Sit With Us

Software Requirements Specifications

Will Crain, David Glenn, Ryan Mitchell, Hazem Tashkandi

October 3rd, 2017

CHANGE HISTORY

Date	Versio n	Description	Author
September 19, 2017	0.01	Initial list of Functional and Nonfunctional Requirements	Team
September 21, 2017	0.02	Initial Use Case Diagrams published	Team
September 26, 2017	0.03	Revised Use Case Diagrams, start of Activity Diagrams	Team
September 28, 2017	0.04	Added more Activity Diagrams, Use Case Diagrams revised	Team
October 1, 2017	0.05	Finished draft of Activity Diagrams, began write up of requirements, including Table of Contents, Purpose, Glossary, and Project Description	Team
October 2, 2017	0.06	Continued work on Requirements document, Activity Diagrams revised, Class Diagram published. Sorting of essential parts of program finished.	Will, Ryan
October 3, 2017	0.1	Revising of Activity Diagrams and Use Case Diagrams, writing of activity diagram descriptions, writing of use case descriptions. Finalization of formatting.	Team

TABLE OF CONTENTS

I. INTRODUCTION

Purpose

Sit With Us is an Android social networking app that will allows to spontaneously meet new people at their current location. The purpose of this app is to make finding new people to develop temporary or long-lasting friendships with easier. The app will allow people to form spontaneous meetups and search for other nearby people and meetups to join using GPS.

Glossary

Block: An action a user invokes on another user that hides the user's profile and activity from the other user.

Block list: A list of other users that a particular user has blocked.

Description: A freeform text field designed to allow users to enter a brief description of themselves. This will not be used for matching algorithms, but will be displayed to other users during the matching process to shore up any gaps left by interests.

Developer: A software developer who works on creating the app.

Friend List: The list of contacts in the user's device that have accounts in this app associated with their contact number.

Interest: A textual tag corresponding to a hobby, used to match with other users who share interests.

Meetup: Two or more users currently matched by the app, either through the search function or through the friends list. Can be closed or open to additional users.

Profile: A page owned by a user containing (a) photo(s), interests, and a text field filled out by the user.

Server: The device that runs software used to connect users and manage user data.

Status: Text set before the user searches for a meetup. This text conveys why a user wants to meet up, what a user wants to talk about, or what is on the user's' mind.

User: A person who downloads the app to find other users. User1 refers to the user performing the action, while User2 refers to a user who receives the result of that action without initiating the action themselves.

Scope

The app will run on the Android platform. The app will utilize the Android framework to develop and easy-to-user user interface that facilitates meeting new people spontaneously. The app will use GPS to determine the location of the user and will access the user's contact list to make it easy to include their friends in meetups as well as make it easy to save the contact information of people a user meets.

The project will require a backend server to store user information and to connect users to each other. The server will have to use user feedback from previous meetups to determine which pair of users would mostly produce a meetup that goes well.

Goals

The final goal of the application is to have everything mentioned in this requirements document functioning precisely as intended, with each feature being implemented in an efficient, easy-to-use way. We hope users will be able to use our application to form lasting friendships, by finding users similar to themselves.

Our goals with the project are, in descending order of priority, having profiles implemented and working with ability to view another profile, matching users with profiles, giving the user the ability to manage their profile after creation, implementing meetups (both the meetup itself and the ability to provide feedback as a result of the meetup), having a meetup history users can view, implementing contact list integration, allowing a user to invite contacts to meetups, implementing GPS integration to allow users to more easily find each other after matching, implementing a block list, and finally, optimization. Optimization will ideally be performed throughout the course of the application's development, rather than at the end of the development cycle.

One stretch goal is to have a working machine-learning based algorithm for matching users in a more effective way than simply comparing similarity of tags. Since time is limited, it is likely there will not be enough time to fully teach the algorithm, the goal is to have at the very least a framework in place that learns behind the scenes and uses user feedback from matching to train itself. The implementation of this goal is dependent on time and difficulty of the implementation itself. Another is the implementation of sponsored locations, which are currently on the roadmap but not a priority feature.

II. PROJECT DESCRIPTION

Profile Creation

Once the app has been downloaded, the user is prompted to create an account or log into an existing account on the initial screen. The user can also be able to reactivate a deactivated account at this point.

Once the user has chosen their name, username, password, and phone number (which will most likely be the device they are currently on), the account will be created and the user will be logged in.

Profile Management

The user can edit their account information like their description, profile picture, phone number, and their interests. The user can also choose to deactivate their account so it can no longer be logged into.

By default, a user's "friend's list" will be their phone contacts. There will be a separate block list available for people to block users.

Meetup Creation

The user can create their own meetup, so they can invite people already in their contacts who have the app to also look for people to meet.

Meetup Search

Users can set their status indicating why they would like to meet up with someone or what they want to talk about before searching for people to meet.

Users and existing meetups can browse for other users and/or currently active meetups who are also looking for people to join. A user or meetup will receive a list of other users and/or currently active meetups who are also looking for people to join. From there the user can view the profiles of the those users and view the current status of those users. A user can toggle all in case they are willing to match with anyone. The user can also toggle which users the user is willing to join in a meetup.

Once a combination of two user/meetups both toggle that they would like to join the other user/meetup, the two both receive a notification that the other is willing to join them and asking them if they would like to stop looking for matches to meet up with the other user/meetup. If both sides agree to meet up, they are removed from the search pool and form one combined meetup.

Meetup Management

After creating a meetup, a user can send invites to other users in their friends list to join the meetup.

When in a meetup, users can send messages to other users in their current meetup. Users in a meetup can also search for other users/meetups to join. A user can also choose to leave their current meetup.

Meetup History

The user can view a list of their previous meetups. The user can view the profiles of each user in that participated in meetup. They can also provide feedback on the meetups or the individual people to aide in finding desireable people from that user to meet.

View Profile

The user can view the profiles of other users. This allows a user to screen potential matches before toggling if they are willing to meet up with that person. The profile will contain information including a profile picture, the user’s name, the description of the user, and their interests.

A user can also choose to add the user of the profile to their block list to prevent meeting up with this user again. Additionally, the user can toggle a box that states whether it is okay to share phone number information with the user or not. If both users agree to share phone number information, the phone number information is displayed on the profile.

Settings Management

The settings menu allows the user to perform several miscellaneous functions. The user can deactivate, but not delete, their account. The account would be reactivated if the user attempts to access the app again. The user can also contact the developers through the settings menu.

Developer Management

The developer screen will be inaccessible to normal users. It will allow users with administrative privilege to view feedback sent to developers by users in the settings menu.

III. REQUIREMENTS

Functional Requirements

Profile Creation and Management

- A user needs to be able to create a profile
- A user needs to be able to manage and modify his or her profile, including preferences, tags, photos, and interests
- A user needs to be able to suspend his or her account
- A user needs to be able to disassociate his or her phone number from his or her account
- A user needs to be able to manage a Friends List
- A user needs to be able to manage a Block List

Group Search and Management

- A user needs to be able to search for groups
- A user needs to be able to cancel a search
- The application needs a way to grab the user's GPS location
- A user needs to be able to select to utilize his or her current location, or a nearby sponsored location
- A user needs to be able to match groups with a high degree of success
- A user needs an option to provide group feedback upon leaving a group
- A user needs an option to share contact info
- A user needs a way to view group history
- The application needs a method to track a user's group history

Settings and Miscellaneous

- A user needs a method of contacting the developers to provide feedback
- A user needs a way to report other users
- The application needs fake reporting options, such as "I didn't like this user", to reduce abuse of the report function
- The application needs to appropriately filter reports to prevent overloading developers with reports
- The application needs to properly punish offending users

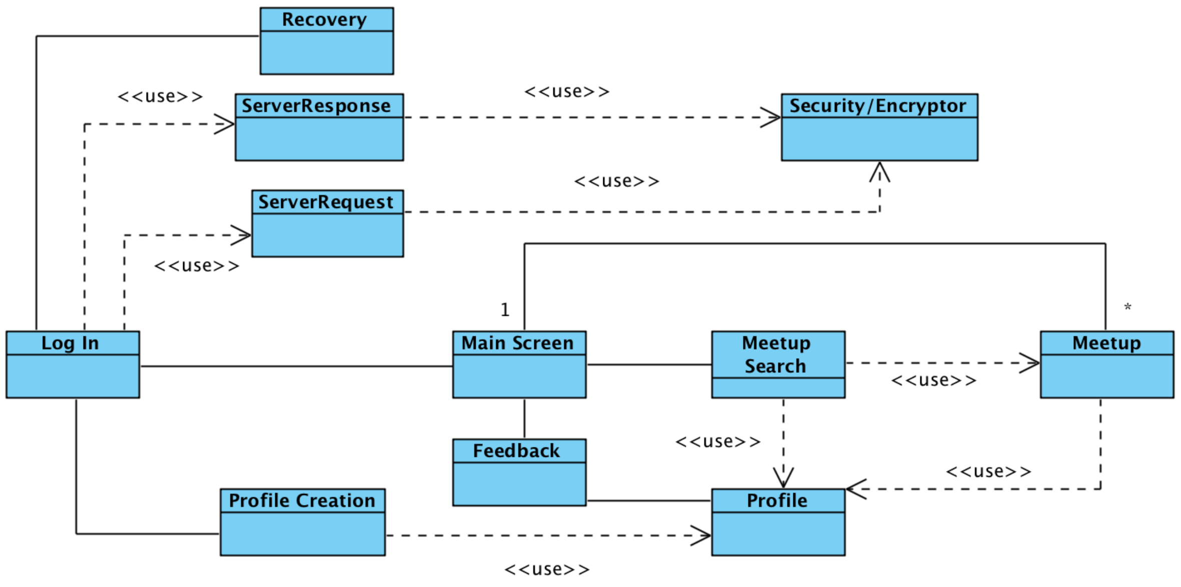
Developer Management

- The developers need a way to view report feedback

Non-Functional Requirements

- User passwords must be hashed appropriately
- The application needs a database that can hold usernames, passwords, profiles, etc
- The application must remain functional on older/lower-end Android phones
- The application must not unnecessarily drain battery
- The application must perform tasks (such as group searching, matching) in a reasonable amount of time

IV. CLASS DIAGRAM



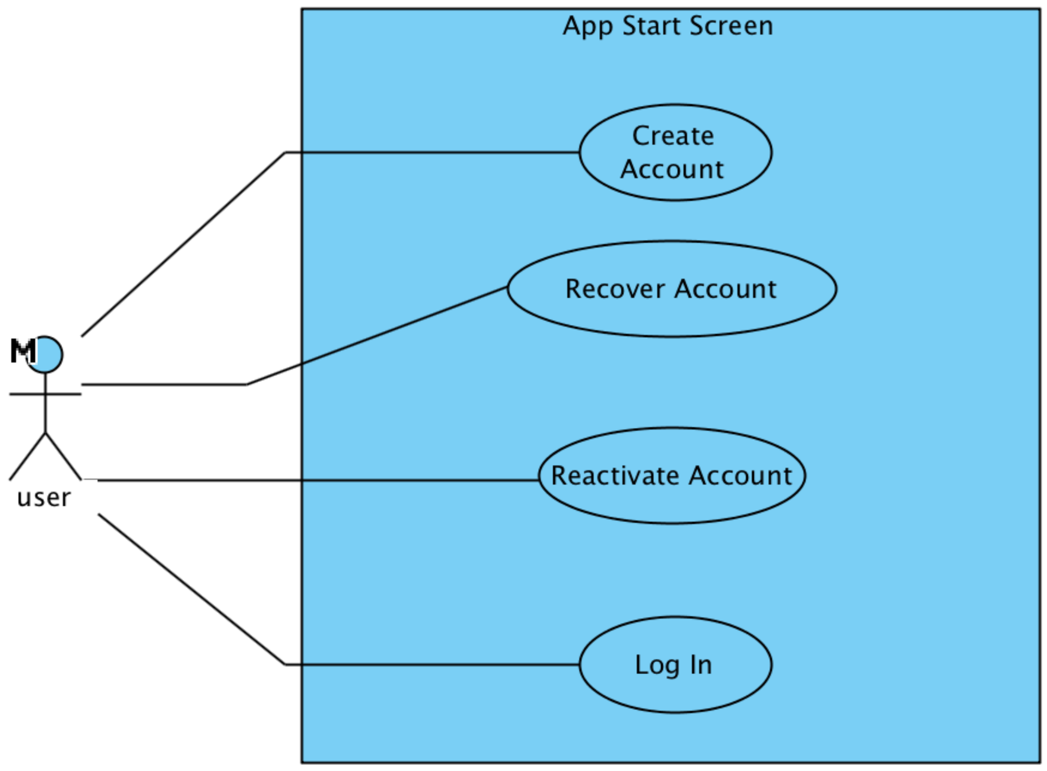
The diagram shows the initial draft of interaction between high-level classes.

The Security/Encryptor class will be used to encrypt and decrypt messages between the app and the server. The messages take the form of ServerRequest and ServerResponse. Classes like Log In, Profile Creation, and the Meetup Search will use ServerRequest and ServerResponse.

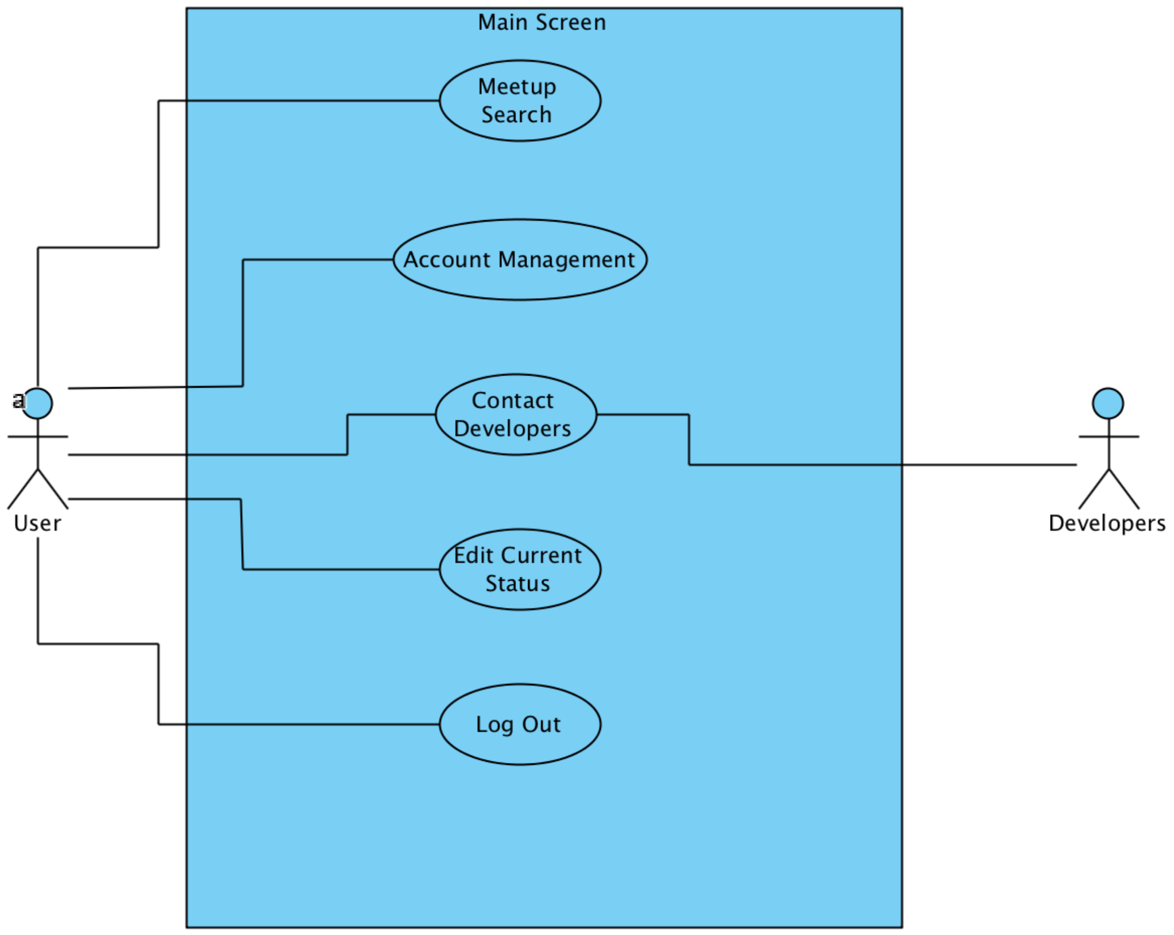
Meetup represents a group of users currently active in a meetup. Profile will hold the data for a particular user.

V. USE CASE DIAGRAMS

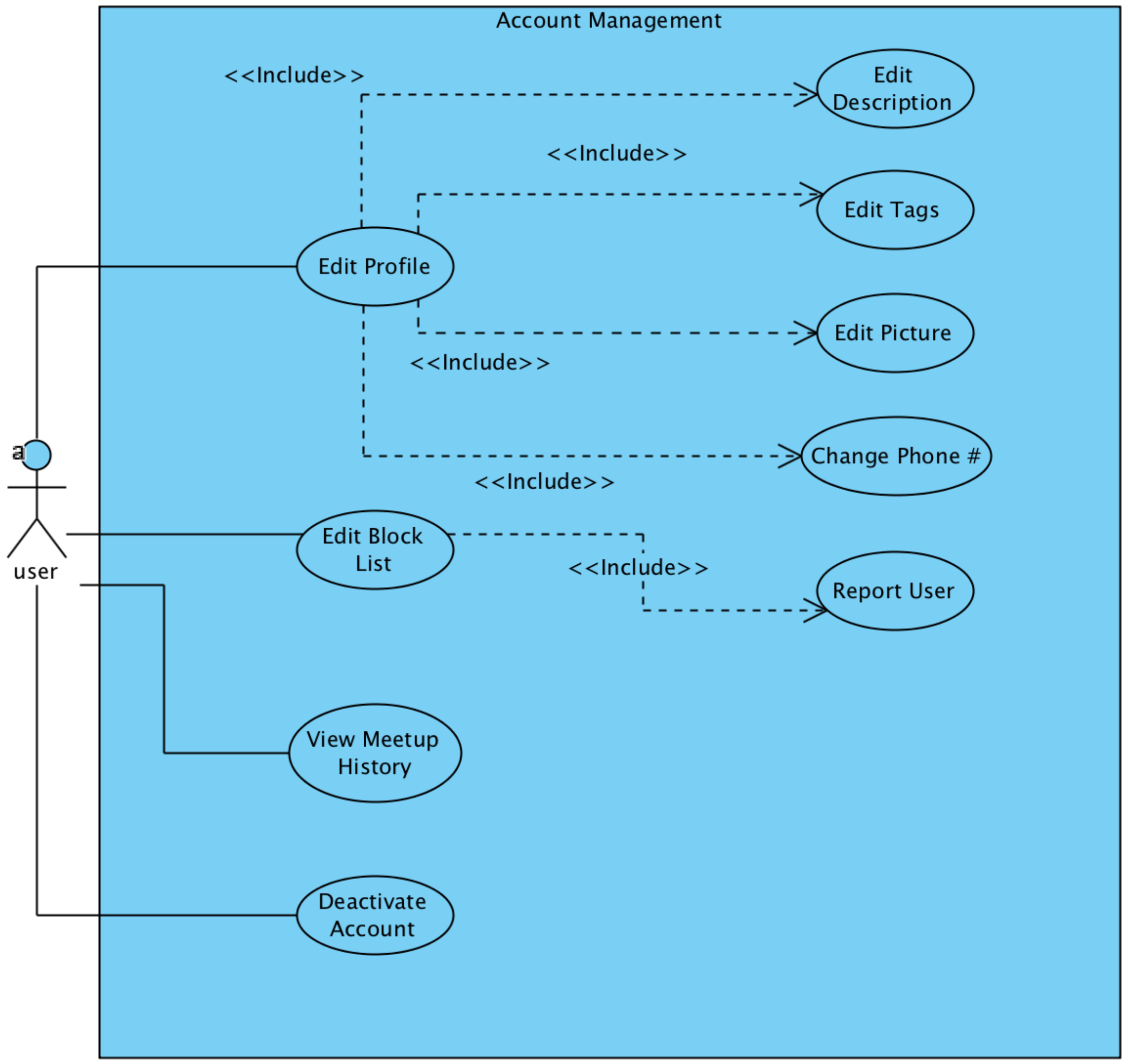
Log-In Use Case Diagram



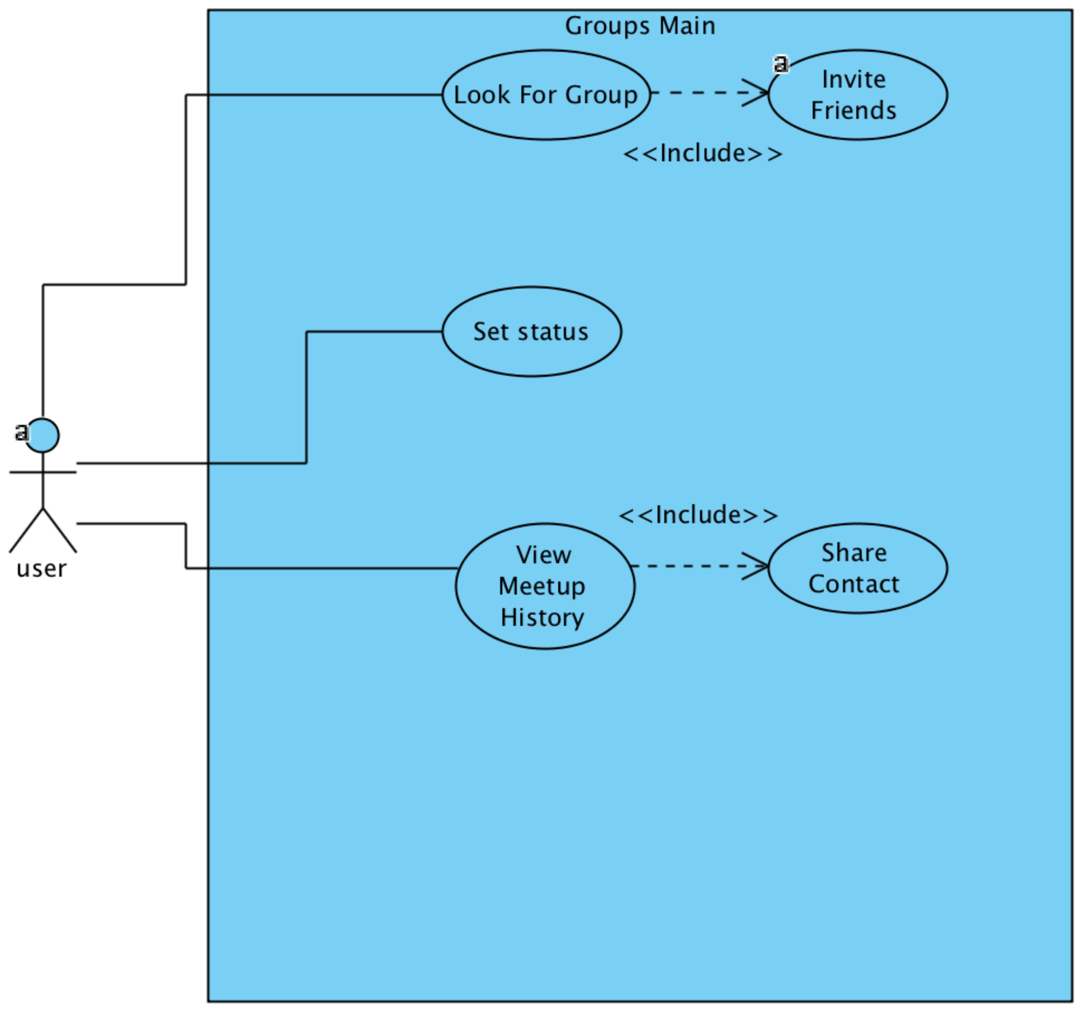
The Log-In Use Case encompasses activities a user can perform before logging into the application or creating an account. These are limited in scope to creating accounts, recovering/reactivating accounts, and logging in to an already extant account.



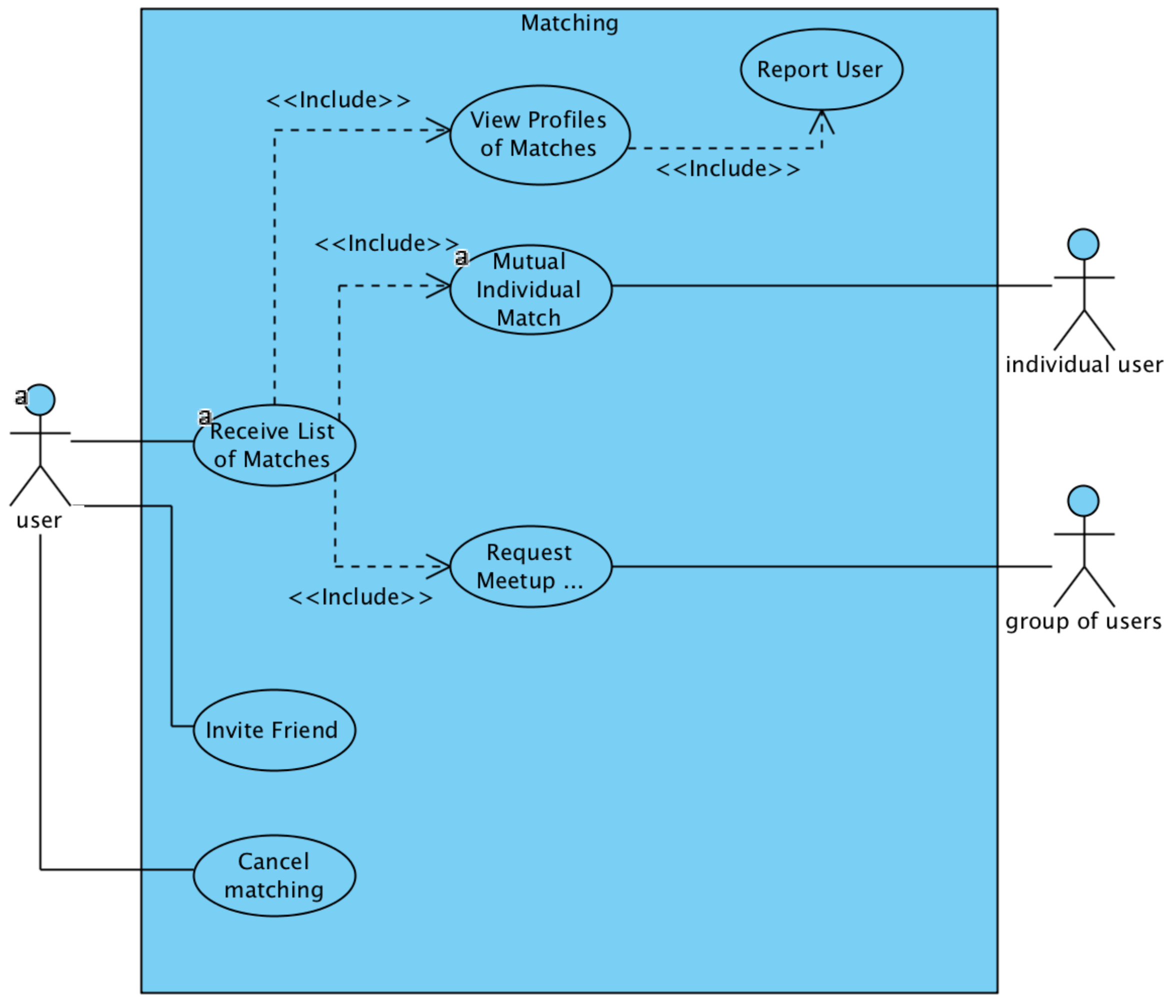
The **Main Screen** use case outlines what a user will be doing when on the primary screen of the application. These cases include searching for a meetup, managing one's account, contacting the developers with feedback, and editing one's current status for meetup searching. Logging out is also available, should the user wish to do so.



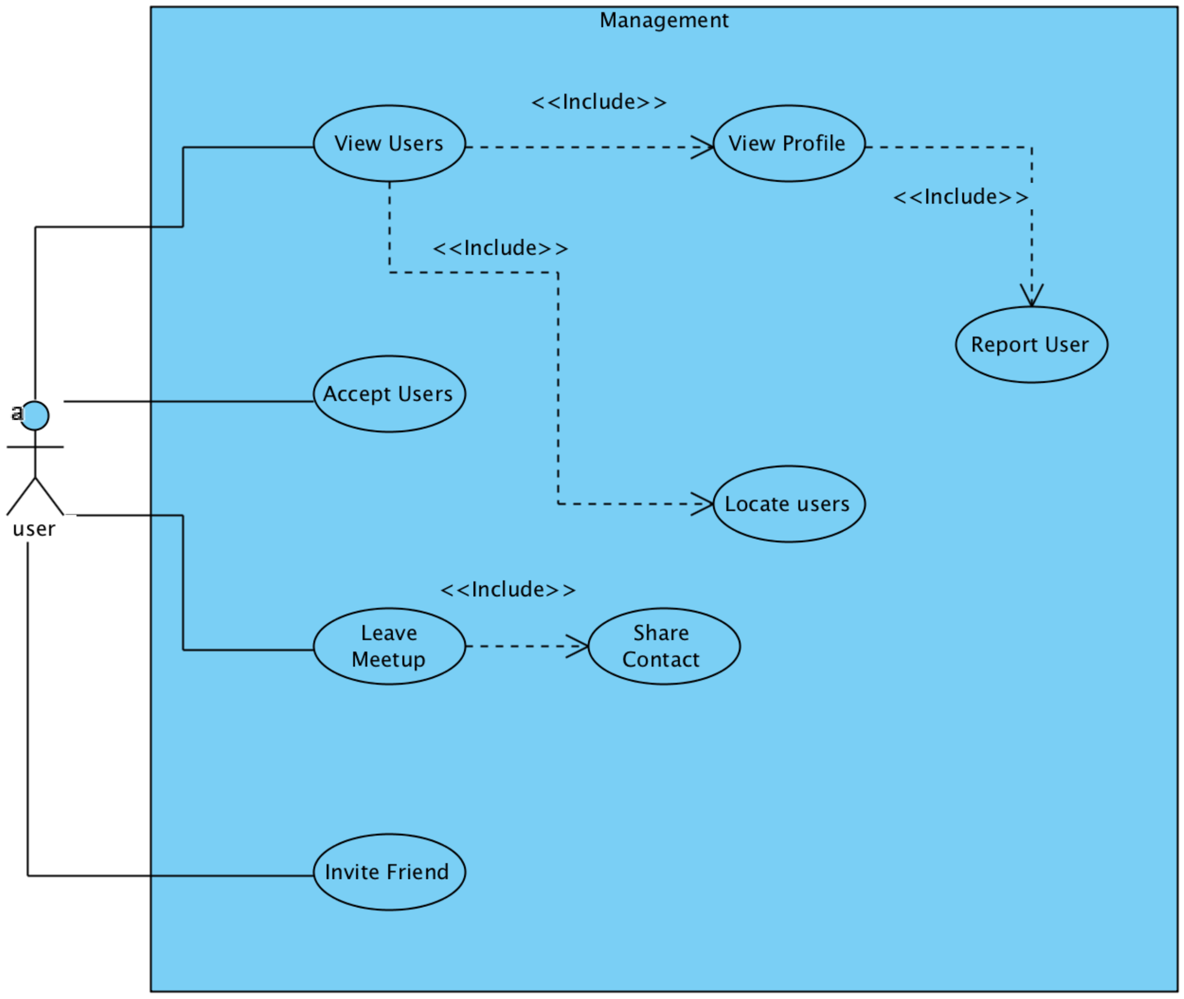
The **Account Management** use case encompasses what a user can do when managing his or her account. Options include editing one's profile, editing one's block list, viewing one's meetup history, and deactivating one's account. Reactivation is available on the main screen.



The **Meetups Main** use case encompasses the group/meetup related activities that a user can perform before or after searching for a meetup, without actually requiring the user to search for a group or be a part of a group. These are spread across different screens, but encompass the same general idea.



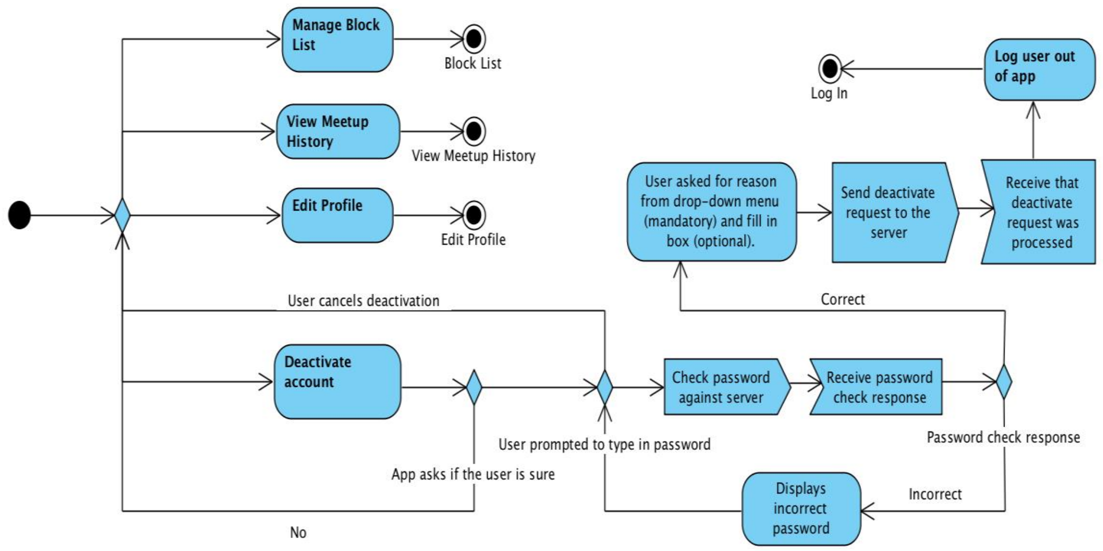
The **Meetup Matching** use case encompasses the activities a user can do during a search for a meetup. In addition to the standard matching, a user can invite friend(s) to immediately form a meetup, should the friend(s) accept.



The **In A Meetup** Use Case encompasses activities a user can perform whilst in a group. These are limited to accepting new users, viewing users currently in one's group, locating these users, leaving meetups, and inviting friends to the meetup. We hope to implement the ability for meetups to search for individual users or other meetups, but this is currently a stretch goal.

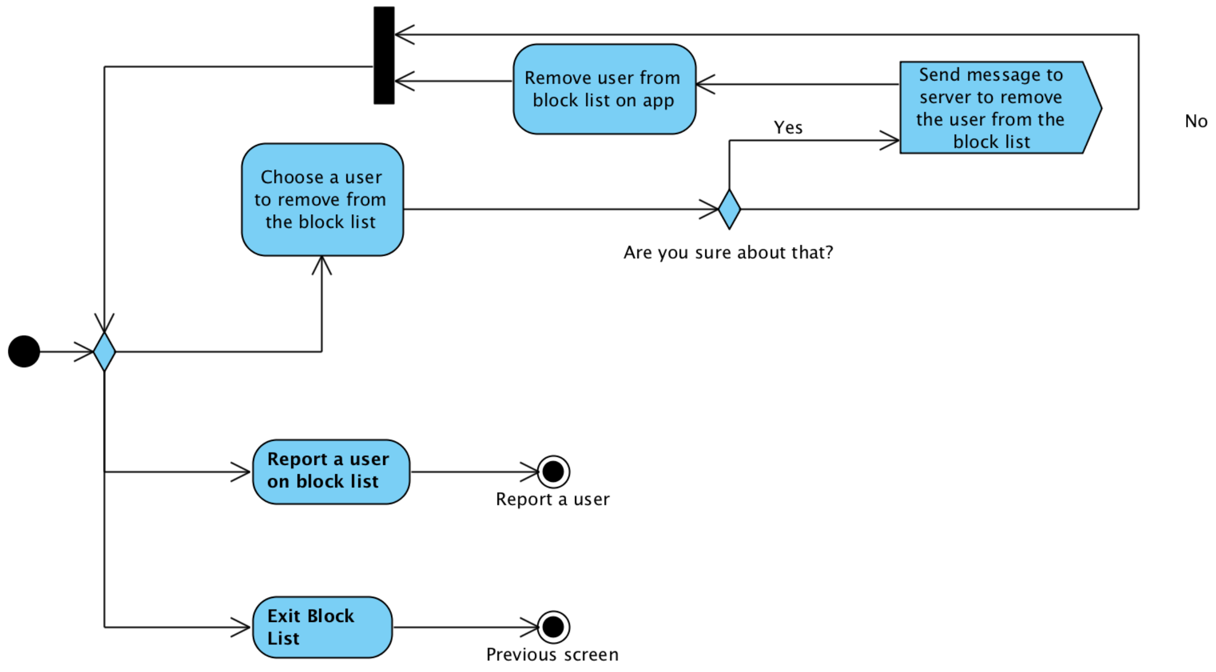
VI. ACTIVITY DIAGRAMS

Account Management Activity Diagram


Manage Block List
Block List
View Meetup History
View Meetup History
Edit Profile
Edit Profile
Log In
Log user out of app
User asked for reason from drop-down menu (mandatory) and fill in box (optional).
Send deactivate request to the server
Receive that deactivate request was processed
User cancels deactivation
Correct
Deactivate account
Check password against server
Receive password check response
User prompted to type in password
Password check response
App asks if the user is sure
Displays incorrect password
Incorrect
No

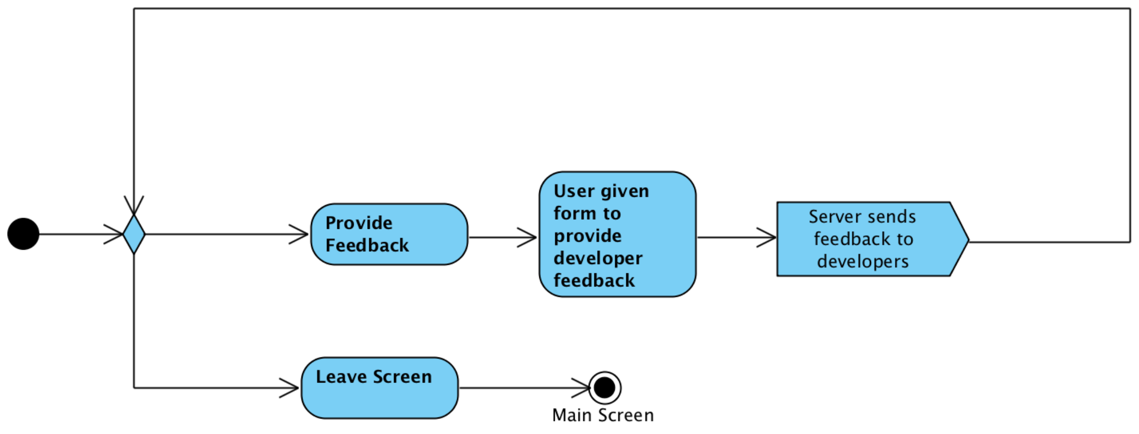
The user can choose to manage their list of blocked users, view their meetup history, edit their profile information, deactivate their account.

If the user chooses to deactivate their account, a dialog box asks for confirmation of whether the user is sure they desire to deactivate their account. The user must then type in their password and the server verifies if that password is correct. If the password is not correct, the user is prompted again to type in their password. Once the user types in their password correctly, they are prompted to optionally fill out a reason for deactivation. Next, a deactivation request is sent to the server, the server processes the deactivation request, and the app receives that the server processed the request. The user is then logged out of the app.


Remove user from block list on app
Send message to server to remove the user from the block list
Yes
No
Choose a user to remove from the block list
Are you sure about that?
Report a user on block list
Report a user
Exit Block List
Previous screen

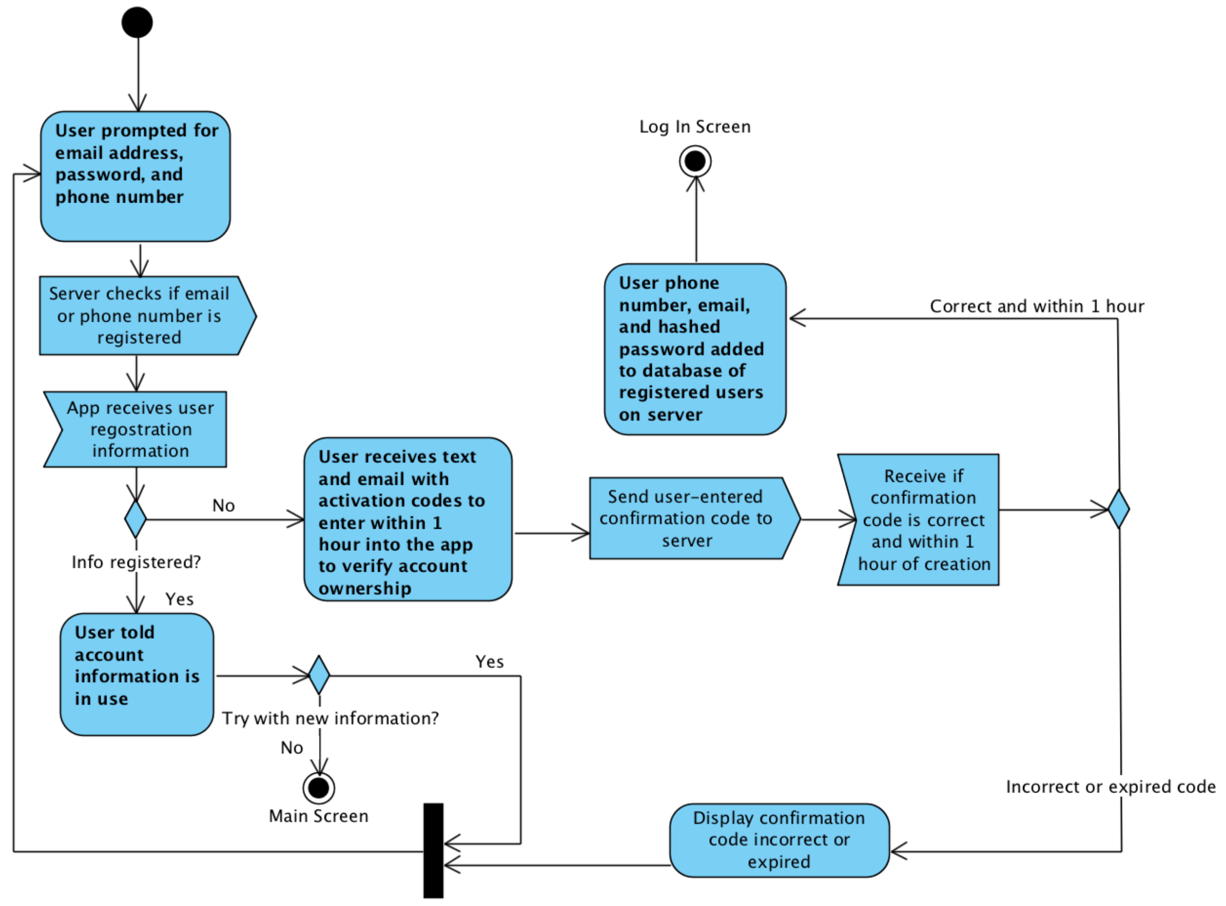
The user can choose a user to remove from their block list. A confirmation dialog then confirms if they would like to remove the user from their block list. If the user wants to remove the blocked user from the block list, the server is notified on this and the blocked user is removed from their block list. A user will also be able to report a user on his or her block list, without having to first remove the user from their block list.

Contact Developers Activity Diagram

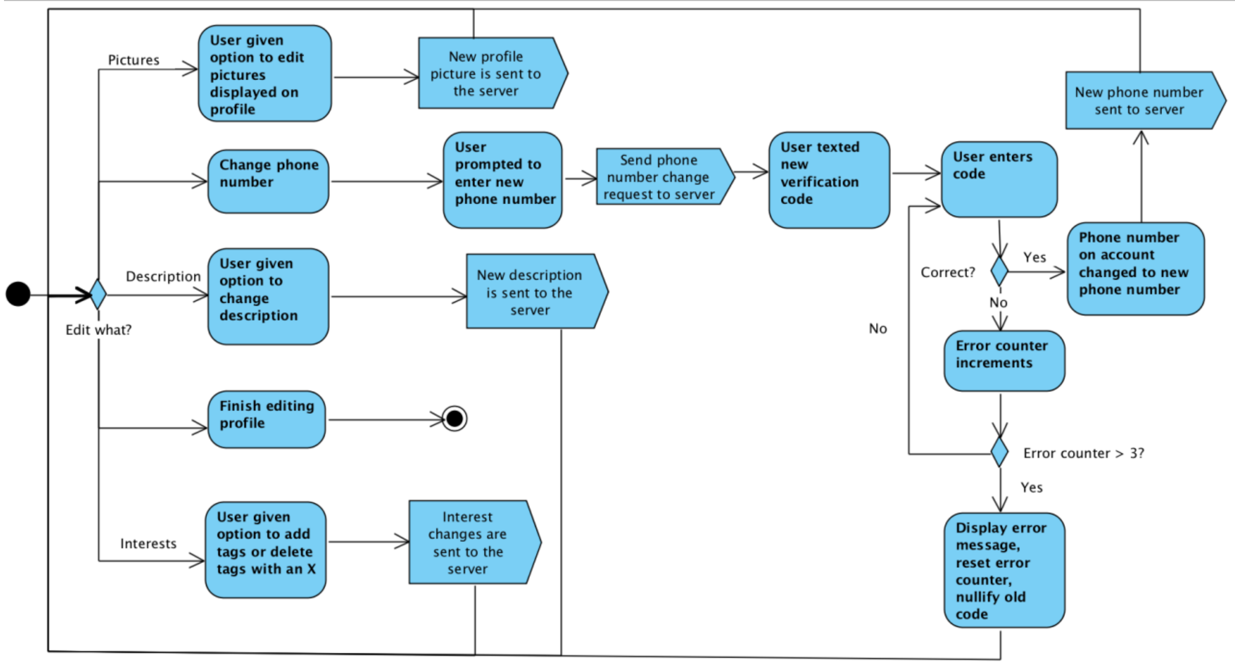

Provide Feedback
User given form to provide developer feedback
Server sends feedback to developers
Leave Screen
Main Screen

The user can choose to provide feedback on the app and is prompted to enter their feedback into a text form. The feedback is sent to the server when the user finishes filling out the form.

Create Profile Activity Diagram

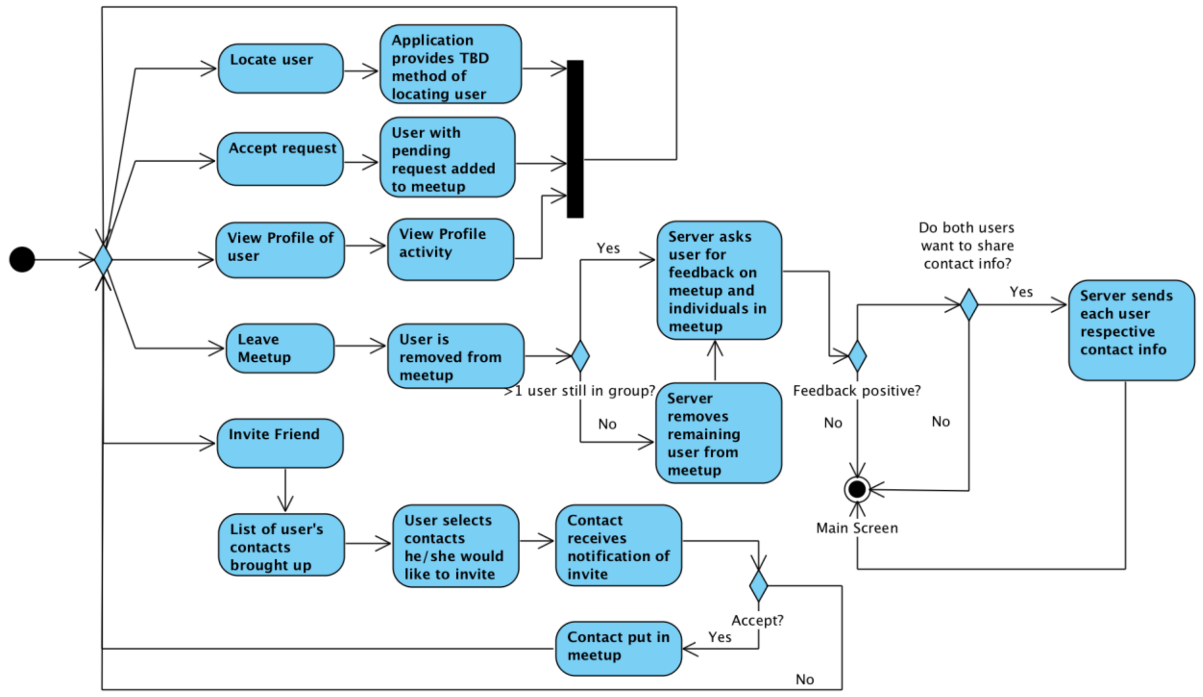

User prompted for email address, password, and phone number
Server checks if email or phone number is registered
App receives user regostration information
No
Info registered?
Yes
User told account information is in use
Try with new information?
Yes
No
Main Screen
User receives text and email with activation codes to enter within 1 hour into the app to verify account ownership
Send user-entered confirmation code to server
Receive if confirmation code is correct and within 1 hour of creation
Log In Screen
User phone number, email, and hashed password added to database of registered users on server
Correct and within 1 hour
Incorrect or expired code
Display confirmation code incorrect or expired

The user is prompted for their email address, password, and phone number. The server will check if the email or phone number is current associated with a user account. If either is associated with a current user account, the user is informed of this and is allowed to restart account creation or leave account creation. Otherwise, the user will receive email and phone confirmation codes that they must input into the app to verify that the email and phone number provided is their email. The confirmation codes expires in an hour. When the user types in their confirmation code, the server checks if the codes have been entered within an hour from when it was created and if the codes are correct. If so, the account is created and the user is sent to the login screen.


Pictures
User given option to edit pictures displayed on profile
New profile picture is sent to the server
Change phone number
User prompted to enter new phone number
Send phone number change request to server
User texted new verification code
User enters code
New phone number sent to server
Correct?
Yes
Phone number on account changed to new phone number
No
Error counter increments
Error counter > 3?
Yes
Display error message, reset error counter, nullify old code
No
Description
User given option to change description
New description is sent to the server
Edit what?
Finish editing profile
Interests
User given option to add tags or delete tags with an X
Interest changes are sent to the server

The user can choose which information they would like to change. For all information besides phone number, the changes can be made in the app and then the server updates the user profile. For the phone number, the user enters the desired new phone number and a phone number change request is sent to the server. The server then texts a verification code to the new number. The user has three attempts to type in the correct verification code. If the verification code is correct, the new phone number is sent to the server and the phone number is changed. Otherwise, the user fails and is taken back to edit profile.

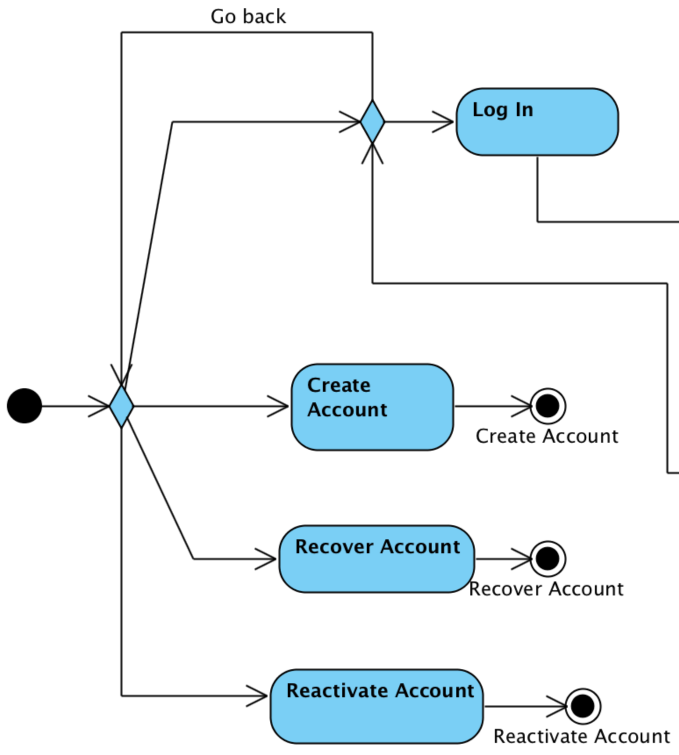
Meetup Management Activity Diagram


Locate user
Application provides TBD method of locating user
Accept request
User with pending request added to meetup
View Profile of user
View Profile activity
Leave Meetup
User is removed from meetup
>1 user still in group?
Yes
No
Server asks user for feedback on meetup and individuals in meetup
Server removes remaining user from meetup
Feedback positive?
No
Do both users want to share contact info?
Yes
No
Server sends each user respective contact info
Main Screen
Invite Friend
List of user's contacts brought up
User selects contacts he/she would like to invite
Contact receives notification of invite
Accept?
Yes
No
Contact put in meetup

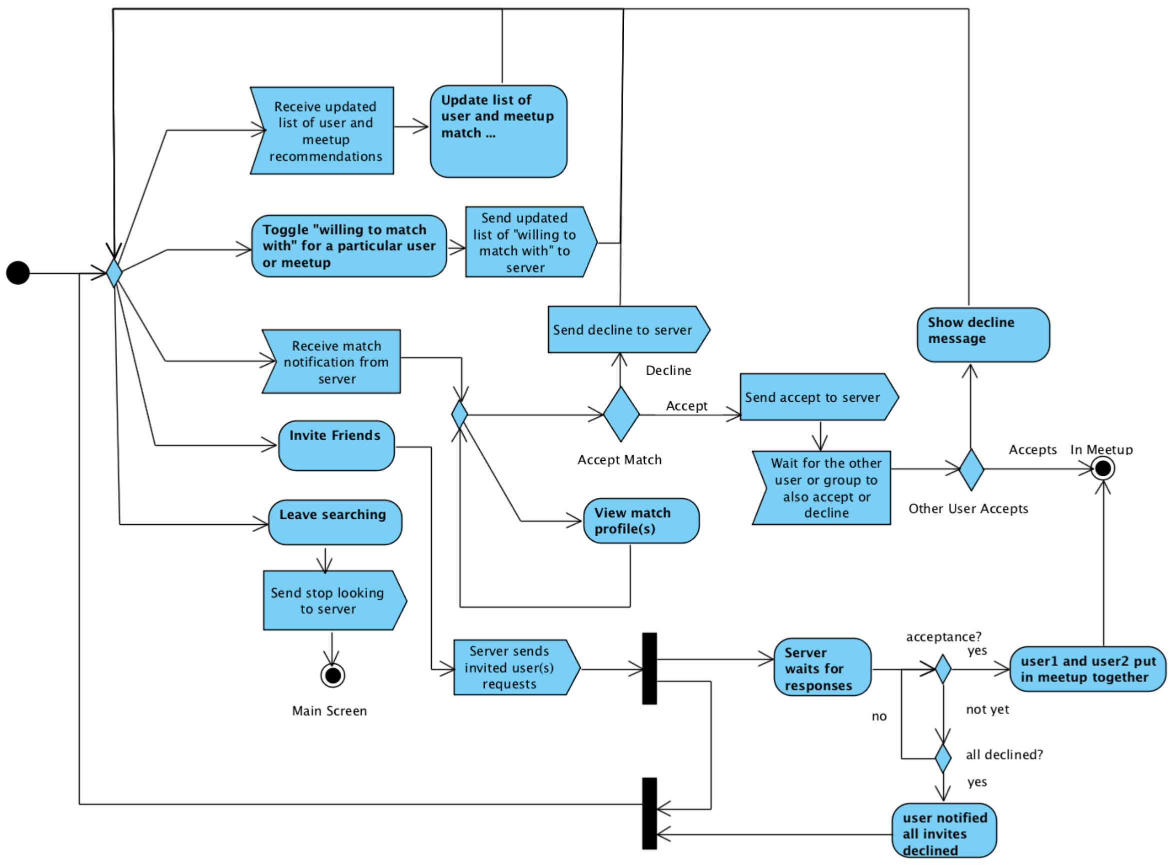
While in a meetup, the invite users in their friends lists to join the meetup, view profiles of users in the meetup, and leave the meetup. Users can also locate other users who are in the meet up. The method used to locate people has not been decided yet. Potential methods are using gps to create a compass, making the screen glow a certain color that users hold up to find each other, and allowing people to text each other through the app. The user can also search for other users to bring into the meetup in the background.

When a user invites a friend to the meetup, a list of the user's friends comes up. The user selects all the people they want invited to the meetup. The server receives this list and sends invite notifications to the friends. When a friend accepts the request, they are put into the meetup.

When a user leaves a meetup, if the meetup is now empty, it is closed and finished. The user is then prompted to send feedback to the server about how the meetup went. This feedback can be used later on to help sort potential future meetups.

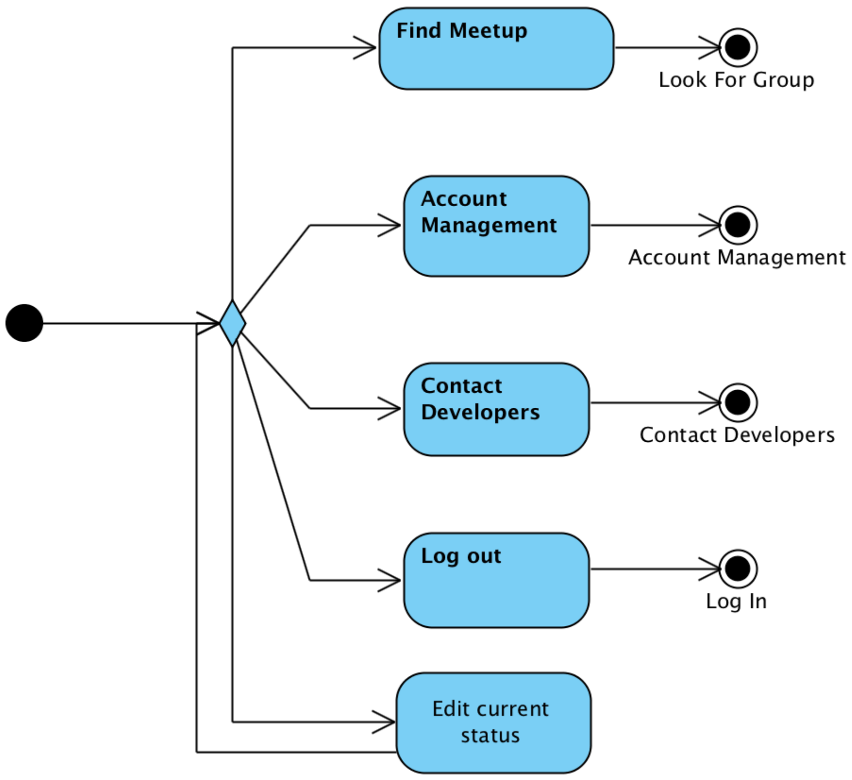


The user can choose to create their account, log into an account, recover an account, or reactivate an account. If the user chooses to log into an account, the user types in their login information and the information is sent to the server. If the information is valid, the server allows the user to log in and is taken inside the app. If the information is not valid, the user is prompted to retry entering their information.

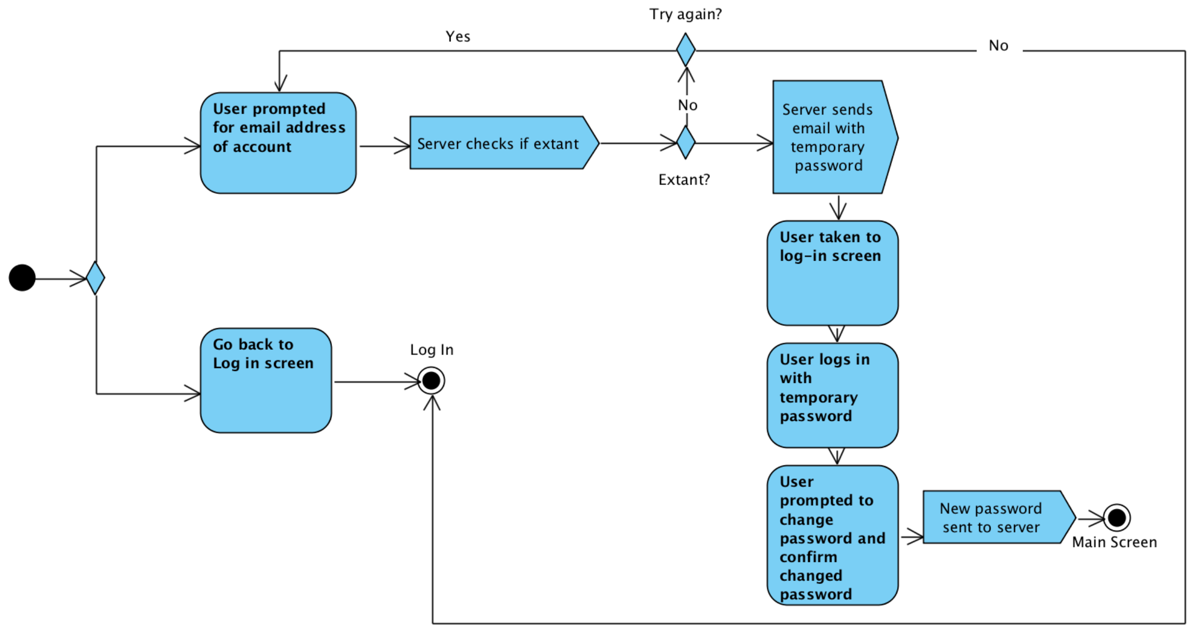

Receive updated list of user and meetup recommendations
Update list of user and meetup match ...
Toggle "willing to match with" for a particular user or meetup
Send updated list of "willing to match with" to server
Send decline to server
Show decline message
Receive match notification from server
Decline
Accept
Send accept to server
Invite Friends
Accept Match
Wait for the other user or group to also accept or decline
Accepts
In Meetup
Other User Accepts
Leave searching
View match profile(s)
Send stop looking to server
Main Screen
Server sends invited user(s) requests
Server waits for responses
acceptance?
yes
user1 and user2 put in meetup together
no
not yet
all declined?
yes
user notified all invites declined

When searching for a meetup, the user will continuously receive a list of user and meetup recommendations from the server. The user can toggle which users and/or meetups they are willing to join. The user will receive match notifications from the server when a different user/meetup that this user is willing to meetup with also is willing to meetup with this user. If the user wants to go finalize the match, the user confirms to meetup. If not, the searching continues. If a user/meetup does not confirm to meetup, then a decline messages is sent to the other user. The user can also choose to stop searching so they will no longer match with other users. Users can invite friends during this stage, to start their own meetup. Users will send out invites to contacts, and then go back into the normal meetup searching. When a contact accepts, the user will be pulled from wherever they are in the activity and placed into the “Meetup Management” activity, along with their invited contact.

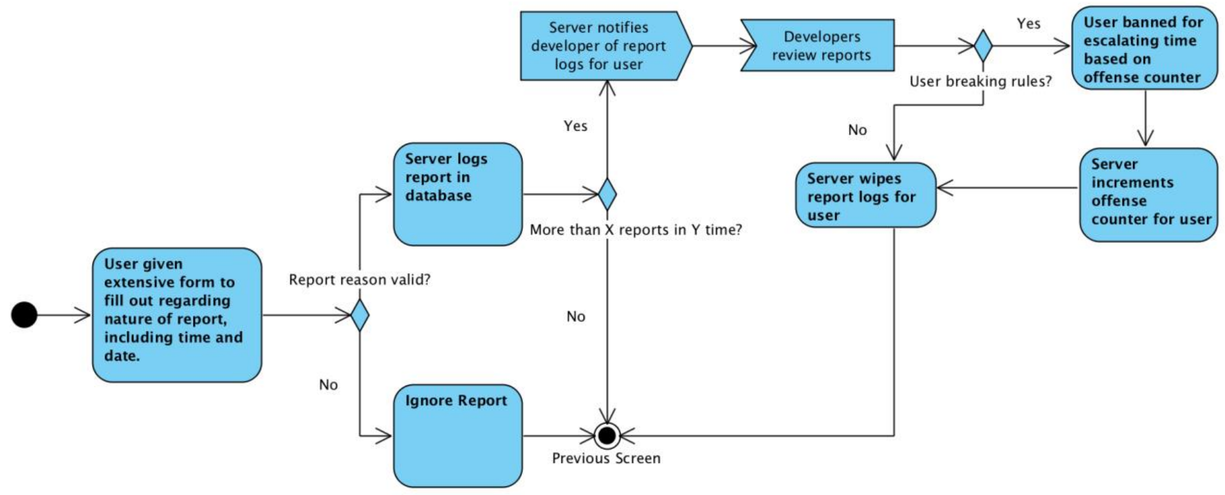
Main Screen Activity Diagram



From the main screen, the user can choose to start searching for a meetup, manage their account, contact the developers, log out, and change their status.

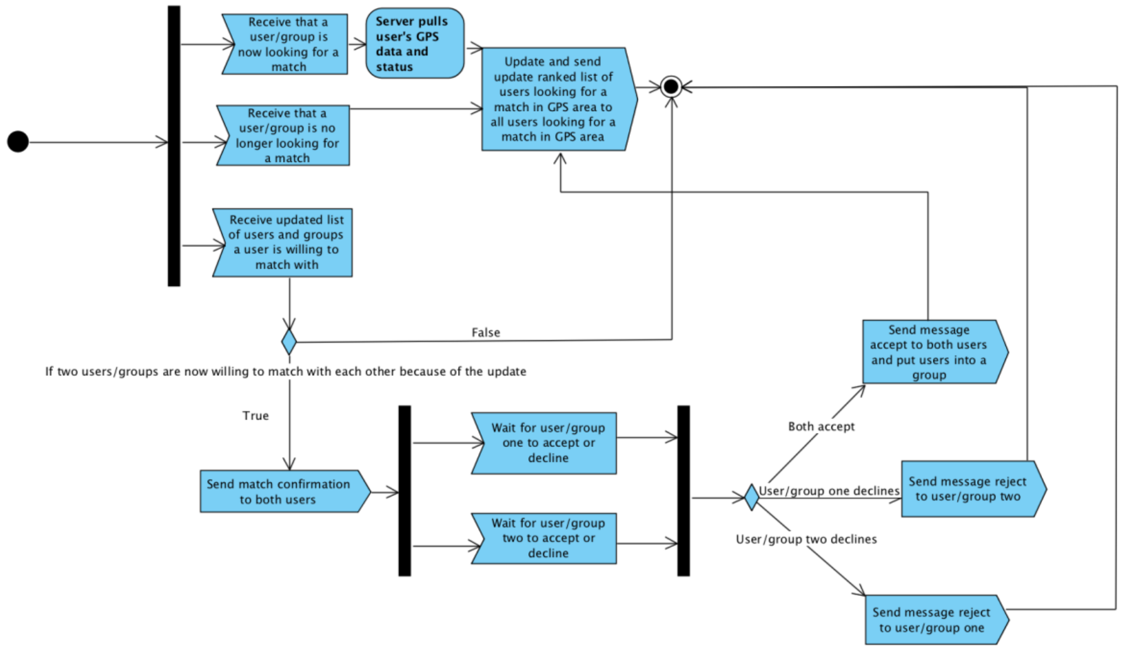

Try again?
Yes
No
User prompted for email address of account
Server checks if extant
No
Extant?
Server sends email with temporary password
User taken to log-in screen
Go back to Log in screen
Log In
User logs in with temporary password
User prompted to change password and confirm changed password
New password sent to server
Main Screen

To recover an account when the password is forgotten, the user sends a recover account request to the server for a specified email. The server sends a temporary password to the email associated with the user account. The user is then taken to the login screen. After the user logs in using the email and temporary password, the user is prompted to then change their password. The new password is sent to the server after confirmation. Account reactivation happens the same way.


User given extensive form to fill out regarding nature of report, including time and date.
Report reason valid?
No
Server logs report in database
Ignore Report
Yes
More than X reports in Y time?
No
Previous Screen
Server notifies developer of report logs for user
Developers review reports
User breaking rules?
Yes
No
User banned for escalating time based on offense counter
Server increments offense counter for user
Server wipes report logs for user

To report a user for abuse of the app, the user selects the "report user" option from the other user's profile. The possible options for reporting include for misconduct such as hateful speech, unwanted lewdness, or stalking. The menu will also include dummy options such as "I did not like the user" to prevent useless reports.

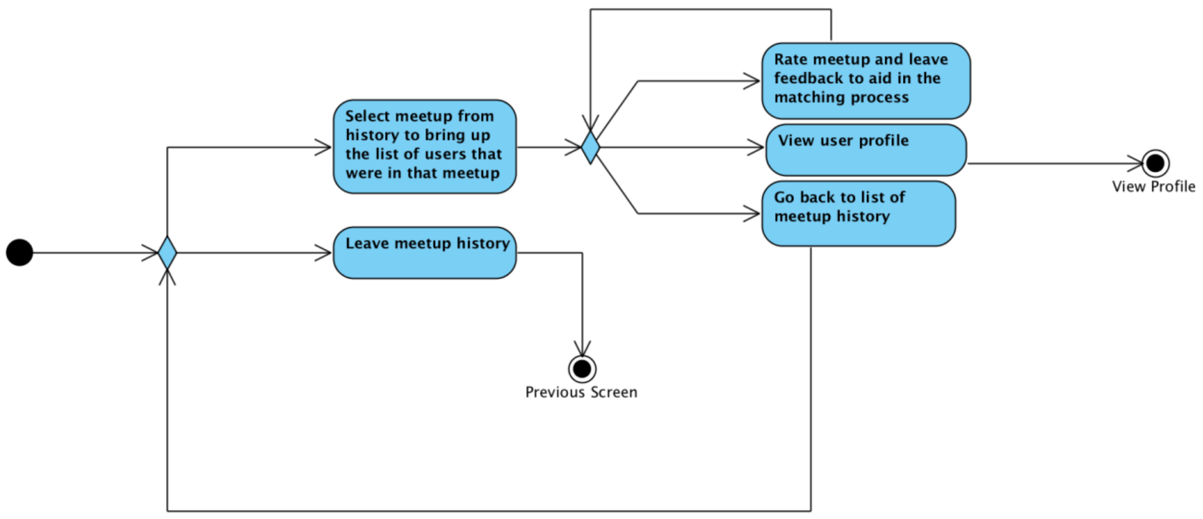
Server Meetup Search Activity Diagram


Receive that a user/group is now looking for a match
Server pulls user's GPS data and status
Update and send update ranked list of users looking for a match in GPS area to all users looking for a match in GPS area
Receive that a user/group is no longer looking for a match
Receive updated list of users and groups a user is willing to match with
False
If two users/groups are now willing to match with each other because of the update
True
Send match confirmation to both users
Wait for user/group one to accept or decline
Wait for user/group two to accept or decline
Both accept
Send message accept to both users and put users into a group
User/group one declines
Send message reject to user/group two
User/group two declines
Send message reject to user/group one

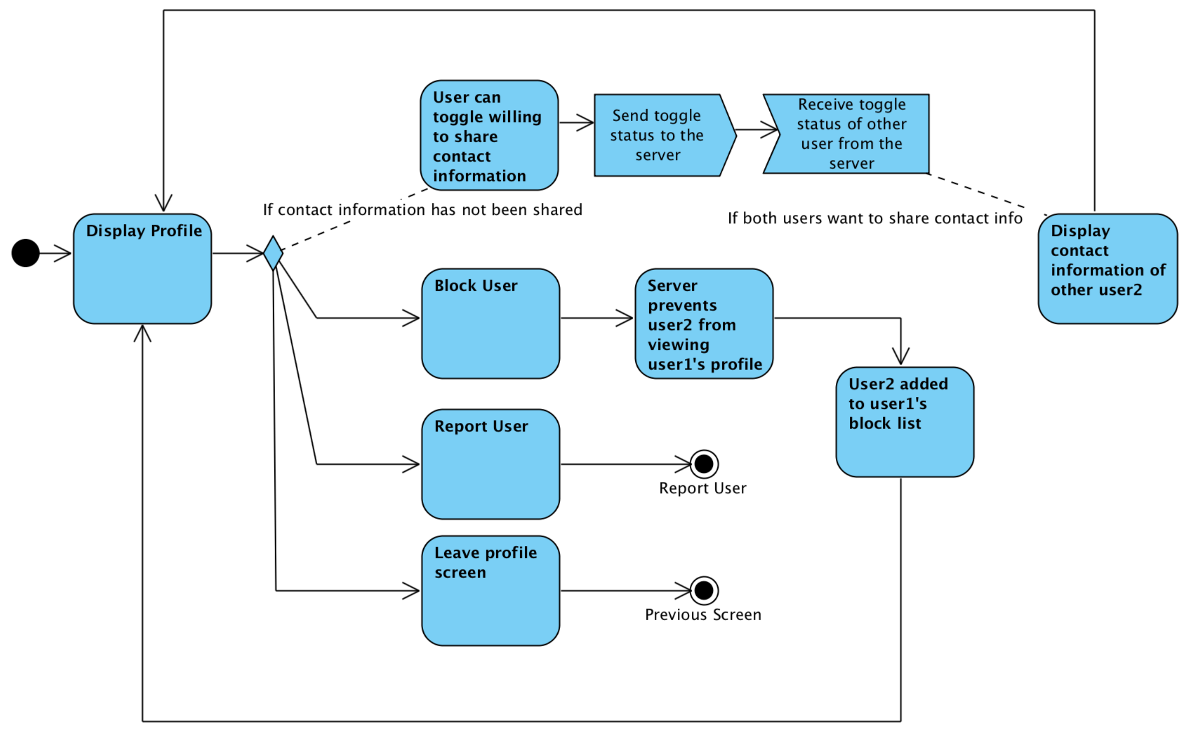
For creating matches, the server will send a list of ranked meetups that a user could match with to a user who sends "a start searching request" to the server. This will pull the user's GPS data and status, to be used in the searching process. The server also adds that user to the list of users looking for a meetup. If a user cancels searching for a meetup, the user will be removed from the list of users looking for a meetup.

When users change the list of users they are willing to match with, the server checks if two users are now willing to match with each other. If so, confirmation notifications are sent to each user. The server waits for both users to accept or decline. It should also timeout if the a user does not respond. If both users/meetups accept, then they are added to one meetup and are removed from searching for a meetup. If a user declines, the other user is sent a decline notification.

View Meetup History Activity Diagram


Rate meetup and leave feedback to aid in the matching process
Select meetup from history to bring up the list of users that were in that meetup
View user profile
View Profile
Go back to list of meetup history
Leave meetup history
Previous Screen

The app displays a list of previous meetups and the user can choose a meetup to further inspect. From there, the user can view the profile of anyone user who participated in the group, or leave feedback on the group or individuals in the group.


User can toggle willing to share contact information
Send toggle status to the server
Receive toggle status of other user from the server
Display Profile
If contact information has not been shared
If both users want to share contact info
Display contact information of other user2
Block User
Server prevents user2 from viewing user1's profile
User2 added to user1's block list
Report User
Report User
Leave profile screen
Previous Screen

The app displays the profile of the viewed user either from a cache or from pulling the data from the server. The user from there can block the viewed user, report the user, and toggle whether the they want to share contact information with the viewed user. If both users toggle that they desire to share contact information with each other, then contact information is displayed instead of the share toggle.