Mmes
Linda Wang, Caitlin Mehl, Aradhana Sinha
Tiffany Wong & Connie Siu
are proud to present
HOGWARTS
The
MARAUDER'S
MIT

Design Document

Overview

In J.K. Rowling's series *Harry Potter*, the Marauder's Map is a magical object that lets characters automatically see the location and movement of other characters on a map. This allows whoever owns the map to both seek out people they want to find, and avoid others they don't want to run into.

Often at MIT, students find themselves in the first scenario where they would like to find a friend for collaboration, event hangouts, or other purposes. We would like to create a "magical" Marauder's Map for MIT, to allow users to find friends nearby by publishing their locations and seeing the locations and statuses of other users.

Problem Statement

Though students would like to know where their friends are, it is often hard for a user to collect this information on their own. Current solutions that exist for meeting friends include the methods in the following table:

Current Solution	Downside
Calling or texting friends	The user has no way of knowing who to contact first or who might be in the area. MIT students are also often reluctant to be the one to reach out first.
Planning ahead	At MIT, people usually do things at the last minute.
Foursquare	Exists mainly to broadcast where you have been, not where you currently are.
Facebook	Not a clear way to see which friends are near you.
Swarm	Does not include a map and is for everyone. Marauder's MIT will be more visually useful and only for MIT students
WeChat	Not enough privacy: allows everyone to see where you are which is creepy

Purpose and Goals

Our app is designed to retrieve users' locations and to broadcast this information to their friends in an easy-to-read format. The key purpose of this app is to let a user know which of their friends are in a specific area at a specific time. The user can then get in contact with these specific friends to meet up with them. This solves the problem of not knowing which friends it would be most worthwhile to contact first and saves the user time and effort.

Essence of the Design

Design Criteria

Safe:

Only available on MIT campus, only available to friends, only available with certificates

Scalable:

As more users join, the app needs to remain usable. To solve issues created by crowding, we can use the following approaches:

1. To make the visuals functional, we can enable a zoom-in feature on the map.
2. To help find a certain user, a search by name function would be useful.

Intuitive:

To encourage people to join and continue using the app, we hope to make the user interface as intuitive as possible by using pre-existing tropes. For example, users will be able to zoom in on the map with the customary finger-spreading motion. Green will encode the available status, and red will encode busy.

Key Concepts

The main concepts we plan on implementing for our application include the following:

Apparition

When a user checks in, he or she is known to have "apparated". This concept allows users broadcast to their friends where they currently are, or where they plan on being in the future. The main purpose of the Marauder's MIT application is to broadcast where a user currently is, and the concept of Apparition allows us to do this.

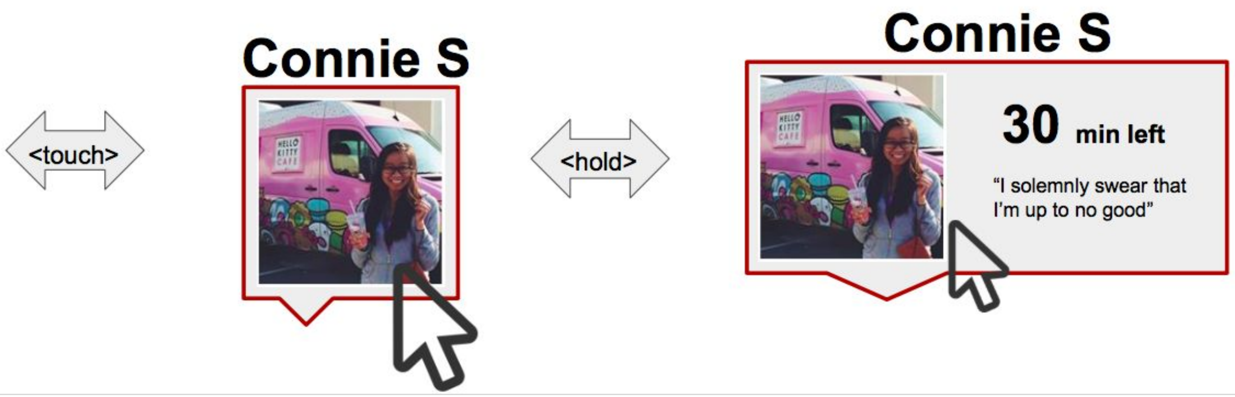
In order to apparate, a user needs to allow the Geolocation API to access his or her location. If this location is within the borders of MIT campus, the Marauder's MIT map will display the user's profile picture at that place and will allow the user to check-in with a given duration, availability and a text status explaining why the user is there. Once a user checks-in, his/her friends will be able to view the location, availability and status on their own maps.

After implementing our MVP, we will consider adding a "time-travel" feature which allows a user to manually check in to a location they intend to be in the future. The map would then have the option to show where users intend to be at a certain time in the future and a user can use this information to make plans.

Friends

Due to the fact that people may not want everyone on MIT campus to know where they are, we are introducing the concept of "Friends". Users can send a friend request, accept or ignore a friend request, and delete other users. Exchange of location information is essentially limited to between friends. This means that when a user apparates to a location, this is shared only with other users on their friends list, and when a user checks their Marauder's MIT map, they can only see the profiles of where their friends are located. The motivation behind the friends concept is to allow users to know and choose exactly with whom they are sharing their location information.

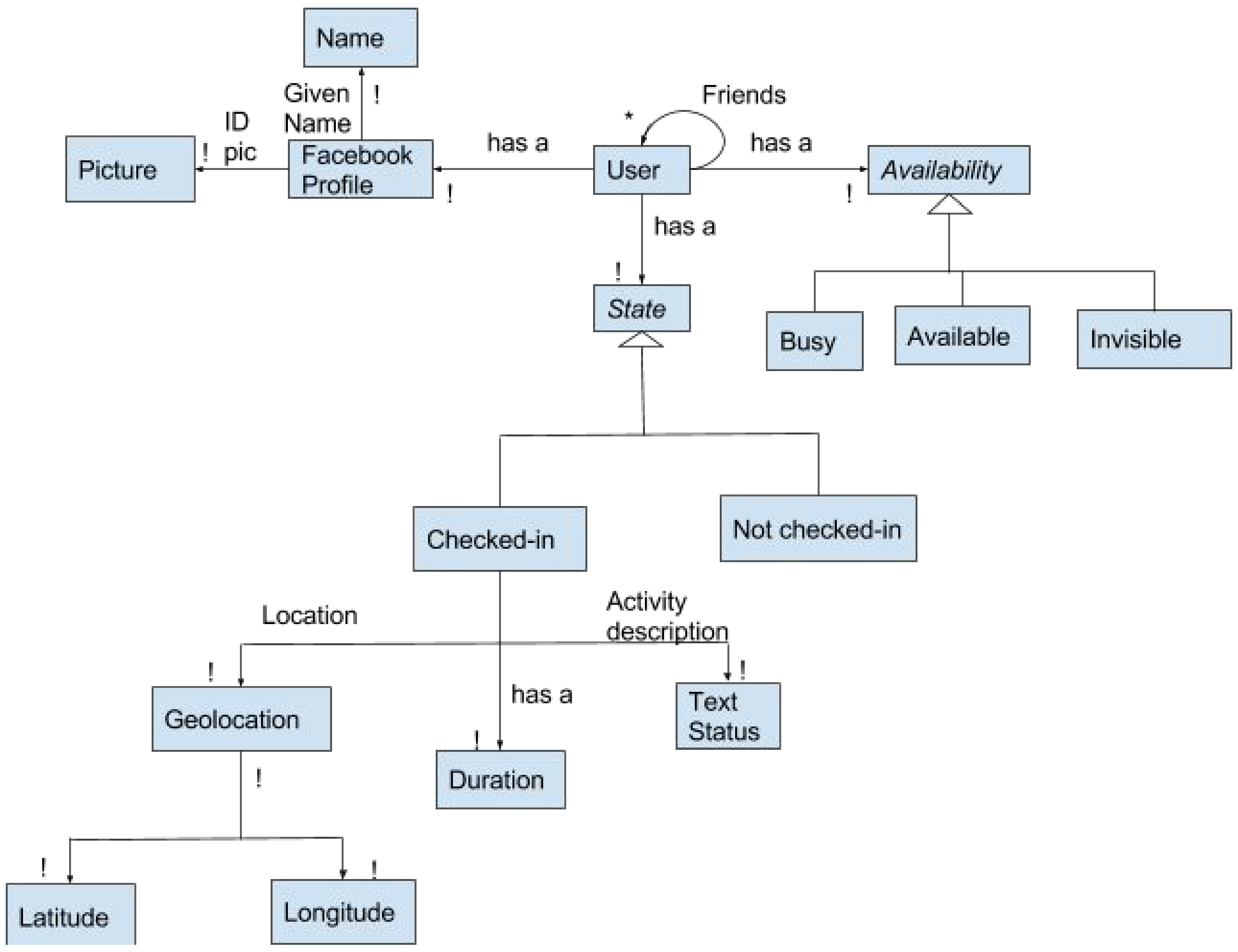
Status



MIT students are known for their collaborative spirit, and students often love working together on psets, even if it's not necessarily the same pset. Students may want to distinguish between broadcasting their location in order to hang out, and letting people know where they are if they want to work together. If a user doesn't want to hang out or work with anybody, they can choose not to check-in to the app at all and their location will not be broadcast to their friends. Thus, a user will always be in one of three states - available, busy, and invisible. In addition to a state, a user's status will include a text, which is defaulted to "I solemnly swear that I'm up to no good". A user has the option to edit this text to show a bit of personality or elaborate on their state: e.g. "working on 170 pset, anyone wanna join? :)"

When user touches an image, the image enlarges and the friend's name appears. When the user continues to press and hold the image, the status and the time left that the friend will be at that location for will also show up.

Data Model



Textual Constraints:
All users must have an availability (available, busy, invisible) but a user can only have an availability of “available” or “busy” if they are checked-in and if a user is not checked in they must have an availability of “invisible.”

Pithy Insights:
A user can have the status of being “invisible” but they don’t have the option of explicitly choosing to be. “Invisible” automatically happens if a user is not checked-in.

Security Concerns

Since our app involves storing and broadcasting the locations, we need to make sure this information is not leaked or abused. To mitigate this, the first security technique we need to implement is storing a user’s information in a protected account. Logins will be done with MIT email and Facebook to ensure that this app is only available to members of the MIT community. The concept of friends is also designed to protect information. A user’s location will only be broadcast to their friends, and becoming friends with someone is a mutual agreement. This should prevent a user’s information from getting leaked to potentially malicious people. Users can also remove friends that they no longer want to broadcast their location to. If users are constantly being harassed/friended by the same people, they can report or have them blocked from the site.

Standard Web Attacks: The majority of the functionality of our site does not require the user to input text queries, so many of the text based standard attacks are mitigated. The only time text input is available is when users input text based statuses, and during login. During these times we will make sure we to sanitize all text inputs and check for buffer overflow issues. For all other requests that happen on the app we will use CSRF tokens to ensure that all requests going through to the server actually came from the client.

Threat model:
A user that is not logged in cannot view anything on the site except for the log in page.

A user that is logged in can view the location of friends and can check-in to a location based on their Geolocation. Users can only view the location of friends that have mutually accepted the friend agreement. Users can set fake locations during check-in by spoofing their geolocation, but if they do this repeatedly other users can report the user for bad behavior.

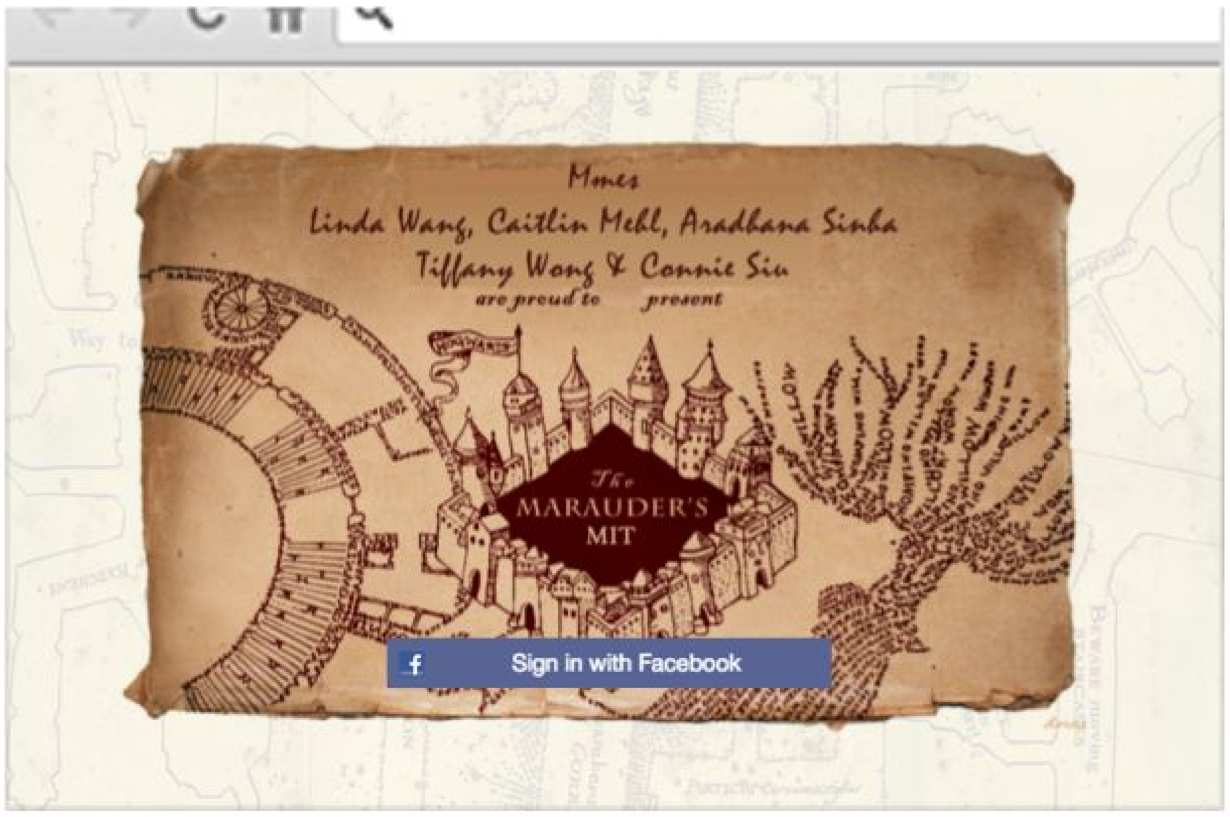
A user that manages to log in to another user’s account will have access to all of his/her friends’ data. Other users can still report the user if the find out this is happening. Users can also stop checking-in to locations if they suspect that they are being followed.

User Interface

We are aware of the fundamental difference of usage styles of mobile sites and desktop browser sites. We would like to model our app after the Google Maps app. The idea is that since a desktop browser has more space, we would like to provide a sidebar with constant information such as a list of suggested friends, or a list of pending friend requests. A mobile browser inherently has less space so we would like that space to mostly be taken up by the map.

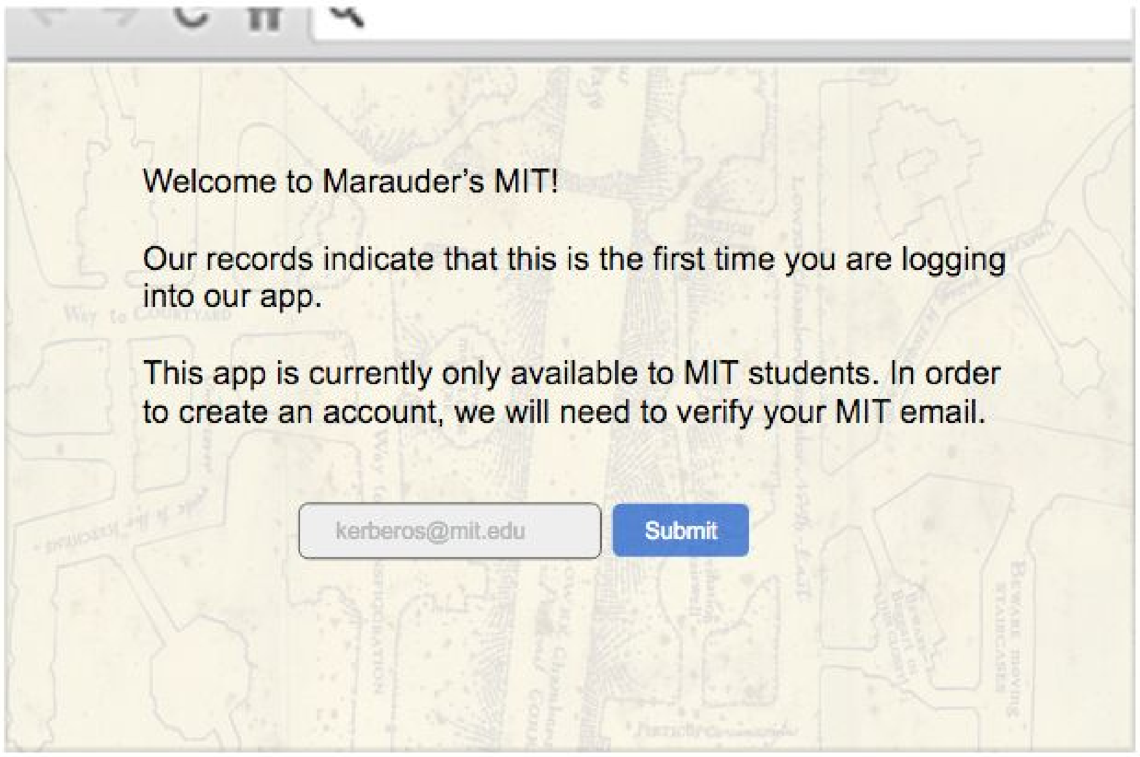
Note: We've specifically made the decision to not have a user profile page, since we don't think it's necessary. All of our information comes from facebook, where users are already friends and can already see each other's profiles.

Login /Register:


Mmes
Linda Wang, Caitlin Mehl, Aradhana Sinha
Tiffany Wong & Connie Siu
are proud to present
The MARAUDER'S MIT
Sign in with Facebook

Using Facebook for authentication gives us the advantage of not having to ask users to register. Regardless of whether they have logged in for the first time or not, users will simply "Sign in with Facebook".

Verify Email:

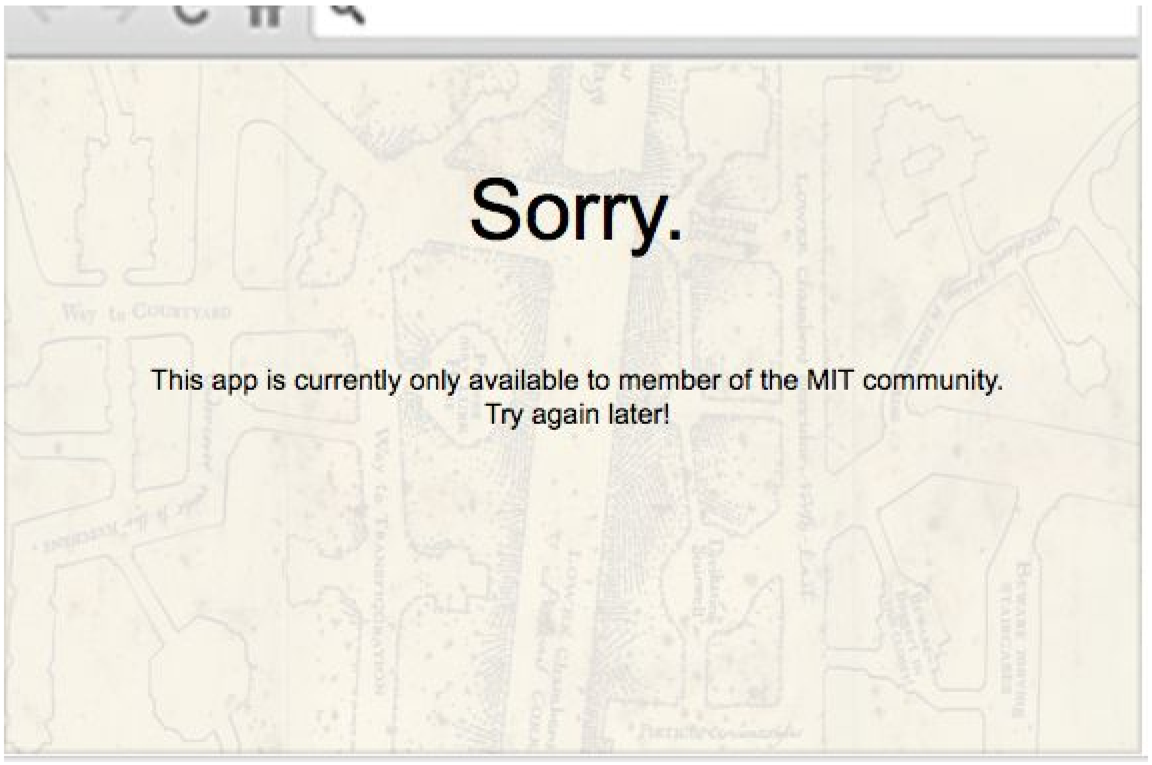

Welcome to Marauder's MIT!

Our records indicate that this is the first time you are logging into our app.

This app is currently only available to MIT students. In order to create an account, we will need to verify your MIT email.

However, we also want to limit this app to only registered MIT students. When a user signs in for the first time (if we do not have a record of this user in our database) we will ask the user to verify his/her MIT email. When the email is verified, we will create a record for this person in our database.

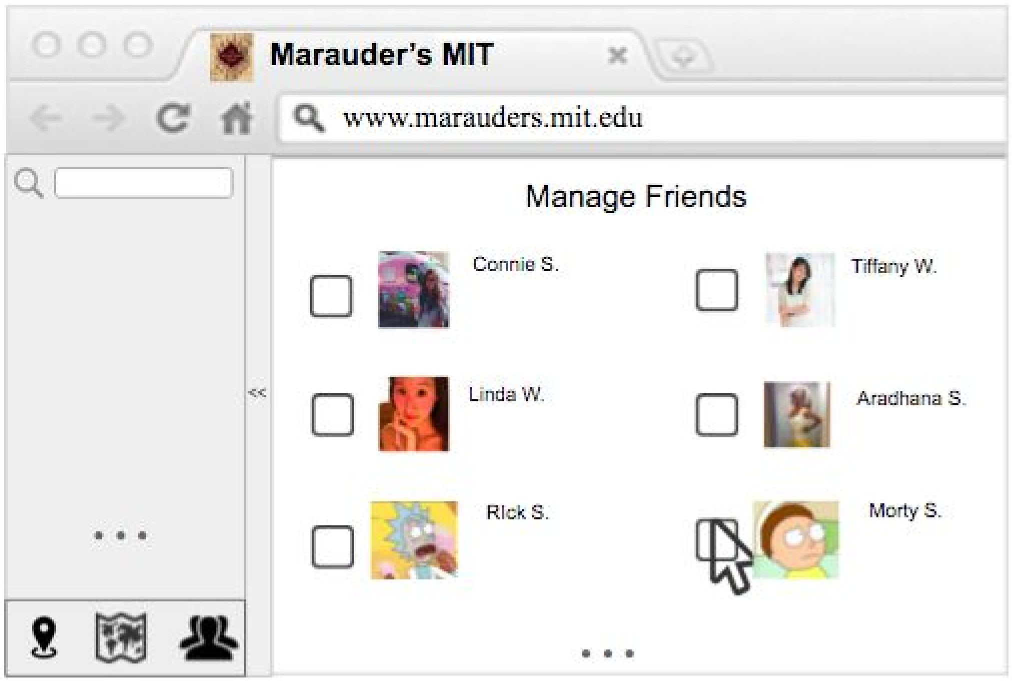
Verify Email Error:


Sorry.

This app is currently only available to member of the MIT community.
Try again later!

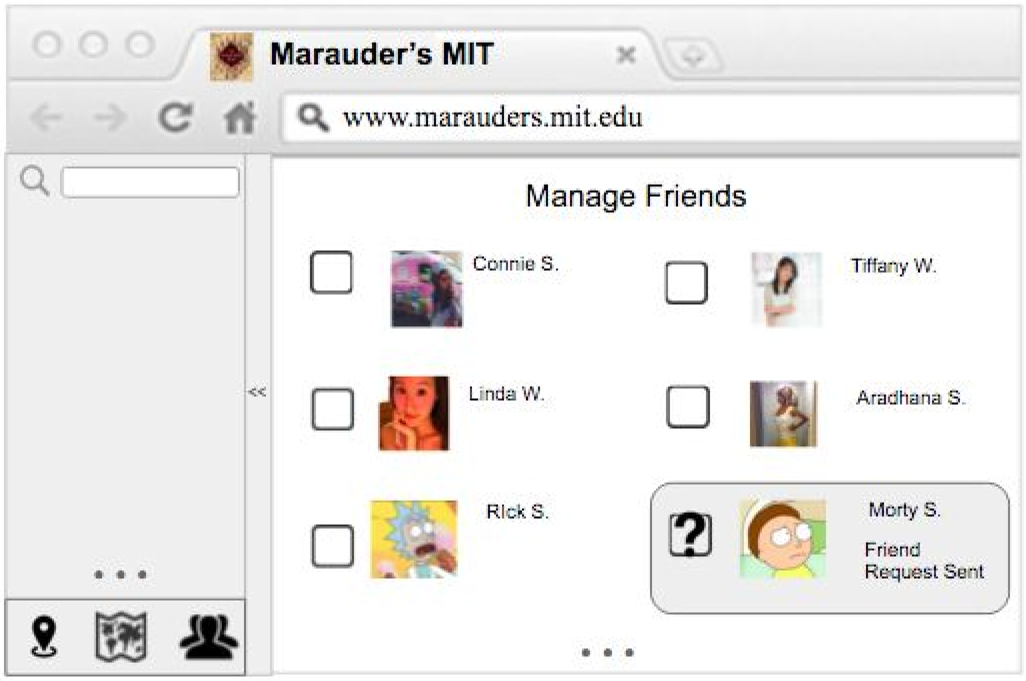
We will only make the app available to members of the MIT community with a @mit.edu email address.

Manage Friends:



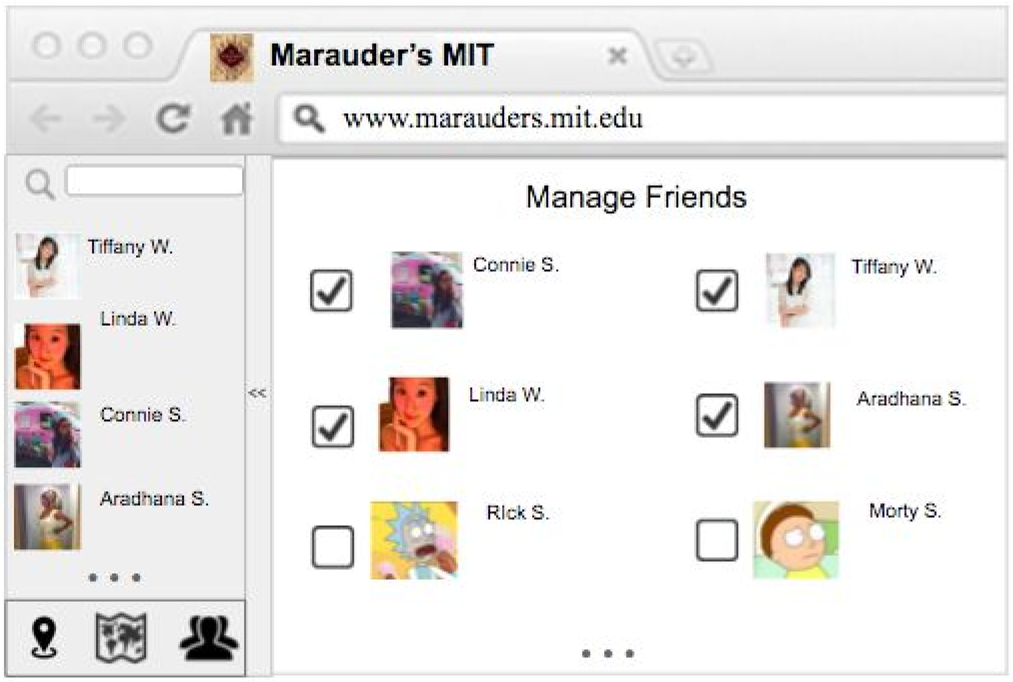
For users who have never signed in before, it's important that they first manage their friends. The first screen they see will be the "manage friends" page, accessible at any time by the people icon on the left menu. The "manage friends" page shows the user his facebook friends.

Send Friend Request:



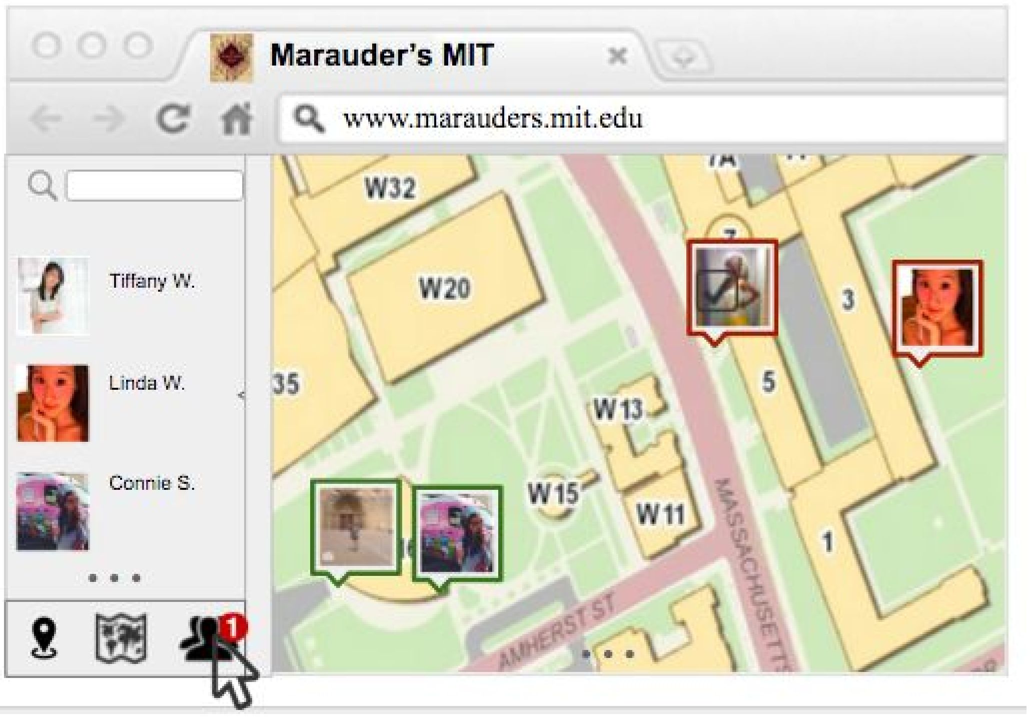
By clicking on a friend, a request to connect to the user on this app is sent. To undo the request, the user can simply click once again.

View approved friends / Remove friends:



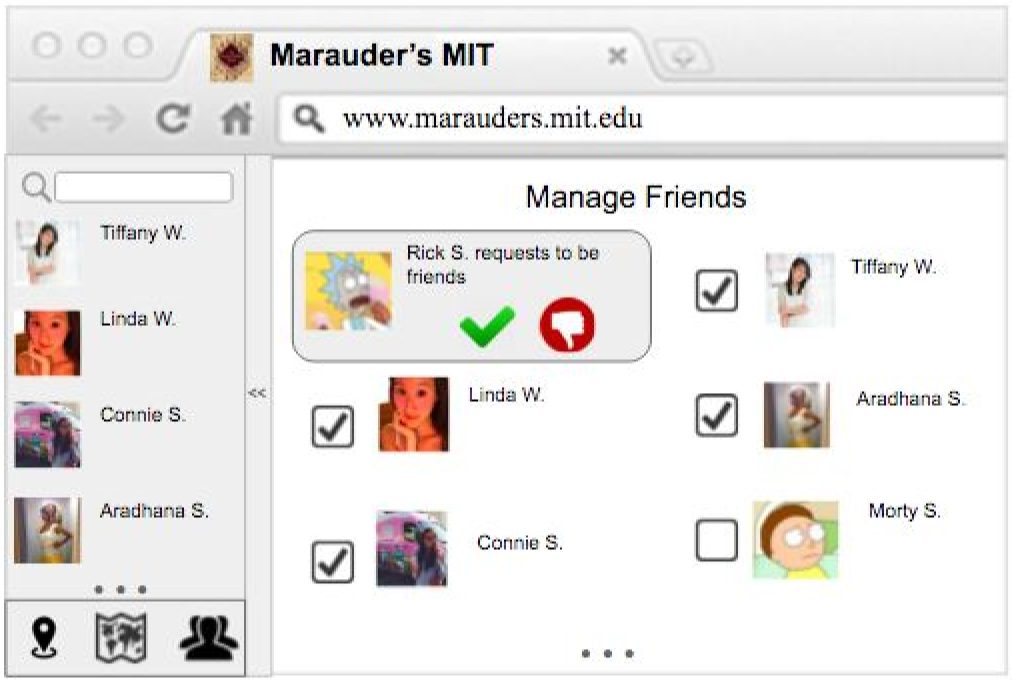
The user may see what friends have responded to her friend requests. To remove friends, the user simply needs to uncheck the check box.

Friend Request Notifications:

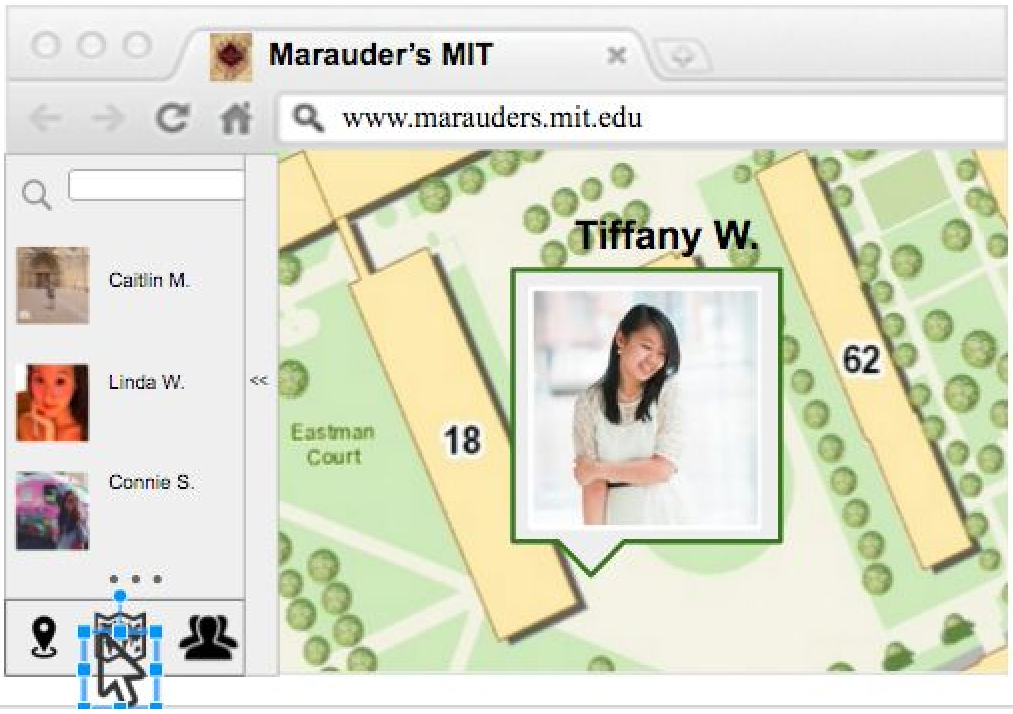


When the user receives a friend request, she is notified by a red icon on the screen.

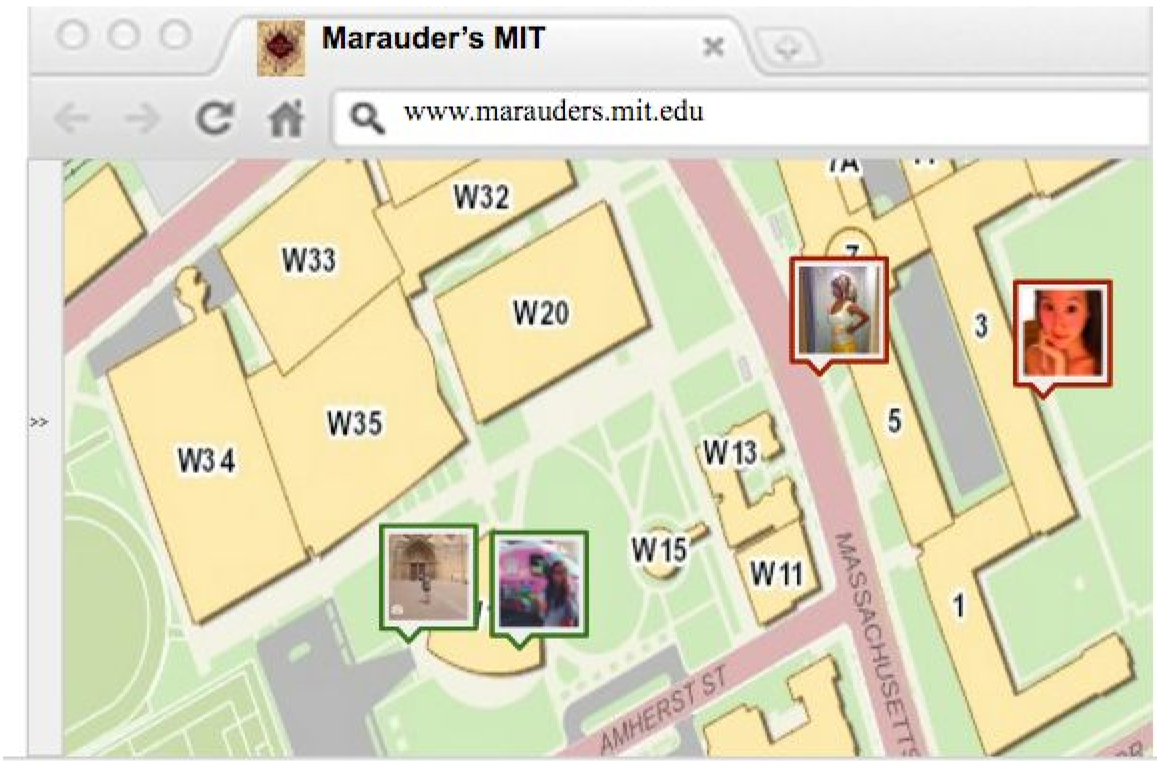
Accept / Reject Friend Request:



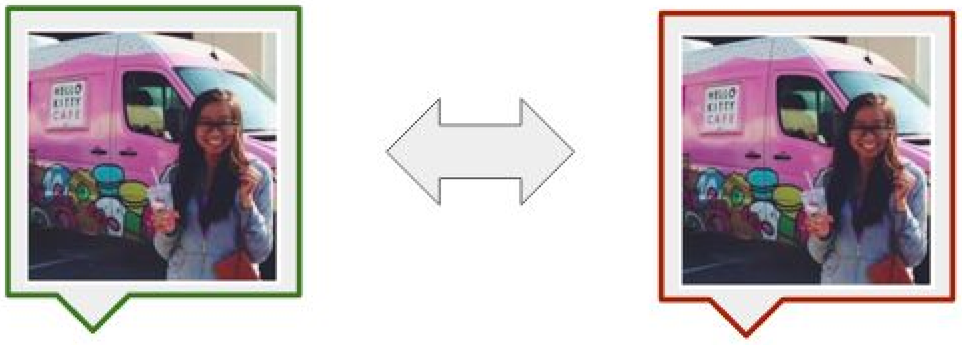
Full Map Page:



By clicking on the map icon, the user switches to the map page. The Map icon shows the user where she is.

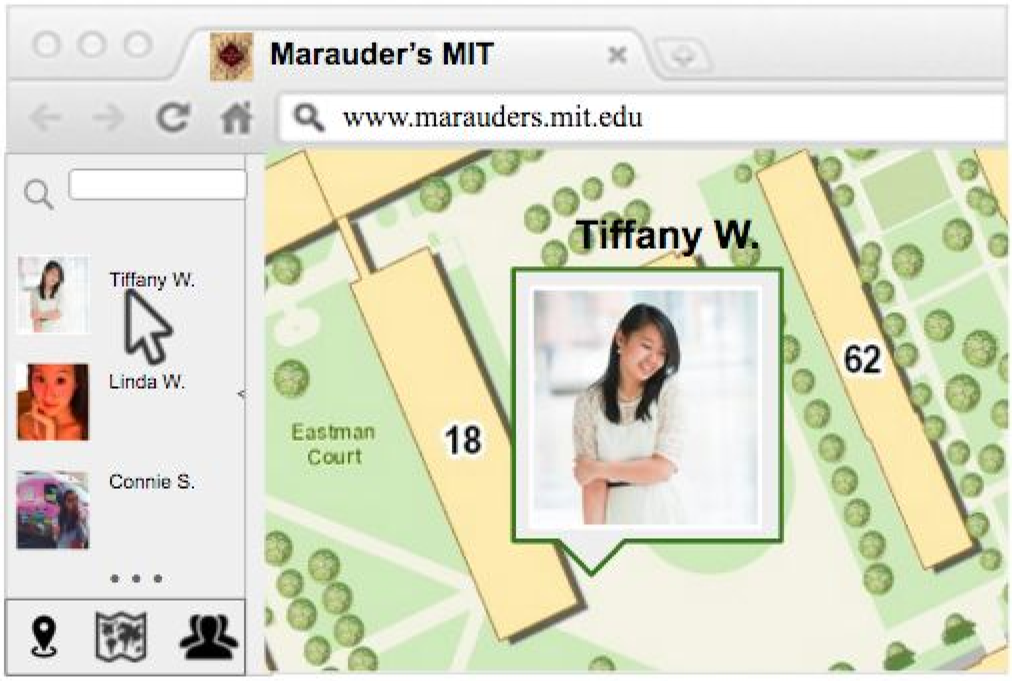


Users are visible from a bird's eye view on the map.

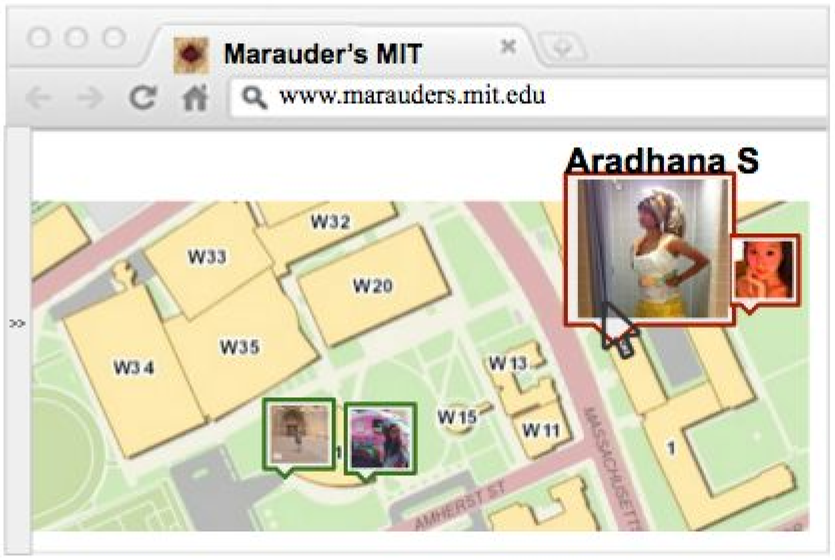


The user updates their status from "available" to "busy" when they apparate, which is displayed with either green or red in the birds-eye view.

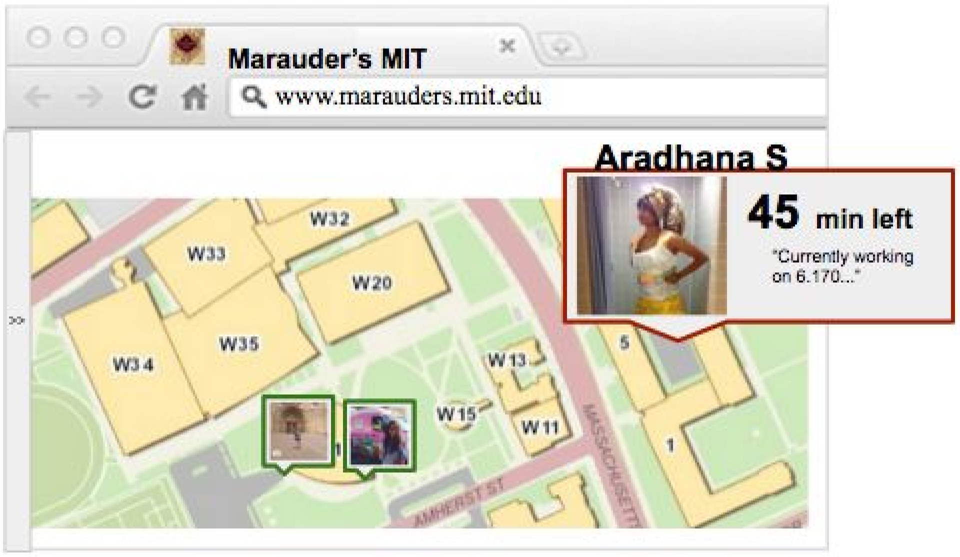
Finding Friends:



To find a specific friend on the map, the user may navigate to her friends list and click on a friend. The friend will immediately come into focus on the main map.

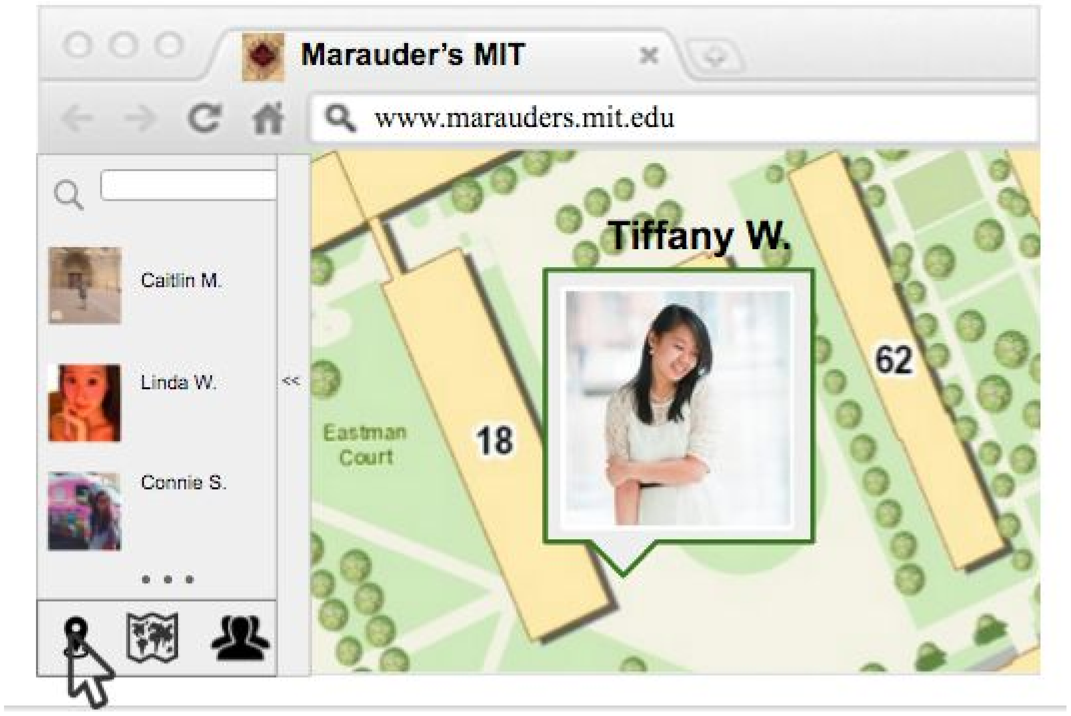


By clicking or pressing on the user, the picture enlarges and the user's name appears

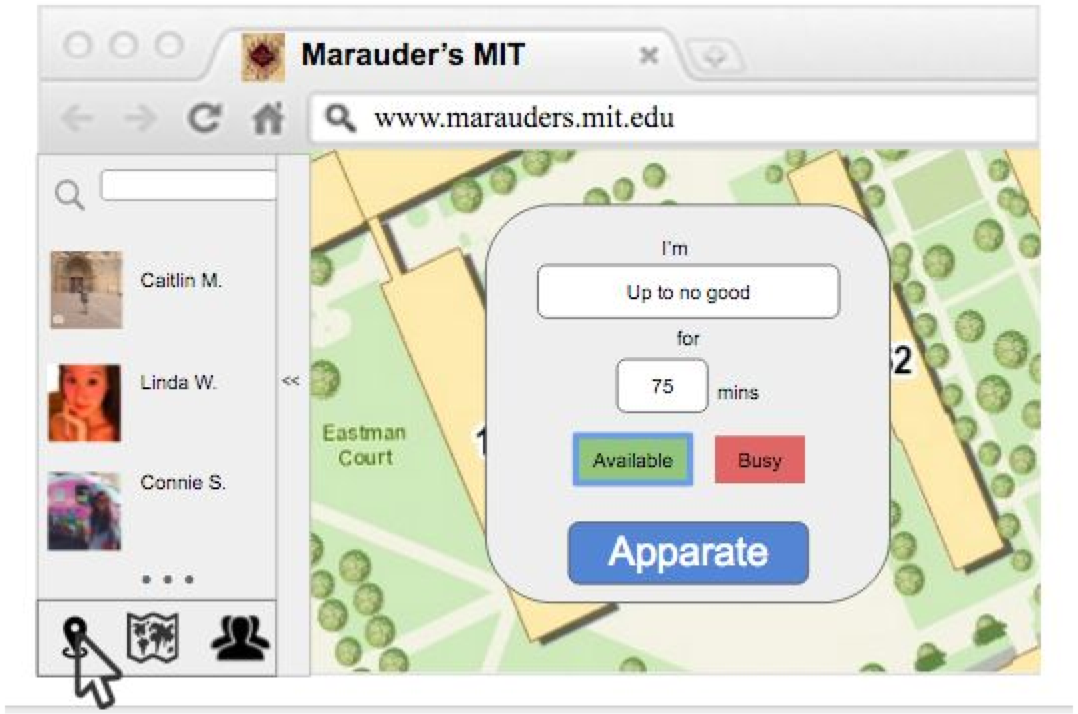


By holding down on the user's image, more information about the user's status and how long they have left at their location is displayed.

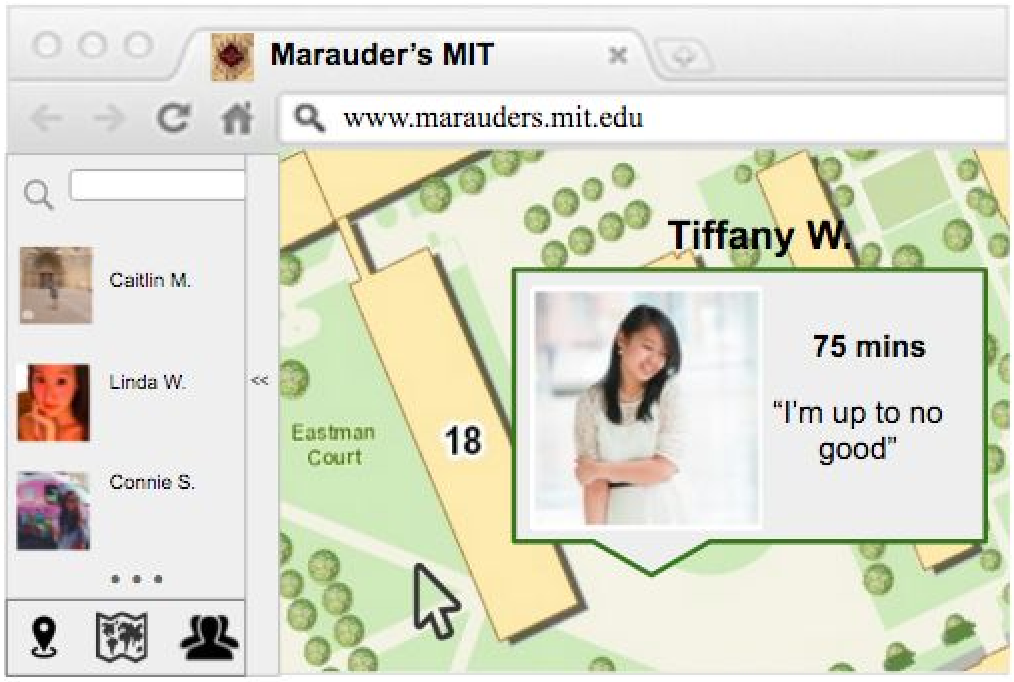
Apparate / Set Status:



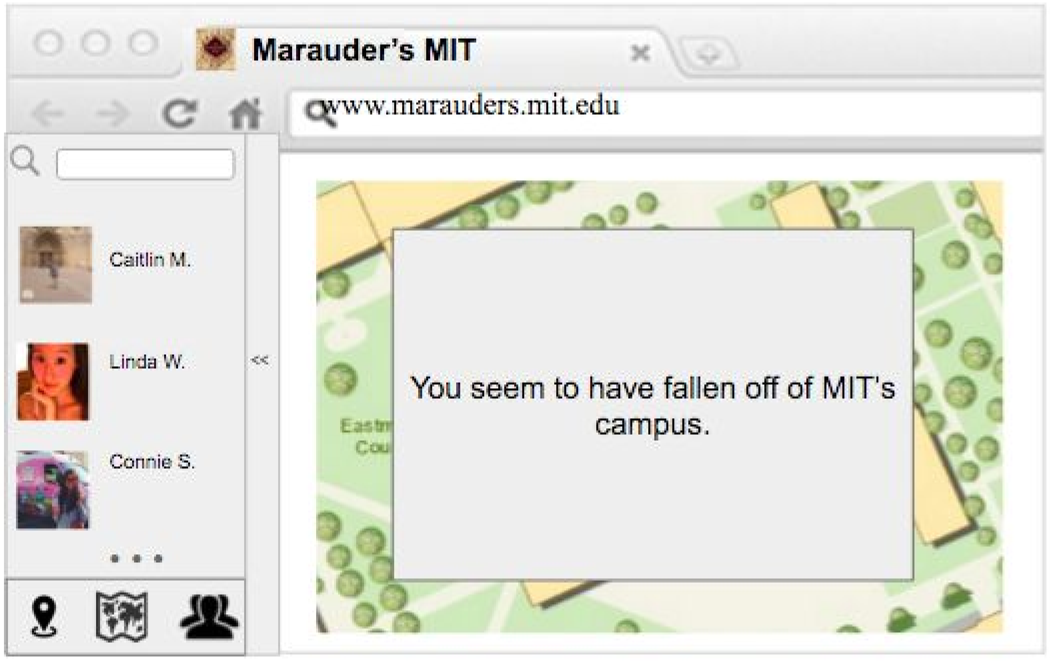
The user can elect to apparate by clicking the marker icon on the left menu. The icon at first refocuses the user to their current location.



After refocusing, the app gives the user the option to set a text, duration, and status of available or busy. Then he or she may apparate.



After an apparate, the user's details are updated.



You seem to have fallen off of MIT's campus.

If the user is not on campus, the user can't see any of her friend's locations.

Design Challenges and Mitigations

User and Concept Issues:

Login and Registration

Facebook currently has a large and loyal user base, and we had a lot of discussions about whether to use Facebook to login to the Marauder's MIT.

- If we use Facebook, we bypass the registration process of having a user input their personal information. We would also be able to retrieve a lot of un-inputtable information such as a user's list of friends. However, this would limit the user-base to only people who have Facebook profiles. We would also have to have an extra step to make sure the user is an MIT student.
    - For this extra step, we considered checking if the user's Facebook profile has an "@mit.edu" email address connected to it. This is a problem because a lot of people don't have their MIT email associated with Facebook.
    - We also considered requiring a user to input their "@mit.edu" email when they register with facebook.
- If we don't use Facebook, we would have to have the user go through the entire registration process of adding their name, choosing a password, etc. We would also not have very much information about them, such as who they are already friends with.

Because we decided to use Facebook to help us with suggesting friends, we decided to require a user to have a Facebook account to register and have them use it to log in. We decided since a lot of people don't have their MIT address associated with their Facebook account, we will have them input their "@mit.edu" email the first time they register for the app.

Adding Friends

The Marauder's Map would be useless if a user has no friends, and one of our biggest design challenges was deciding how to suggest friends to the user. We came up with a few ideas:

1. Because we are requiring Facebook login, we can use a user's Facebook friends as a starting set of possible friends.
    a. We can allow a user to send friend requests to any of their Facebook friends, even if they are not using Marauder's MIT. This means we will not have to do any filtering on the user's list of friends and in addition, this will help us advertise to people who are not yet using the app. However, we won't be able to tell which of the friends are MIT affiliated, meaning the user may be sending friend requests to people who are not allowed to use the application.

    b. We can filter the Facebook friends to those who are already using the application. This will remove our advertising to potential users, but will not annoy people who can't use the app. This is pretty computationally expensive. We would either:
        i. Calculate this every time a user goes to the "add friends" page.
        ii. Calculate this once and update whenever a user's friend joins or deletes Marauder's Map.
        iii. Calculate this once and update every couple of weeks or so.
    c. We can filter the Facebook based on people that have MIT listed in their Education or in their Work fields. The downside to this option is that it might not be exactly accurate because some people may not display that information on their Facebook, or we might accidentally include alumni who are no longer in the area.
2. We could also suggest friends based on who is currently nearby the user. This will help people get to know new people, but it might also be kind of creepy.

We decided to go with the Facebook friends because it would be less creepy to be suggested to be friends with people you already know. We also decided to filter based on MIT community status because even though it might not be exactly accurate, it's the best way to not clutter a user's manage friends page. If users are sent a friend request but are not on the app yet, we can send them a notification on Facebook. This is also a good way to help expand our user base.

Checking In

Having users and friends is important, but the map would also be useless if nobody shows up on the map. We also had a lot of discussions about how to get a user's location information, and when to show this:

- We can get a user's location using the HTML5 Geolocation API, which is incredibly accurate. However, this is only available if a user has the browser open to Marauder's MIT and will be removed as soon as the user closes the browser.
- We can get a user's location by having them input it themselves. This is would require more effort on the user's part, and could cause trouble if this conflicts with the Geolocation information. This could also be bad if a user tries to be dishonest and say they are somewhere they are not.

We decided to use a combination of both. Using the Geolocation API, we can get a user's current location. Then if the user chooses to, they can manually check-in to that specific location. This way we still have the accuracy of the Geolocation API but still can retain the user's location if the browser is not open. This will also let the user be more careful about the information s/he is broadcasting. The user's location information will not be shown unless they choose to check-in to the app.

If a user is not checked in, can they see their friends? We considered this question and thought:

- If yes, it'd be a little creepy because they would be receiving more information than they are providing
- If no, this would provide incentive for them to check in.

We considered the case where a user doesn't know where their friends are, and wants to use Marauder's MIT to find out. In this case, it would not make sense for them to have to check in where they currently are to find out where they want to be, so we are allowing a user who is not checked in to view the map.

Finally, once a user is checked in, when do they check out? We had a few design decisions to choose from here as well:

- Automatic checking out a user, either when:
    - They close the browser. This could be a problem if the user doesn't stay on the Marauder's Map page the entire time they are at a location. It is pretty likely for a user to open the tab, scope out their friends, and close it.
    - A certain amount of time has passed, either inputted by the user or a default of 4 hours or so.
- Having a check out button. A user could change their mind about how long they stay at a location, but they would have to remember to check out every time they leave.

We decided it would be better to say that a user is not somewhere they actually are, than to say that they are somewhere they are not. Because users a probably lazy and forgetful, we decided to require a user to input a time when they check in. We may decide to include a check out button as an additional option beyond our MVP.

UI Issues:

Clown Car Problem

To solve issues created by crowding, we will use the following approaches:

1. To make the visuals functional, we can enable a zoom-in feature on the map. In a fully zoomed out map, the map will display a certain user and a number indicating how many other users are also there. For example, "Connie Siu + 10 others".
2. To help find a certain user's location on the map, a search by name function will be created.

Elevation Issues

(Feature beyond MVP) To help discern which floor a user is located on, we can create different levels for the map for each floor. Broadcasting location by floor could be achieved by manual check-ins or by using the phone to determine which MIT wifi router it is currently closest to.

Other Issues:

Achieving Critical Mass of Users

The success of this app hinges on there being enough users to make joining the app worthwhile. This means that joining the app should be simple and hassle-free. Additionally, as an app designed to help people connect with each other, it would be useful to let people invite their friends to join.

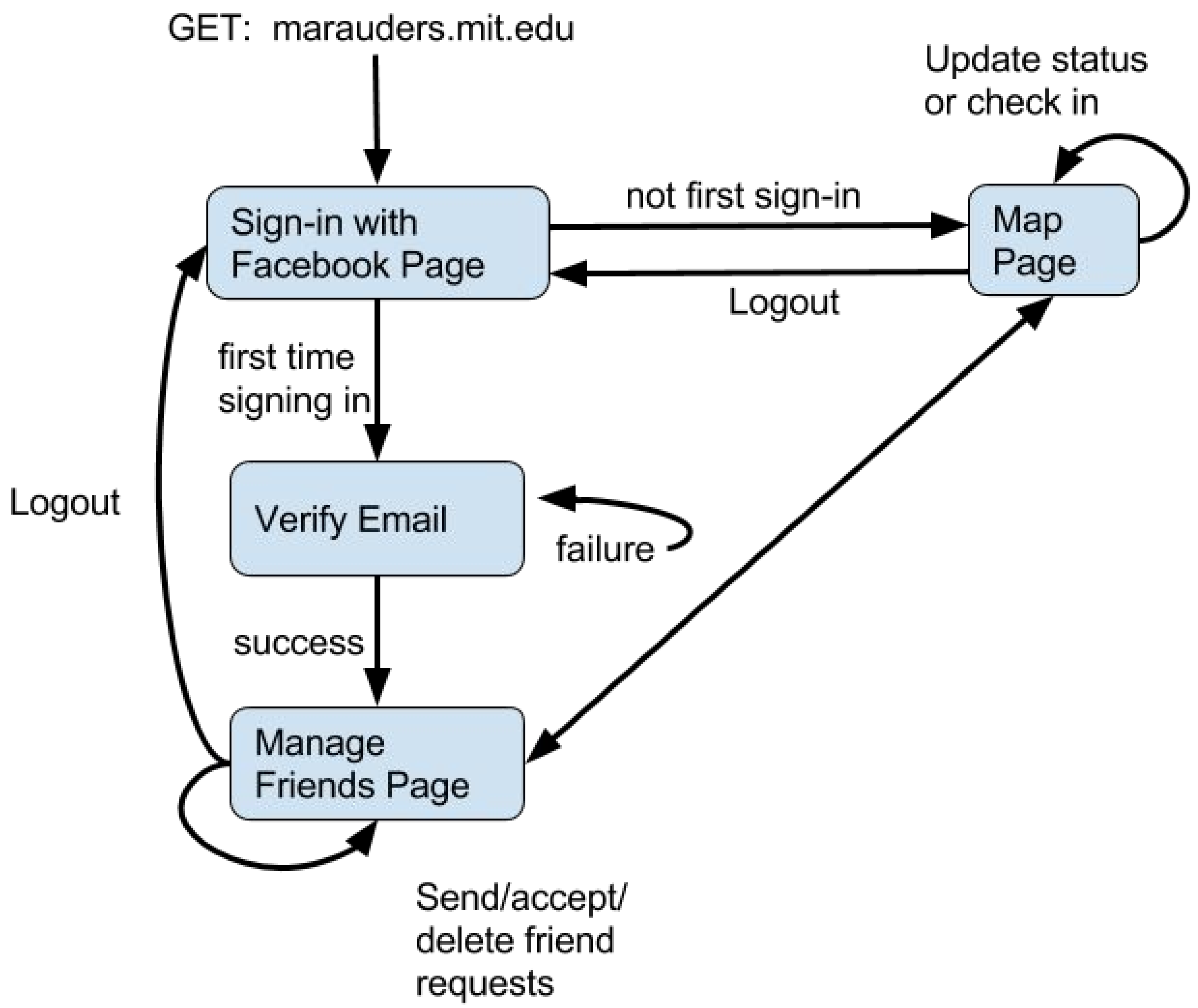
One major challenge will be getting enough users to join our app. We could accomplish this in a number of ways: with marketing, with incentives, or by integrating the app with Facebook. We've decided to integrate the app with Facebook because we believe it will be the best way for the application to go viral and requires little effort on our part. We hope that as Facebook users see their friends using the application, they will see the benefits of joining and will sign up for the application as well.

Stalking or Harassment

To ensure the safety and comfort of users, we will enable users to:

1. Share location with only their friends. This allows a user to know and edit exactly who they are sharing their information with.
2. Report harassment from other users. If a user is reported for harassing behavior, site managers will contact the user reported. Repeated offenders will be blocked from the app.

State Diagrams


GET: marauders.mit.edu
Update status or check in
Sign-in with Facebook Page
not first sign-in
Map Page
Logout
first time signing in
Logout
Verify Email
failure
success
Manage Friends Page
Send/accept/ delete friend requests