

# CivicConnect

## Community Service Social Network

CS307 - Design Document

**Group 11**

Aysu Saglam

Heidi Teng

Avishi Goyal

Jammy Wang

Roohee Urs

# Index

<b>Purpose</b>	<b>4</b>
<b>Functional Requirements</b>	<b>5</b>
User Registration and Authentication	5
Curation and Recommendations	5
Event Creation by Organizations	6
Event Attendance and Participation	6
Community Member Posts	7
Messaging	7
Notifications	7
<b>Non-Functional Requirements</b>	<b>8</b>
Performance and Scalability	8
Security and Data Privacy	8
Maintainability and Monitoring	8
<b>Design Outline</b>	<b>9</b>
High Level Overview	9
Interactions Between Components	10
UML Overview	12
Design Issues	14
Functional Issues	14
Non-Functional Issues	17
<b>Design Details</b>	<b>19</b>
Class Design Diagram	19

Description of Classes and Interactions Between Them:	20
1. Volunteer	20
2. Event Organizer	20
3. Event	20
4. Post	20
5. Comment	21
6. Message	21
7. Message Room	21
8. Notification	21
Sequence Diagrams:	22
1. Sequence of Events when user tries to log into system	22
2. Sequence of Events when posting a comment under a post	22
3. Sequence of Events when sending a message to another user	23
4. Sequence of Events when subscribers are notified that an event organizer has created a new event	23
5. Sequence of Events when making a post	24
<b>UI Mockups</b>	<b>25</b>
Sign Up and Login Pages	25
User Feed	26
User and Organization Profiles	27
User and Organization Posts	28
Messages	29

## Purpose

For those who are not already a part of service organizations/initiatives or options within their community, it can be difficult to seek out community service opportunities. One of the easiest internet mediums for people to navigate and gain information from are social media platforms. There are several notable platforms that expand beyond traditional social media content, such as those for career building and networking, finding novels and films, and even sharing running statistics: Why not create a platform centered around community service that allows for people to get community service recommendations based on their profile and allow users to post about events they have partaken in. The purpose of building this network is to encourage individuals to discover and engage with their communities, creating an environment of community service that is tailored to an individual's interest and fostering a culture of giving back that goes beyond the scope of traditional social media.

There are several existing platforms that accomplish different things through a similar recommendation-based or social media format. For example, users of Netflix can navigate through the app/site and be given recommendations and media for shows/movies based on their specific interests and previously watched. Another platform, LinkedIn, follows the more traditional social media format, but interestingly, not all posts in a user's "feed" come from their following, but rather the liked posts of those they are connected with and recommendations from the algorithm. When the broad functionality of these two examples is extrapolated, you get the intended functionality of CivicConnect -- a social media platform that recommends users community service listings based on their interests and previous attendance.

## *Functional Requirements:*

### **(a) User Registration and Authentication**

1. As a user, I would like to register and authenticate securely so that I can access my personalized profile and service recommendations.
2. As a user, I would like the option to reset my password if I forget it so that I can regain access to my account.
3. As a user, I would like to have different options to sign up, including signing up with a Google Account or a traditional username and password sign up.
4. As a user, I would like my account information populated according to how I last saved it.
5. As a user, I would like to have some type of two-factor authentication for my account to increase security.
6. As a user, I would like to link my email and phone number to my account to get notifications to both for anything in relation to the application.
7. As a user, I would like to link outside social media accounts to my CivicConnect account so that I can reach a larger audience with my posts.

### **(b) Curation and Recommendations**

1. As a Community Member, I would like to specify my availability (e.g., weekends, weekdays, specific times) and location so that I receive relevant service opportunities.
2. As a Community Member, I would like to specify my interests and hobbies so that I receive relevant service opportunities.
3. As a Community Member, I would like to receive push notifications when an opportunity that matches my profile has been posted.
4. As a Community Member, I would like to bookmark/save postings to look back on later.
5. As a Community Member, I would like to filter my feed based on transportation requirements (e.g., walking distance, public transport) to get from my location to the posted events.
6. As a Community Member, I would like to be able to search events by any parameters, including type of event, time, and date.
7. As a Community Member, I would like to receive notifications when events that I have bookmarked/saved are close to reaching maximum capacity.
8. As a Community Member, I would like to see the Organization's profile and read their bio, find their socials, and look at their events; As an Organization, I would like to view the users' profiles who have RSVP'd to my event.

9. As an Organization, I would like to add the following parameters to event postings: type (e.g., environmental, education, health), time, date, and location.
10. As an Organization, I would like to have a sign up where Community Members can sign up for our newsletters or promotional emails.

(c) Event Creation by Organizations

1. As an Organization, I would like to upload images and videos for my event posts so that I use visuals to describe the event.
2. As an Organization, I would like to attach captions to my posts regarding events to provide logistical details.
3. As an Organization, I would like to see how many users have bookmarked/saved the event I have posted so that I can quantify interest before the event happens.
4. (If time allows) As an Organization, I would like to schedule posts for future events and to plan them in advance.
5. As an Organization, I would like to edit features of event postings that have already been made.
6. As an Organization, I would like to be able to delete event postings.
7. As an Organization, I would like to create a post and be able to make it a daily, weekly, biweekly, or monthly event.
8. As an Organization, I would like to notify Community Members previously interested in my events or Members with similar interests about a new posting from us.

(d) Event Attendance and Participation

1. As an Organization, I would like to see the number of users who have RSVP'd to my event so that I can prepare for the correct number of participants.
2. As an Organization, I would like to allow users to share their event participation on other social media platforms outside of CivicConnect.
3. As an Organization, I would like to put a cancellation constraint on the event where Community Members can only cancel until 24 hours before the event.
4. (If time allows) As a Community Member, I would like to leave feedback/rate the event after attending so that others can read about my experience.
5. As a Community Member, I would like to be able to RSVP for an event or cancel an existing RSVP for an event.
6. As a Community Member, I would like to see the events for which I have RSVP'd.
7. As a Community Member, I would like to be notified if there have been any changes made regarding the events for which I have RSVP'd.
8. As a Community Member, I would like to have an in-app calendar where I can see what dates and times I have booked for.

(e) Community Member Posts

1. As a Community Member, I would like to create posts with multiple images/videos about events I have attended so that I can share my experiences.
2. As a Community Member, I would like to be able to specify the location of the post/update
3. As a Community Member, I would like to attach captions to my posts regarding events to provide more detailed accounts of my experience.
4. As a Community Member, I would like to create a posting on a linked social media account regarding my event attendance so I can spread awareness.
5. As a Community Member, I would like to be able to comment on other peoples' posts as well as my own along with replying to other comments on posts.
6. As a Community Member, I would like to be able to react to peoples' posts.
7. As a Community Member, I would like to like peoples' comments on their posts.

(f) Messaging

1. As a user, I would like to privately message other users with text, links, or photos and access my message history with other users. As a Community Member, I would like to privately in-app message the Organization with any questions or concerns I might have regarding the events.
2. As a user, I would like to be notified when I receive a new message so that I can reply and continue the conversation.
3. As a user, I would like to search for others by name so that I can start a conversation with them.
4. (If time allows) As a user, I would like to be able to react to messages with emojis by double-tapping on the message.
5. (If time allows) As a user, I would like to be able to see if another member is typing.
6. As a user, I would like to be able to create group chats with several users that are attending the same event to coordinate plans.

(g) Notifications

1. As a user, I would like to be notified for upcoming events and community interactions with my posts and be able to toggle these notifications..
2. As a user, I would like to set the notification time for the event in advance such as 'notify before one day' or 'one week'.
3. As a user, I would like to be able to see when people are notified that I am interested in a post.

## *Non-Functional Requirements:*

### 1) Performance and Scalability

1. As a Developer, the platform should be able to handle 10,000 simultaneous user requests without performance degradation.
2. As a Developer, I want the platform to utilize efficient caching mechanisms for frequently accessed data so that response times are minimized and the server load is reduced. I would like the platform to respond to actions (loading a page, searching for opportunities, and viewing event details) within 1000 milliseconds so that users can quickly find and engage with relevant community service opportunities without frustration or delays.

### 2) Security and Data Privacy

1. As a Development Team, our platform will use HTTPS with TLS 1.3 for secure connection.
2. As a Development Team, we will have regular security and vulnerability assessments that will be conducted to ensure ongoing protection against emerging threats.
3. As a Development Team, We would like our platform to utilize authentication processes to support multi-factor authentication (MFA) so that the platform can add additional layers of security to protect against unauthorized access.
4. As a user, I would like to know that my password is securely saved in a database using password encryption, specifically with RSA 256 encryption.
5. As a user, I would like to have constraints for my password to make it more secure such as the password must include a number and a capital letter.

### 3) Maintainability and Monitoring

1. As a Development Team, our platform will have automated deployment pipelines so that new features, updates, and security patches can be rolled out quickly with minimal downtime.
2. As a Development Team, our platform will support version control and rollback features, so in the case of any deployment issues or errors, the system can quickly be reverted to a stable state.
3. As a Development Team, regular maintenance windows will be communicated in advance to all users so they are aware of scheduled downtime. This will minimize their disruptions to their access and use of the platform.
4. As a Developer, the platform will have well-documented code so that future updates, bug fixes, and feature additions can be implemented efficiently without impacting the existing functionality.



## Design Outline

### *High-Level Overview:*

The CivicConnect platform will allow organizations to create posts regarding community service events. These posts will be pushed to community members based on parameters such as their location, interests, past event attendance, and more. The platform will be built using a client-server model in which the frontend (client) interacts with the backend (server) via API calls to perform tasks such as user registration, event management, and post recommendations. The backend will be responsible for the storage and organization of data as well as the recommendation logic, while the frontend will act as an easy-to-use interface for users.

### Components of the System:

1. **Frontend (Client)** - The frontend will handle all user-facing interactions and act as the interface for the software. It will provide necessary forms for user registration, authentication, profile creation, posting, activity feeds, and chatting. It sends requests to the backend, and will render responses accordingly.
2. **Backend (Server)/API Endpoints** - The backend will be responsible for providing APIs for user registration, user authentication, general data storage, and the event recommendations. It will manage interactions with the database and send responses to the frontend.
3. **Database** - The database will be responsible for storing data regarding user profiles, event listings, user posts, messages, and any additional information. The information will be stored in a relational schema.
4. **Machine Learning Recommendation System** - The ML recommendation system will be responsible for generating personalized community service recommendations based on a community member's profile parameters and past event attendance. The system will process such user data to generate accurate and relevant suggestions.
5. **3rd Party Authentication/Security** - The authentication system will be responsible for ensuring user authentication and authorization. It will provide the necessary support for multiple log-in methods (multi-factor authorization) and secure storage of log-in credentials.
6. **Notification System** - The notification system will be responsible for sending users notifications about any relevant topic such as upcoming events, updates, and messages they have received.

## *Interactions Between Components:*

### User Registration and Authentication

1. The user will interact with the **frontend** to register via a form interface.
2. The **frontend** will send the registration data to the **backend** API, which will interact with the **authentication system** to authenticate the user.
3. The **authentication system** will send a secure token to the backend, which will validate this token and store the user's profile information in the **database**.
4. Once a user is registered, when they attempt to log in, the **frontend** will send their credentials to the **backend**, where they will be **authenticated** in a similar process.

### Community Service Recommendations

1. The **frontend** sends a request to the **backend** for community service opportunity recommendations based on a user's profile.
2. The **backend** will then interact with the **machine learning model** to generate recommendations based on the user's profile information and past attendance.
3. Then, the **backend** will return this information to the **frontend** which will be responsible for properly displaying this information via the activity feed.

### Event Creation and Participation

1. Organizations will use the **frontend** to create posts about community service events they are hosting.
2. The **frontend** will then send the event details and related date to the **backend**, which will store the data in the **database**.
3. When prospective attendees interact with the **frontend** to RSVP to an event/bookmark it, these interactions will be sent to the **backend** so that the community member's RSVP and bookmark statuses can be updated.
4. Notifications for a user's bookmarked events once they have reached near capacity/are nearing in date, will be sent to the **frontend** via the **notification system**.

### Messaging

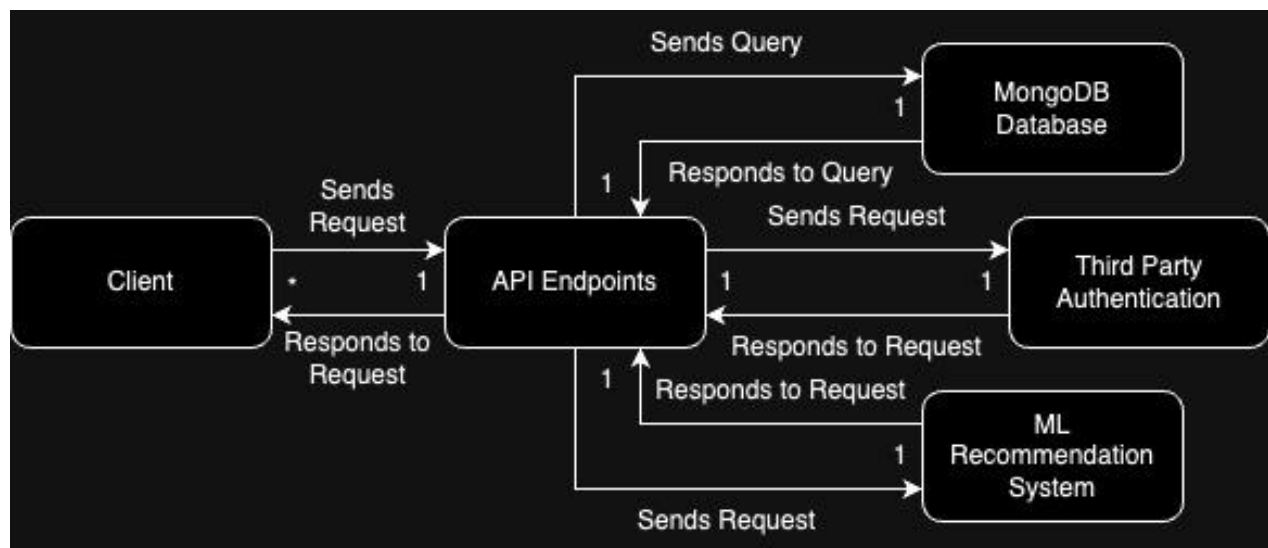
1. Through the **frontend** users can send messages that will be ultimately sent to the **backend API**.
2. The **backend** will store the message in the **database** and retrieve it when a conversation is ongoing.
3. Notifications sent regarding messages will be handled via the **notification system** and sent to the **frontend**.

### Event Feed and Post Interactions

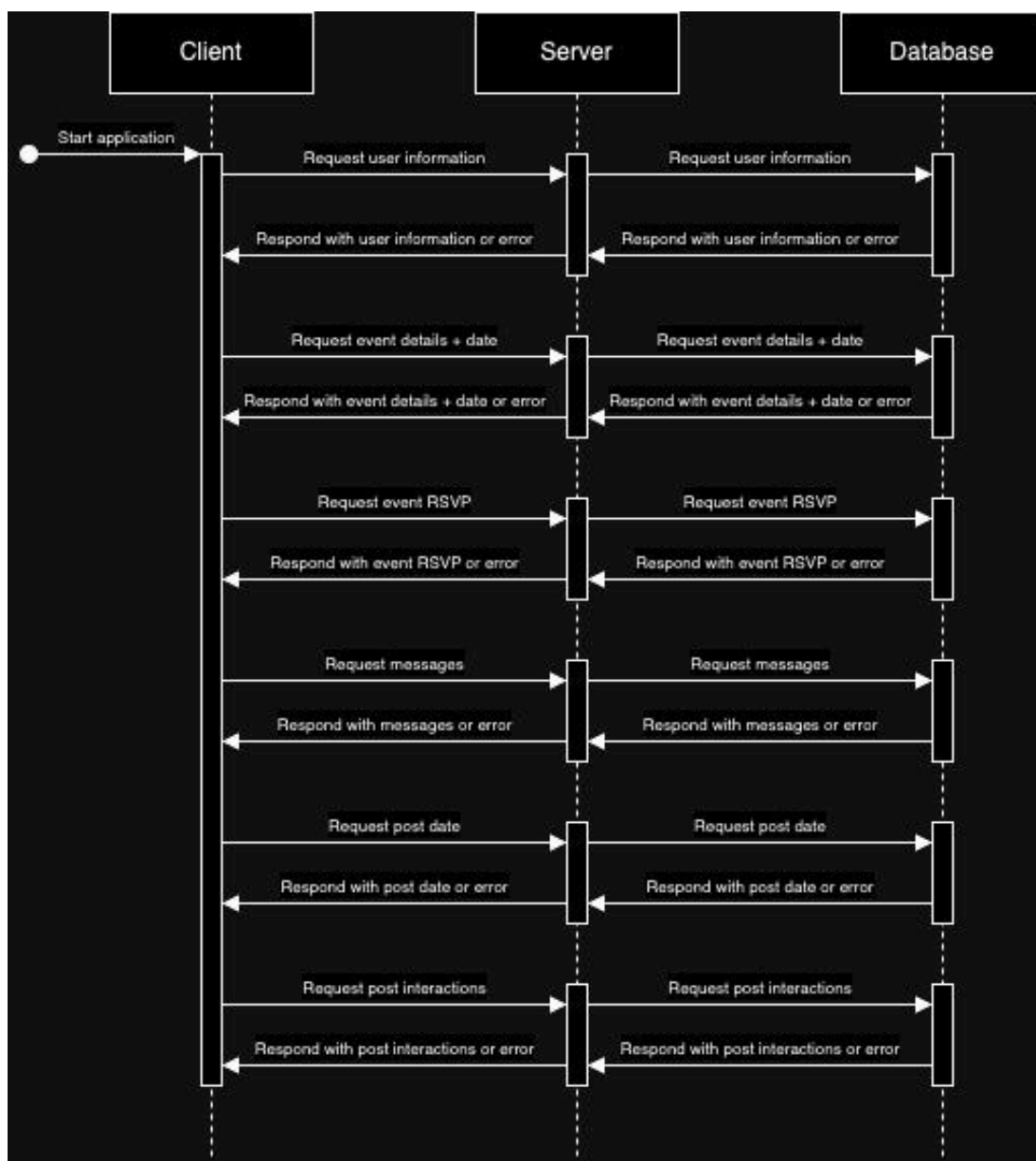
1. The **frontend** will request feed (post) data from the **backend**, which will retrieve this information from the **database**, and send it back to be displayed on the **frontend**.
2. Users can then interact with posts (via likes, comments, RSVPs, and bookmarks), which will be processed by the **backend** and updated in the **database**.
3. When a post has been interacted with in the frontend, a **notification** will be sent to the poster.

### *UML Overview:*

This diagram gives a high-level overview of the components of CivicConnect. There will be many clients that can send requests to our singular backend/API Endpoints, which can then respond to client requests. Our backend will have three connections. It will interact with our singular database to send queries and receive responses. It will also send requests and receive responses from our third party authentication system, which will authenticate users. Lastly, our backend will send requests to our ML Recommendation System, which will respond with recommendations sent to the backend.



This sequence diagram describes interactions between the client, server, and database. After a user starts the application, they will log in, which will cause the client to send a request to the server. The server will handle the request and send a query to the database, which will then respond with the data from the database. The server will then send that to the client and log the user in. Similar to that, the client can send several requests to the server such as requesting event details or event RSVPs, requesting messages and message history, requesting post dates, or requesting post interactions, including bookmarks, likes, or comments. To respond to such requests, the server will send queries to the database and receive the data from the database. The server will then respond to the client with the information requested.



## Design Issues

CivicConnect has been designed to focus on user accessibility, flexibility, and security. The following are some design issues that were made during the course of planning our platform, which include the creation of accounts, messaging between volunteers and event organizers, RSVP systems for event planning, and the frameworks chosen for efficient and secure development.

### *Functional Issues:*

1. What credentials and information will the app require during the creation of a new account?
  - a. Option 1 - Username, Password
  - b. Option 2 - Email, Username, Password
  - c. Option 3 - Email, Username, Password, Phone Number
  - d. Option 4 - Email/Google account, Username, Password
  - e. Option 5 – Name, Phone Number, Username, Password, Email/Google Account, City, Age, Reputation, Interests

Choice: Option 5

Justification: We decided to allow users to enter their email or sign in through Google and create a username and password. Using their email/Google account will allow the users to recover their account if they forget their other login credentials. Each username is unique; therefore, there can be no overlapping information, as everyone has their own identity within CivicConnect. Other metadata information is collected, as seen later in our class diagram. Age, Reputation, City, and Interest information is gathered and stored in the database, which can prompt events that fit their interests once an account is created.

2. What other accounts/credentials can users link with their existing CivicConnect accounts?
  - a. Option 1 - Facebook
  - b. Option 2 - LinkedIn, Facebook
  - c. Option 3 – Facebook, LinkedIn, Twitter, Instagram

Choice: Option 3

Justification: While an initial account already contains a phone number and email, we encourage our volunteers to link their existing CivicConnect accounts with their social media accounts. There are a variety of social media platforms that exist, and this gives users the chance to connect with others on other platforms to create and maintain relationships. This may also encourage other users on other platforms that do not utilize CivicConnect to create an account and become involved in the community service community.

3. What type of communication is allowed within Community Members and between Organizations and Community Members?
  - a. Option 1 - Community Members can only send private messages to Organizations.
  - b. Option 2 - Community Members can only send private messages to other Community Members.
  - c. Option 3 - Community Members can send either private or public messages to both Organizations and other Community Members, and vice versa.
  - d. Option 4 - Community Members can send public messages to Organizations.

Choice: Option 3

Justification: Allowing both public and private messages allow for flexibility, as private messages can be used for one-on-one conversations, whereas public messages can be used as announcements, share of information, and open discussions for an entire group of members in a message group. Public messages also build a sense of community, allowing one to communicate with one another in a community setting. Both private and public messages make it easier for community members to connect with organizations and other members.

4. How can the RSVP concept be incorporated into CivicConnect?
  - a. Option 1 - Community members can RSVP to events via an in-app RSVP feature.
  - b. Option 2 - Community members can RSVP to events by participating in a public poll, liking an announcement message in a message group, or filling out a survey created by the organization.
  - c. Option 3 - Community members can manually RSVP by sending a private message to the Organization conducting the event.

Choice: Option 2

Justification: A public poll, similar to an announcement message, or a survey allows organizations to track RSVPs transparently, making it easier for community members to also see how many people have expressed interest in an event, which can persuade more members to participate if they see a friend attending, or if there is a high number of people participating. In addition, organizations can customize polls to gather additional information about attendees, such as dietary restrictions or transportation needs, if necessary. This option offers a balance between the convenience of an in-app RSVP feature and the transparency of a public message.

5. What information associated with an event is visible on the event posting?
- a. Option 1 - Name, Description, Location, Date & Time,
  - b. Option 2 - Name, Location, Date & Time
  - c. Option 3 - Name, Location, Date & Time
  - d. Option 4 - Name, Location, Date & Time, Purpose, Registration Link, Description

Choice: Option 4

Justification: Displaying the name, location, date and time, purpose, registration link, and description of the event gives users maximum information about an event. Users can view all information relevant to an event before deciding whether or not they will attend. Making important information about the event easily accessible also helps Organizations by reducing the potential number of questions from volunteers about events.



### *Non-Functional Issues:*

1. What platform is the app being developed for?
  - a. Option 1 - iOS
  - b. Option 2 - Android
  - c. Option 3 - Web App

Choice: Option 3

Justification: We will be developing a web app because it is more versatile. Users can easily access the platform from any device with a web browser, including computers, smartphones, and tablets. This eliminates the need for users to download and install specific apps for different operating systems.

2. Which framework should be used?
  - a. Option 1 - React
  - b. Option 2 - Angular
  - c. Option 3 - HTML

Choice: Option 1

Justification: We chose React because all members of the team are familiar with React and have used it extensively in the past. React is also component-based and allows for efficient web development, which can improve code organization and maintainability. In addition, as the platform grows, it can be easily modified, and certain parts of our frontend can be reused, making it easier to scale and maintain.

3. Which database will the application use?
  - a. Option 1 - MongoDB
  - b. Option 2 - MySQL
  - c. Option 3 - Oracle Database

Choice: Option 1

Justification: We chose to use MongoDB as it provides more flexibility than an SQL database. Additionally, MongoDB's document-oriented structure allows for flexible data modeling, making it easier to adapt to changing requirements and accommodate diverse data types. It also can scale horizontally, meaning that it can handle large data requests and high traffic load.

4. Are we going to be using HTTP or HTTPS for our web app?
  - a. Option 1 - HTTP
  - b. Option 2 - HTTPS

Choice: Option 2

Justification: We are going to utilize HTTPS, as it protects user data from being compromised during transmission, and users are also more likely to trust these websites, which indicates a commitment to security. In addition, various search engines give preference to websites that begin with HTTPS, which can improve our site's visibility.

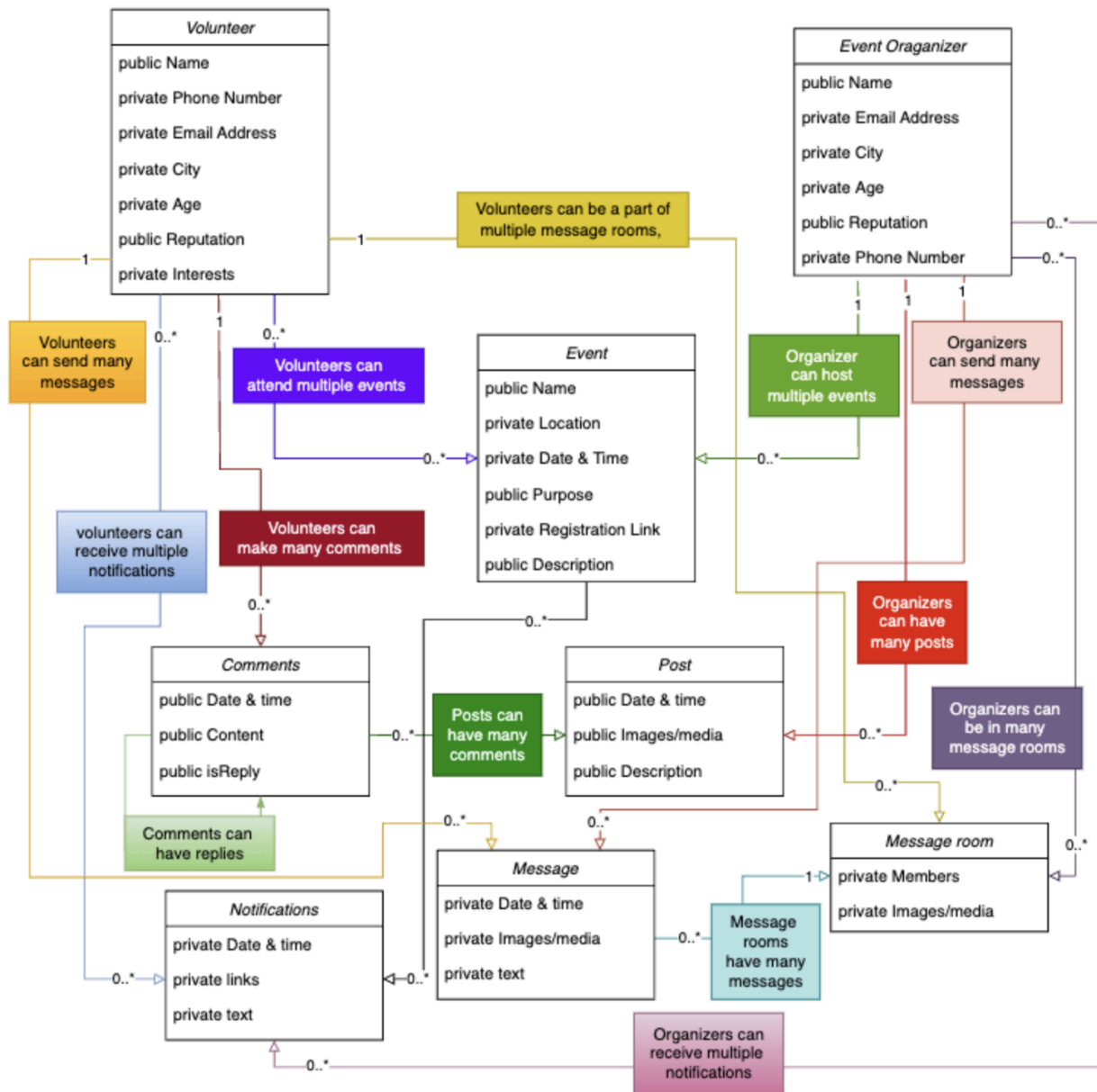
5. What kind of recommender system will we use to curate user feeds?
  - a. Option 1 - Rule-based recommender system using SQL
  - b. Option 2 - Time-based recommendations (most recent)
  - c. Option 3 - Recommender system that uses machine learning

Choice: Option 3

Justification: Using a recommendation system that uses machine learning to curate user feeds helps in displaying posts that are most relevant to the user. This ensures that a user's feed is based on their preferences and past event attendance, which overall contributes to a personalized experience for each user.

## Design Details

### *Class Design:*



### *Description of Classes and Interactions Between Them:*

There are eight classes: Volunteer, Event Organizer, Event, Post, Message Room, Message, Notification, and Comments. Our classes are designed based on the objects within our application, and each class has a various number of attributes that represent the characteristics associated with each class.

### **1. Volunteer Class**

- Attributes: name, phone number, email address, city, age, reputation, and interests
- A Volunteer object is created when someone signs up for our application, and chooses the “Volunteer” role. During the signup process, personal details are provided by the user, and the account will then be created and stored in the database.
- Volunteers can send and receive many messages, attend multiple events, make many comments, receive multiple notifications, and be part of many messaging rooms.

### **2. Event Organizer Class:**

- Attributes: phone number, email address, city, age, reputation, and name
- Event Organizer class is created when a user signs up for the platform and selects the “Event Organizer” role. This role is selected by representatives or individuals who wish to host and manage events, and similar to the volunteer class, personal details are provided and stored in the database.
- Each event organizer could have the ability of hosting multiple events, sending and receiving multiple messages to their volunteers, making many posts, receiving multiple notifications, and can be in many messaging rooms to promote event(s).

### **3. Event Class:**

- Attributes: name, location, date & time, purpose, registration link, and description
- An Event Object is created when an Event Organizer creates an event in the system. The event’s details are stored in the database, and made available for volunteers to view and sign up.
- Each Event can have multiple volunteers attending, and volunteers can make many Comments on the event page, and event organizers can view a list of attending volunteers, and send messages or updates to them.

### **4. Post Class:**

- Attributes: date & time, images/media, and descriptions
- A Post Object is created when either an Event Organizer or Volunteer makes a post on the platform, typically related to events, service opportunities, or general community information.
- Each post can have various comments from users, and posts can include images and other media to make them more engaging.

### **5. Comment Class:**

- Attributes: date and time, content, and isReply.
- A Comment Object is created when a user (volunteer or organizer) leaves a comment on a post of an event.
- Each comment can have many subcomments (i.e, replies to comments), and similar to Instagram, comments can have varying numbers of likes.

#### **6. Message Class:**

- Attributes: date & time, images/media, and text
- A Message Object is created when a user (volunteer or event organizer) sends a message in the message room. Each message can include text and media attachments.
- Messages can be sent by Volunteers and Event Organizers, and they belong inside specific message rooms.

#### **7. Message Room Class:**

- Attributes: members and images/media
- A Message Room Object is created when an Event Organizer or a Volunteer creates a Message Room, similar to a direct message or new chat room, and can have multiple volunteers and event organizers as members.
- Message rooms can contain multiple messages, and have both Volunteers and Event Organizers as members.

#### **8. Notification Class:**

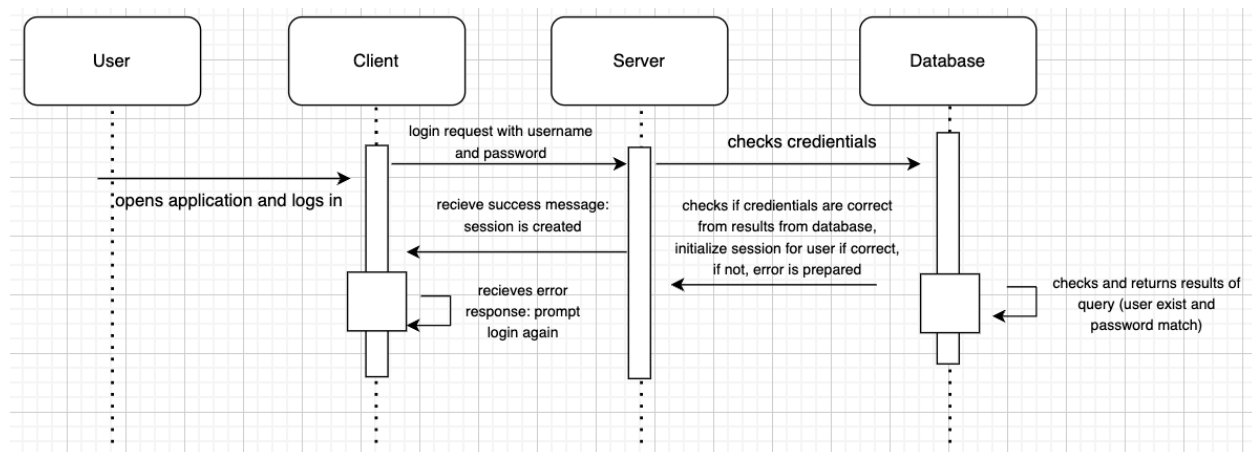
- Attributes: date & time, links, and text
- A notification object is created when certain actions trigger an alert, such as a volunteer signing up for an event, a new message being received, or a reminder for an upcoming event. Notifications are created automatically by the system based on user activity or schedule.
- Notifications can be made by volunteer and event organizations, where their respective audience can receive notifications about reminders, future event information, etc.

## Sequence Diagrams:

The following sequence diagrams illustrate the critical processes in our application, including user login, comments on posts, and sending messages between users. The following diagrams represent the information in a client-server architecture. In each sequence, the client will send a request to the server when a user initiates an action through the frontend user interface. The server then handles the request and interacts with the database to retrieve, store, or verify data. When the server finishes processing the necessary information and the database finishes working with the metadata and the current database, it will send a response back to the client. This response will update the user interface and display the appropriate information or actions.

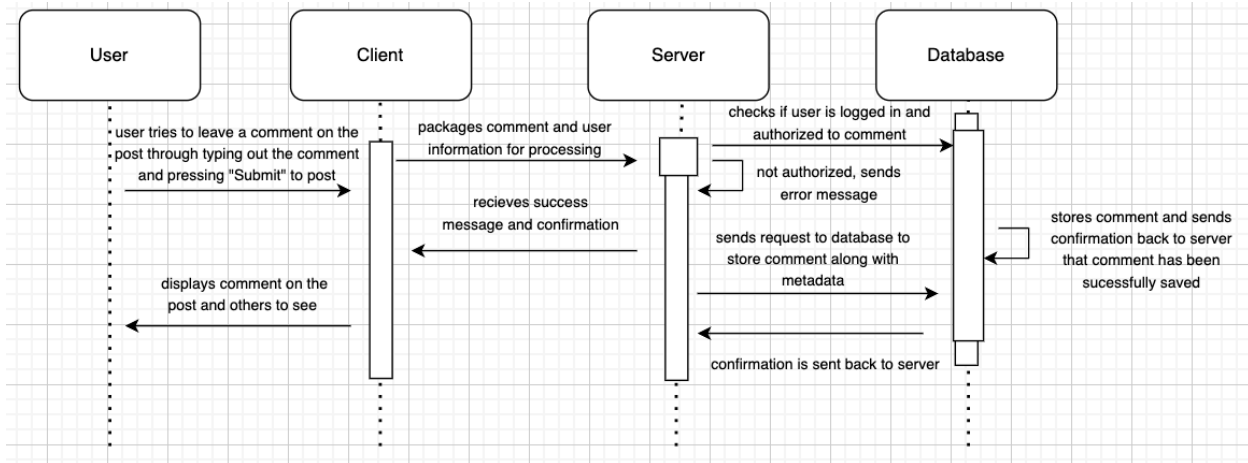
### 1. Sequence of Events when user tries to log into system

This sequence diagram illustrates the login process in a system. The user initiates the process by entering their credentials, which the client sends to the server. The server then queries the database to verify the credentials and either creates a session for successful login or returns an error for incorrect details, prompting the user to retry.



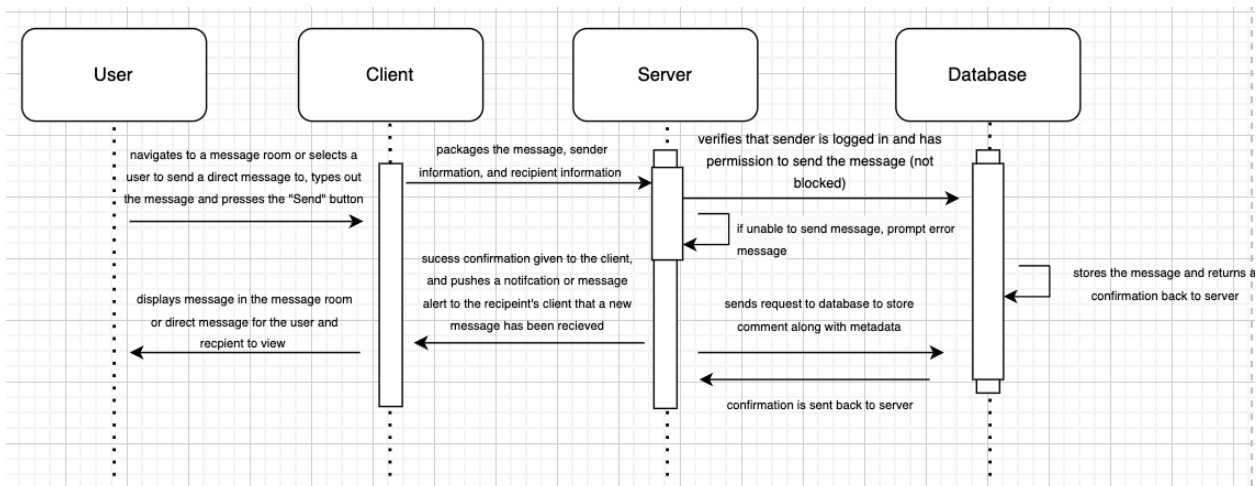
### 2. Sequence of Events when posting a comment under a post

This sequence diagram shows the process of a user posting a comment. The user writes a comment and submits it through the client, which sends the user and comment data to the server. The server checks if the user is logged in and authorized to comment, then sends the comment to the database for storage. If successful, a confirmation is returned to the client, and the comment is then displayed for others to see. If the user is unauthorized, an error error is returned.



### 3. Sequence of Events when sending a message to another user

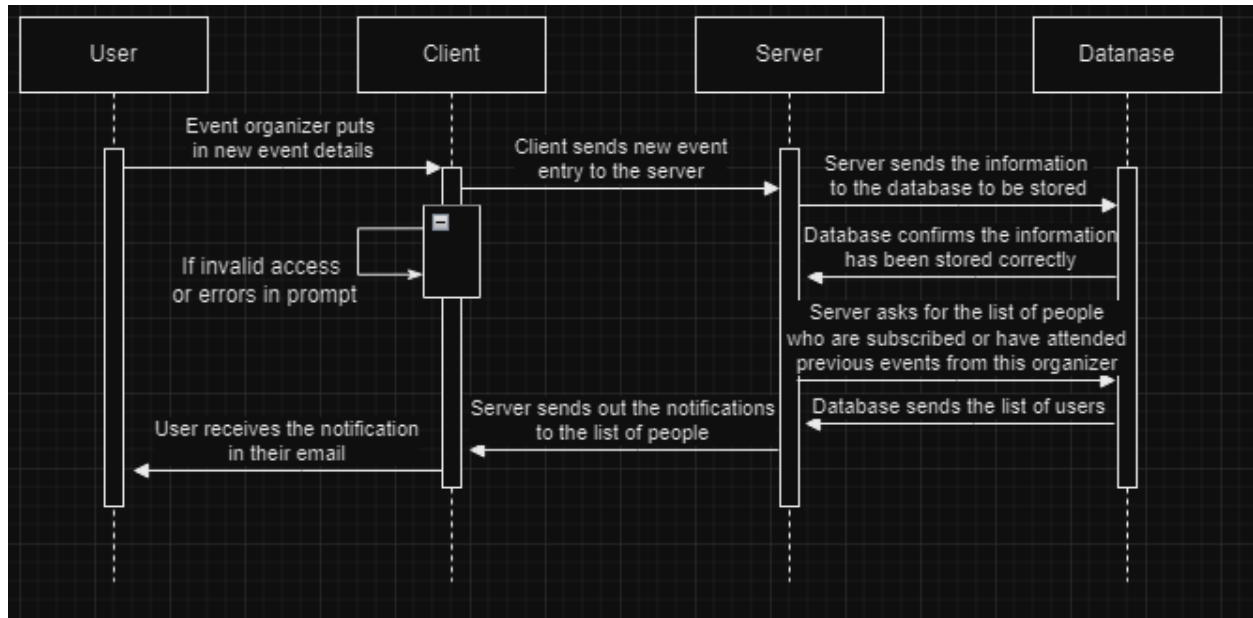
This diagram outlines the process of sending a direct message. The user navigates to a message room or selects a recipient, types a message, and presses "Send." The client will package the message, sender, and recipient information, and send it to the server. The server will verify if the sender is logged in and allowed to send messages, then requests the database to store the message. Upon success, the server sends confirmation to the client, which notifies the recipient and displays the message in the room. If the message can't be sent, an error is returned.



### 4. Sequence of Events when subscribers are notified that an event organizer has created a new event

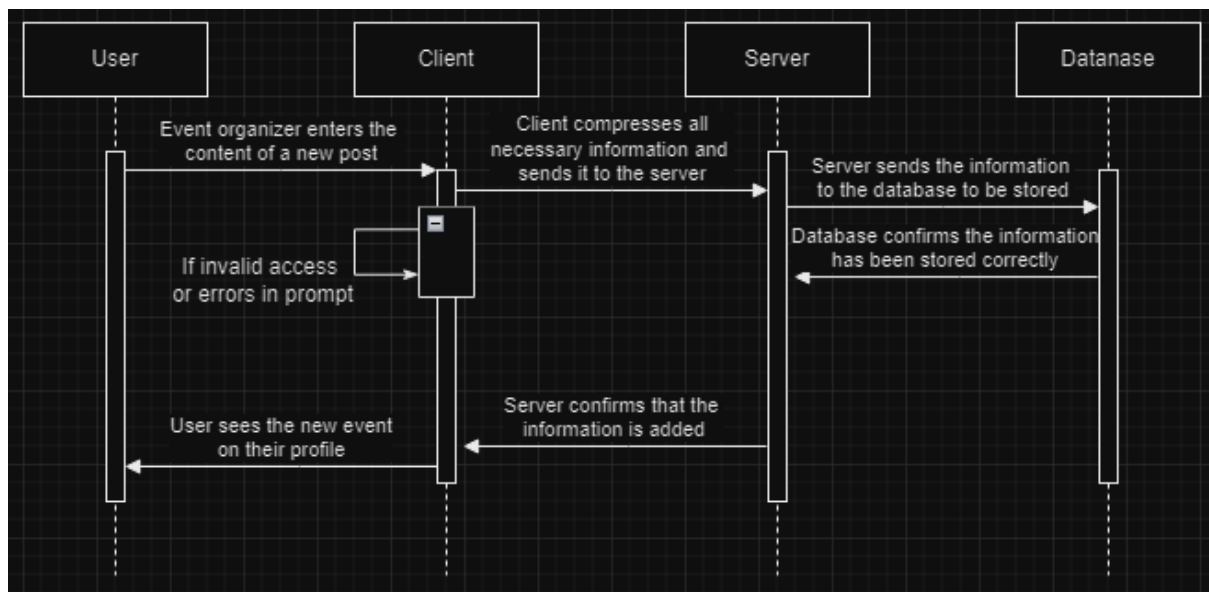
This sequence illustrates the process of an event organizer creating a new event. The organizer submits event details through the client, which sends the event information to the server. The server stores the event in the database and confirms that the data has been correctly saved. The server then requests a list of users who are subscribed or have attended previous events by the organizer. The database will then return the list, and the

server sends a notification to those users, which is received through email. If the submission encounters errors or invalid access, the system prompts the organizer with an error message.



##### 5. Sequence of events when making a post

This sequence diagram shows the process of an event organizer creating a new post. The organizer enters the content of the post through the client, which compresses and sends the information to the server. The server stores the post in the database and receives confirmation that the information was successfully saved. The server then confirms that the post was added, and the user can see the new event on their profile. If there is an error or invalid access during the process, an error prompt is displayed to the organizer.

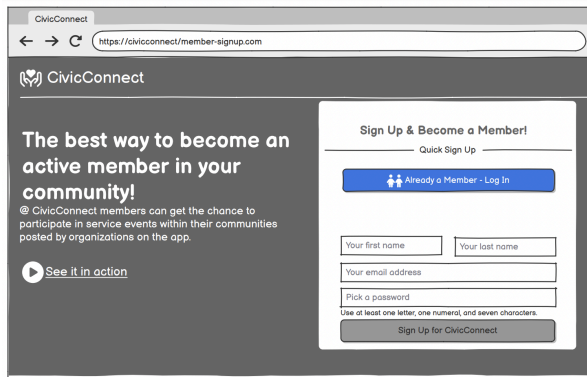




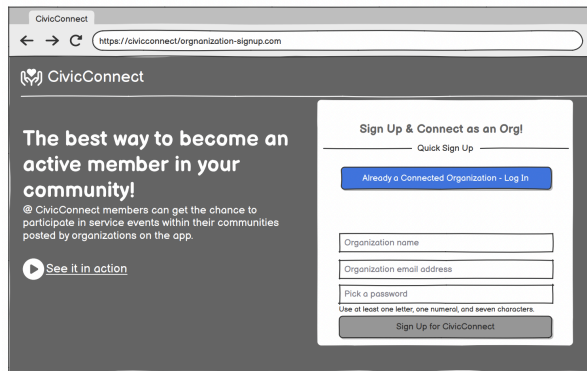
## UI Mockups:

### Sign Up & Login Pages

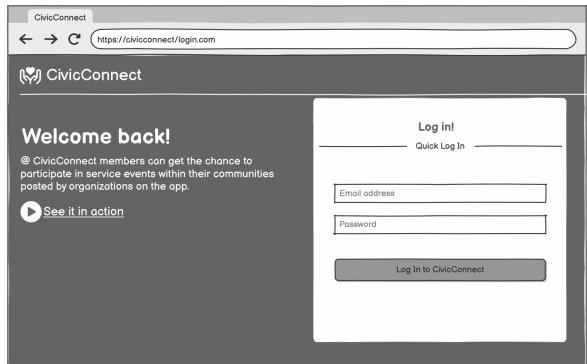
To account for the distinction between a community member and an organization as users for CivicConnect, we opted to create two different 'sign up' pages for the differing parties. The community member sign up will prompt a member to log in if they have an existing account, or sign up by filling in the required fields. The organization sign up will do the same, with slightly differing required fields. If either party has an existing account, they are prompted to sign in via a button at the top of the page.



This mockup shows the 'Sign Up & Become a Member!' page for CivicConnect. The browser address bar shows 'https://civicconnect/member-signup.com'. The page features the CivicConnect logo and a heading 'The best way to become an active member in your community!'. Below this is a sub-heading 'Sign Up & Become a Member!' and a 'Quick Sign Up' section with a blue button labeled 'Already a Member - Log In'. The main sign-up form includes fields for 'Your first name', 'Your last name', 'Your email address', and 'Pick a password' (with a note: 'Use at least one letter, one numeral, and seven characters'). A 'Sign Up for CivicConnect' button is at the bottom.



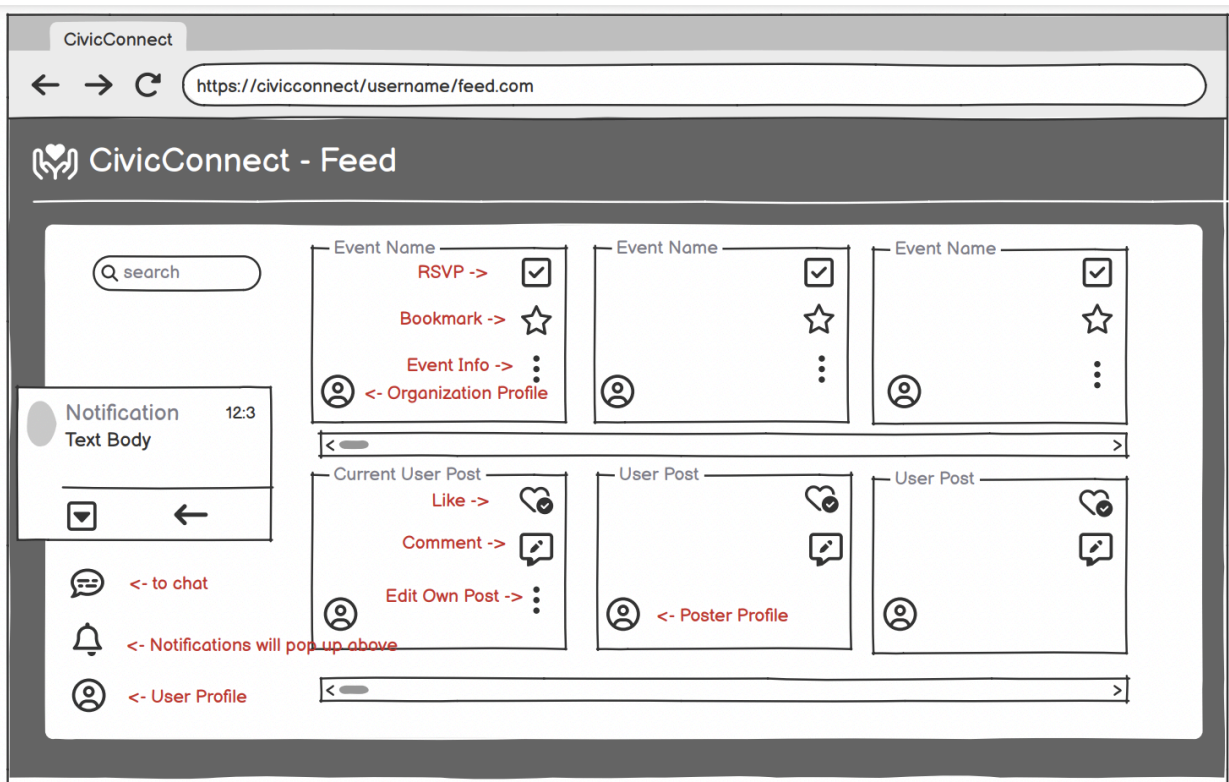
This mockup shows the 'Sign Up & Connect as an Org!' page for CivicConnect. The browser address bar shows 'https://civicconnect/organization-signup.com'. The page features the CivicConnect logo and a heading 'The best way to become an active member in your community!'. Below this is a sub-heading 'Sign Up & Connect as an Org!' and a 'Quick Sign Up' section with a blue button labeled 'Already a Connected Organization - Log In'. The main sign-up form includes fields for 'Organization name', 'Organization email address', and 'Pick a password' (with a note: 'Use at least one letter, one numeral, and seven characters'). A 'Sign Up for CivicConnect' button is at the bottom.



This mockup shows the 'Log in!' page for CivicConnect. The browser address bar shows 'https://civicconnect/login.com'. The page features the CivicConnect logo and a heading 'Welcome back!'. Below this is a sub-heading 'Log in!' and a 'Quick Log In' section. The main login form includes fields for 'Email address' and 'Password', followed by a 'Log In to CivicConnect' button.

## User Feed

When a user (community member) signs into CivicConnect, they will be initially routed to the 'user feed' of the platform. On the left-hand side of the feed, a search bar is present that allows users to search for posts based on certain tags and potential included text. By selecting icons on the left hand side, a user can either navigate to (1) their messaging chats (2) their notifications or (3) their own profile. In terms of the posts on the page, there are two dynamic slides that show differing posts, the first shows events being listed by organizations and the second showcases posts made by users. For event posts, users can either RSVP or bookmark them. For user posts, users can like, comment, and edit a post (if it is their own).



## User and Organization Profiles

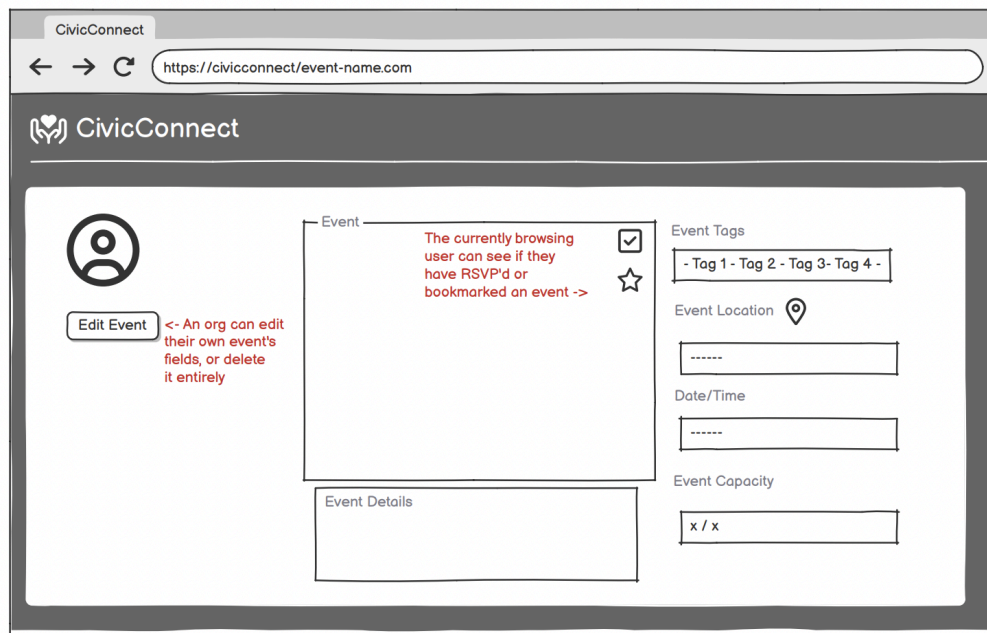
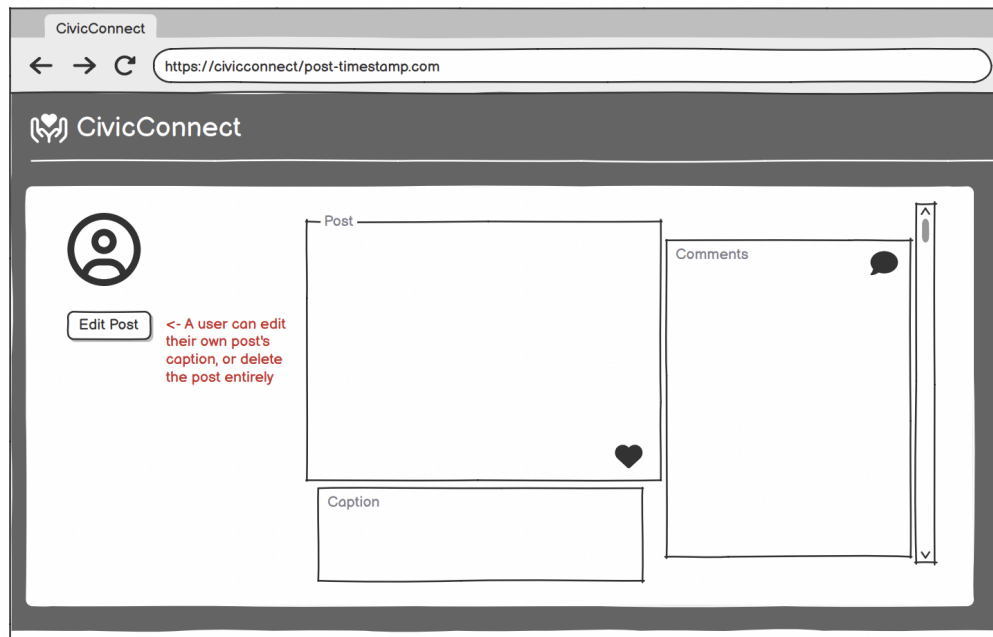
Users and Organizations can each view their own profiles on the CivicConnect platform. When an organization first logs into their platform, they will be routed to their profile. On one's profile, you can view their profile information and the posts they have made. If on your own profile, you have the right to edit components of your profile.

The screenshot shows a web browser window with the address bar displaying `https://civicconnect/username.com`. The page header includes the CivicConnect logo and the text "Some text". The main content area features a user profile for "Username - Name". On the left, there is a profile picture placeholder, a "+ Create a post" button, and a list of interests: "Interest 1 - Interest 2 - Interest 3 - Interest 4 - Interest 5 - Interest 6 - Interest 7 - Interest 8 - Interest 9 -". Below the interests, there is a section for "Upcoming and Bookmarked Events" with a list: "Event 1 - Event 2 - Bookmark 1 -". An "Attendance Rating" is shown with a star icon, and a "Location" field is present. To the right of the profile information, there is an "Edit Profile" button and a red note: "<- A user can edit there own profile fields". Further right, there are three "User Post" placeholders, each with a vertical scrollbar on its right side.

The screenshot shows a web browser window with the address bar displaying `https://civicconnect/organization.com`. The page header includes the CivicConnect logo. The main content area features an organization profile for "Organization Name". On the left, there is a profile picture placeholder, a "+ Create a post" button, and a list of tags: "Interest 1 - Interest 2 - Interest 3 - Interest 4 - Interest 5 - Interest 6 - Interest 7 - Interest 8 - Interest 9 -". Below the tags, there is a "Location" field. At the bottom left, there is a "General Organization Info" section. To the right of the profile information, there is an "Edit Profile" button and a red note: "<- An org can edit there own profile fields". Further right, there are three "Event" placeholders, each with a vertical scrollbar on its right side.

## User and Organization Posts

Both user and organization posts can be navigated to in a detailed view. In this view you can see the media of the post, the associated text/caption, comments (if applicable), and any other data associated with the post that is front-facing. If it is a post you have uploaded, you have the rights to edit the post and/or delete it.



## Messages

Through the feed, a user can navigate to their messages. They will be able to message text/images/video to other users and can navigate to their differing messages on the right hand side of the screen. They can either manually scroll and search through their messages, or, they can search for messages via text search.

