Team 17
Ariya Lau
Aaron Lynn
Daniel Sanchez
E.J. Wennerberg
Michael Zhang

Purpose

GPS navigation has become a staple in modern life, allowing people to get to where they need to go in a timely manner and in a much simpler way. However, not many applications take advantage of an interpersonal approach to support group meetings.

Google Maps offers a “share location” feature which allows users to see the location of the people they are meeting with. However, users are unable to schedule a meeting with others directly through the app. Apple’s Find My Friends also allows users to see the location of friends and family, but also lack an events feature as well as support for Android phones.

The purpose of our application is to allow users to coordinate meetups with friends and colleagues and ensure that everyone travels safely to their respective destinations. Users can create events, send meeting requests, and get event notifications to make coordinating meetings easier.

Functional Requirements

1. User Account
   <u>As a user:</u>
   - I would like to be able to register for an account.
   - I would like to be able to login and manage my account.
   - I would like my password to be reset if I forget it.
   - I would like to be able to edit my profile.
   - I would like to store a picture on my profile.
   - I would like to be able to search for other users using their phone number and email.

2. Viewing Events
   <u>As a user:</u>
   - I would like to be able to view a list of all the events I need to attend today, this week, and this month.
   - I would like to view the list of the attendees and be able to view their profiles.
   - I would like to accept or decline invitations to events.
   - I would like to see a list of events that I have declined.
   - I would like to rejoin an event even if I originally declined.

3. Managing Events

   As an attendee:

   - I would like to suggest a meeting place to the event admin.
   - I would like to exit from an event at any given point in time, including on the way to the event.

   As an event admin:

   - I would like to create an event.
   - I would like to add other users to an event.
   - I would like to optionally specify additional details regarding location, such as room number.
   - I would like to set permissions on which attendees are allowed to modify the event and the attendees list.
   - I would like to change details of an event.
   - I would like to see recommendations for common nearby meeting places.
   - I would like to choose a recommendation as the meeting place.
   - I would like to cancel a meeting at any given point in time, including on the way to the event.
   - I would like to add a plain-text note to an event.
   - I would like to attach arbitrary documents to an event.

4. Notifications

   As a user:

   - I would like to receive notifications about events that I have been invited to.
   - I would like to optionally receive notifications/reminders about changes to an event.
   - I would like to be able to add individual events to my Google calendar.
   - I would like to sync all events to my Google calendar.
   - I would like to notify other attendees when I arrive home safely.
   - I would like to be notified when other attendees arrive home safely.

5. Mapping system

   As a user:

   - I would like to send my location to other users without being attached to any events.

   As an attendee:

- I would like to see the location of the attendees as the event is about to start.
- I would like to see my estimated time of arrival to the event location.
- I would like to know when I have to leave to arrive on time.
- I would like to see everyone else's estimated time of arrival to the event location.
- I would like to see my recommended route to the event.
- I would like to see everyone else's personal recommended route to the event.
- I would like to display my location to others even if I have left an event earlier than everyone else.
- (if time allows) I would like to be recommended nearby parking locations.

Non Functional Requirements

1. Client Requirements
   As a developer:
   - I would like the application to be run on Android phones running Android 6.0 (Marshmallow) and above.

2. Server Requirements
   As a developer:
   - I would like the server to save user data to a relational database.
   - I would like the server to save event and invitation data to a relational database.
   - I would like the server to support real-time communication of location data between server and client(s).

3. Design Requirements
   As a developer:
   - I would like the APIs to be able to be expanded upon or reduced very easily.
   - I would like our codebase to be flexible so that we can add new functionalities without having to redesign prior code.

4. Performance Requirements
   As a developer:
   - I would like the server to be able to handle 1000 users.
   - I would like the application to launch quickly and smoothly.

- I would like the map function to support real-time location updates with less than 30 seconds intervals between screen updates.

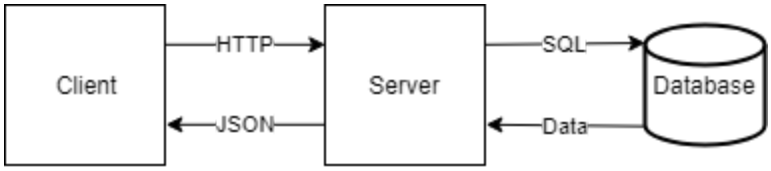
5. Appearance Requirements
   <u>As a developer:</u>
   - I would like the application to be organized in a logical way so that users can easily navigate the app.
   - I would like navigation between pages to run smoothly and contain no lag.

6. Security Requirements
   <u>As a developer:</u>
   - I would like my personal information such as email, password, phone number, current location, and profile picture to be securely and responsibly handled.
   - I would like to give users the option to opt-in to enabling location services for our device.
   - I would like to allow users to disable sharing their location when they opt to leave an event early.

Design Outline

This project will be an Android application that utilizes GPS navigation to allow users to coordinate meetups and ensure other users arrive safely to their respective destinations. This application will use the client-server model, where one server manages the exchange of information between multiple clients. The server will store and access user and event information in a database and redirect geographic information between users.

High Level Overview



Android Client

- The client will be how the user interfaces with our system.
- The client sends authenticated HTTP requests to the server.
- The client receives and parses JSON objects in HTTP responses from our server and formats and displays this data to the user.

Server

- The server handles HTTP requests from the client.
- The server checks that the user is authorized to access the requested data.
- The server makes SQL queries to the database if necessary.
- The server responds with the requested information in JSON.

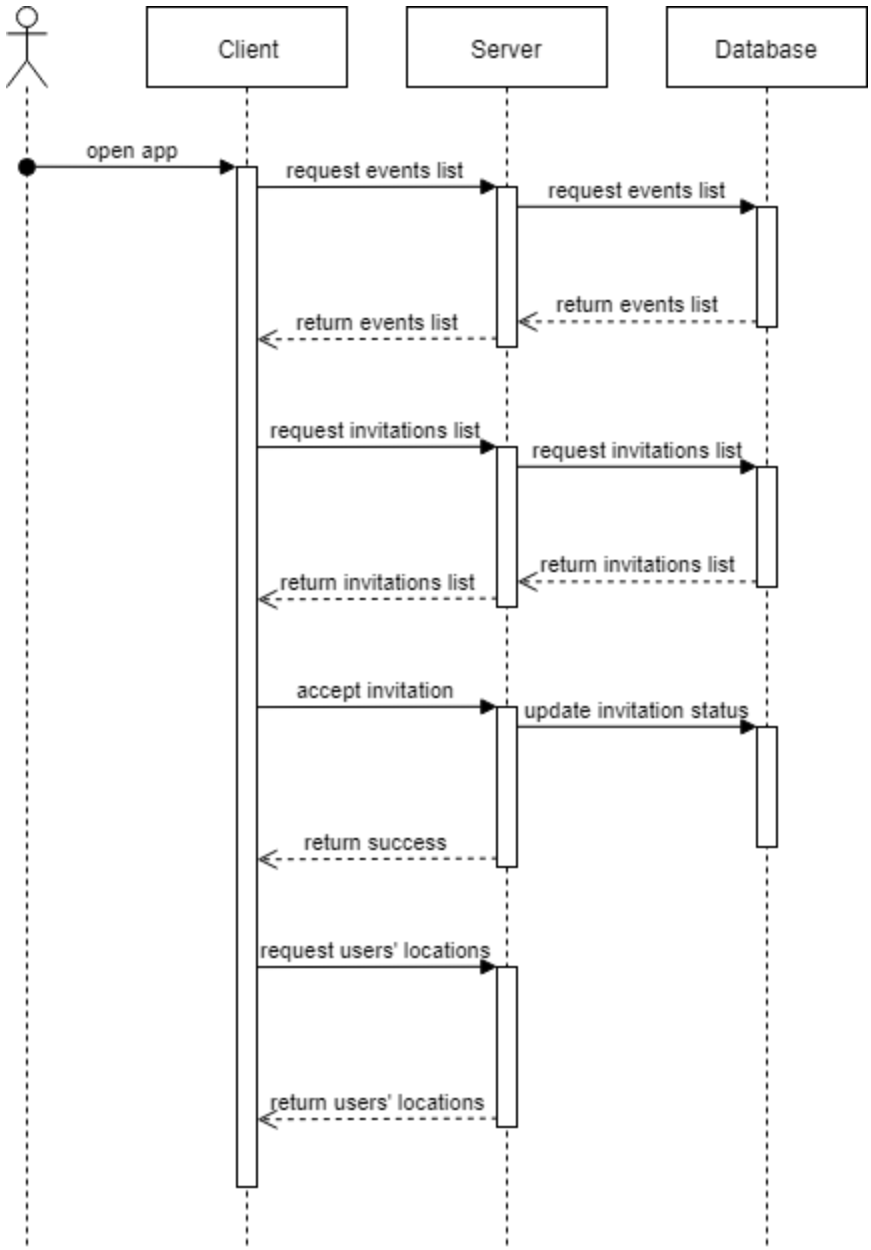
Database

- The database stores all user account and event data.
- The database stores relationships between users and events.

Detailed Overview

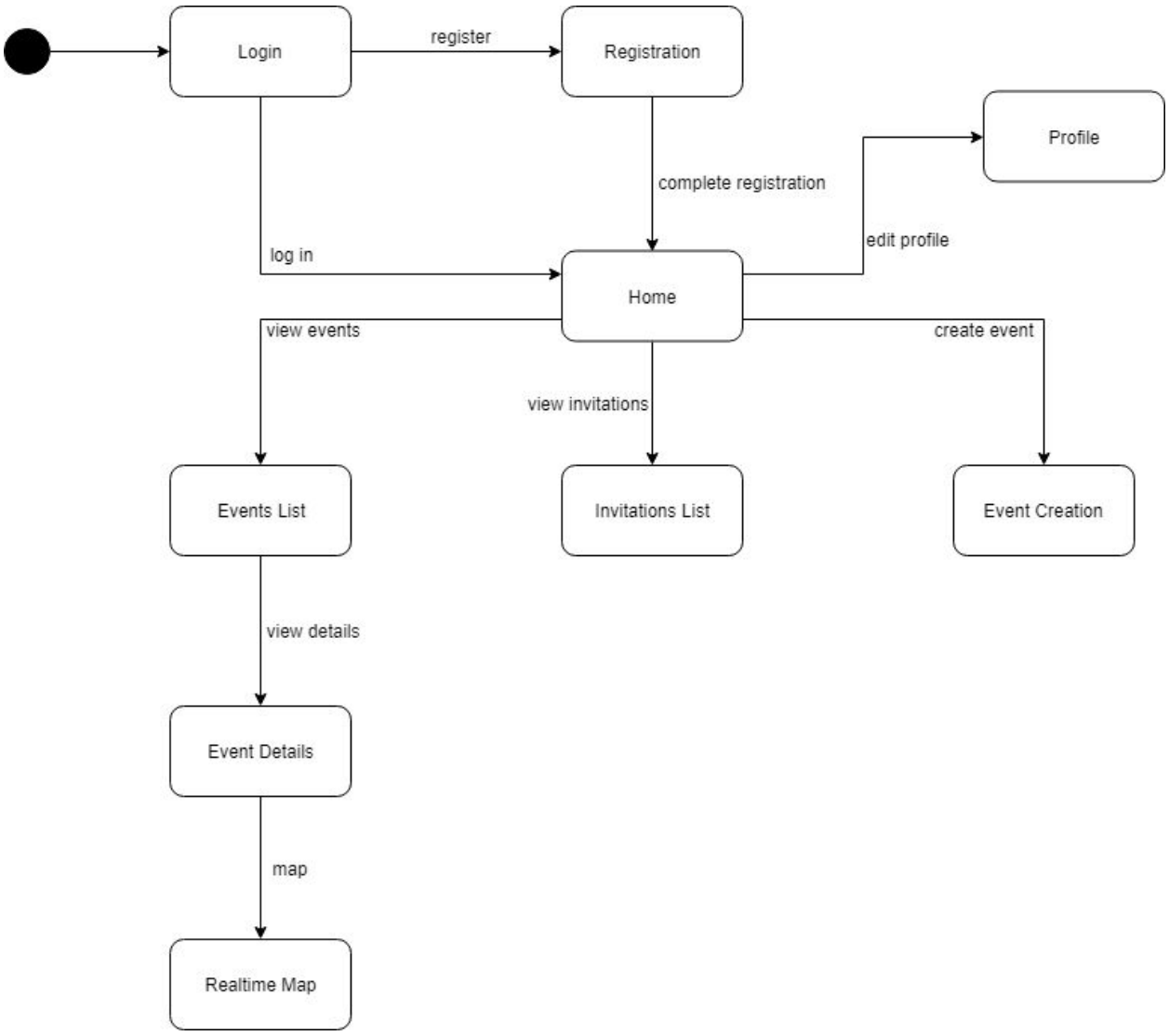
Sequence of Events Overview

The following sequence diagram shows a regular use of the application. The client is initiated immediately after the user opens the app. All of the user's request will be done through the client. For most requests, the client will send an information request to the server; the server will then send a query or a series of queries to the database to retrieve the information, then organize it in an appropriate format and send it back to the client. The exception to this is geographic information. In this case, rather than querying the database, the server will send location requests to the clients of interest and will return this information to the requesting client.


Client
Server
Database
open app
request events list
request events list
return events list
return events list
request invitations list
request invitations list
return invitations list
return invitations list
accept invitation
update invitation status
return success
request users' locations
return users' locations

UI State Overview

The following diagram shows an overview of the different states of the application's user interface. When the user first opens the application they will be taken to the login screen. In case they don't have an account, they will have the option to register and create one. Once the login is complete or the account is registered, the user will be redirected to the app's home page, where they can choose one of the many options the application gives to them by simply opening the menu and tapping the option. From any of the states it is possible to return to the previous state, up until the home page.



Design Issues

Functional Issues

1. What types of notifications should users receive?
    - Send notifications to users for everything
    - Send notifications based on users preferences
    - Send users notifications for certain things, allow them to customize other aspects

   Choice: send notifications based on users preferences
   Justification: Users will enjoy the customizability of the app. Sometimes users prefer to check on the app themselves rather than receiving notifications. In this case they can simply turn all their notifications off.

2. How should users login to their account?
    - Username and Password
    - Email and Password
    - Phone Number and Password

   Choice: Username and Password
   Justification: It matches users' expectations to login using username and password.

3. What details would be included in creating an event?
    - Event name, date, time, attachments, notes
    - Event name, date, time
    - Event name, date, time, attachments

   Choice: Event name, date, time, attachments, notes
   Justification: name, date, and time are the most important aspects of a meeting. Sometimes there are flyers associated with an event that can provide users more details. If there is not a flyer associated with an event or if the admin would like to provide additional information it is useful to have a notes sections.

4. How will we recommend locations to users?
    - Show them the most used locations in the country the user is in.
    - Show them the most used locations in the state the user is in.
    - Show them the most used locations based on the radius around where the user currently is.

Choice: Show them the most used locations based on the radius around where the user currently is.
Justification: The radius around where the user currently is provides locations that are most accessible to the user. Most of the time people would not drive to the other side of the state or to another country to meet up with people.

5. How will users recommend a location to an event admin?
    - Create a location recommendation and send a notification to the admin.
    - Create a location recommendation and let the admin view it in his recommended locations.
    - Create a location recommendation and put it in a separate recommended locations section.

   Choice: Create a location recommendation and send a notification to the admin.
   Justification: It ensures the admin receives the recommendation and prevents the addition of another interface.

Nonfunctional Issues

1. What language will our Android app be written in?
    - Java
    - C++
    - Kotlin

   Choice: Java
   Justification: The Android framework gives strong preference to JVM languages, and all of our group members already know Java. Additionally, the vast majority of Android development resources assume using Java.

2. What OS versions will our Android app support?
    - 4.4 KitKat
    - 5.0/5.1 Lollipop
    - 6.0 Marshmallow
    - 7.0/7.1 Nougat
    - 8.0/8.1 Oreo

   Choice: 6.0 Marshmallow
   Justification: The Android 6.0 release added new and better functionalities regarding notifications, accessing location data, and runtime permissions. While some users do not have access to newer phones that can support Android 6.0, we can still reach 71% of all Android users while also taking advantage of these

new features.

3. What framework/language will our backend server use?
    - Django (Python)
    - Flask (Python)
    - Express (Node.js/JavaScript)
    - Ruby on Rails (Ruby)
    - Spring Boot (Java)

   Choice: Django (Python)
   Justification: While all of our members already know Java, none are familiar with Spring Boot. Two of our members both have prior experience with Django, and Python is a relatively easy language to learn.

4. How will our backend server be hosted?
    - Amazon Web Services
    - DigitalOcean
    - Google Cloud Platform

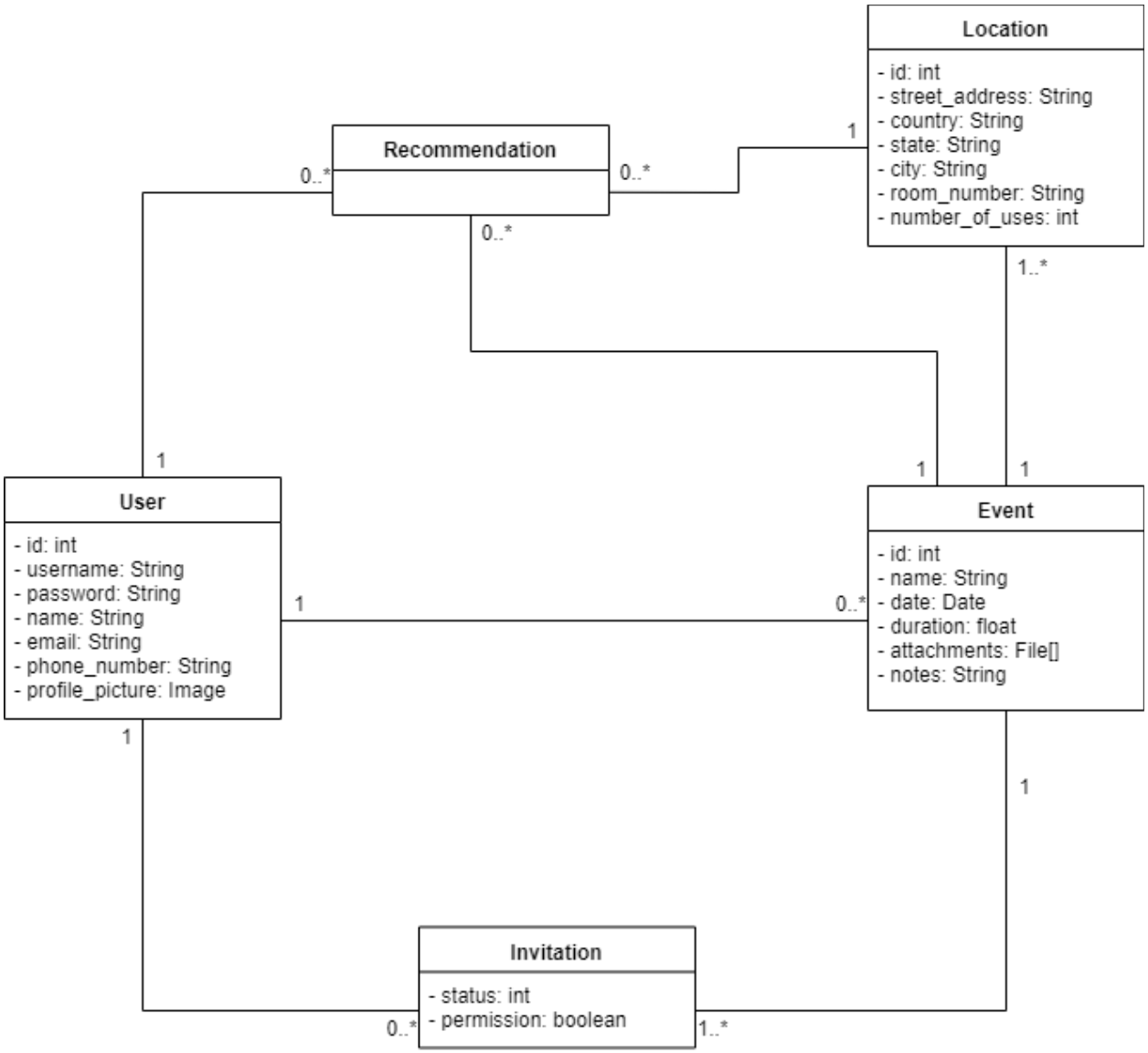
   Choice: DigitalOcean
   Justification: DigitalOcean provides the simplest platform and pricing model. While AWS and GCP can provide more features, our project does not need the extra complexity.

5. What database our backend server use?
    - MySQL
    - SQLite
    - PostgreSQL

   Choice: PostgreSQL
   Justification: DigitalOcean provides a pre-built image for deploying Django applications which also deploys a PostgreSQL database.

Design Details

Class Design



Description of Classes and Interactions

Classes are based on the elements that are relevant to the application and what information the application needs to function properly. They can be easily translated into tables to be stored in our relational database.

User

- Represents the real users of the application.
- Is created when a real user registers for an account in the application.
- Each user has a unique id to distinguish them from other users; this id is generated by the application when an account is created.
- To execute the login, each user has a username and a password.
- In order to ease the identification of users by other users, each user has a name and an email account, and optionally a phone number and profile picture. These features will allow other users to search for a specific user in the application more easily.

Event

- Represents the events the users organize and attend.
- Is created when a user creates a new event.
- To distinguish events with similar attributes, each event has a unique id, generated by the application right when it is created.
- Events will have a name for users to identify it by.
- Each event has a date and a duration, representing the information of the event's time.
- Events can optionally include notes and attachments, in order to give the users additional information that might be relevant to the event.
- A user that creates an event will be from then on associated to that event as its admin, and will have complete permissions to change the information of said event.

Location

- Represents real life locations users will use to host events.
- Is created when, during the creation of an event, a user chooses a place that is not currently present in the system.
- Similar to users and events, locations have a unique id to distinguish them from each other.
- To properly identify the real life location a location represents, each location has a street address, city, state and country.

- Optionally and if applicable, locations can have a room number, which gives users even more detailed information.
- Each location keeps track of the number of times it has been used in order to, later on, recommend the nearby most popular locations for events to a user as they create an event.

Invitation

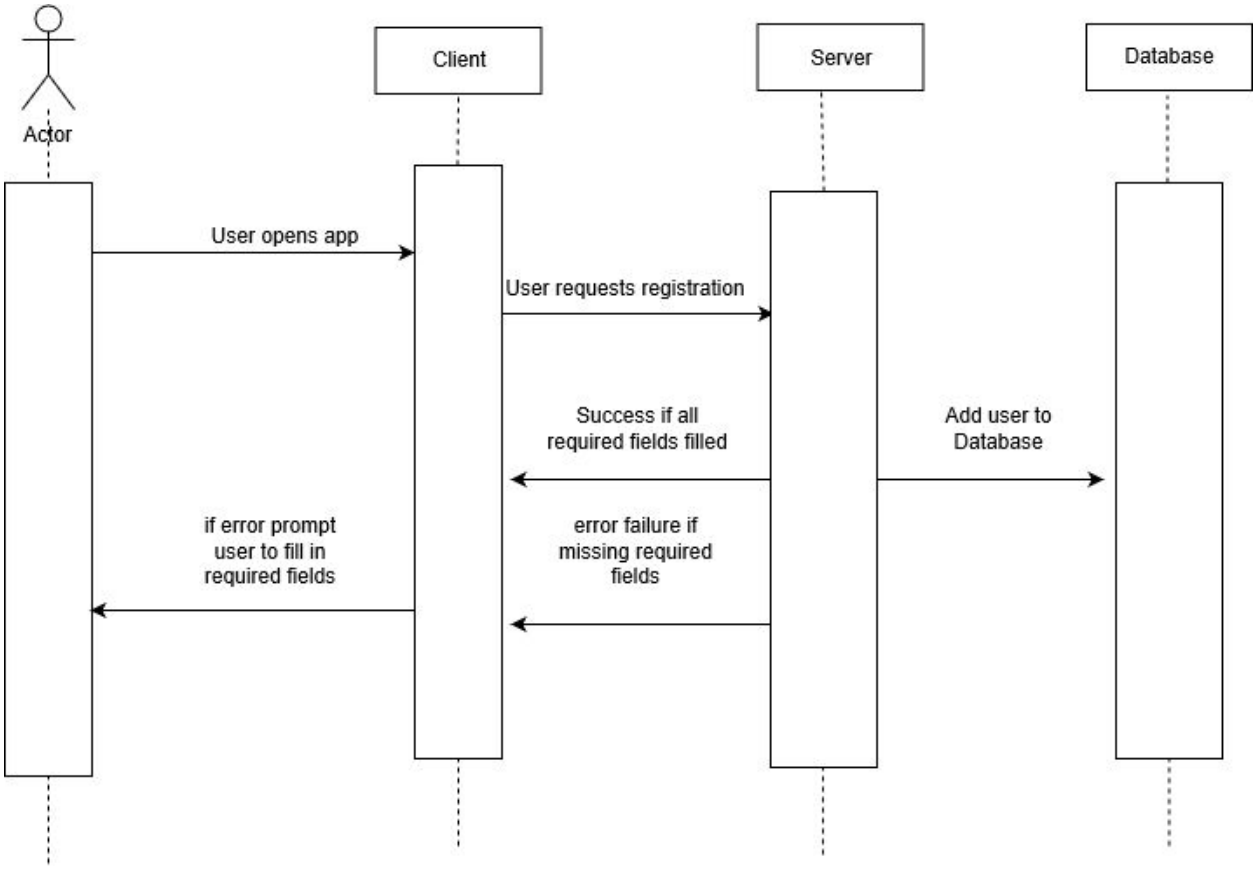
- Is created when the user that serves as event admin adds other users to the event.
- Identifies the users that will attend the event other than the event admin, known as attendees.
- Each invitation is identified by its attendee and its event.
- Invitations have a status, which identifies whether the user will be attending the event or not; users can change this status anytime.
- Each invitation has a permission, which determines whether the user can make changes to the event in a similar fashion to the event admin or not.

Recommendation

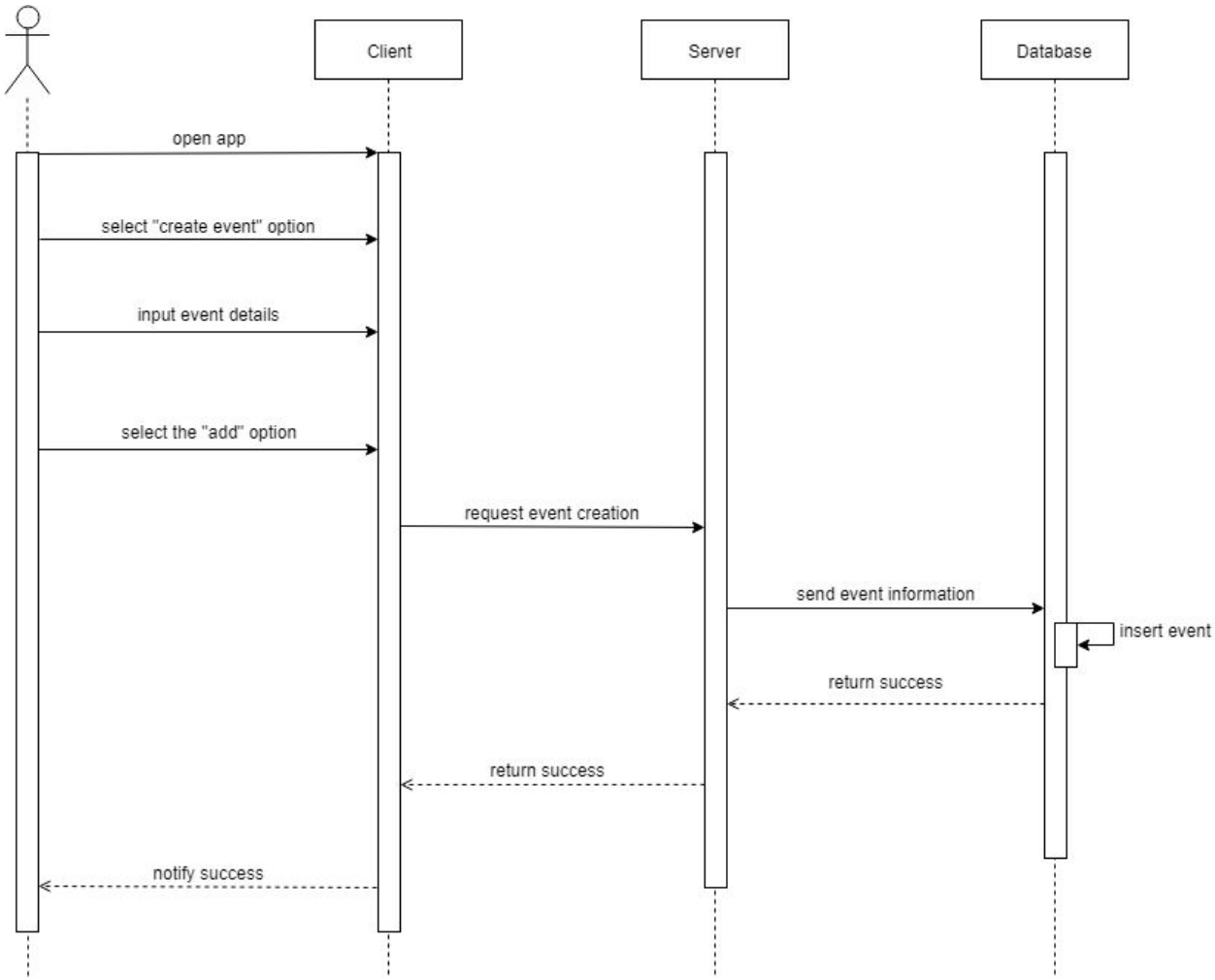
- Represents the recommendations that attendees make to an event admin about possible locations for an event.
- Is created when a user makes a recommendation.
- Recommendations are identified by the user that made them, the event they are about and the location that is being recommended for said event.
- An event admin will be able to view a recommendation for their event, and can then make changes depending on their decision.

Sequence Diagrams

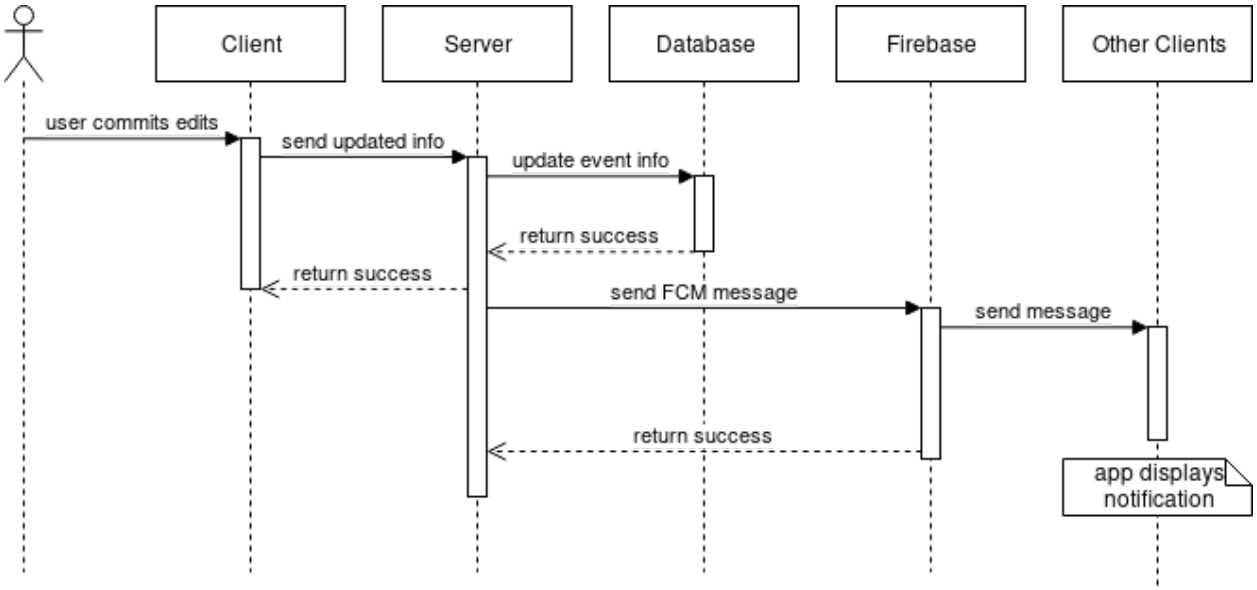
- Sequence of events when a user creates an account



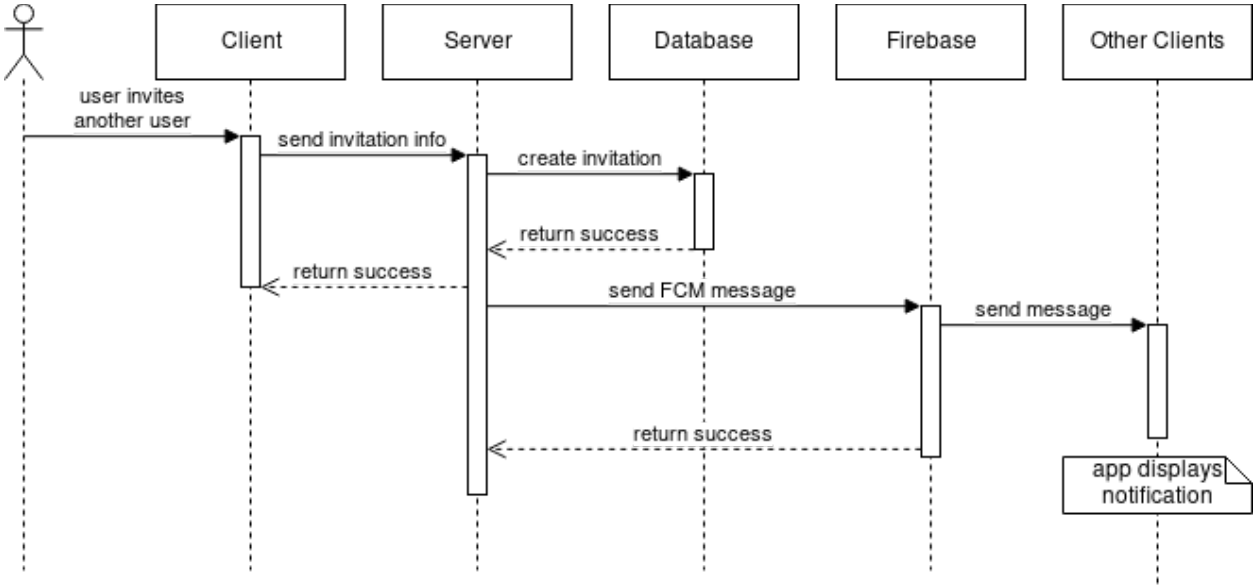
- Sequence of events when a user creates an event



- Sequence of events when a user edits an event

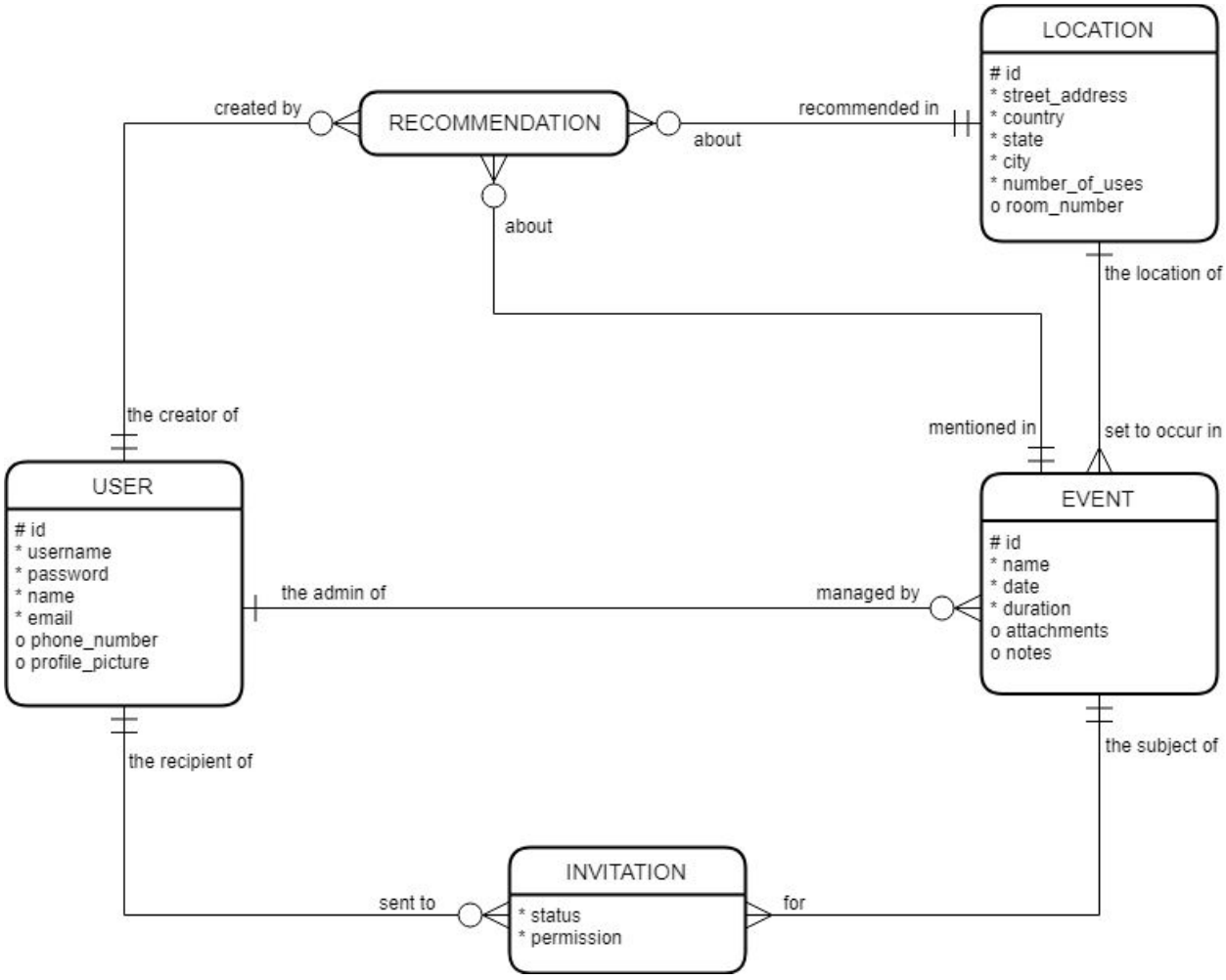


- Sequence of events when a user invites another user



Database Design

Information relevant to the application will be stored in a relational database created in the PostgreSQL database management system. Each class in the application will correspond to a table in the database. The following entity-relationship model represents the structure of the database. The unique attributes and relationships will be represented in the database using primary keys and foreign keys, respectively.



Endpoints

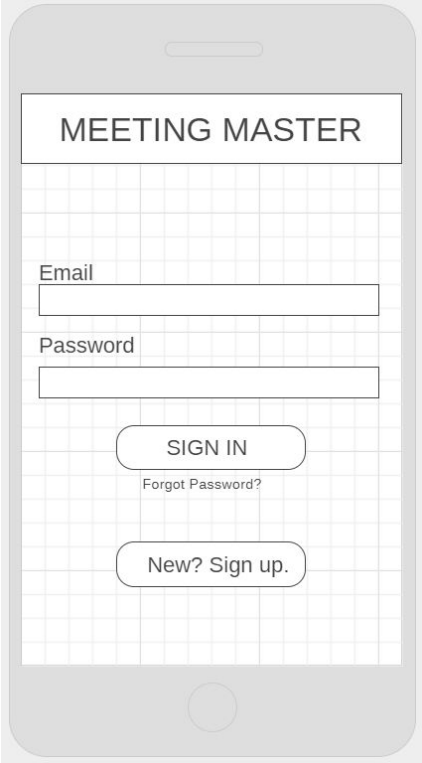
The endpoints describe the communication methods and protocols between the client and the server in our system, specified by classes/entities. A few classes have the attribute pk, which corresponds to that class's unique id.

User	/user/ /user/{pk} /user/{pk}/invitations	- GET, POST - GET, PATCH, DELETE - GET
Invitation	/invitation/ /invitation/{user_id}/{event_id}	- GET, POST - GET, PATCH, DELETE
Event	/event/ /event/{pk} /event/{pk}/invitations /event/{pk}/recommendations /event/{pk}/recommendations/{location_id}	- GET, POST - GET, PATCH, DELETE - GET - GET, POST - GET, PATCH, DELETE
Location	/location/ /location/{pk}	- GET, POST - GET, PATCH

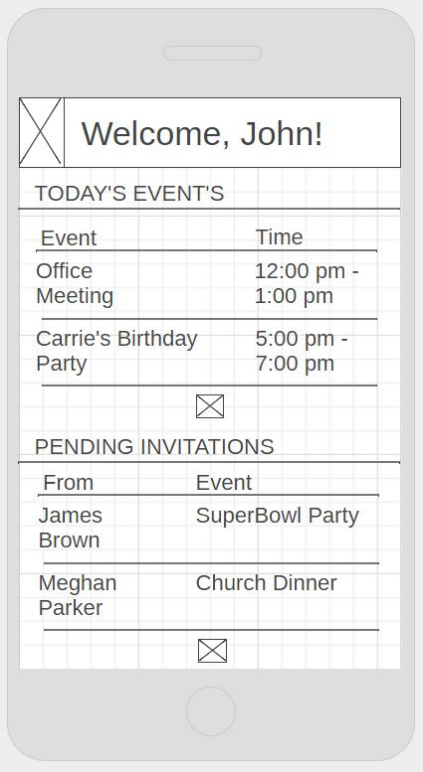
UI Mockups

The following images show a mockup of the system's UI. Colors will be determined during development.

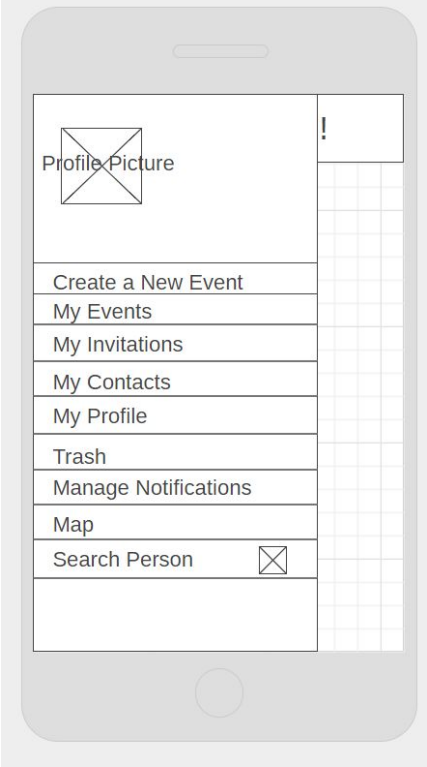
- Login page



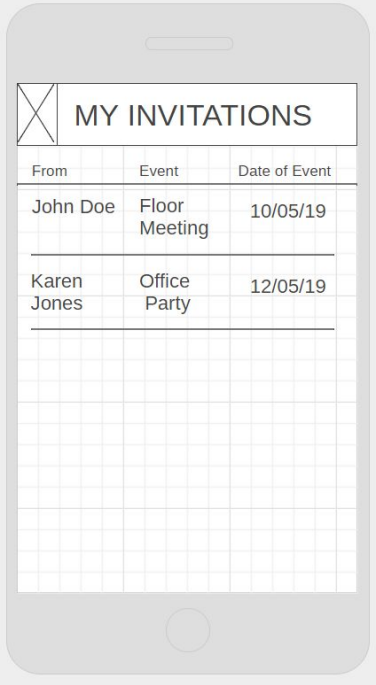
- Homepage



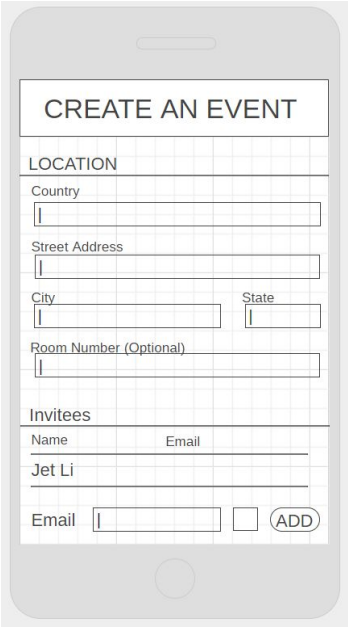
- Menu Opened

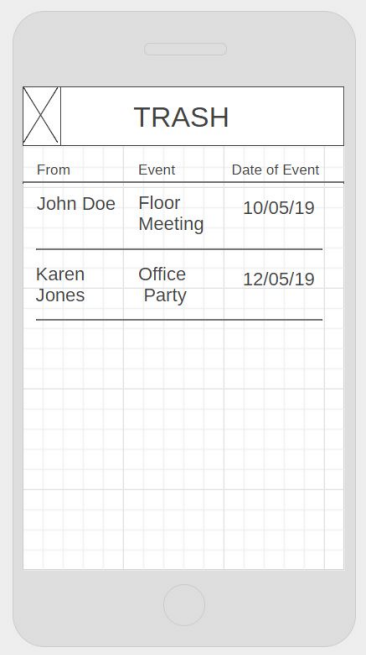


- My Invitations/ Trash/ My Events

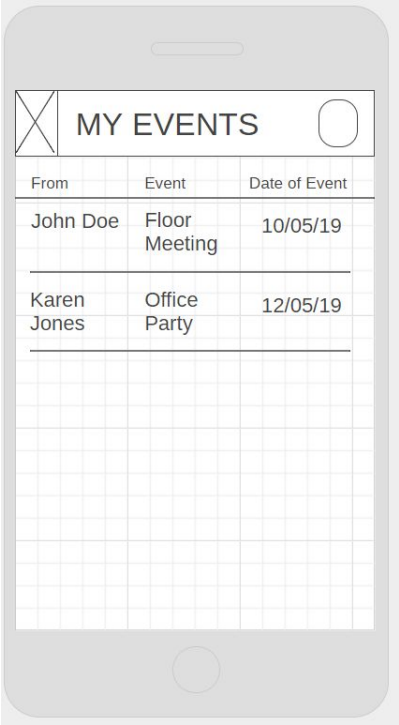


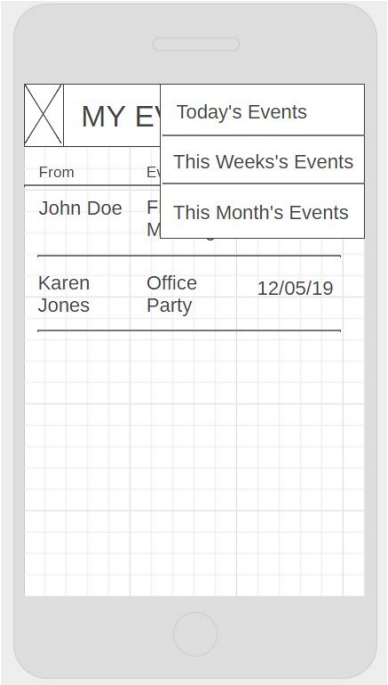
- Create an Event



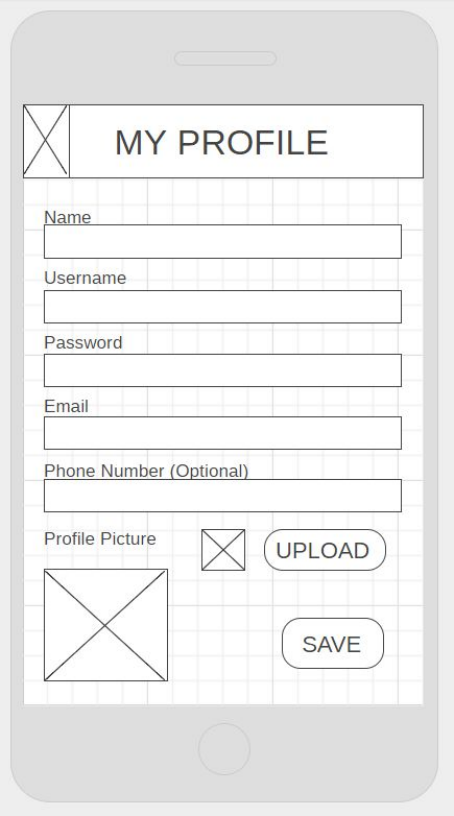


Square box next to add is search button

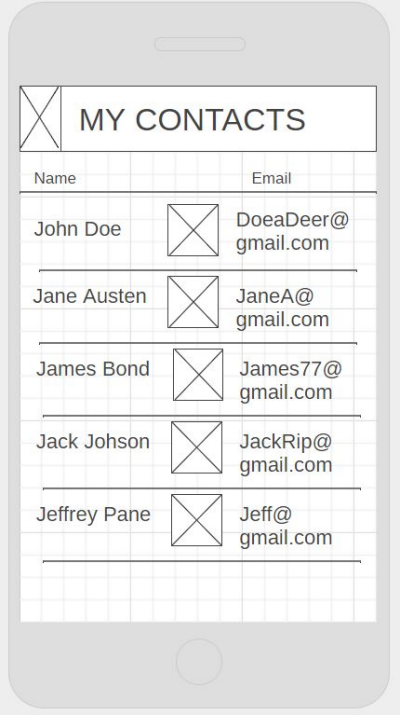




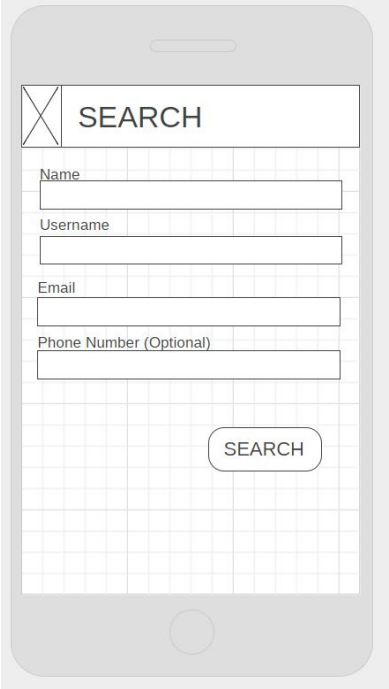
- My Profile



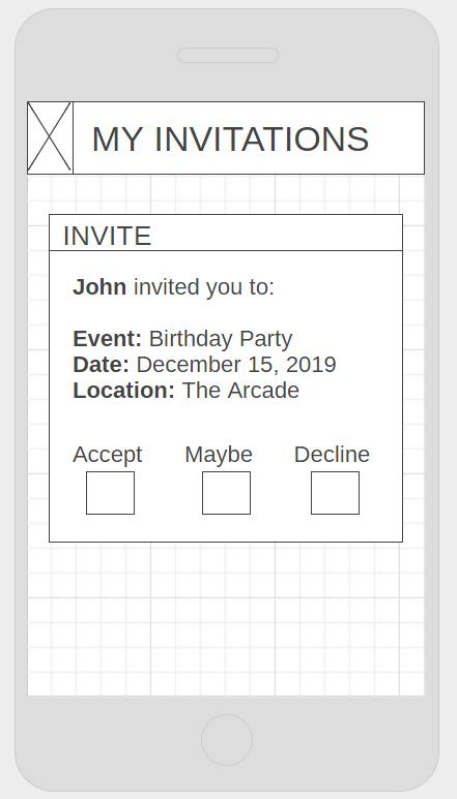
- Contacts



- Search Person



- Invite



- Map