6.170 Project Design

Team members: Yixin Li, Yihua Li, Michael Wee, Faith Keza
Website name: Trippy

1. Overview(Purpose and goals)

1.1 Brief Description

An app that lets users share their availability, places of interests and budget range, create and organize trip groups and plans with friends.

1.2 Key goals and purpose

- Facilitate the formation of trip groups through easy sharing of information;
- Facilitate the communication inside a trip group to plan the activities and logistics of a trip.

1.3 Motivation for development

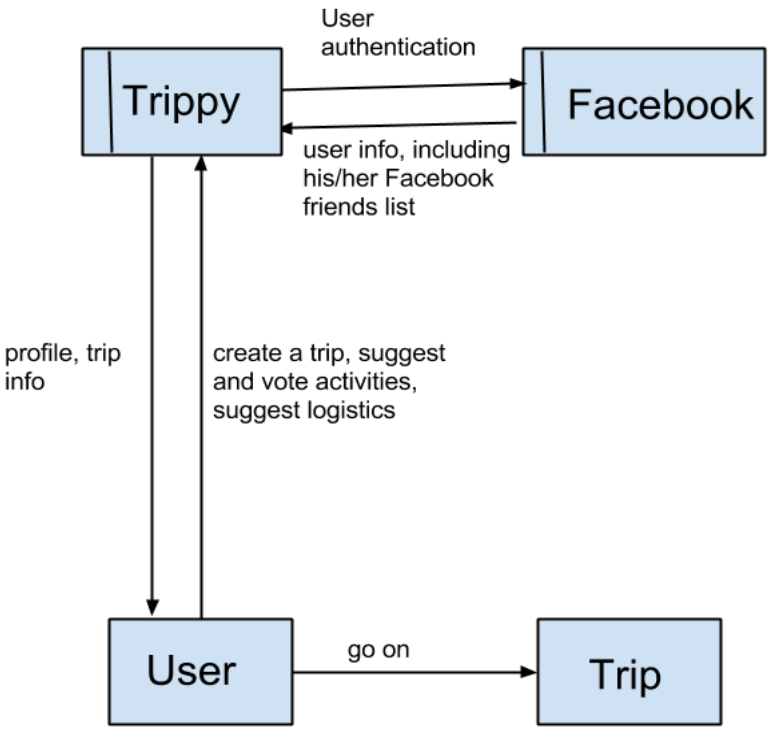
Friends use email, text, Facebook messages, phone, and in-person communication to plan and organize trips. This gets very hard as number of the group members increase, with many moving parts such as dates, destinations, budgets, activities, and so on. The existing solution includes following:

- Doodle / WhenIsGood
  These websites allow users to create a poll of availability for an event. Then users are polled to determine the best time and date to meet. Also, the websites show common free times among all users. They make the scheduling of events easy. However, they have no specific relation to travel.
- Triporama
  Triporama let allows you to plan a trip and to invite friends and family members to join. Users can also create trip plan and then invites others to join the trip group. However, the users need to create a plan first without any knowledge of other's availability, interests or budgets. In addition, the site looks over-complicated to use.
- TripIt Travel Organizer
  TripIt automatically creates a detailed itinerary after user forwards travel confirmation email to the app. It helps one individual organizes his/her trip. However, it does not help to coordinate group travel.

As a result, the workflow for a user to organize a trip with friends would be: sending emails/making phone calls asking friends' interests and availability; perhaps using Doodle to find a common free time; setting up a trip plan Triporama, inviting friends to join and add any activity; finalizing the trip plan maybe through TripIt or by sending emails/calls.

Our solution is different from above solutions in that we aim to provide an easy and integrated way for friends to share trip-related details such as when they are free, their interests and rough budget range. As a result, before creating a trip group, a user has a general idea of his/her friends' availability and interests. Also, we aim to improve simplicity in group discussion of the logistics of the trip and which activities to do for a trip. So there is higher probability that the trip becomes reality.

2. Context diagram



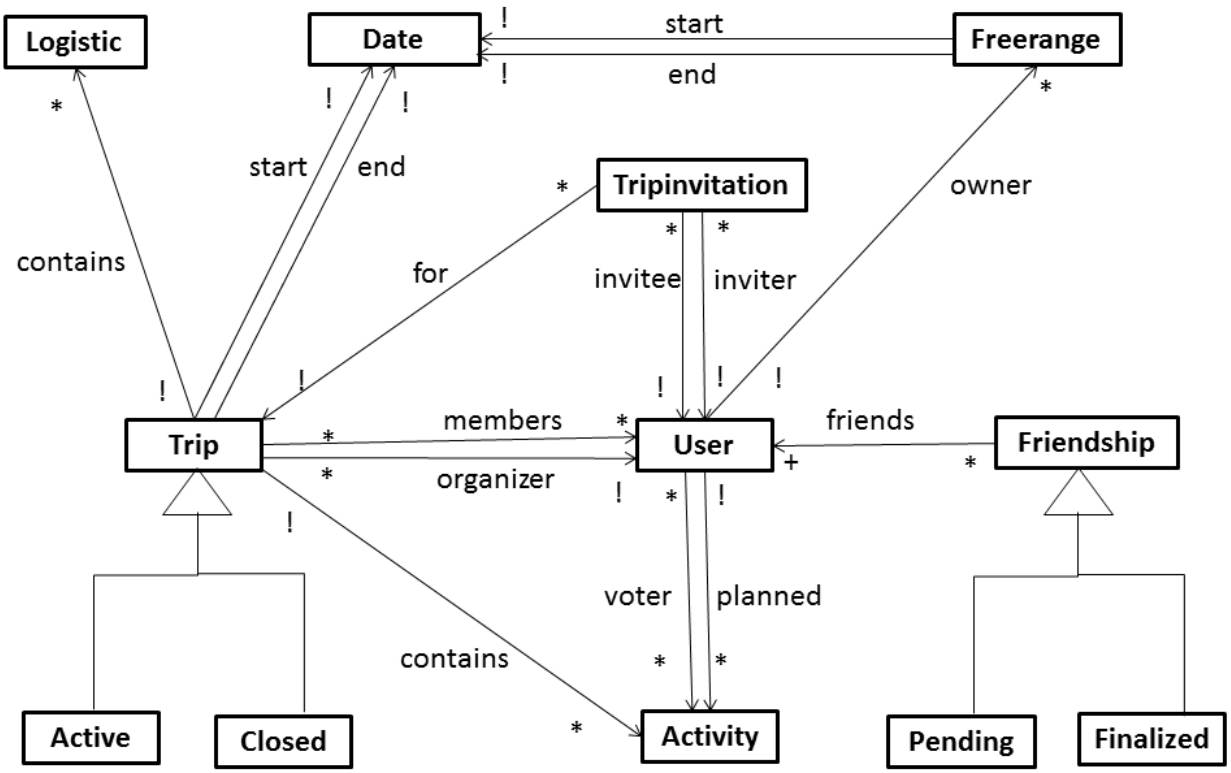
3. Key Concepts

Trip: A journey that a group of people wish to go on together. Each trip has a destination, dates, estimated cost per participant and participants.
User: An individual user who desires to plan a trip with friends and use the app.
Activity: A proposed activity for the group to do on the trip.
Logistic: A detail of the trip that is related to operational aspect of the trip, such as flight, hotel.
Vote: A user votes on activities for the group to go on.

Additions:

Price: The cost of an activity, travel plan, and overall cost for the trip.
Free range: A block of continuous dates during which the user is free for travelling.

4. Data model



Note: A friendship consist of two users as friends with each other.

5. Feature descriptions (Changes are in italic)

- **User authentication**: Users could log in through Facebook and sign out.
- **Friendship management**: Users could invite Facebook friends *and add other app users as friends by entering their email address. Users could also decline/accept friendship invitations.*
- **Profile sharing**: Friends could see each other's profile, including free dates, destinations interested in going and acceptable budget range.
- **Trip creation**: Users could start a trip and invite friends to join that trip *with the option of filtering friends to invite based on their specified free range dates and their acceptable budgets. Users could decline/accept trip invitations.*
- **Trip finalization**: *Users could finalize a trip at a certain point, not allowing any more changes.*
- **Activity planning**: Users suggest activities to do on a trip and vote for their desired activities. Users can upload pictures and descriptions of various activities to convince their friends to vote for it on the trip since users could *rank activities based on the number of votes. Activities have prices associated with them that add into the budget.*
- **Logistics organizing**: Users can sort out final logistics of the trip including hotel information, flight bookings, final dates, etc.

6. Design challenges

1. Friendship management: how could users add and confirm friends with each other?
   Option 1: Users manually invite other users in the site as friends by typing in the email address of friends. The app sends friend invitations to those users. This friendship becomes finalized when the receiver accepts invitation.
   Cons: Users' friends might not have registered in the site. However, as we would need to have friendship model, this makes seeing other friend's profile information easier.
   Option 2: through Facebook: We could use Facebook API for not only user sign up, but also friend invitations. If a user sign up through an invitation, her/she could see friend invitations he/she has and act on them. And the inviter is also his/her friend on Trippy.
   Pros: users could easily invite a large number of friends without performing any additional friend adding action.
   Cons: Some Trippy users might not have Facebook accounts.
   We choose Option 2 because the user does not manually add friends and makes our app more usable. We will also implement Option 1 as an additional feature since this allows our app to have a way of adding friend without relying on Facebook.

2. Can the creator of a trip invite user who is not his/her friends to join the trip? Can member of a trip invite their friends?
   Option 1: Only allow the creator of the trip to invite his/her friends into the trip.
   Pros: This keeps the size of trip relatively small and there is a stronger sense of trust among the members of the trip.
   Cons: A friend of the creator of the trip might also want to invite his/her friends to join in the trip. If so, then the creator needs to add his/her friend's friend as friend.
   Option 2: Allow all members of the trip to invite their friends.
   Cons: In real life, there are situations where friends of the creator of the trip want to let their friends who are not friends of trip to join a group.
   Pros: Since the members of the trip might not be direct friends with each other, there is less trust and unity among the members.
   We choose the second option. The first option, while insures that everyone in the trip are friends with each other, restrict the range of people in the trip.

3. Should a trip has an associated state such as active/finalized. And how should a trip becomes finalized.
   Option 1: A trip becomes finalized when its date has been past.
   Pros: Users don't need to take care of closing a trip.
   Cons: The app needs to periodically check if the trip date has been past and this places additional requirement on the app. In addition, malicious users could change the date and therefore close the trip.
   Option 2: The trip creator could close a trip after he sees that the majority of trip member has agreed on relevant trip details. Only the creator can close a trip.
   Pros: This correspond to real life situation.
   Cons: Members other than the creator of the trip might believe that the details of a trip

aren't finalized yet.
We chose the second option because we don't want the system to check it the trip dates have passed automatically. Beside, a trip should be considered closed if all details are settled. And we want the organizer to have more power than the rest of the group members.

Code Challenges:

- Completing Facebook integration with our backend so Facebook friends can be invited to sign up, and the relationship between our backend and Facebook are correctly represented.
    - Option 1: We find the user's friends by getting the user's friend list through Facebook API and return all those who have already registered in the app.
    - Option 2: Use a Friendship model in our backend
        - Cons: the app needs to remember the invitations the user sends to his/her Facebook friends and any Friendship relation in the database.
    - We choose option 2 because user gets all his/her friends through Friendship without relying on Facebook anymore.
- Implementing activity ranking based on number of votes
    - Option 1: Rank activities dynamically in the front end every time a trip page is loaded
        - Pros: May be easier to implement
        - Cons: Additional work has to be done each time a trip is loaded
    - Option 2: Rank activities on change in the backend
        - Pros: A sort is done once per vote in the backend, then displayed; no more work needed to be done for page loads
        - Cons: Because it is a voting system it may be conceivable that the number of votes up or down may outweigh the number of views, leading to more computation
- Implementing the filter of adding friends to a trip based on whether their free ranges contains the trip dates
    - Option 1: First find an union of a friend's free ranges and then find out if the trip range is contained in that union.
        - Pro: It seems easy to take the union of several free ranges.
        - Con: Might be hard to check if a range is contained in a set of free ranges..
    - Option 2: For the trip range, check if each of the trip date in the range is contained in any of the friend's free ranges. If all dates are in some free ranges, then the friend is free during the trip.
        - Con: Seems to have more running time than option 1 as we need to loop through free ranges multiple times.
    - Choose option 1 since it seems implementable and take less time than option 2.

7. Security concerns

7.1 key requirement:

- Only current members and creator of a cabal could view cabal information and vote on activities.
- Only authenticated users could see their private data.

7.2 Threat model

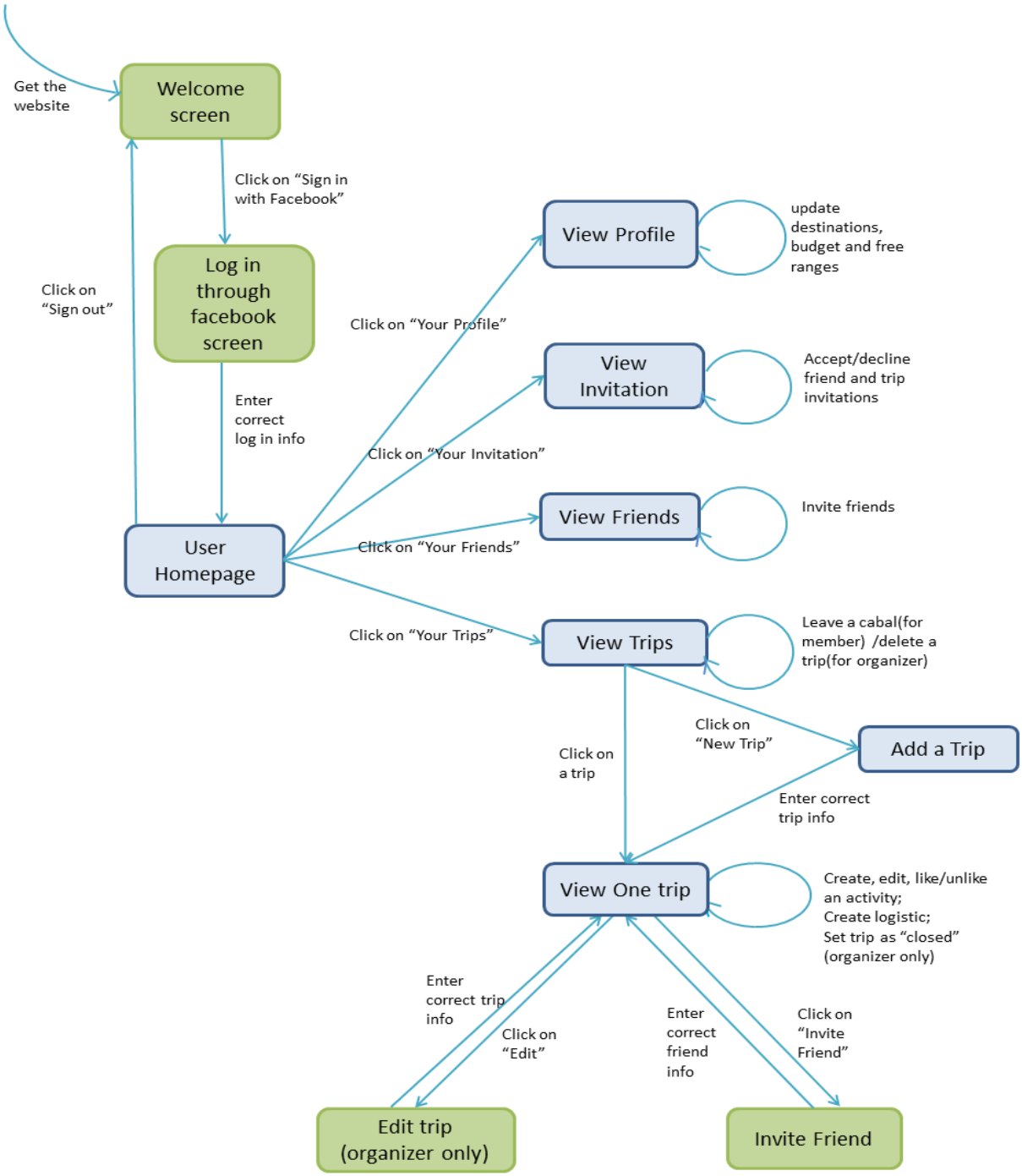
- Unauthenticated users or users who don't belong to a trip may try to view or edit a trip.
- Authenticated users enter information via forms, which can allow for vulnerabilities such as SQL injections.
- Malicious member hacks into another user's account or viewing their private data (their email, invitations and trips) without authorization.
- Malicious users could send spam friend requests.
- Malicious users could also create lots of trip and spamly send trip invitations to their friends.

7.3 Mitigation of Attacks

- **Access control:**
  There is sitewide protection that prevents users from accessing any part of the website without being logged in, except for the login and signup pages.
  In addition, every page checks that the currently logged-in user is allowed to view the current page.
- SQL injection:
  Only Rails ActiveRecord methods are used to access the database. Since ActiveRecord methods sanitize the database queries automatically, there should be no risk of SQL injections.
- XSS:
  Rails escapes HTML strings by default. Trippy should not be vulnerable to basic XSS attacks by not using any raw HTML strings.
- CSRF:
  Since the line "protect_from_forgery with: :exception" is included in the Application Controller, Rails automatically protects the website from CSRF attacks.

8. User interface

8.1 Wire Frame for the app



Note: All blue boxes are reachable from other blue boxes through the navigation bar(the user homepage).

8.2 Wire Frame for major pages

Attached as PDF(TRIP.PY_Wireframes.pdf)