3.1.2: DataFound
[DataFound==True]
3.1.3: displayDataInAcendingOrderOfDistance()
4: selectLocation()
4.1: getDirectionsToLocation()
4.2: directions
4.3: displayDirections(directions)
3.1.4: System displays error message "Data not Found"

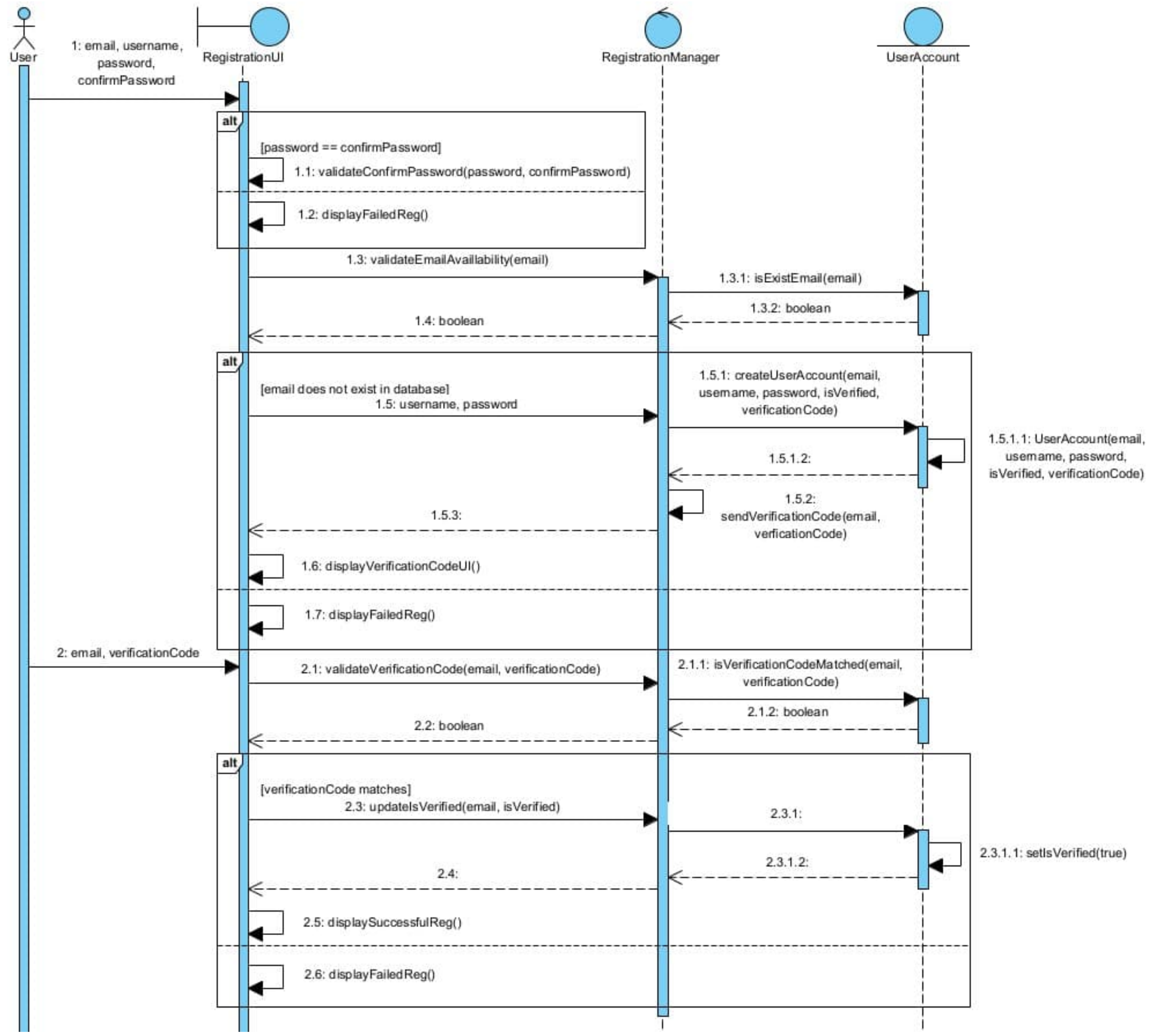
UC6: Estimate Cash Return


User
Cash4TrashUI
CalculatorManager
Material
1: User selects type of recyclable material
alt
[weight valid input]
2: User enters weight of recyclable material
2.1: estimateCash() : Double
2.1.1: getPrice(): float
2.2: displayEstimate() : Double
2.3: displayError
3: Re-enter weight of recyclable material

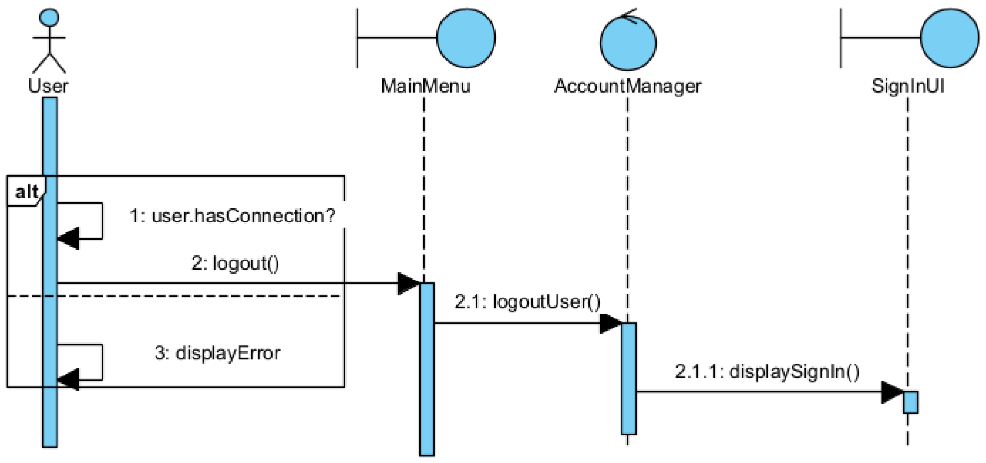
UC7: search for bins using filters



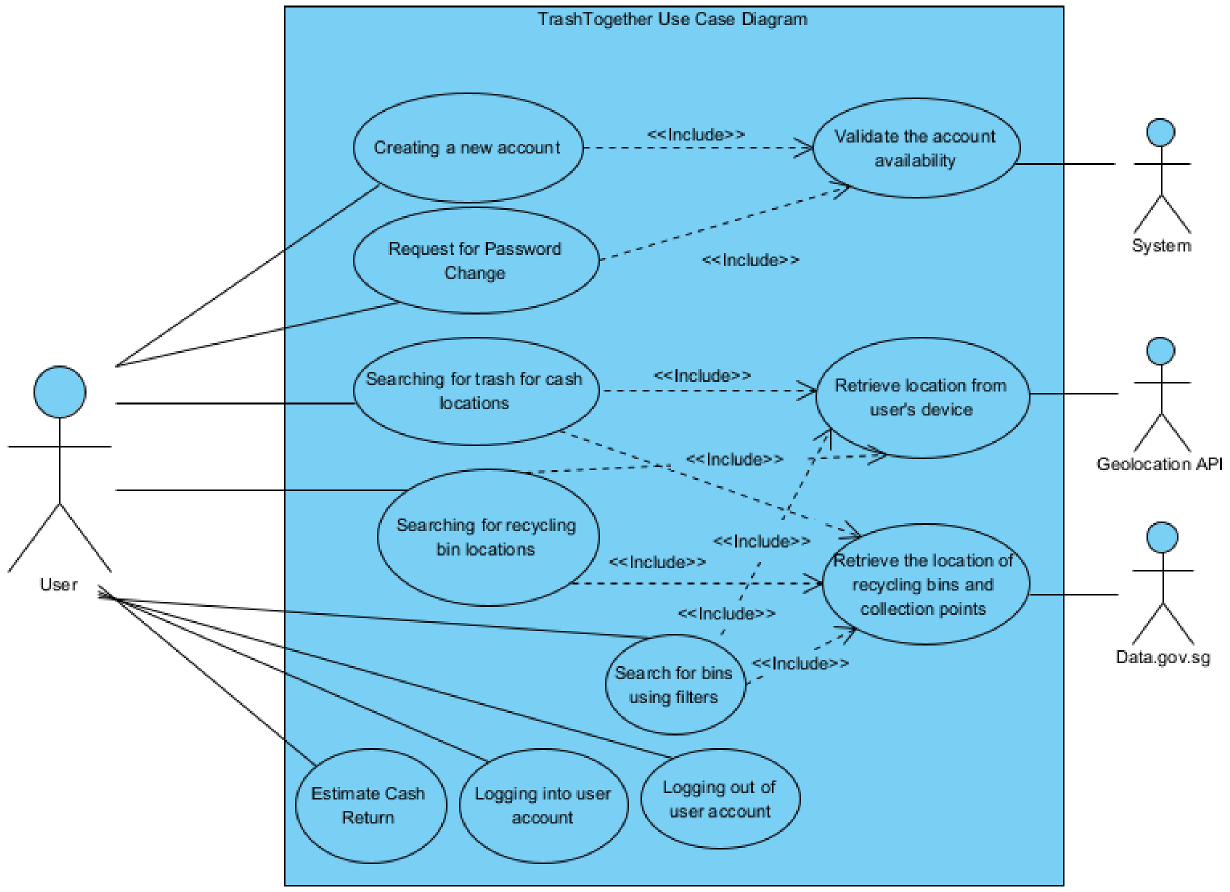
UC8: logging In


User
RegistrationUI
RegistrationManager
UserAccount
1: email, username, password, confirmPassword
alt
[password == confirmPassword]
1.1: validateConfirmPassword(password, confirmPassword)
1.2: displayFailedReg()
1.3: validateEmailAvaillability(email)
1.3.1: isExistEmail(email)
1.3.2: boolean
1.4: boolean
alt
[email does not exist in database]
1.5: username, password
1.5.1: createUserAccount(email, username, password, isVerified, verificationCode)
1.5.1.1: UserAccount(email, username, password, isVerified, verificationCode)
1.5.1.2:
1.5.2: sendVerificationCode(email, verficationCode)
1.5.3:
1.6: displayVerificationCodeUI()
1.7: displayFailedReg()
2: email, verificationCode
2.1: validateVerificationCode(email, verificationCode)
2.1.1: isVerificationCodeMatched(email, verificationCode)
2.1.2: boolean
2.2: boolean
alt
[verificationCode matches]
2.3: updateIsVerified(email, isVerified)
2.3.1:
2.3.1.1: setIsVerified(true)
2.3.1.2:
2.4:
2.5: displaySuccessfulReg()
2.6: displayFailedReg()

UC9: logging Out



Use Case Diagram


TrashTogether Use Case Diagram
Creating a new account
<<Include>>
Validate the account availability
Request for Password Change
<<Include>>
System
Searching for trash for cash locations
<<Include>>
Retrieve location from user's device
Geolocation API
<<Include>>
Searching for recycling bin locations
<<Include>>
<<Include>>
Retrieve the location of recycling bins and collection points
User
<<Include>>
Search for bins using filters
<<Include>>
Data.gov.sg
Estimate Cash Return
Logging into user account
Logging out of user account

Any use case need <<Extend>>?

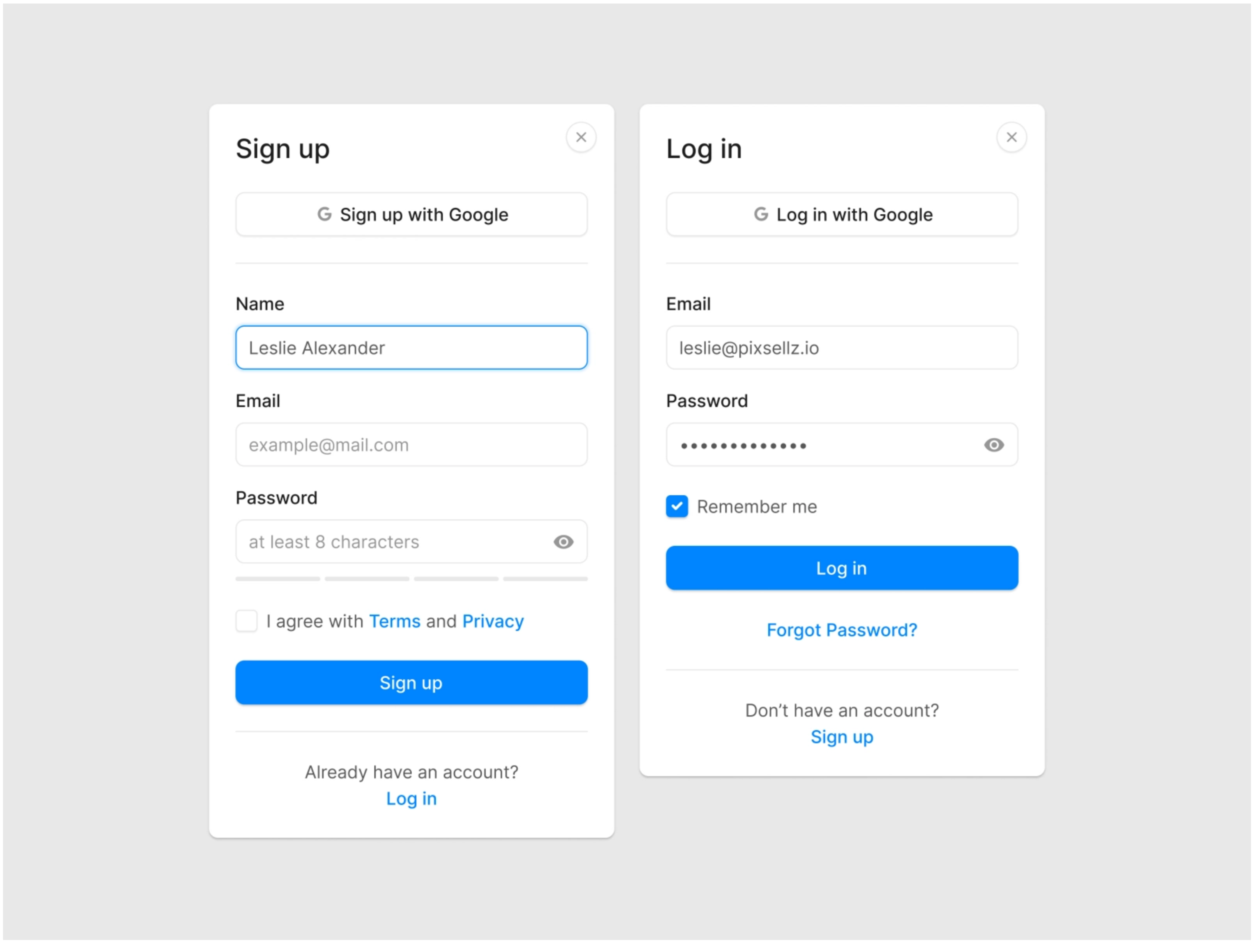
Not recommended to have floating bubble
Flutter, android app, Firebase,
Front end only display
Back end calculation on cloud

Class Diagram And Dialog Map

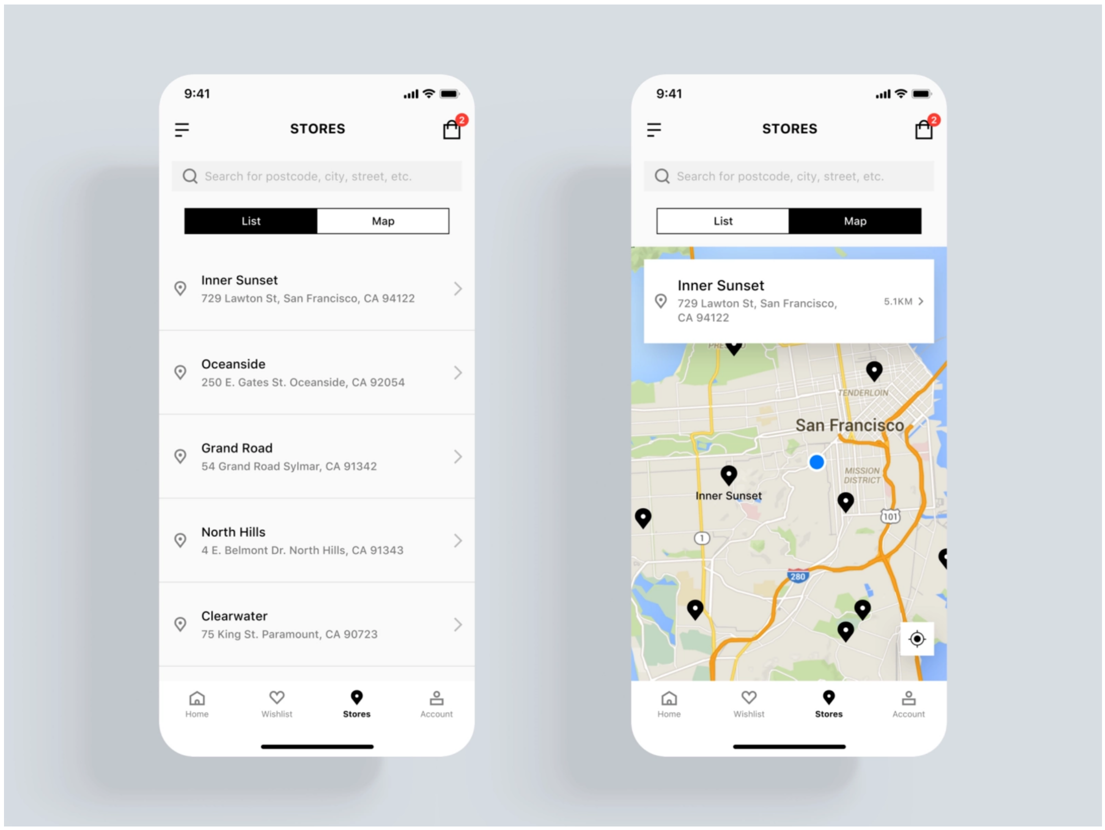
https://app.diagrams.net/#G1umbv5-YboZAMt0YwoQ9VXcEp8EI5NA8q

UI design mockup:

Log in / Sign up page



Searching for locations:



9:41

STORES

Search for postcode, city, street, etc.

List | Map

Inner Sunset
729 Lawton St, San Francisco, CA 94122

Oceanside
250 E. Gates St. Oceanside, CA 92054

Grand Road
54 Grand Road Sylmar, CA 91342

North Hills
4 E. Belmont Dr. North Hills, CA 91343

Clearwater
75 King St. Paramount, CA 90723

Home | Wishlist | Stores | Account

9:41

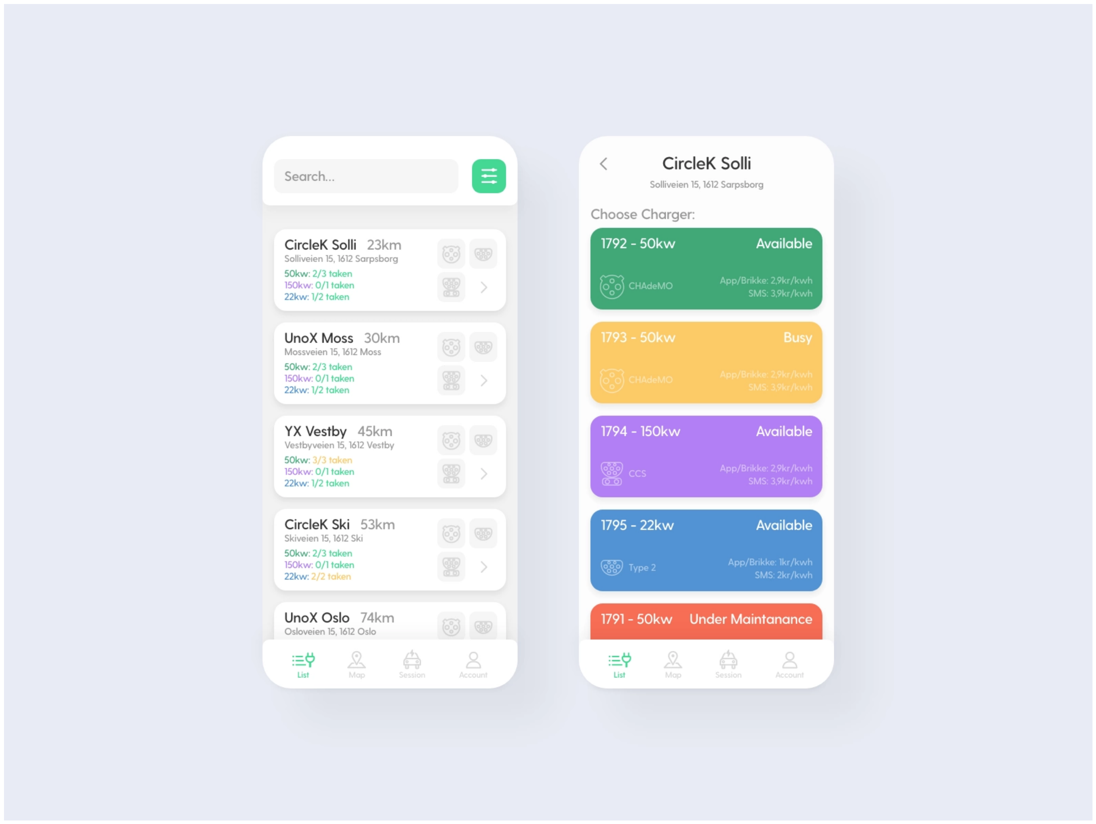
STORES

Search for postcode, city, street, etc.

List | Map

Inner Sunset
729 Lawton St, San Francisco, CA 94122
5.1KM

Home | Wishlist | Stores | Account



Search...

CircleK Solli 23km
Solliveien 15, 1612 Sarpsborg
50kw: 2/3 taken
150kw: 0/1 taken
22kw: 1/2 taken

UnoX Moss 30km
Mossveien 15, 1612 Moss
50kw: 2/3 taken
150kw: 0/1 taken
22kw: 1/2 taken

YX Vestby 45km
Vestbyveien 15, 1612 Vestby
50kw: 3/3 taken
150kw: 0/1 taken
22kw: 1/2 taken

CircleK Ski 53km
Skiveien 15, 1612 Ski
50kw: 2/3 taken
150kw: 0/1 taken
22kw: 2/2 taken

UnoX Oslo 74km
Osloveien 15, 1612 Oslo

List | Map | Session | Account

CircleK Solli
Solliveien 15, 1612 Sarpsborg

Choose Charger:

1792 - 50kw Available
CHAdeMO
App/Brikke: 2,9kr/kwh
SMS: 3,9kr/kwh

1793 - 50kw Busy
CHAdeMO
App/Brikke: 2,9kr/kwh
SMS: 3,9kr/kwh

1794 - 150kw Available
CCS
App/Brikke: 2,9kr/kwh
SMS: 3,9kr/kwh

1795 - 22kw Available
Type 2
App/Brikke: 1kr/kwh
SMS: 2kr/kwh

1791 - 50kw Under Maintanance

List | Map | Session | Account