

Team 14 - CS 307 Design Document

Connor Brown, Hunter David, Steve Eisner, Baxter Govan, Ryan Klinedinst, Victor Pan

**Table of Contents**

**Design Outline 3**

Components 3

High-Level Overview 3

Component Interaction Diagram 3

UML Diagram 4

**Design Issues 5**

Functional Requirements 5

Non-Functional Requirements 6

**Design Details 8**

Class Design 8

Descriptions of Classes and Interactions 9

Sequence Diagram 11

UI Mockup 15

**Design Outline**

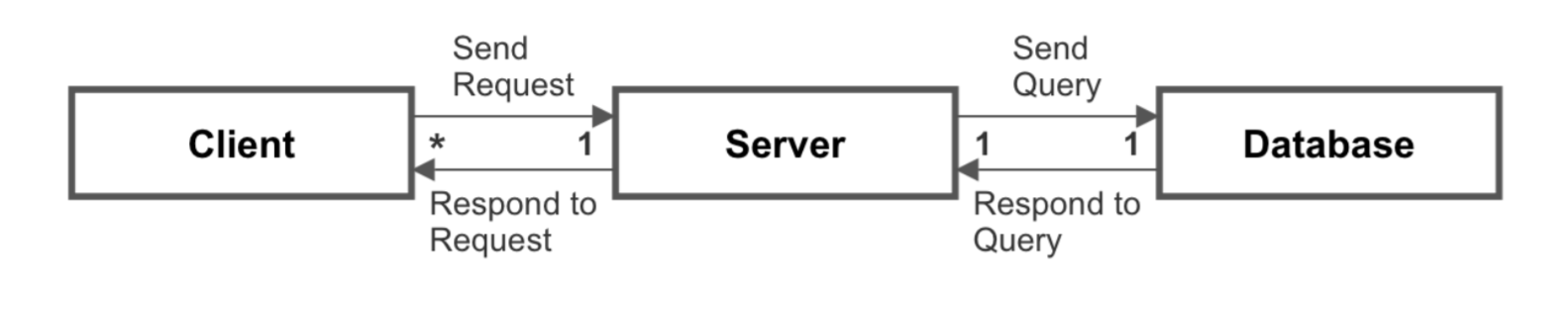
**Components**

1. Client
   1. Provide UI interface for users to interact with the system
   2. Send HTTP requests to the server based on user interaction with the UI
   3. Update the UI for the user based on the HTTP response received from the Server
2. Server
   1. Receives and handles HTTP requests through an API
   2. Validates requests
   3. Sends queries to the database to add, modify, retrieve, or delete information
   4. Generates responses and sends them back to Clients
3. Database
   1. Stores all data for the platform in various, connected tables
   2. Completes queries sent from the server and sends appropriate data to the server

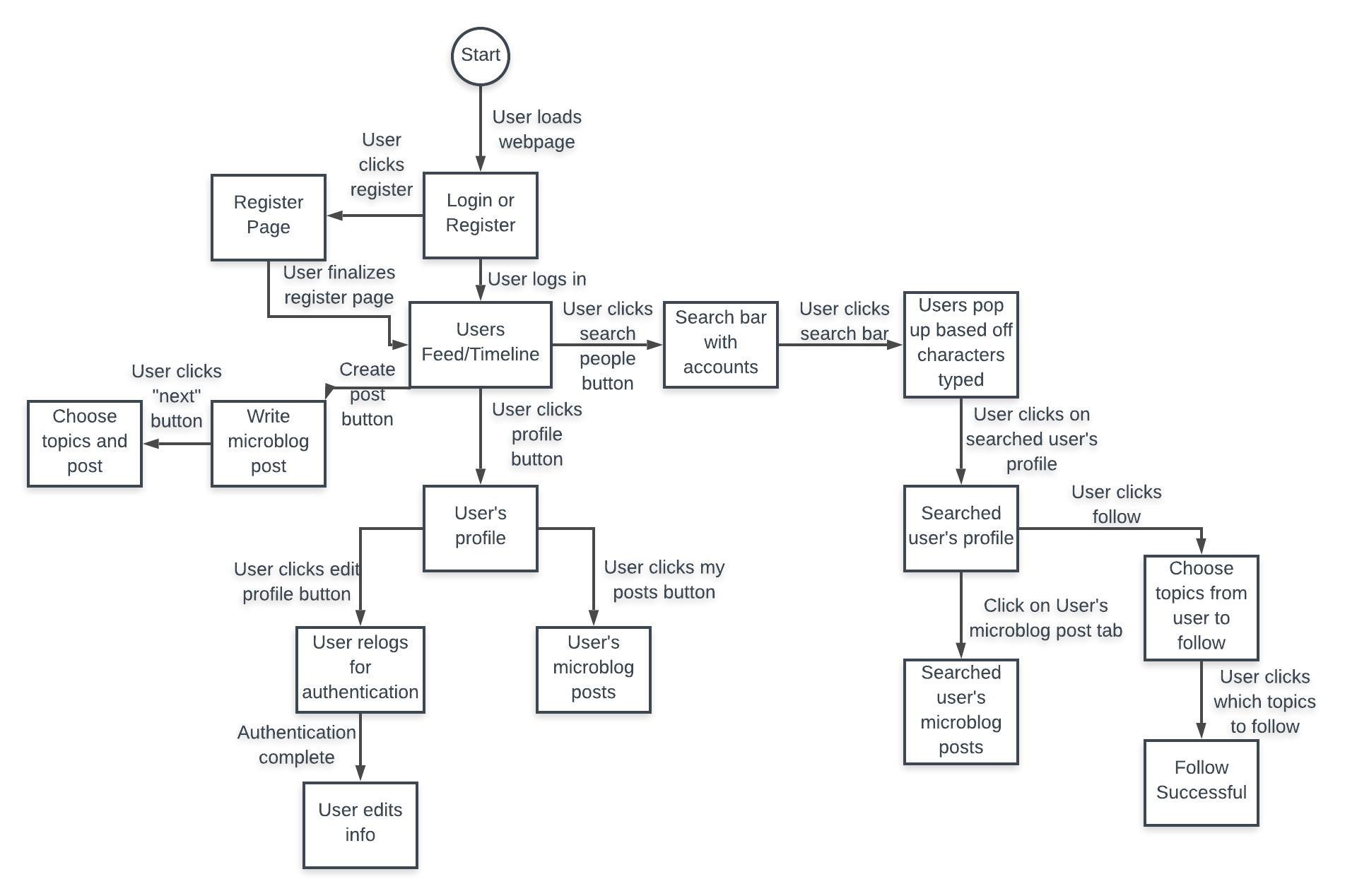
**High-Level Overview**

This project is a web application, accessed via webpage, that allows users to write microblogs. Users may tag their microblogs with any topics they wish and may also view the microblogs’ of other [user, topic] combinations they choose to follow. This application will use a client-server model in which one server handles requests from many different users using Spring Boot in Java. The server will accept requests, access and interact with an SQL database, then provide feedback to the clients in a timely manner.

**Component Interaction Diagram**

****

**UML Diagram**



**Design Issues**

**Functional Issues**

How should a user’s topics be organized when another user attempts to follow him/her?

* Solutions
  + Option 1: Alphabetical order
  + Option 2: Priority based on number of tagged posts with each topic
  + Option 3: Random order
* Choice: Option 2
  + Justification: When following another user, it is typical to follow them for the content that they produce. As such, it is best to display the topics they use most frequently as those are the topics that best represent their content. Alphabetical order and random order would be easier to display to a user but could misrepresent the content of the account they wish to follow as it may display topics that are not used often by the account and may make it so that it is harder to see post that you may be interested in.

What information do we need for signing up for an account?

* Solutions
  + Option 1: Username, password, email address
  + Option 2: Username and password only
  + Option 3: Username, password, email address, phone number
* Choice: Option 1
  + Justification: Basic account info includes a username and password. Email will allow for extra verification when signing up. The email will also be used for resetting the password when needed. We decided phone would only be useful for extra security, which we considered not necessary. In addition, sending verification code through SMS is a paid service. We do not plan to add this cost to our project. Therefore, username, password and email should be enough for the sign up purpose.

How should users find people to follow?

* Solutions
  + Option 1: Have a recommended page based on user’s followed topics
  + Option 2: Have a “popular” section including a list of most popular users of all time
  + Option 3: Have a “trending” section with a list of the popular users in the last 24 hours, along with a search bar to find particular users
* Choice: Option 3
  + Justification: Since Hot TopiX is a social app, users need to be able to find users to follow for content. Users generally want to see what is currently popular, so having a trending page with the most relevant profiles in the last 24 hours allows users to find the best content currently available. There will also be a search bar, to allow users to search for other users by profile name.

How should users login to the app?

* Solutions
  + Option 1: No login required
  + Option 2: Third-party authentication
  + Option 3: Enter email and password
* Choice: Option 3
  + Justification: To have a secure profile, each user will be required to login with their email and password before being able to access the features and functionality of Hot TopiX

How should topics be displayed in a user’s post?

* Solutions
  + Option 1: Within the post text
  + Option 2: Above the post’s text
  + Option 3: Below the post’s text
* Choice: Option 2
  + Justification: Since the goal of Hot TopiX is to display posts that users are interested in, it is necessary to make the topics visible to the user instantly. To do this, it is best to display the topics above the post as it will make it easier when scrolling to see posts with the topics you are interested in first. Having topics in the post could make it hard to read, and having topics below the posts text would make them less visible as a user is more likely to read the post through before seeing the topics.

**Non-Functional Issues**

Which hosting service should we use?

* Solutions
  + Option 1: AWS
  + Option 2: Google Cloud
  + Option 3: Azure
* Choice: Option 1
  + Justification: We will host our server using AWS. A benefit of hosting with AWS is that it gives our application increased scalability. Our team members are also well versed in using AWS.

What backend should we use?

* Solutions
  + Option 1: Spring Boot (Java)
  + Option 2: PHP
  + Option 3: Node.js (JavaScript)
* Choice: Option 1
  + Justification: Our team members are all familiar with Java and we want to use Spring Boot because it will reduce development, unit testing, and integration testing time. Also, Spring Boot enables us to develop production-ready web applications very easily.

Which frontend framework should we use?

* Solutions
  + Option 1: React JS
  + Option 2: Bootstrap
  + Option 3: Angular
* Choice: Combination of Option 1 and Option 2
  + Justification: React JS can be used as a base in the development of single-page web apps, as it is optimal for fetching rapidly changing data that needs to be recorded. Our application will need to use a frontend framework that can keep up with the fast paced nature of a social media

Which database should we use?

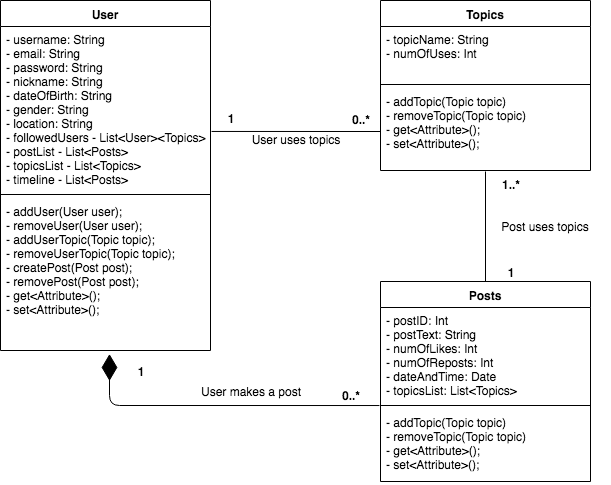
* Solutions
  + Option 1: MySQL
  + Option 2: SQLite
  + Option 3: MongoDB
  + Option 4: Oracle XE
* Choice: Option 1
  + Justification: MySQL focuses on ease-of-use and efficiency makes it an attractive solution for our needs. It’s also free and open source and has many tutorials and guides so learning and working with it will be very easy.

**Design Details**

**Class Design**

**Client Class Diagram**

**Luc**

****

**Descriptions of Classes and Interaction between Classes**

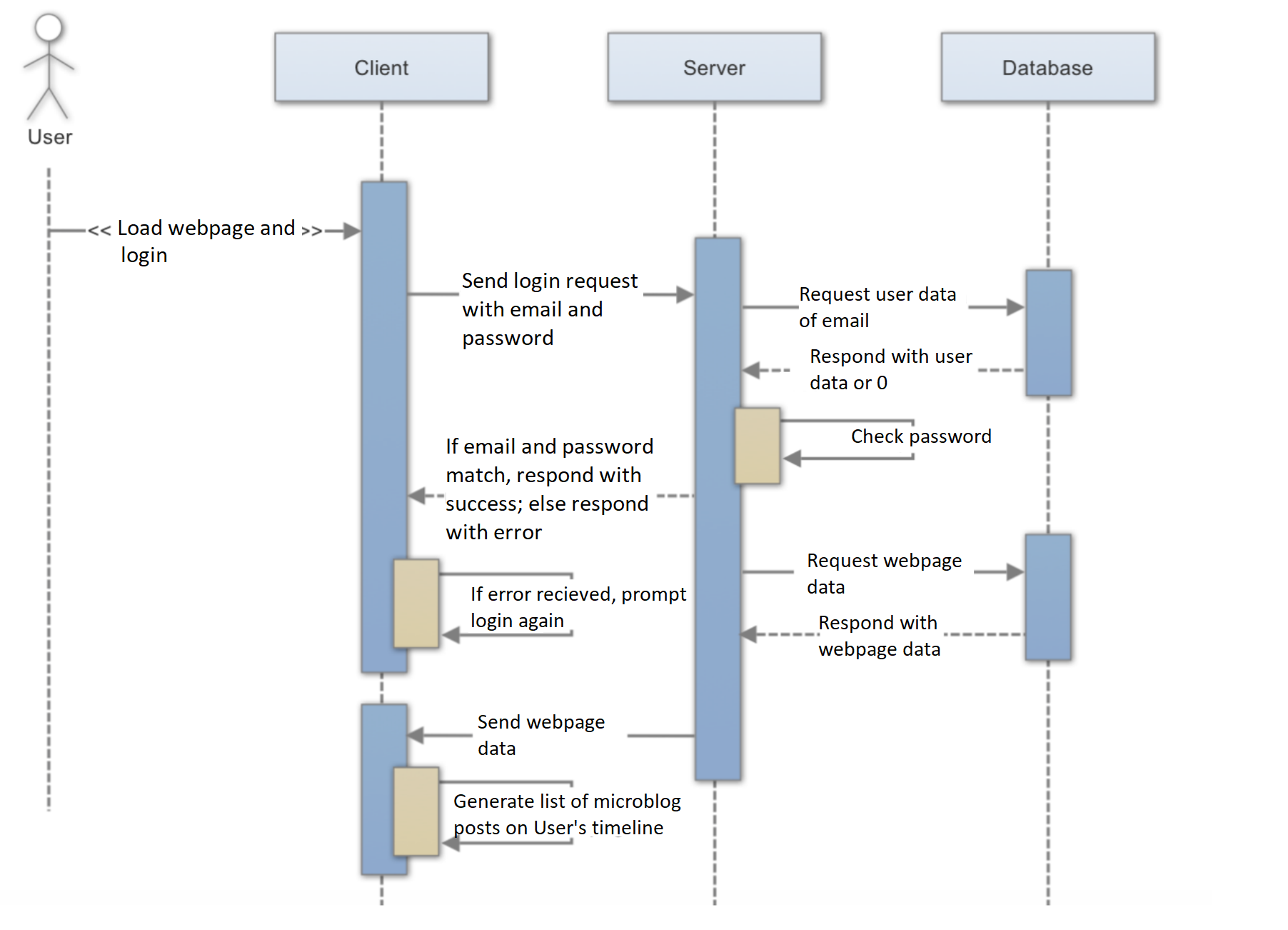
**Client Classes and Interactions**

The classes are designed based on the objects in our application. Each class has a list of attributes which are the characteristics owned by each object. There are also functions that connect the different classes.

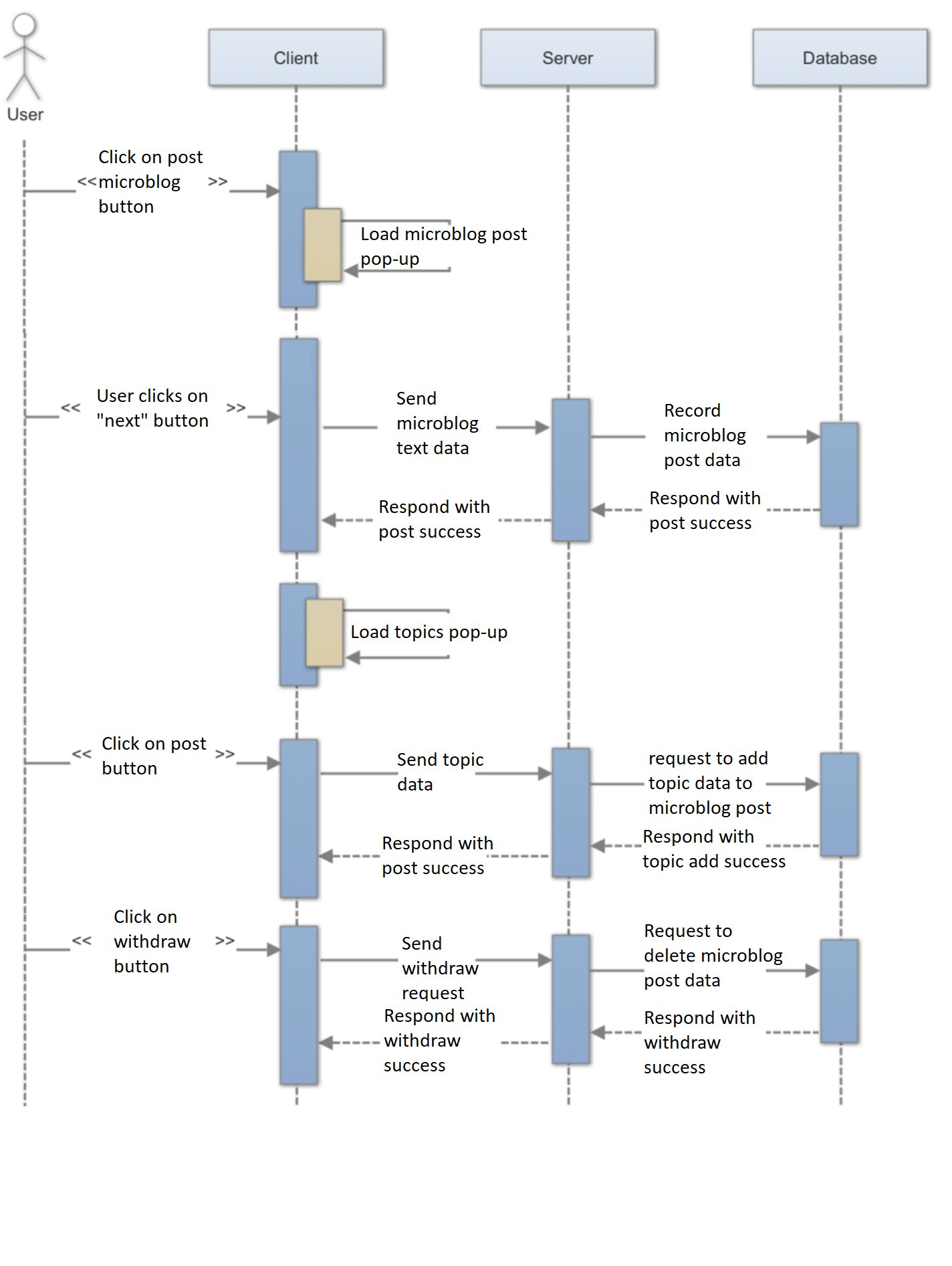
* **User**
  + User object is created when someone signs up on Hot TopiX
  + Each user will have a username, password, and email for login and contact purposes
  + Each user will be able to change their username, password, and email
  + Each user will have the option to add a nickname, date of birth, gender, and location
  + Each user will be able to edit their user information at any time via their profile
  + Each user will be able to follow/unfollow another user and choose which topics they want to view from that user (also an option to follow all topics)
  + Each user will have a custom timeline on their homepage that shows the microblogs from the followed users that also have a tag that the user enabled for the followed user
  + Each user will be able to search for other users
  + Each user will be able to create a post with at least one and no more than five topics
  + Each user will be able to view a timeline of posts from users they follow.
* **Topics**
  + A topic object will be created when a user creates a new custom topic
  + Each topic can be added to a post when the post is created or edited
  + Each topic can be removed from a post
  + Each topic will track how many times it has been used in a post
* **Posts**
  + A post object will be created when a user makes a new post
  + Each post will have at least one and no more than five topic objects associated with it
  + A post will be associated with the user that created the post
  + A post will be editable by its author (user that originally created the post)
  + A user will be able to like a post
  + A post object will track the number of likes
  + A user will be able to repost a post, either with or without an additional comment

**Sequence Diagrams**

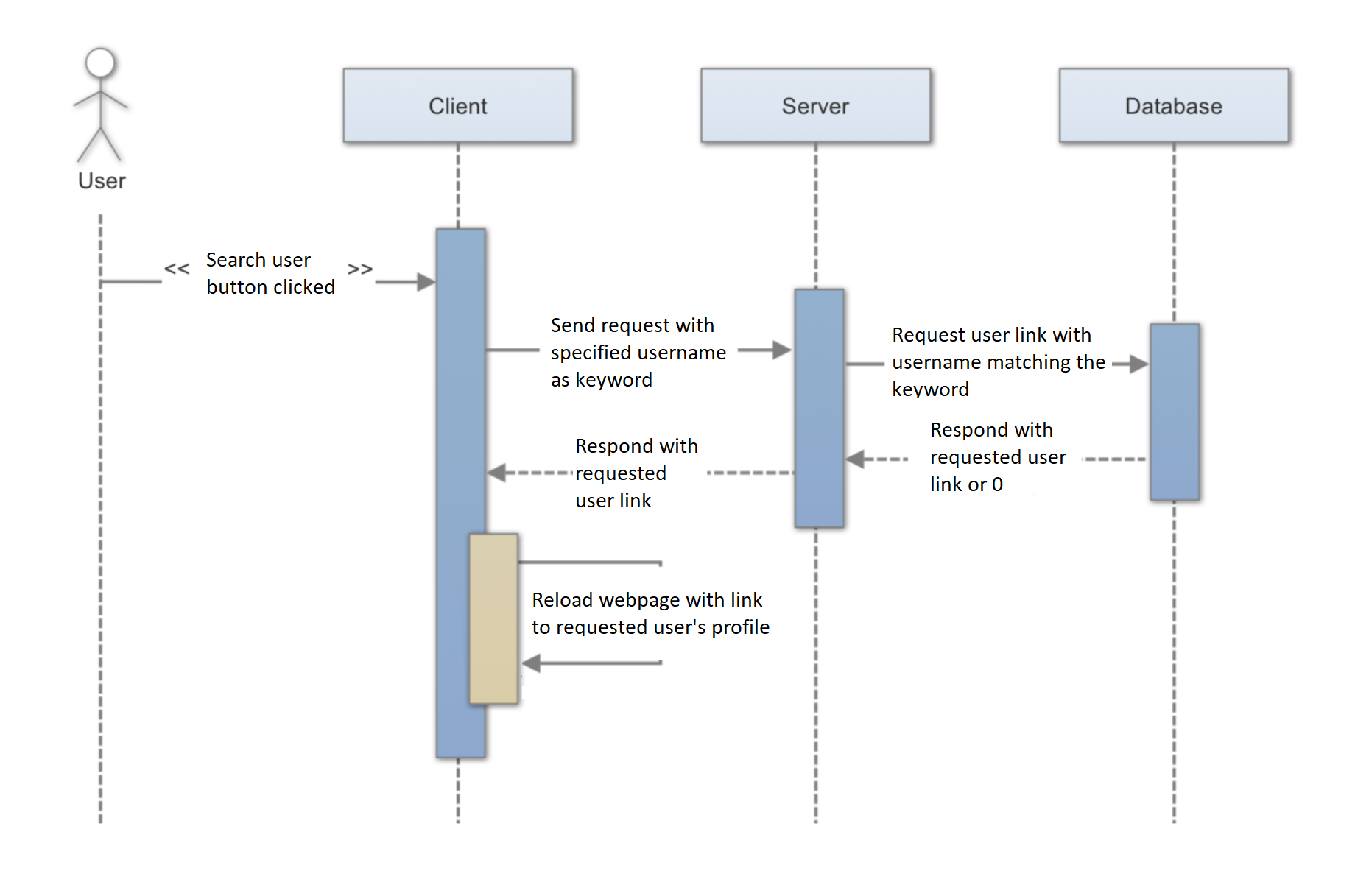
1. Sequence of events when user logs in



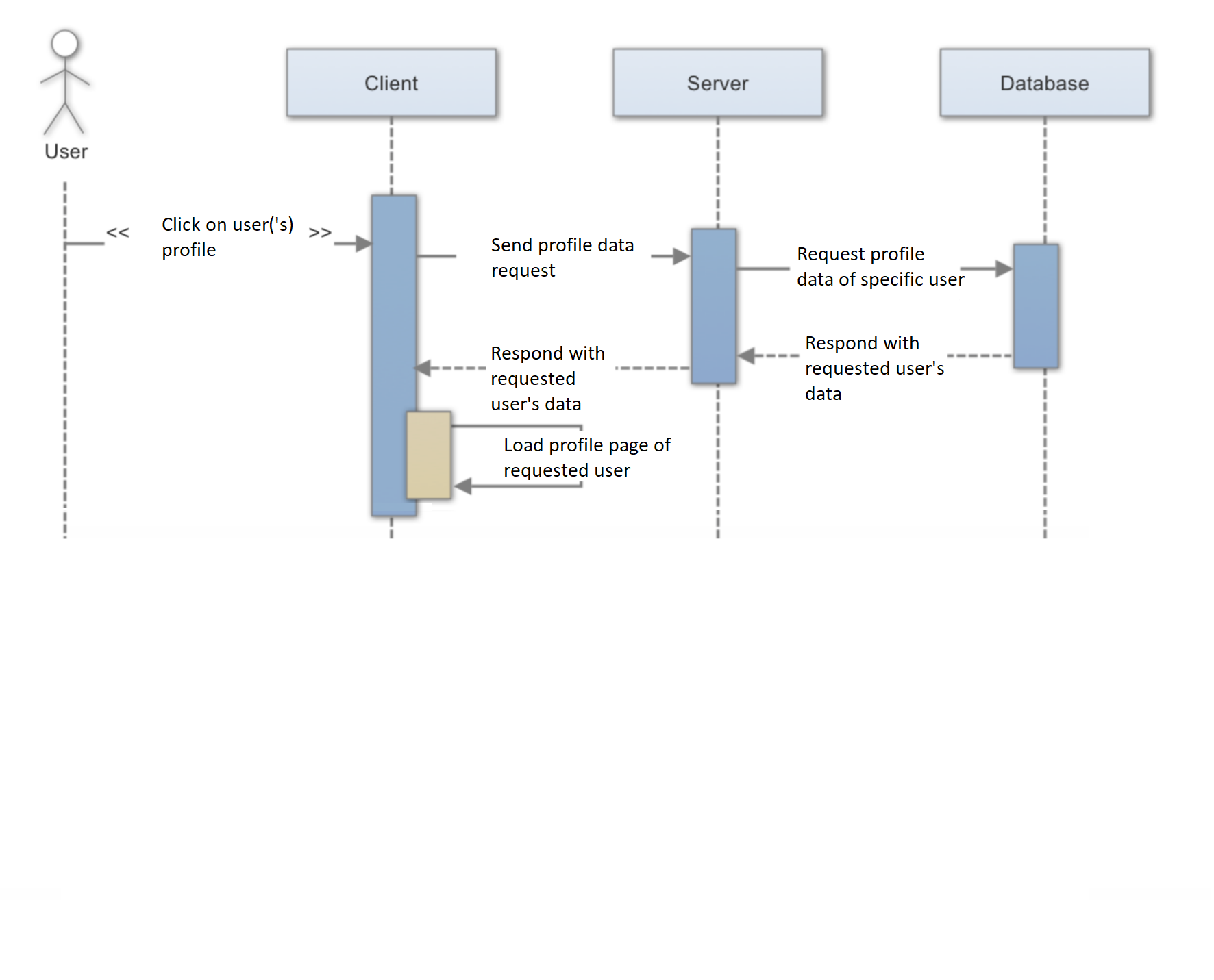
1. Sequence of events when user posts a microblog



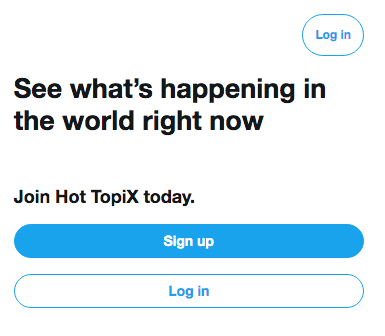
1. Sequence of events when user searches for other users



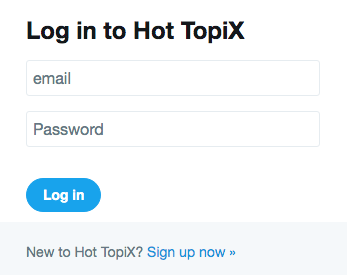
1. Sequence of events when user looks at a profile



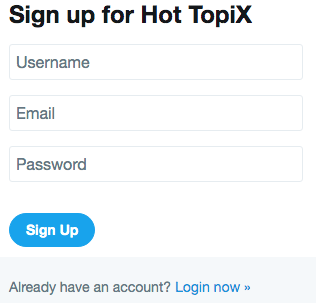
**UI Mockup**



Above is an example homepage where the user chooses to login or sign up for a new account.



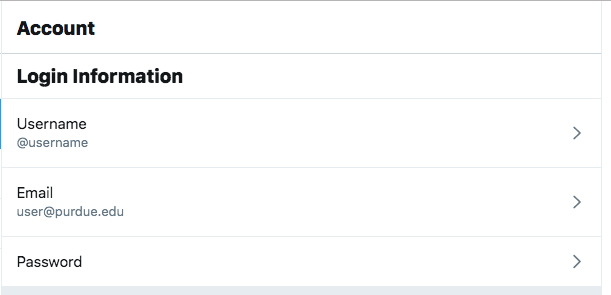
This is an example login page where the user enters their login information, or the user can go to the sign up page if they do not have an account yet.



This is an example sign up page that requires the user to enter a username, email, and password to make an account. There is also an option to go back to the login page if they already have an account. The sign up button will send the information to the server and check for duplication as well as an invalid email.



This is an example of the homepage where the user sees a timeline of microblogs corresponding to who they follow and which topics they follow. There are buttons to search for different topics or people, link to the profile page, logout, and create a new microblog.



When clicking on the profile button, it will look like the homepage, except the timeline would be the microblogs that the user has posted. There will also be a link to edit the account that looks like the example above where the user can edit their username, email, and password. There will also be an option to edit or add a profile picture.