



Sirkel - Design Document

09.12.2018

Team 27

Dan Hipkind, Eli Smith, Trevor Neidlinger, Tianchi Xu, Adam Lula

Index

● Purpose	2
○ Functional Requirements	
○ Non-Functional Requirements	
● Design Outline	6
○ High Level Overview of the System	
○ Detailed Overview of the Client-Server Database System	
○ Broad Overview of Sequence of Events	
● Design Issues	9
○ Functional Issues	
○ Non-Functional Issues	
● Design Details	13
○ Class Diagram	
○ Class Descriptions and Interactions	
○ Sequence Diagrams	
○ Activity Diagrams	
○ UI Mockups	

Purpose

A student's campus involvement is key to a successful college career. University clubs provide smaller communities for students to make friends and work with people who share their passions. However, for many students it is difficult to find clubs that match their interests or have the courage to approach clubs at crowded clubs fairs. We believe that it is important to give each student an equal opportunity to feel welcome and a part of an accepting community.

The purpose of Sirkel is to connect incoming freshman as well as existing students at Purdue University to smaller communities like University clubs who share their passions. Sirkel will provide intelligent suggestions for clubs that each user may enjoy and help connect students to other people with similar interests on campus. With the help of our product, each student on campus will have a more fulfilling college experience and find their Sirkel.

Functional Requirements

1. User Account

As a User,

- a. I would like to be able to register for an account
- b. I would like to be able to login and manage my account
- c. I would like to be able to have reputation points
- d. I would like to verify my email address
- e. I would like to be able to reset my password if necessary

As an Organization,

- f. I would like to be able to create a public profile
- g. I would like to be able to login and manage my account
- h. I would like to verify my email address
- i. I would like to be able to list information about my club on a webpage

2. Event Posting

As a User,

- a. I would like to be able to create an event
- b. I would like to be able to meet up on campus with other students

As an Organization,

- c. I would like to be able to post about upcoming events
- d. I would like to be able to reach a large audience
- e. I would like to be able to reach out directly to students who may be interested in my club
- f. I would like to be able to see which and how many students plan to attend an event

3. Organization Management

As a User,

- a. I would like to be able to connect with a club page and receive information from them

As an Organization,

- b. I would like to be able to set the role and the status for students in my organization
- c. I would like to be able to see which members have paid their clubs dues
- d. I would like to be able to send reminders to members who who still need to pay club dues

4. Suggestion System

As a User,

- a. I would like to be able to connect with other students
- b. I would like to be able to find other users with similar interests

- c. I would like to be able to see a list of clubs on campus
- d. I would like to be able to get suggestions for clubs that I might like
- e. I would like to be able to get notification if there is a new match
- f. I would like to be able to take a quiz to be matched with things that interest me

Non-Functional Requirements

1. Client Requirements

As a Developer,

- a. I would like to be able to access the website on different screen sizes
- b. I would like to be able to access the website from any browser

2. Server Requirements

As a Developer,

- a. I would like the server to handle traffic surges with a cache server
- b. I would like to be able to store data that comes through the server to a local database
- c. I would like the server to be able to store user data and preferences to the database

3. Appearance Requirements

As a Developer,

- a. I would like the UI to be clean and intuitive to use
- b. I would like the application to follow a Sirkel color scheme
- c. (If time allows) I would like the user to be able to customize certain aspects on the UI

4. Performance Requirements

As a Developer,

- a. I would like the application to run smoothly without crashing
- b. I would like the application to be launched in under 5 seconds
- c. I would like the application to support 40,000 users

5. Security Requirements

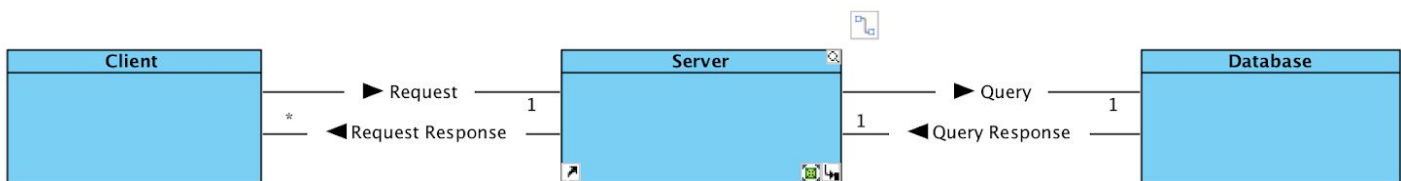
As a Developer,

- a. I would like users to be able to choose what information can be seen by others
- b. I would like users to be banned for poor behavior on a meet up or online
- c. I would like users to be able to set a public place to meet up to avoid possible threats or uncomfortableness
- d. I would like for each user to only be able to create one account to avoid abuse

Design Outline

High Level Overview of the System

Sirkel will be a web application with a many-to-one client-server architecture. We will implement this server using NodeJS with Express. Once implemented the server will be able to accept or deny user requests, access and store data to a database, and relay data back to the user. A database will be populated with the user data and that data will in turn be sent back to users with proper privileges attempting to access it.



Detailed Overview of Client-Server Database System

1. Client

- The client will serve as the interface for the user to interact with our web application
- The client will be completely separate from the server and will make API calls to server endpoints
- The client will receive JSON data from the server and update the user interface accordingly

2. Server

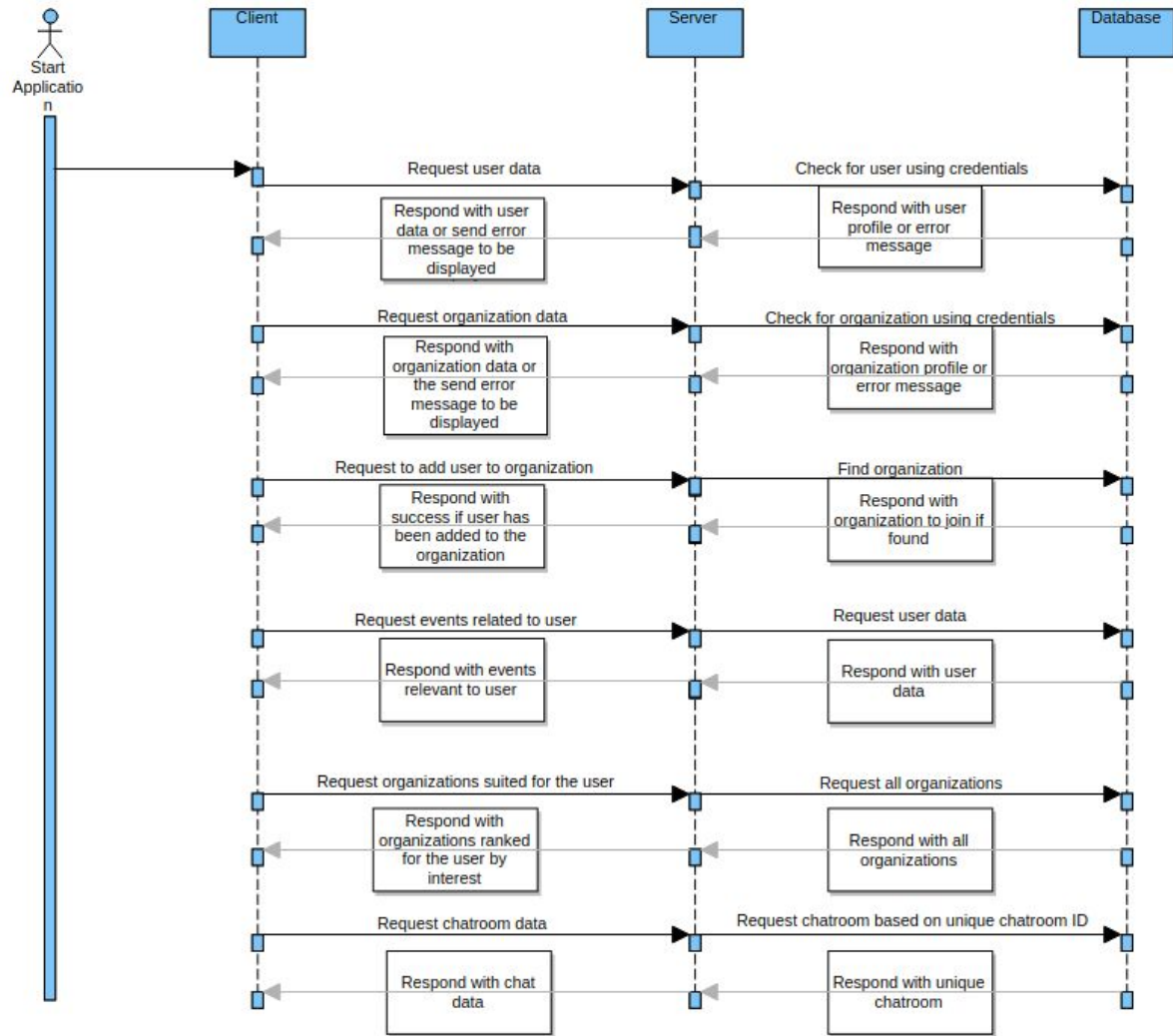
- a. The server will handle API calls from the client and respond accordingly with JSON data
- b. The server will make requests for data from the database
- c. The server will update the database whenever there are valid requests by the client to do so

3. Database

- a. The database will hold all data that must be persistent
- b. The database will not be relational and as a result will store and append data more fluidly
- c. The database will store personal interests as well as their connection status so that we can use algorithm to categorize them

Sequence of Events Overview

The diagram below shows a broad overview of the interactions between the client, the server, and the database. It shows the order of requests and responses made by the three entities. In the diagram, we have the basic flow of requests for information from the database and requests to change data in the database. The typical structure is that the client makes a request to the server, the server then validates the request and makes a request to the database. The database then responds with data or an error. The server then manipulates (or doesn't) the data and sends a response with the data to the client.



Design Issues

Functional Design Issues

Design Issue #1 - How will we give intelligent club suggestions to the user?

Option 1: Have the user take a short interests survey

Option 2: Have the user choose between a selection of clubs to see which they prefer

Option 3: Make suggestions based on clubs the user is already in

Choice: Option 1 and Option 3

Justification: A new user is not likely to know much about individual clubs based on their names alone, so this eliminates Option 2. Option 1 can provide data that would help with matching. Option 3 can provide more options based on what other users in the same clubs are also in.

Design Issue #2 - How should we allow the users to communicate with each other?

Option 1: Real time direct message using websockets with push notification

Option 2: Send message to the server and notificate user when they are online

Choice: Option 1

Justification: One of the most important things in our project is the speed of interaction. Choosing Option 1 gives users quick real time experiences when they are trying to meet new people and get notifications. Option 2 is relatively easy to implement but if users are not online, that means very slow back and force communication, which is outdated.

Design Issue #3 - How can we stop a user from spamming another user?

Option 1: Have a sizeable cooldown to prevent continuous messages.

Option 2: After a certain number of similar messages sent from one user to another, new messages that are similar will be flagged and an incrementing cooldown will be implemented to slow down the spam.

Option 3: Add a block feature

Choice: Option 2 and Option 3

Justification: Option 1, even though it would prevent a lot of spam, would seriously hurt the normal user's experience. With Option 2, we can still implement a cooldown to prevent continuous messages, but it wouldn't hurt the user's experience to the degree that Option 1 would since it is conditional. Option 3 is also a good solution as it can prevent further spamming and at the same time undesirable messages from a single user.

Design Issue #4 - How are we going to filter events that we show each user?

Option 1: Show the user every event posted by every organization.

Option 2: Show the user only events posted by the organizations that they are a part of.

Option 3: Show the user events filtered by their interests.

Choice: Option 1, Option 2, and Option 3

Justification: We will default the activities feed to show only the events posted by the organizations that they are a part of. The user can then select which filter they would like to apply. This gives the user more freedom and multiple ways to explore events around campus.

Non-Functional Design Issues

Design Issue #1 - How can we keep the servers from slowing down if there is a traffic surge due to a large club event

Option 1: Implement a queue for people trying to access that specific event so other people won't be slowed

Option 2: Create a cache server so that the information for that event is easily accessible and won't slow down other users

Choice: Option 2

Justification: A cache server is optimal choice because it will not affect the user experience. Putting users in a queue to use certain functionality of the site is sure to make them quit the application entirely. Using a cache would allow for an optimal and fast experience for all the users. We plan to use Redis for this because of its speed and ability to manipulate data.

Design Issue #2 - How are we going to host the backend?

Option 1: Personal Computer

Option 2: DigitalOcean

Option 3: Amazon Web Services (AWS)

Choice: Option 1 and Option 2

Justification: Hosting on our personal computer is quick and easy way to start when we are developing our project since there will be very few requests coming to the server. Also, it is easier to modify our code base on personal computer. Later on when the project is mature, we can deploy it on DigitalOcean for everyone on the internet to access. We won't choose AWS because we are not familiar with the complex platform they offer.

Design Issue #3 - What Database implementation is most practical for our data

Option 1: MySQL

Option 2: MongoDB

Option 3: Oracle XE

Choice: Option 2

Justification: MongoDB is the most practical choice for our data because it needs to be very fluid. A NoSQL database allows for ultimate flexibility, which allows our users and organizations to not rely on rigid schemas. Given that we must store varying

degrees of data on users and organizations, this database choice is the optimal one. Users and organizations in particular can have varying numbers of interests. This would be more difficult to implement in a SQL database.

Design Issue #4 - How are we going to ensure a user's privacy and security?

Option 1: Encrypt personal data(messages,etc) in our database

Option 2: Store passwords hashed with their salt along with their salt instead of plaintext using Passport.js

Option 3: Use HTTP

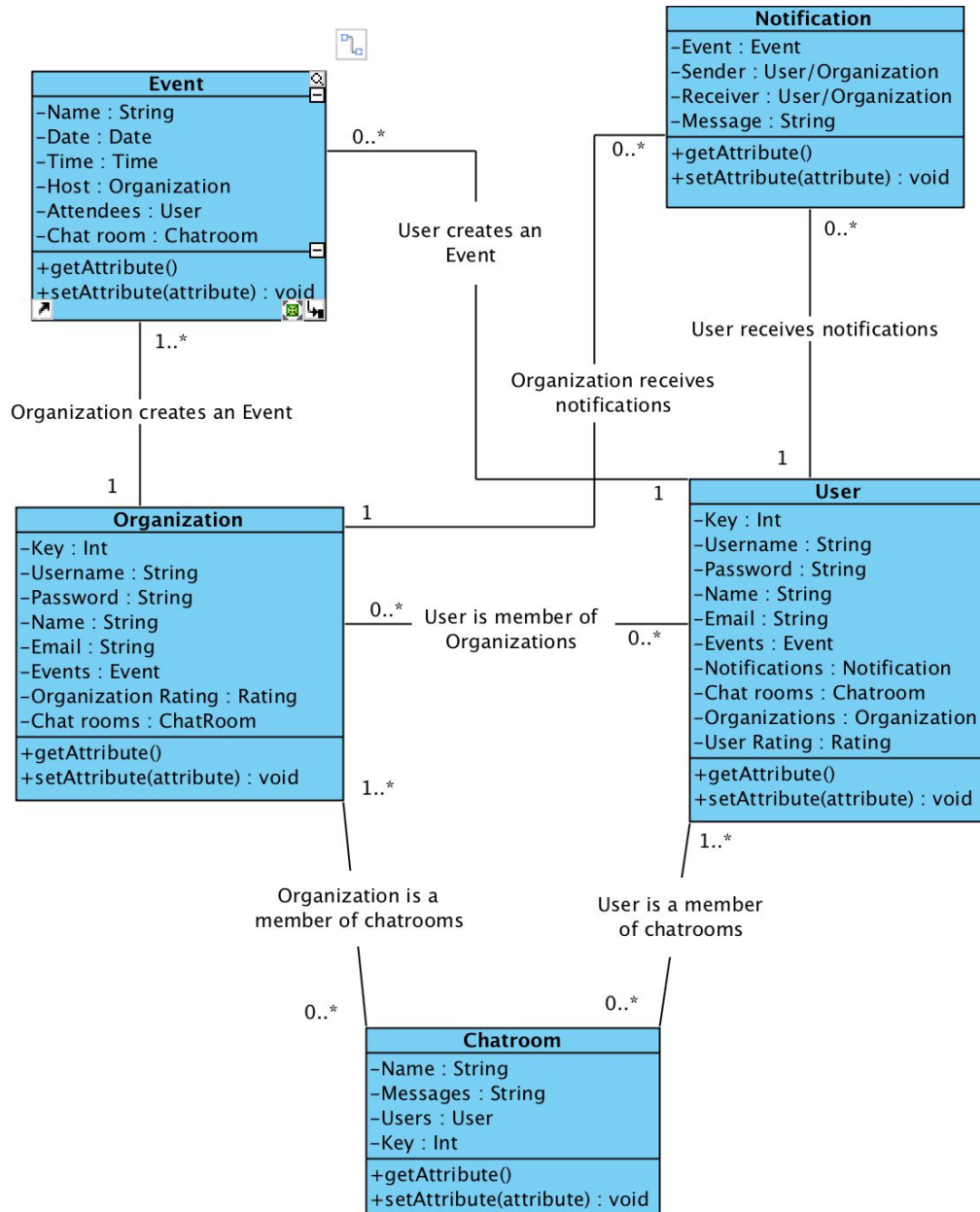
Option 4: Use HTTPS

Choice: Option 1, Option 2, and Option 4

Justification: Option 1, 2, and 4 are all essential to a user's privacy and security. Option 1 and Option 2 ensure a users privacy and security so that if somehow our database was intruded, the user's data and password would still be safe. Option 3 is not a good idea since a third party could pretend to be the owner of our domain. Option 4 is a much better idea since it can ensure safe password and data travel and integrity.

Design Details

Class Diagram



Class Descriptions and Interactions

→ User

- ◆ The User object is created when someone signs up in our product.
- ◆ Each user will have a unique ID
- ◆ Each user will have a unique username and email and a password
- ◆ Each user will have chatting room keys
- ◆ Each user will have a list of organizations they are in
- ◆ Each user will have a reputation rating

→ Organization

- ◆ An organization object is created when a user creates/links a organization
- ◆ Each organization will have whether they are verified or not(through BoilerLink,etc)
- ◆ Each organization will have a list of users
- ◆ Each organization will have the role of each of its users(President, Moderator, etc)

→ Chat Room

- ◆ A chat room will have a name
- ◆ A chat room will have message history
- ◆ A chat room will have which users are in it
- ◆ A chat room will have unique ID

→ Notification

- ◆ A notification will have its event details
- ◆ A notification will have a sender
- ◆ A notification will have receivers
- ◆ A notification will have its message details

→ Event

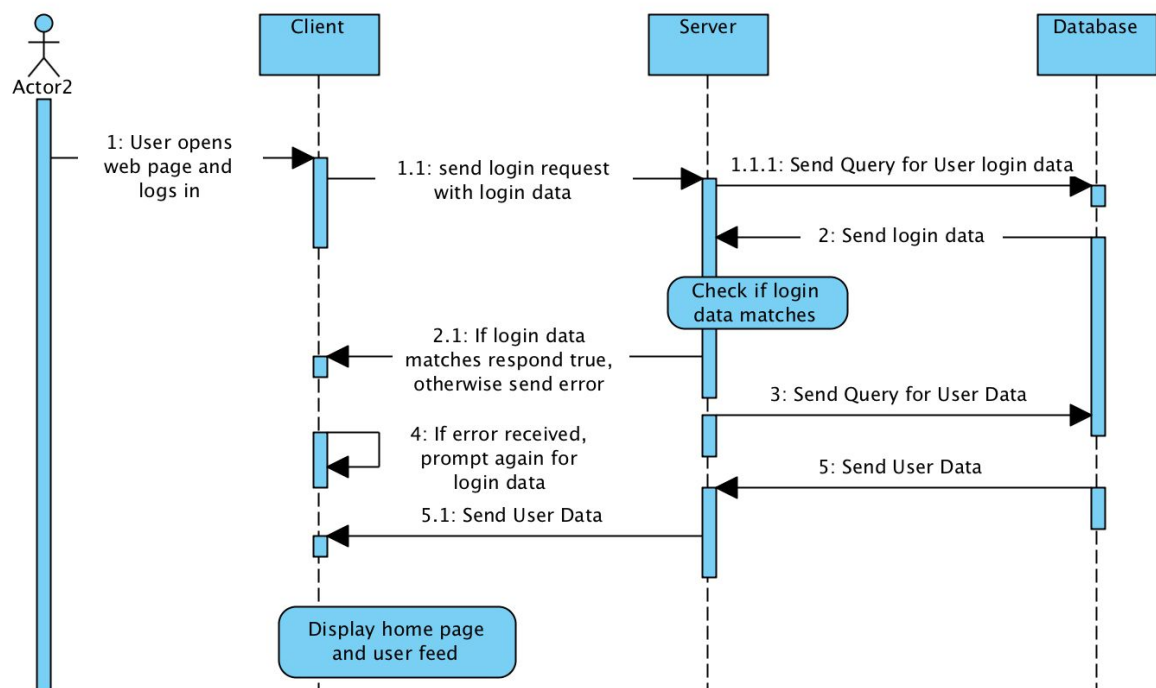
- ◆ A event will have its name

- ◆ A event will have its date and time
- ◆ A event will have its host
- ◆ A event will have its attendees
- ◆ A event will have a chat room

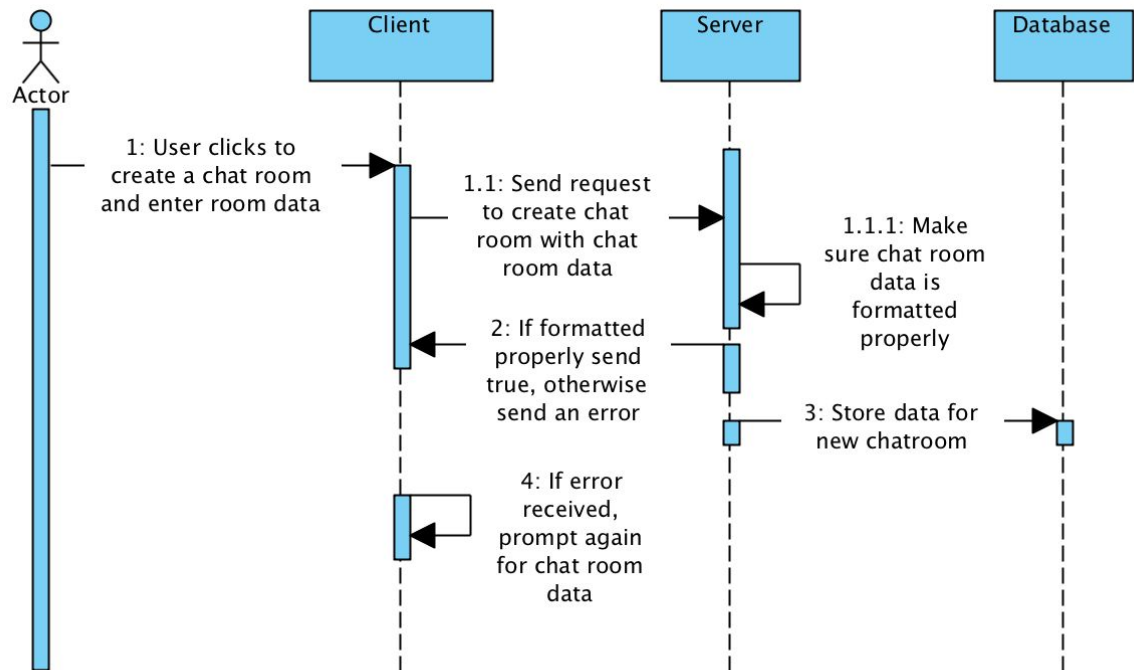
Sequence Diagrams

These diagrams depicted below show the sequence of events for major events in our application, including user login, creating a chat room, logging out, and seeing club suggestions. When a user does an action in the front end, the client will then send a request to the server. The server will then send a query to the database and then the database will send the requested data to the server. The server will then perform computation on the data if needed, then send the result to the client. The client will then process the data and re-render the UI if need be.

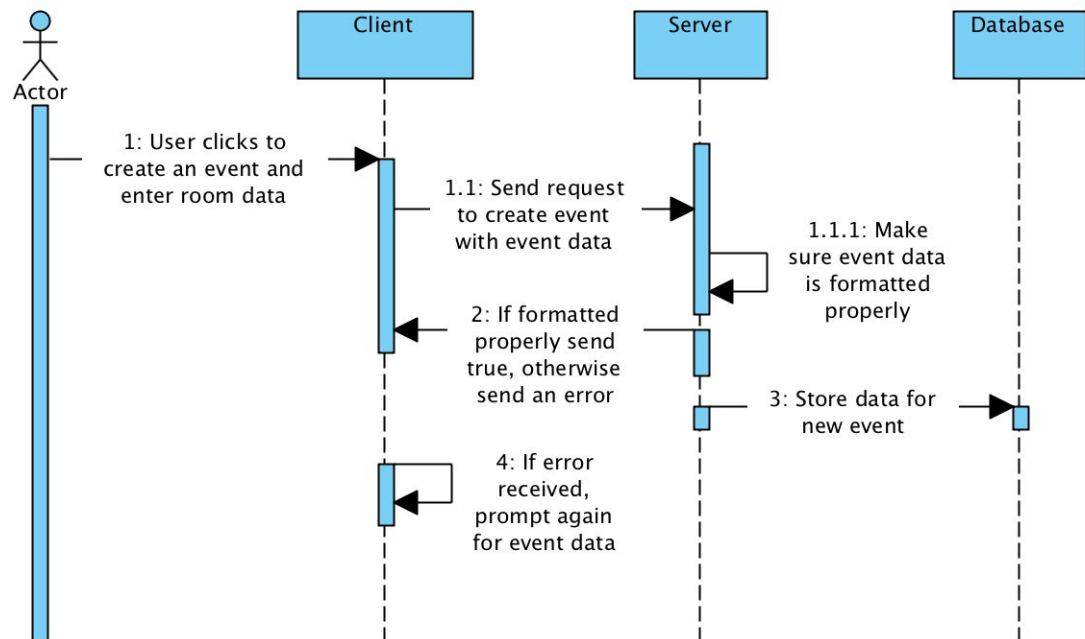
- Sequence of events when user logs into account



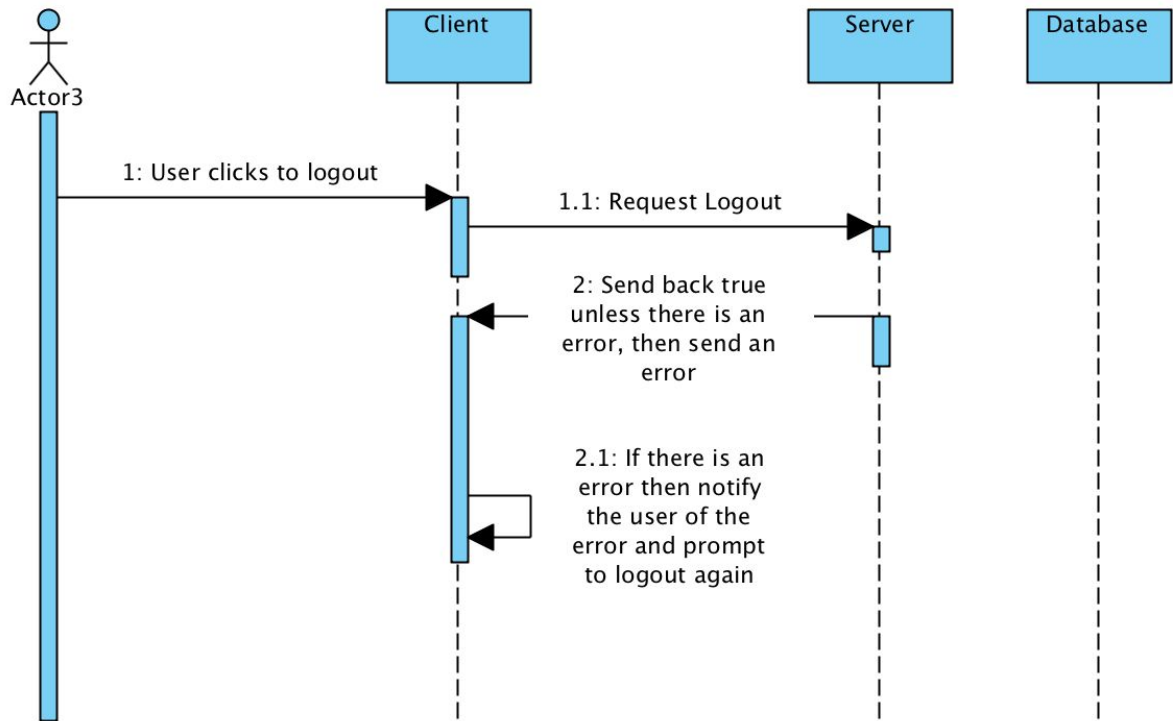
- Sequence of events for creating a chat room



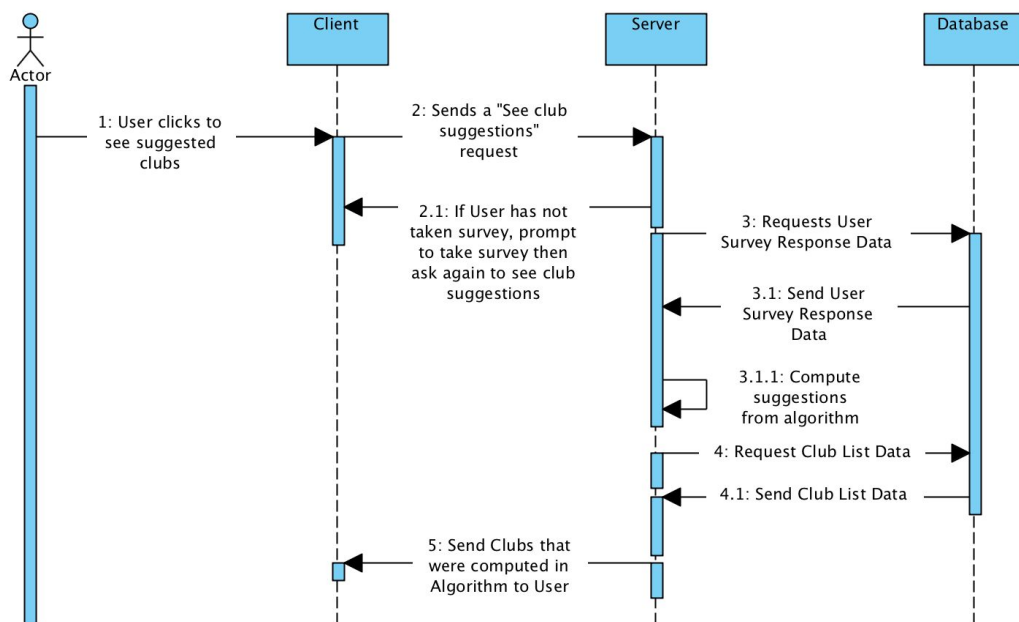
- Sequence of events for creating an event



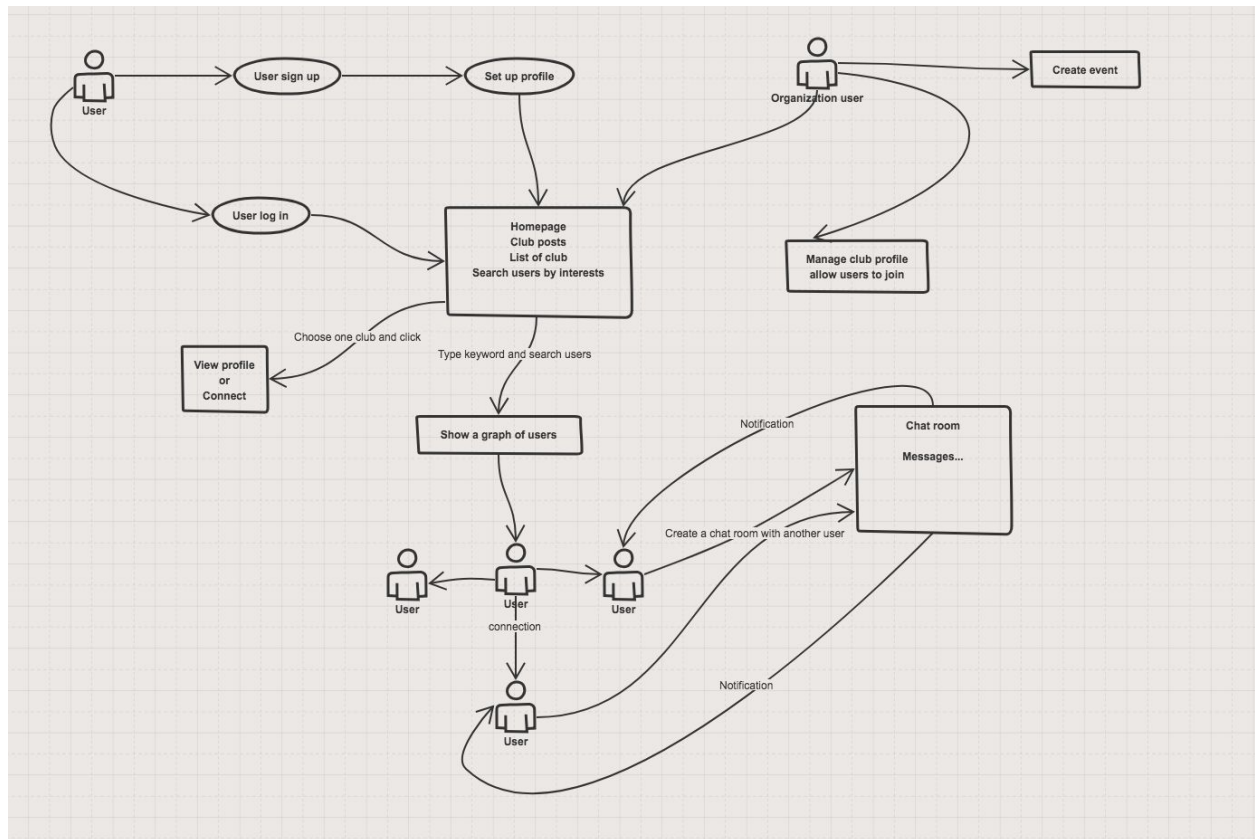
- Sequence of events for logging out




- Sequence of events to see club suggestions



Navigation Flow Map

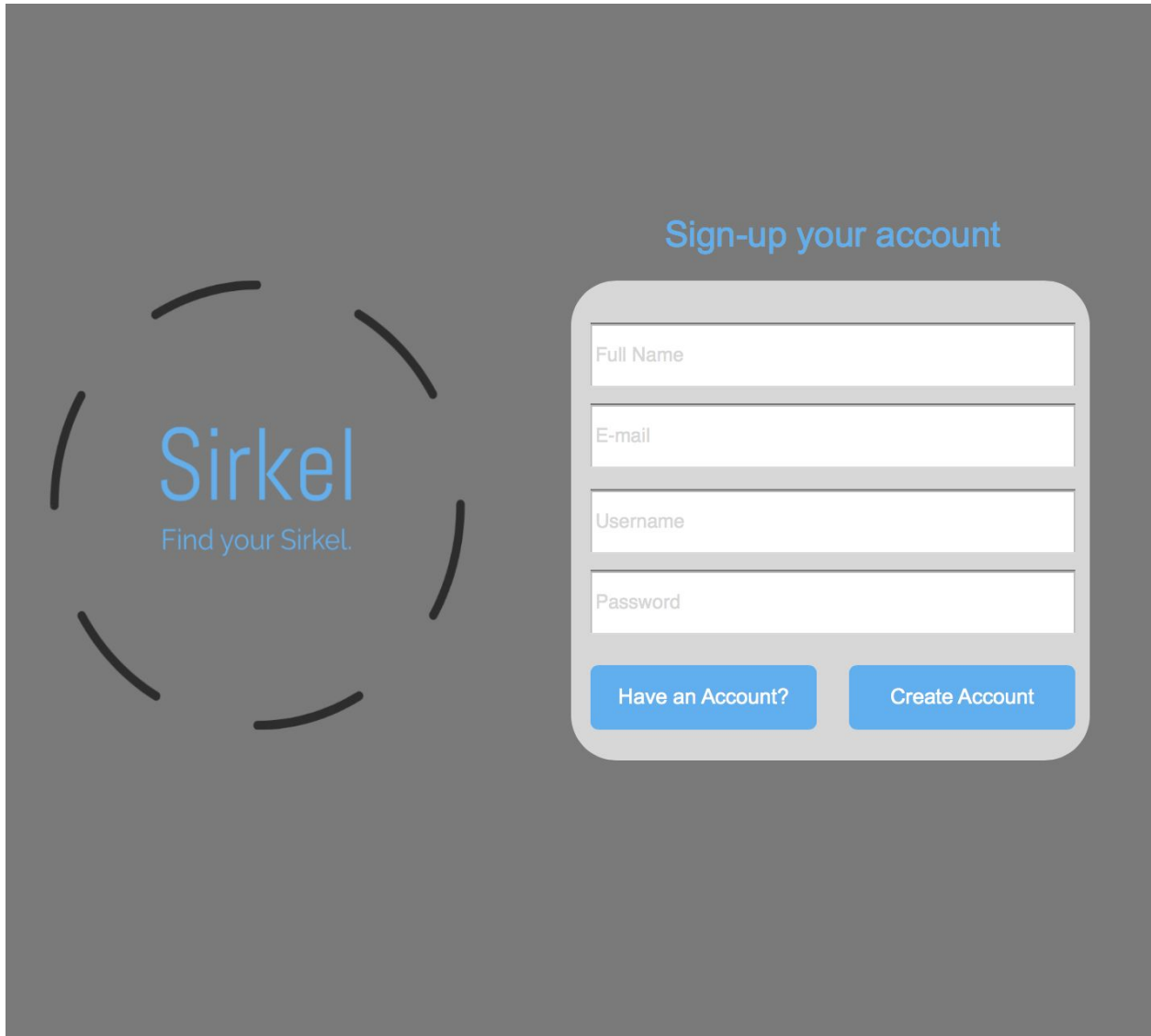


UI Mockup



The image shows a UI mockup for a login page. On the left, there is a circular logo with the word "Sirkel" in blue and the tagline "Find your Sirkel." below it. The logo is surrounded by several curved lines. On the right, there is a login form titled "Log-in to your account" in blue. The form contains two input fields: "Username or E-mail" and "Password". Below the input fields are two buttons: "Register Account" and "Log In".

This is the login page that the user will first see when they go to the website. They will have the option to log in to their account or register a new account if they do not already have one.



The image shows a sign-up page for 'Sirkel'. On the left, there is a logo consisting of the word 'Sirkel' in a blue sans-serif font, with the tagline 'Find your Sirkel.' below it. The logo is enclosed within a circular frame made of several black curved line segments. On the right, the heading 'Sign-up your account' is displayed in blue. Below this heading is a light gray rounded rectangular form containing four input fields: 'Full Name', 'E-mail', 'Username', and 'Password'. At the bottom of the form are two blue buttons: 'Have an Account?' on the left and 'Create Account' on the right.

This is the sign up page when you click on the register account button on the login screen. The user will have to input their full name, e-mail, username, and password then will create account. If someone has an account and accidentally got to this page they have the option to get back to the login page.

The screenshot shows a web application interface for a social media feed. At the top, there is a search bar with a magnifying glass icon, a logo for 'Sirkel' (a circle with a dot inside), and a user status 'Signed in as: lamUser' next to a gear icon for settings. The main content area is titled 'Live Feed - All Clubs' and features a 'Make a Post' button. Below this, there are four club entries, each with a title, a text input field, a 'Reply' button, a 'Send' button, and a timestamp. The clubs are labeled 'First Club', 'Second Club', 'Third Club', and 'Fourth Club'. To the left of the main feed, there is a sidebar with a user profile picture of a cartoon character with orange hair and green eyes, labeled 'Iam User'. Below the profile picture, there is a section 'My Clubs:' with links to 'Club 1', 'Club 2', and 'Club 3'. Below that, there is a 'Sort by:' section with radio buttons for 'All Clubs', 'My Clubs', 'User Post's', 'Clubs and Users', and 'Your Feed'. At the bottom of the sidebar, there is a 'Start Direct Message' button.

Search bar:

Sirkel

Signed in as: lamUser

Live Feed - All Clubs

[Make a Post](#)

First Club

[Send](#) 7:48 PM

Second Club

[Send](#) 7:34 PM

Third Club

[Send](#) 7:01 PM

Fourth Club

Iam User

My Clubs:

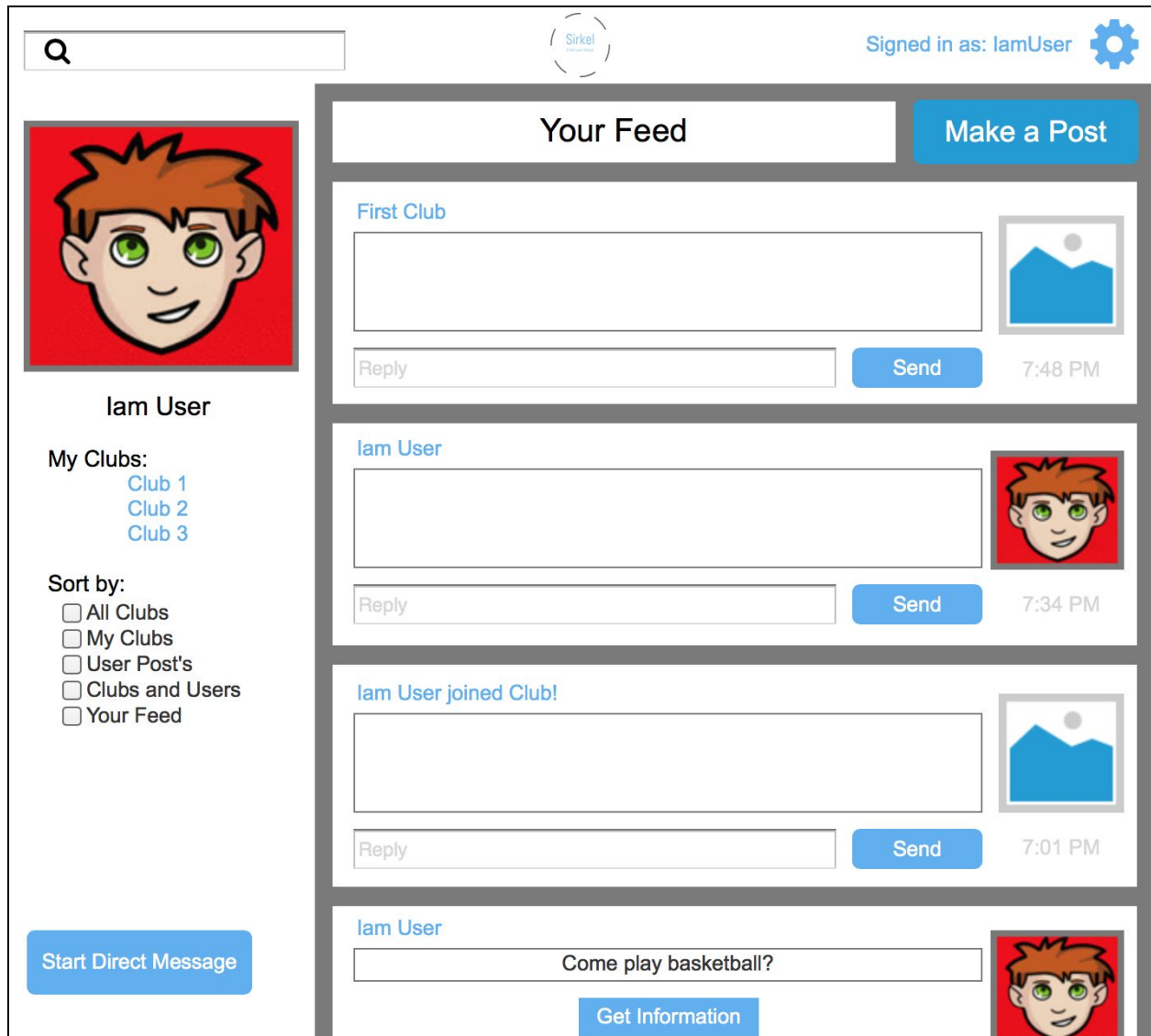
- [Club 1](#)
- [Club 2](#)
- [Club 3](#)

Sort by:

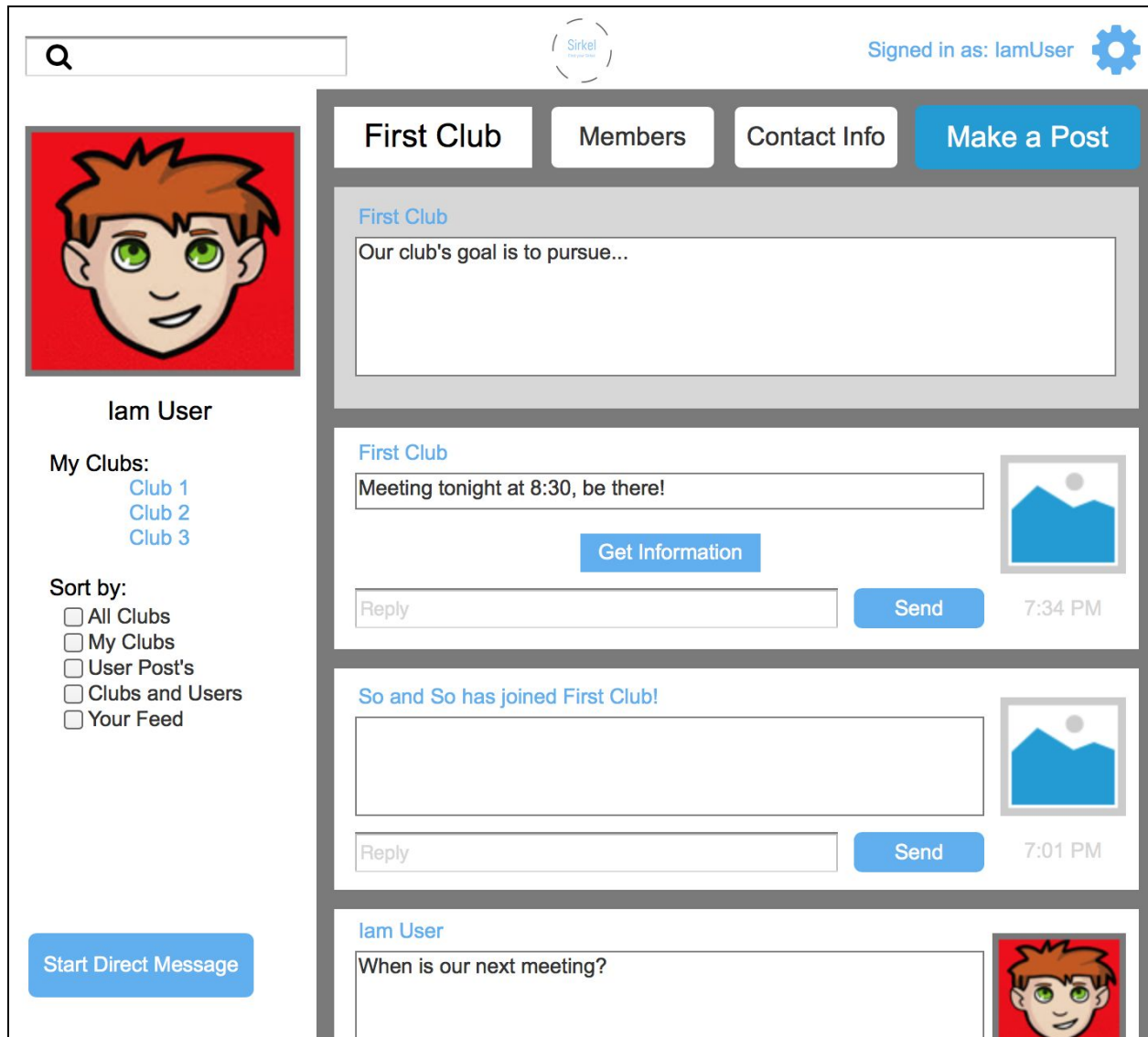
- ☐ All Clubs
- ☐ My Clubs
- ☐ User Post's
- ☐ Clubs and Users
- ☐ Your Feed

[Start Direct Message](#)

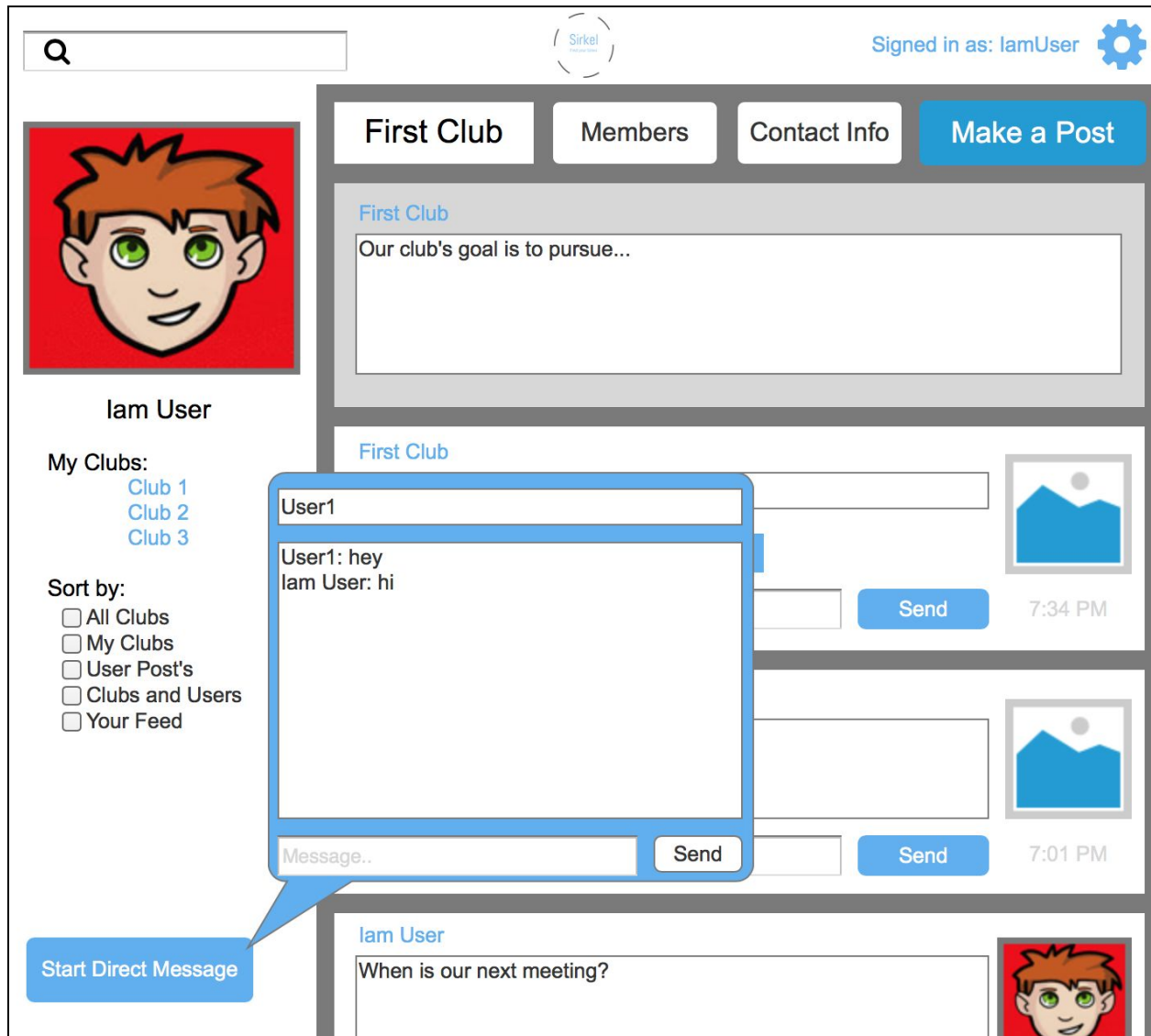
This page is the main feed that the user will see when they get into the website. They will have options to filter the feed to their liking, make a post to the main feed, and start a direct message with a club or person. Some other features include being able to go to settings, and a search bar for searching clubs or people.



This is your profile feed where you will be able to make a post for people to see, see the clubs information that you have joined, and etc. There will also be events that you can create where people can click on Get information to see time, place, etc.



This is what a club's page might look like when you go to it. Similar to what a profile might look like, with a few different features. For example, they will have some sort of mission statement at the top of the page so users know what the club does. There will also be buttons to look at the current members, as well as the contact information for the club.



This is a demonstration of a direct message when someone messages you or you are messaging them. This feature will also allow for you to create chat rooms with multiple people.