# Software Engineering CSC648-848-05 Spring 2023

# Piqued

Team 5

Jose Avila, Team Lead, javila6@mail.sfsu.edu
Andy Shi, Frontend Lead, ashi2@mail.sfsu.edu
Leo Saeteurn, Backend Lead, lsaeteurn@mail.sfsu.edu
Nishit Pachchigar, Github Master, npachchigar@mail.sfsu.edu
Joshua Hayes, Database Master, jhayes10@sfsu.edu
Gautami Kollolu Srinivasa, Document Editor, gkollolusrinivasa@sfsu.edu

Milestone 2
March 30, 2023

History Table

| Version | Submission Date |
|---|---|
| M2V1 | March 30, 2023 |
| M1V1 | March 5, 2023 |

# Table of Contents

## Data Definitions

1. General user: Individuals who do not have a registered account but can search and view public posts on the website. They can also view comments and reactions on public posts.
2. Registered user: Individuals who can log into their account, create, and publish posts on the website. They have the ability to edit and delete their own posts and comments, manage their account information, and view site statistics.
3. Site Moderators: Administrators who manage the frontend and backend of the website.
4. Blog Posts/Articles: Main content pieces on the website, written by users/authors. They contain text, images, and/or videos and can be categorized and tagged.
    4.1. Author: The user who originally posted the article.
    4.2. Published timestamp: A timestamp that represents when an article was originally posted.
    4.3. Updated timestamp: A timestamp that represents when an article was last updated by the author.
    4.4. Images: Visual content that can be used to enhance posts and improve engagement.
    4.5. Categories: Groupings of posts based on topic or theme
    4.6. Tags/Hashtag: Labels which can be added to a post to define its category and determine how it is found through search or recommended to other users.
    4.7. Comments: Rich text responses to posts made by users on the website.
    4.8. Reactions: Emotional responses on posts. This may include likes, emote responses, or any other form of instant feedback.
        4.8.1. Like: A positive response to a post by a registered user initiated by a simple button click.
        4.8.2. Reaction: Any other form of response by a registered user that can be either positive or negative.
        4.8.3. Reaction count: A number which indicates the amount of reactions to a post.
5. Repost: A link to another user's post that appears on a user's profile.
6. Recommendations: Algorithmically generated list of posts based on the user's interests.
7. Feed/For You Page: A landing page for logged-in users which shows a scrollable list of recommended posts.
8. Search: A search function that allows users to search blogs.
    8.1. Search filter: Function to filter out results based on certain criteria. This may be based on the aforementioned categories.
    8.2. Search sort: Function to sort the results based on a certain order.
9. Editor: UI for editing a blog post. This can feature a rich text editor where users can apply formatting to text and insert images, video, and other types of media.
10. Advertising Banners: Display ads that generate revenue for the website.

11. Notification: Represents a notification sent to a user. This can be a new follower, a comment on a post, etc.
12. Profile Page: A customizable page which contains a user's public information and posts.
    12.1. Biography/Bio: A short, user-provided paragraph about the registered user.
13. Navigation bar/Header: A section where links are grouped together for easy navigation.
    13.1. Website logo: Display the company's logo on the website to represent the company
    13.2. Profile link: A link where the user can click on to go to their account page to edit their account.
    13.3. Home page link: A link where the users can navigate back to the main domain.
    13.4. Navigation links: Links to various primary pages or pages commonly accessed by users.
    13.5. Hamburger Menu: A menu which contains navigation items to pages. This may be used on mobile for containing primary navigation links, or to allow users to quickly access followed pages.
14. Footer: An section at the bottom of the page which contains informational links.
    14.1. Company's "about us" page: Information about each of the developers of the website.
    14.2. Company's description: Information about the website and its purpose.
    14.3. Social Media Links: Links to the website's social media profiles.
    14.4. Privacy policy and Terms of Service: Links for legal purposes.

# Prioritized Functional Requirements

1. Priority 1 Requirements :
    1.1. All Users
        1.1.1. All users shall be able to view the website's information
        1.1.2. All users shall be able to contact the site moderators with questions
        1.1.3. All users shall be able to view the privacy policy
        1.1.4. All users shall be able to view the terms and conditions
    1.2. Guest Users
        1.2.1. Guest users shall be able to create an account
        1.2.2. Guest users shall be able to use a unique username to create an account
        1.2.3. Guest users shall be able to only view public posts
        1.2.4. Guest users shall be able to view the contents of public posts
        1.2.5. Guest users shall be able to view a profile's public information
        1.2.6. Guest users shall be able to search for public posts
        1.2.7. Guest users shall be able to discover other public users
    1.3. Registered Users
        1.3.1. Registered users shall be able to log in
        1.3.2. Registered users shall be able to log out
        1.3.3. Registered users shall be able to sign in with both their email and username.
        1.3.4. Registered users shall be able to request to reset a new password if they forget it
        1.3.5. Registered users shall be able to retrieve their username if they forget it
        1.3.6. Registered users shall be able to add a profile picture
        1.3.7. Registered users shall be able to edit their profiles
        1.3.8. Registered users shall be able to change their profile picture
        1.3.9. Registered users shall be able to change their displayed username
        1.3.10. Registered users shall be able to change their email address
        1.3.11. Registered users shall be able to change their DOB
        1.3.12. Registered users shall be able to change what information is displayed publicly to general users
        1.3.13. Registered users shall be able to add contact information
        1.3.14. Registered users shall be able to make contact information private or public.
        1.3.15. Registered users shall be able to change their password
        1.3.16. Registered users shall be able to search for friends by username
        1.3.17. Registered users shall be able to see pending friend requests
        1.3.18. Registered users shall be able to accept friend requests

1.3.19. Registered users shall be able to reject friend requests
1.3.20. Registered users shall be able to remove friends
1.3.21. Registered users shall be able to create blog posts.
1.3.22. Registered users shall be able to add photos on blog posts.
1.3.23. Registered users shall be able to search for posts.
1.3.24. Registered users shall be able to update their posts.
1.3.25. Registered users shall be able to delete their posts.
1.3.26. Registered users shall be able to disable comments for their posts.
1.3.27. Registered users shall be able to add comments on post.
1.3.28. Registered users shall be able to edit their own comments.
1.3.29. Registered users shall be able to delete their own comments.
1.3.30. Registered users shall be able to like/react to other people's comments
1.3.31. Registered users shall be able to comment on posts
1.3.32. Registered users shall be able to react to posts
1.3.33. Registered users shall be able to send private messages
1.3.34. Registered users shall be able to receive private messages.

1.4. Site moderators - creators(us)

1.4.1. Site moderators shall be able to delete posts/articles
1.4.2. Site moderators shall be able to ban accounts
1.4.3. Site moderators shall be able to temporarily block accounts for review
1.4.4. Site moderators shall be able to edit the layout of the platform
1.4.5. Site moderators shall be able to add additional features
1.4.6. Site moderators shall be able to change template layouts for profiles
1.4.7. Site moderators shall be able to remove spam accounts
1.4.8. Site moderators shall be able to answer users' questions and concerns
1.4.9. Site moderators shall be able to create an account
1.4.10. Site moderators shall be able to post on the platform's main page
1.4.11. Site moderators shall be able to make site announcements
1.4.12. Site moderators shall be able to update website
1.4.13. Site moderators shall be able to temporarily suspend the website for maintenance

## 2. Priority 2 Requirements :

### 2.1. Guest users

2.1.1. Guest users shall be able to view comments on public posts
2.1.2. Guest users shall be able to view reactions on public posts

### 2.2. Registered Users

2.2.1. Registered users shall be able to customize profile using the standard format template
2.2.2. Registered users shall be able to make their information private

2.2.3. Registered users shall be able to make some of their information private
2.2.4. Registered users shall be able to make some of their information public
2.2.5. Registered users shall be able to repost public posts

2.3. Administrator Users

2.3.1. Administrator users shall be able to monitor their account
2.3.2. Administrator users shall be able to create authorized users on account
2.3.3. Administrator users shall be able to limit authorized users access on account

2.4. Site Moderators

2.4.1. Site moderators shall be able to reinstate accounts
2.4.2. Site moderators shall be able to review accounts

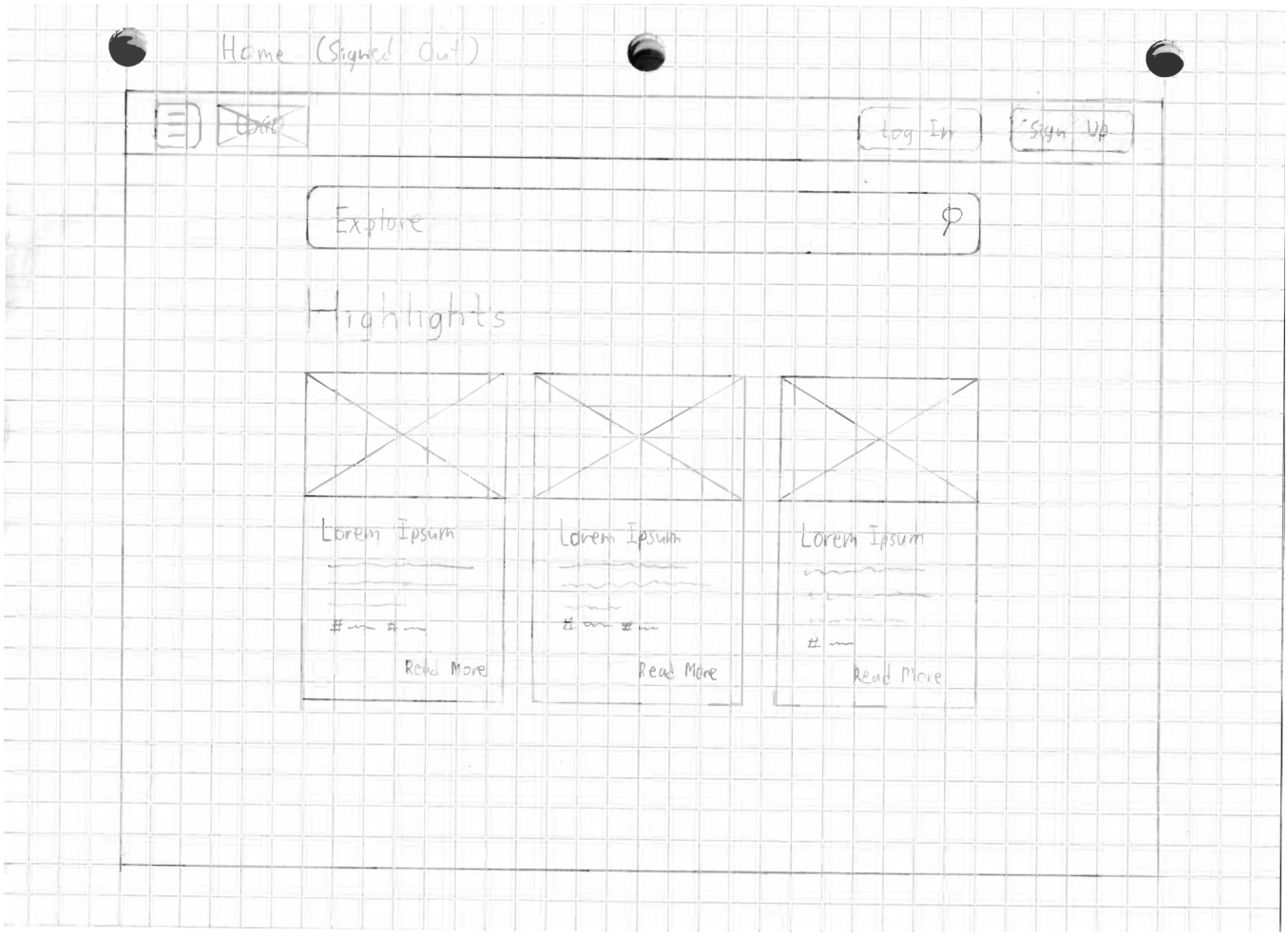
## 3. Priority 3 Requirements:

3.1. Registered Users

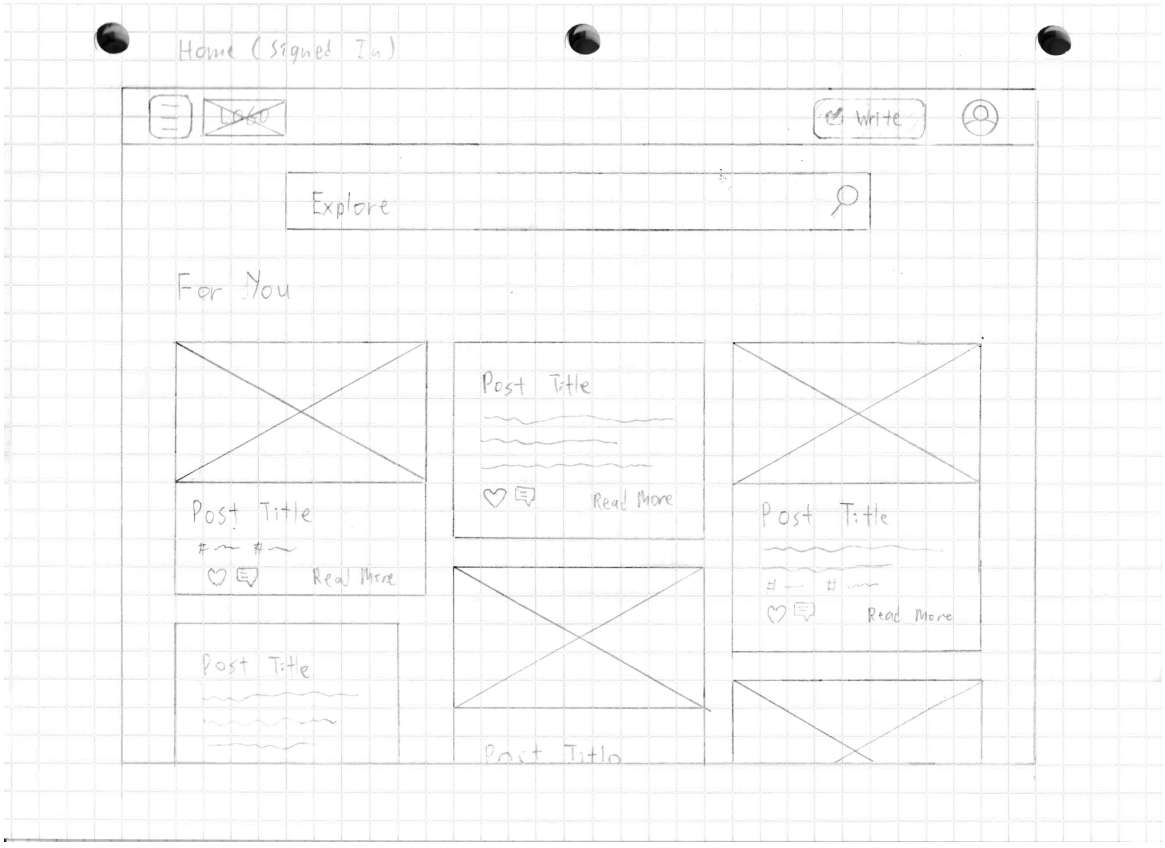
3.1.1. Registered users shall be able to post videos on their blog post.

## UI Mockups and Storyboards (high level only)

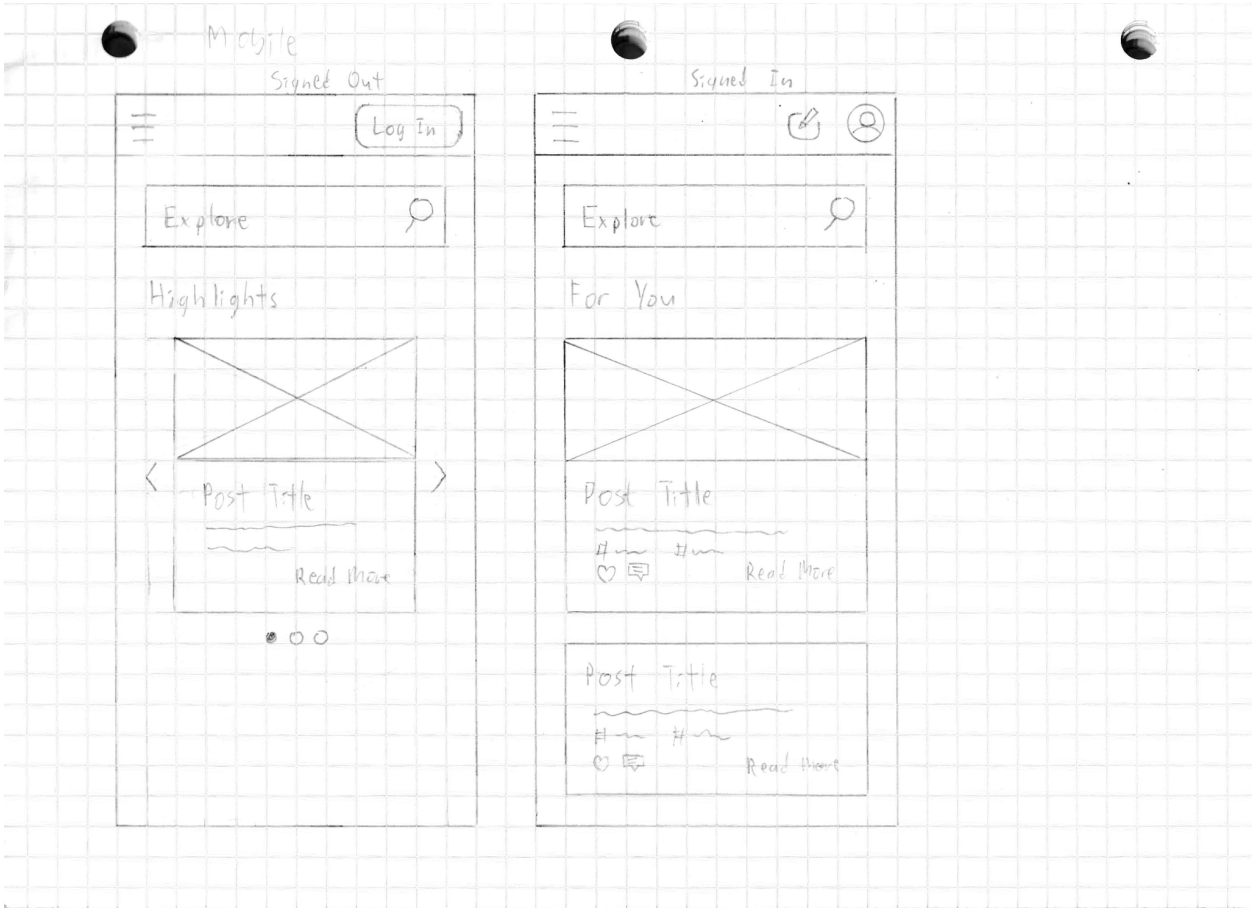
1. Homepage (Logged Out)



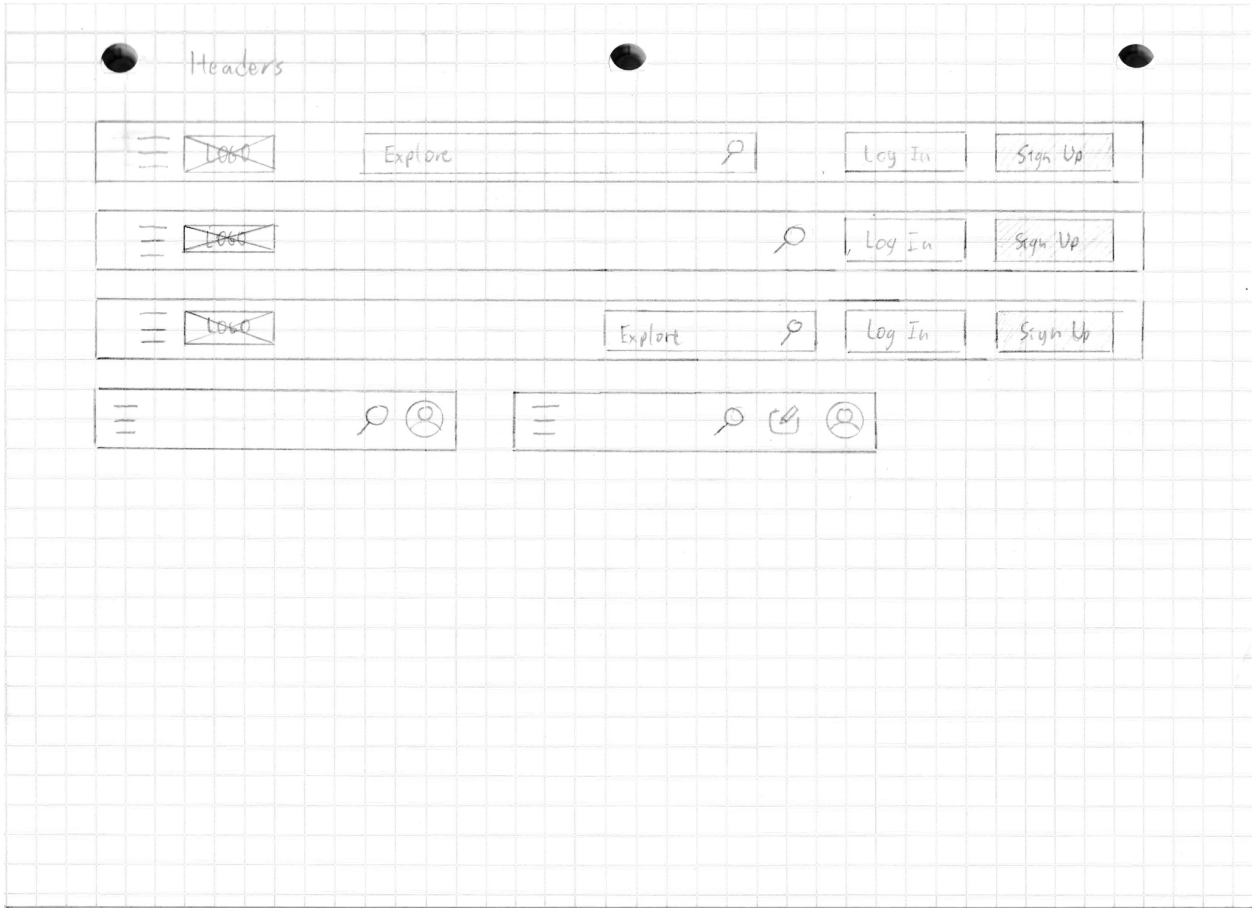
2. Homepage (Logged In)/For You Page



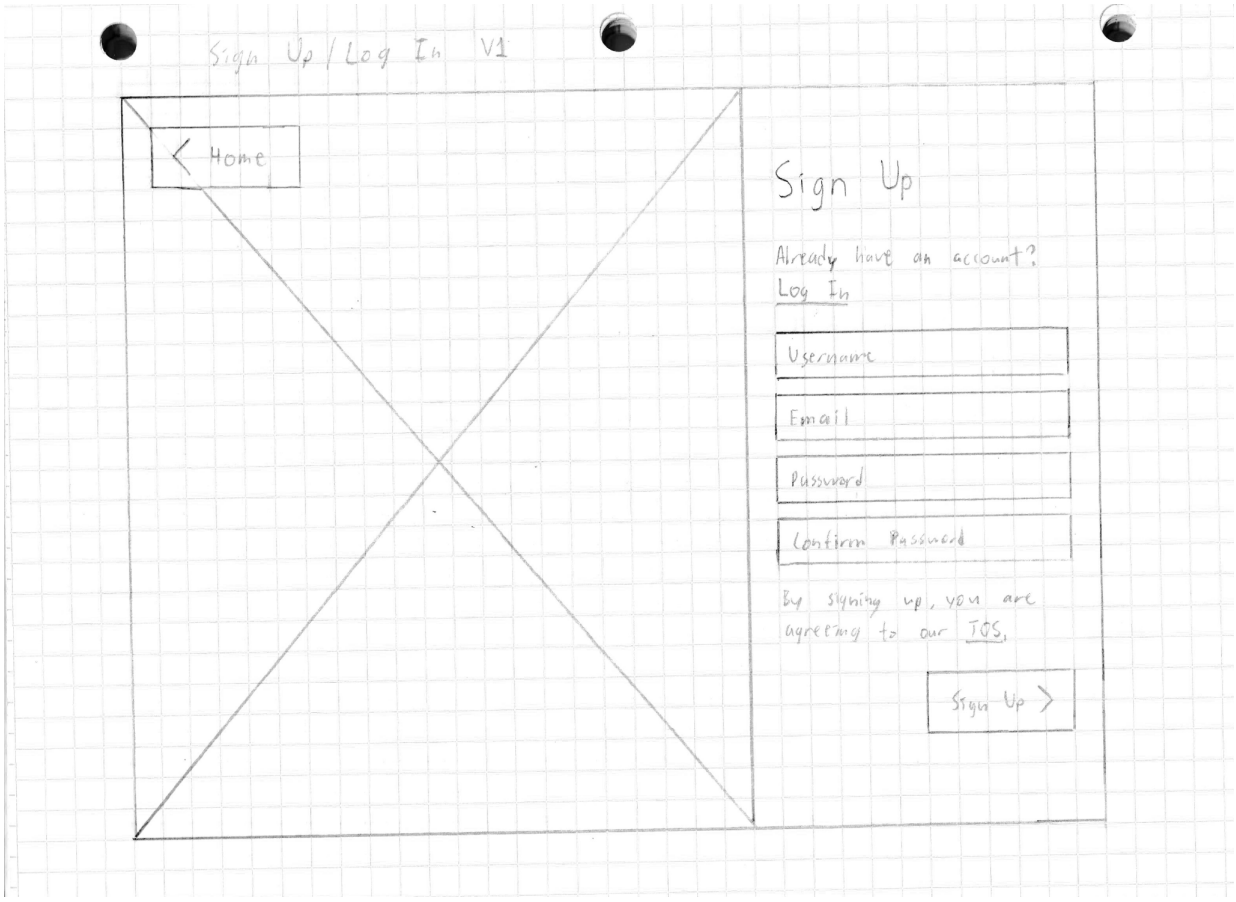
3. Homepages (Mobile)



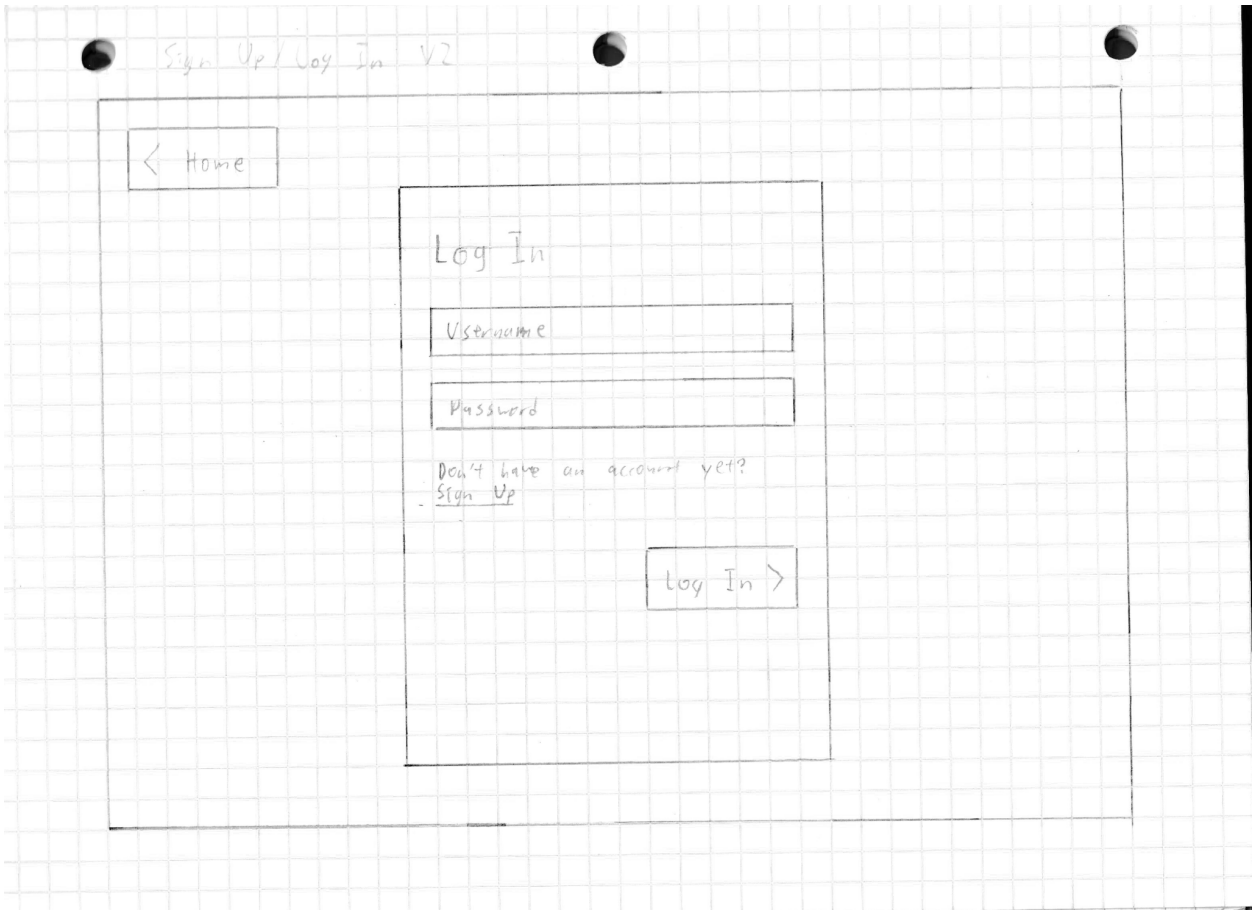
4. Various Navigation Bar Ideas



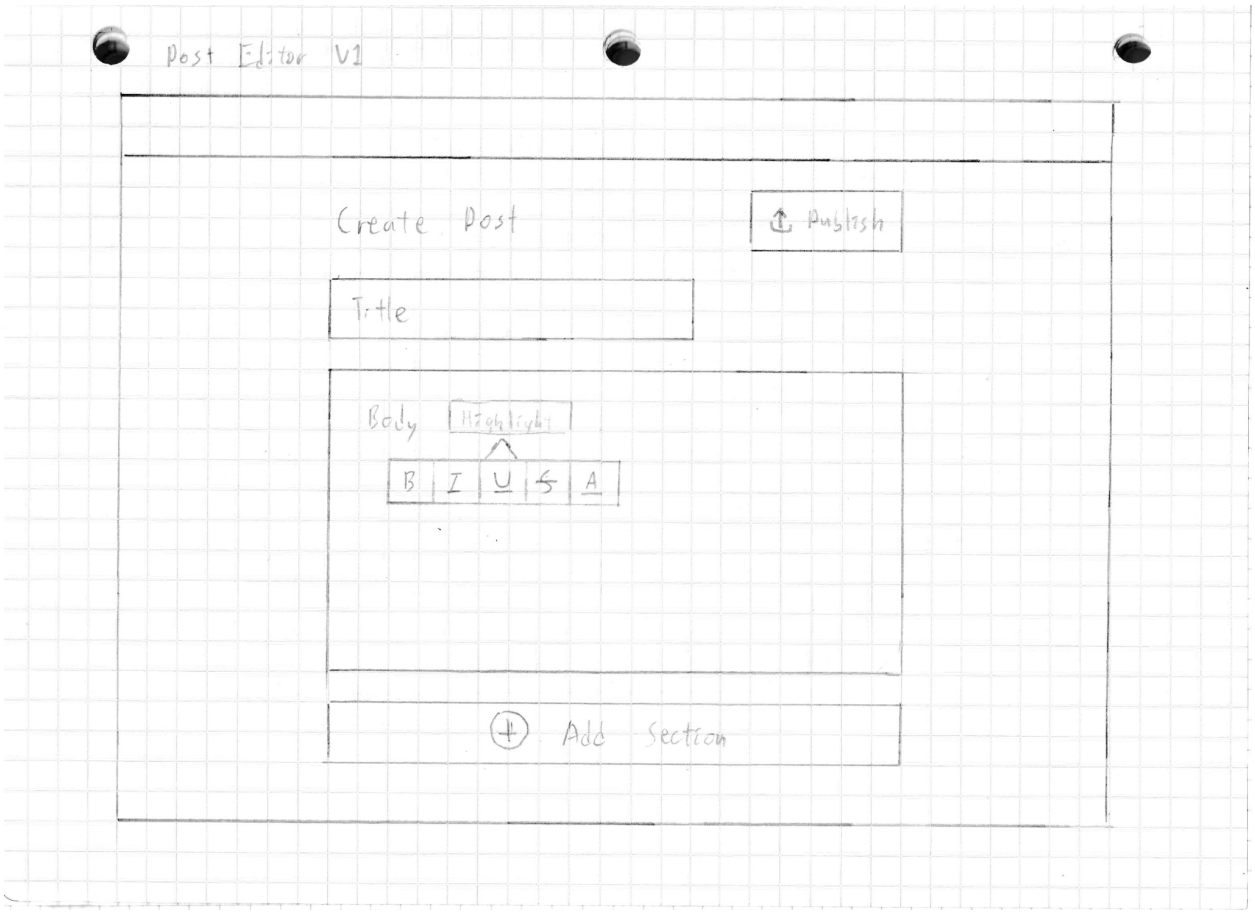
5. Log In/Sign Up Page (Idea 1)



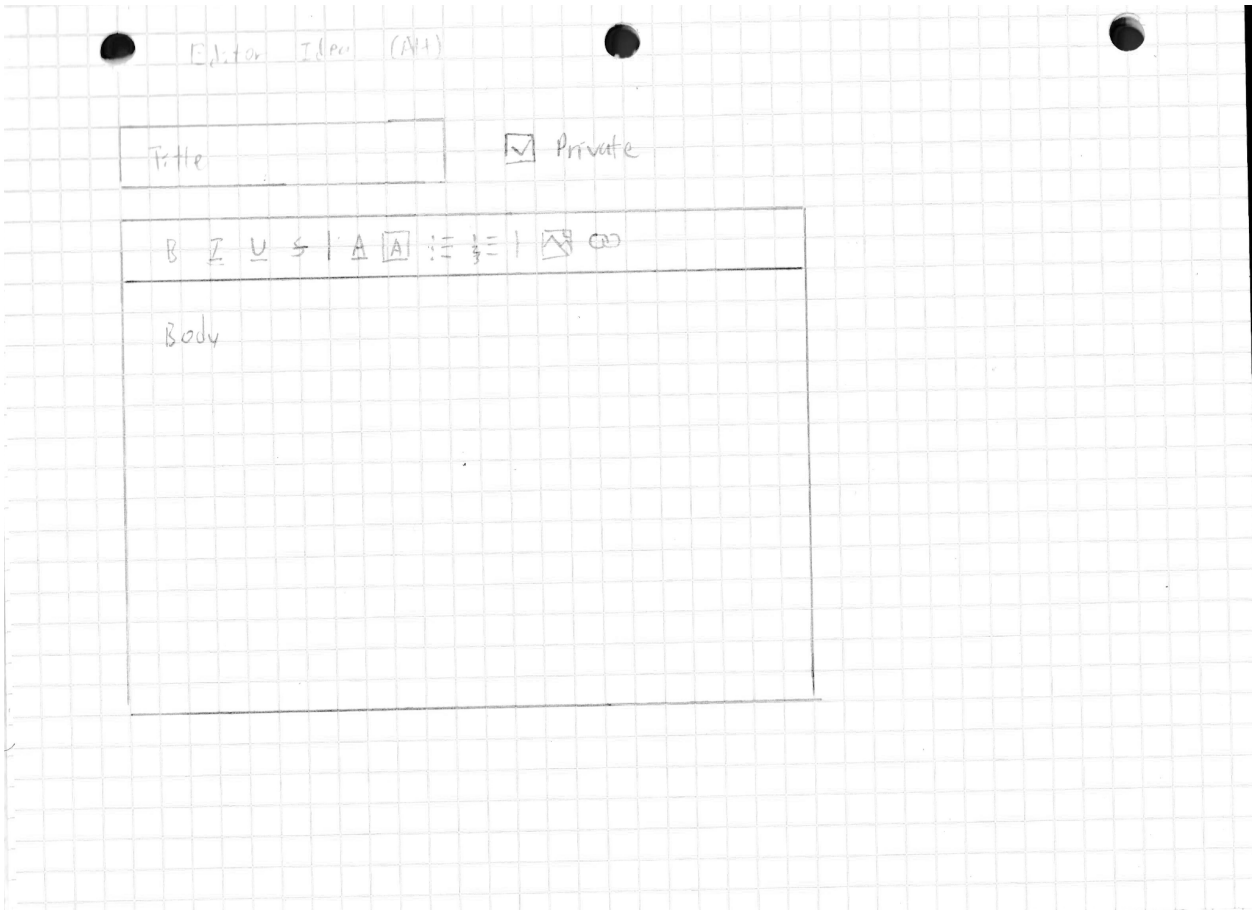
6. Log In/Sign Up Page (Idea 2)



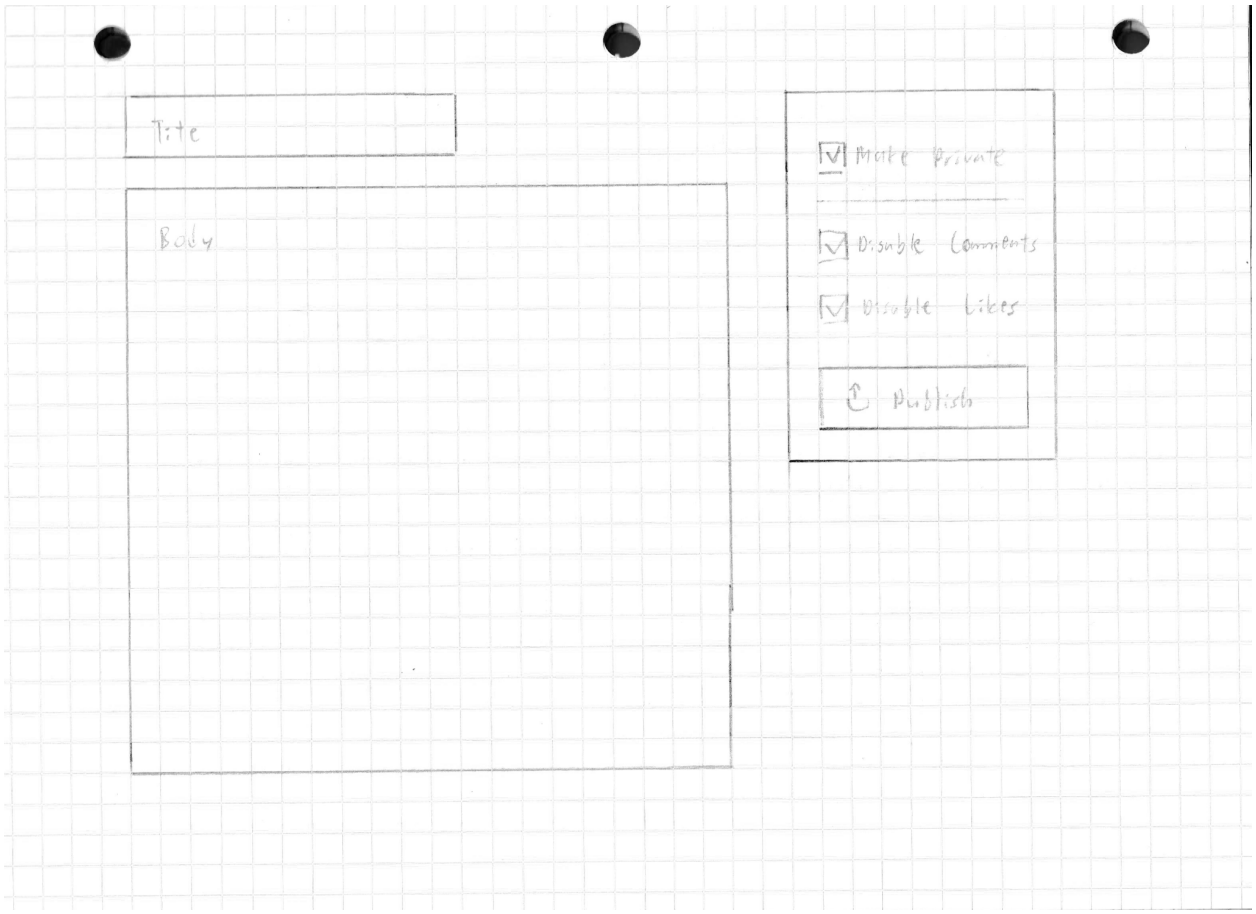
7. Create Post (Idea 1)



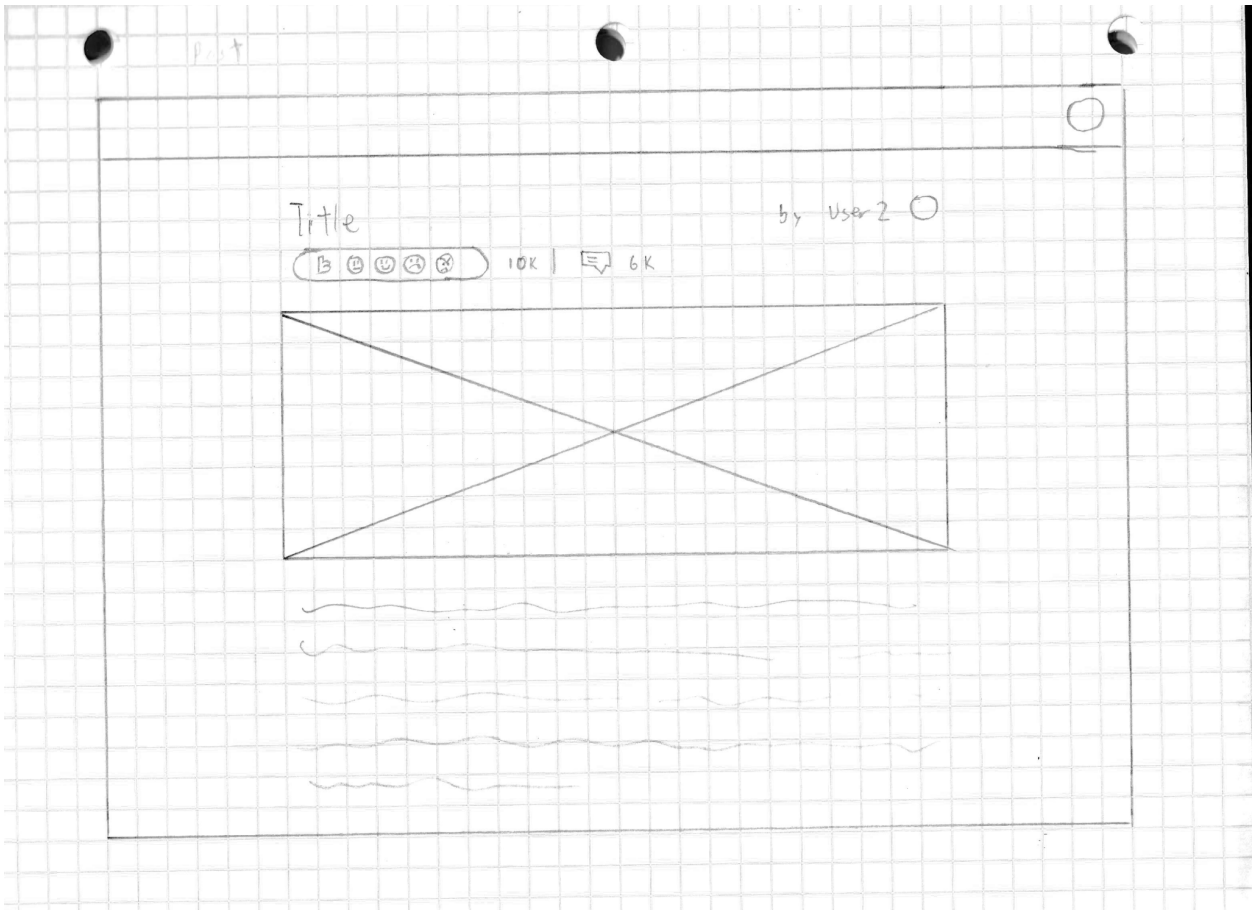
8. Create Post (Idea 2)



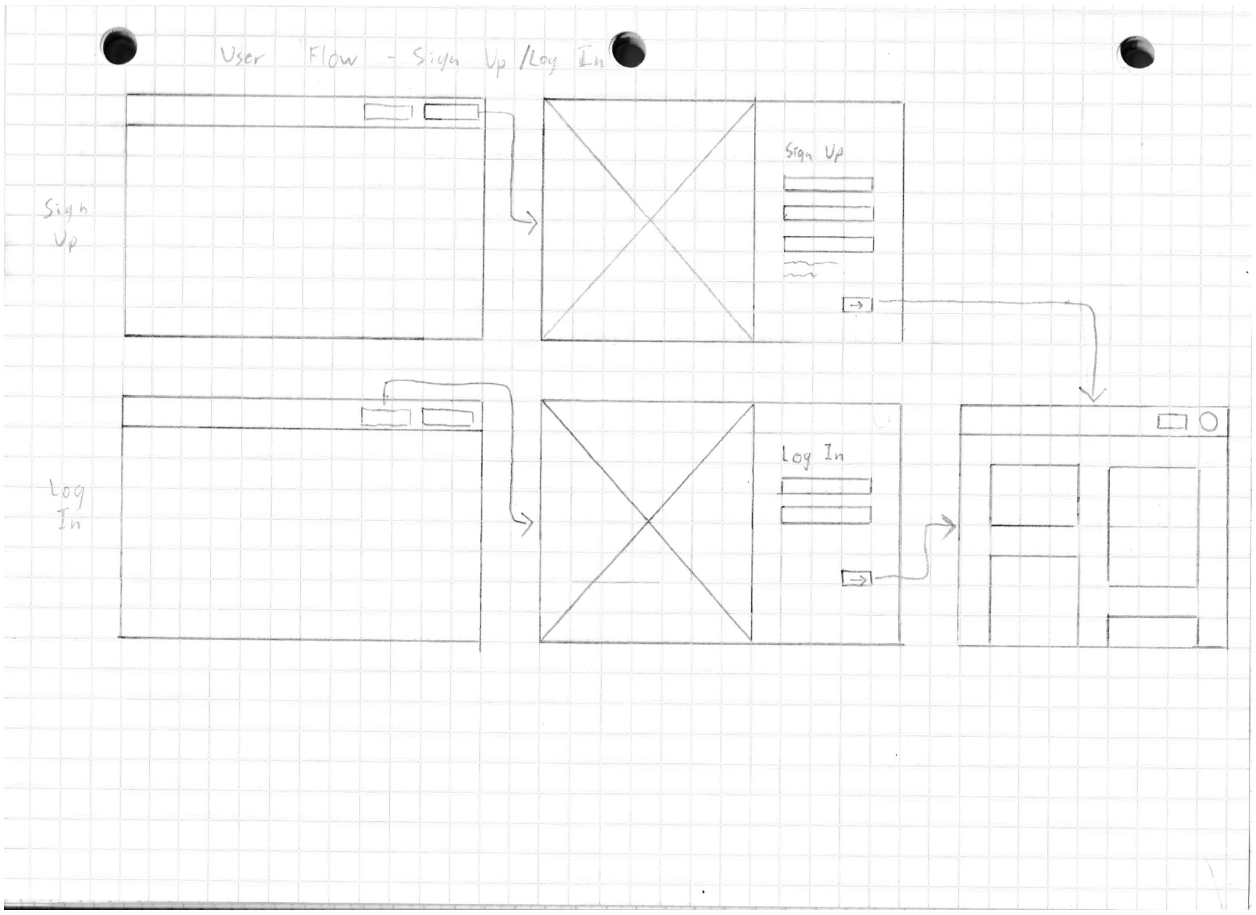
9. Create Post (Idea 3)



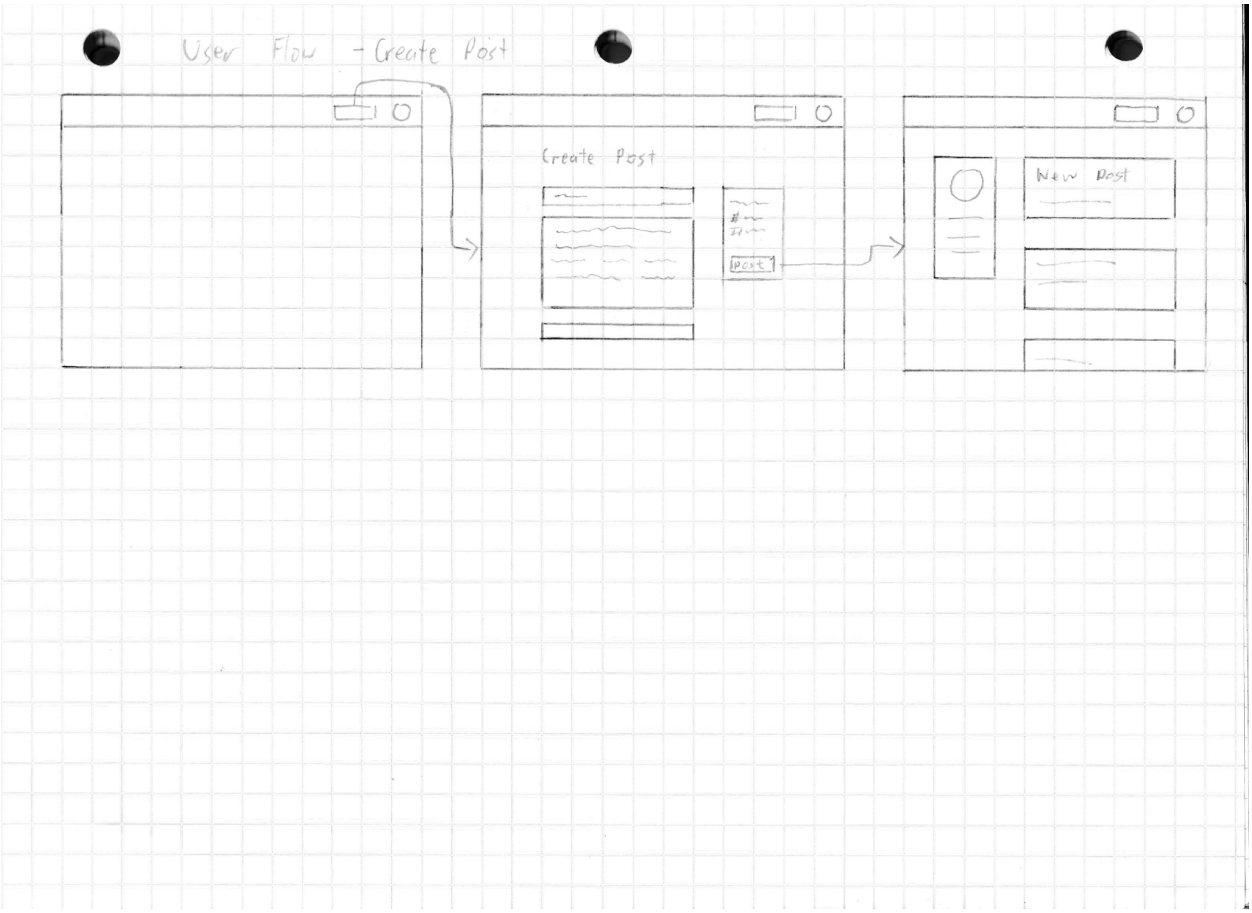
10. Post



11. Log In/Sign Up User Flow



12. Create Post User Flow



# High level database architecture and organization

1. Database requirements
    1.1. Users shall be able to create many posts.
    1.2. A post shall be created by only one user.
    1.3. Users shall be able to create many comments.
    1.4. A comment shall belong to only one user.
    1.5. A post shall have many comments.
    1.6. A comment shall belong to only one post.
    1.7. A user shall be able to have many friends.
    1.8. A friend shall belong to only one user.
    1.9. A user shall be able to send many private messages
    1.10. A private message shall be sent to only one user.
    1.11. A post shall have many reactions.
    1.12. A reaction shall belong to only one post.
    1.13. A post shall have many many photos.
    1.14. A photo shall belong to only one post.
2. Entities and Attributes
    2.1. User: Strong
        2.1.1. userId: key, numeric
        2.1.2. username: alphanumeric, unique
        2.1.3. name: composite, multi-value, alphanumeric
        2.1.4. email: composite, alphanumeric, unique
        2.1.5. password: alphanumeric
        2.1.6. dob: multi-value, date
        2.1.7. profilePicture:
        2.1.8. lastLogin: multi-value, timestamp
        2.1.9. createdAt: multi-value, timestamp
    2.2. Posts: Weak
        2.2.1. postId: key, numeric
        2.2.2. authorId: key, numeric
        2.2.3. title: alphanumeric
        2.2.4. body: alphanumeric
        2.2.5. visibility: alphanumeric
        2.2.6. commentsAllowed: alphanumeric
        2.2.7. createdAt: multi-value, timestamp
        2.2.8. lastModified: multi-value, timestamp
    2.3. Reactions (posts): Weak
        2.3.1. reactionId: key, numeric

2.3.2. postId: key, numeric
2.3.3. userId: key, numeric
2.3.4. reactionType:alphanumeric
2.3.5. lastModified: multi-value, timestamp

2.4. Comments: Weak

2.4.1. commentId: key, numeric
2.4.2. authorId: key, numeric
2.4.3. postId: key, numeric
2.4.4. comment: alphanumeric
2.4.5. createdAt: multi-value, timestamp
2.4.6. lastModified: multi-value, timestamp

2.5. Reactions (comments): Weak

2.5.1. reactionId: key, numeric
2.5.2. commentId: key, numeric
2.5.3. userId: key, numeric
2.5.4. reactionType:alphanumeric
2.5.5. lastModified: multi-value, timestamp

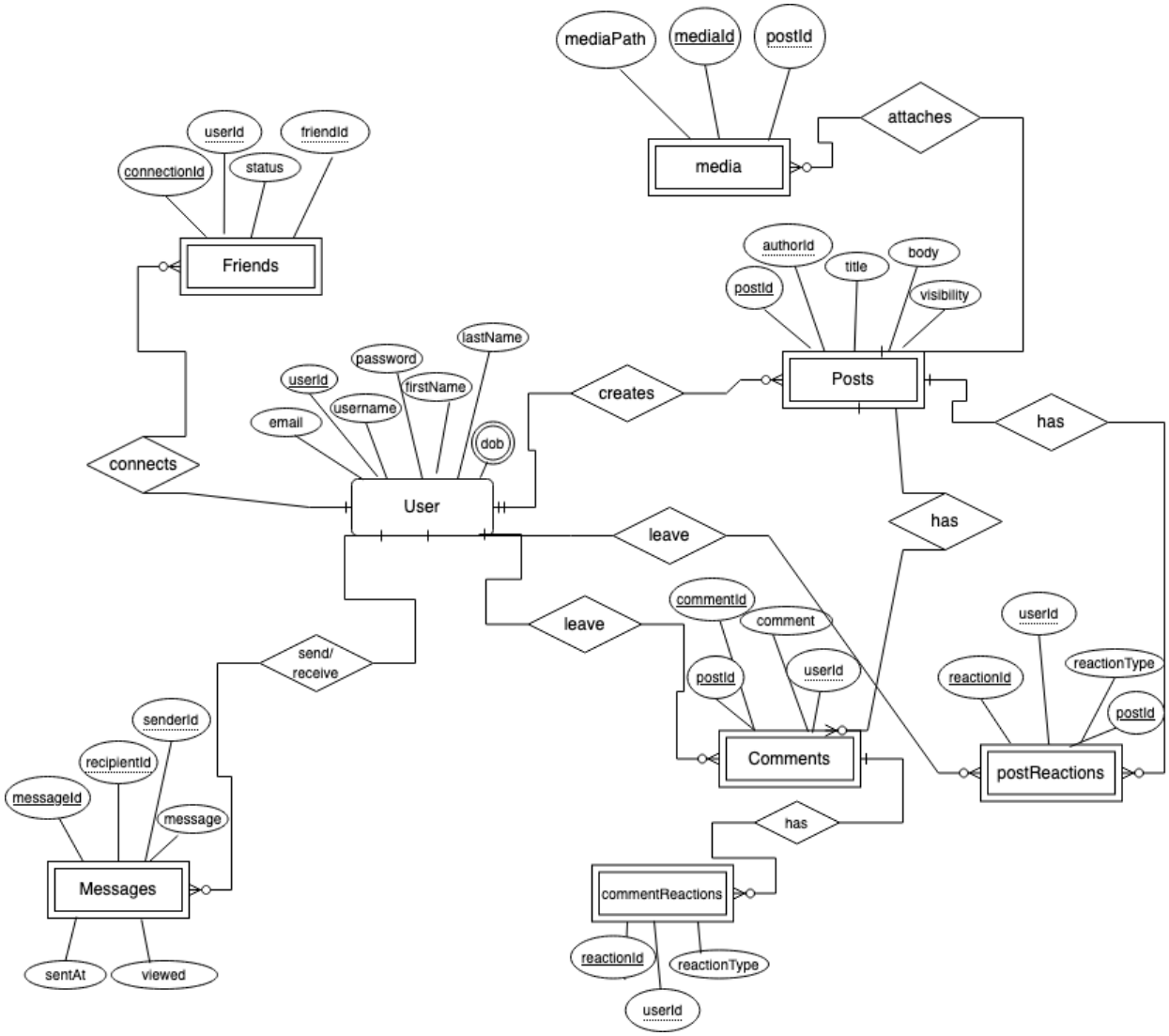
2.6. Friends: Weak

2.6.1. friendId: key, numeric
2.6.2. userId: key, numeric
2.6.3. friend: key, numeric
2.6.4. status: alphanumeric
2.6.5. createdAt: multi-value, timestamp
2.6.6. lastModified: multi-value, timestamp

2.7. Messages: Weak

2.7.1. messageId: key, numeric
2.7.2. senderId: key, numeric
2.7.3. recipientId: key, numeric
2.7.4. message: alphanumeric
2.7.5. sentAt: multi-value, timestamp
2.7.6. viewed: alphanumeric

3. Entity Relationship Diagram (ERD)




4. Define which DBMS you will choose to create the database and why:

   We will be using MySQL as our DBMS for its ease of use and flexibility for setting up/operating on data.

5. Media Storage:

   5.1. Media(images) shall be stored on the file system to reduce overhead on CRUD operations and improve scalability.

6. Search/filter architecture and implementation:

6.1. In the backend, there would be an API function, GET, to render the input from the user, the user can be a guest user or a registered user. In the search bar, the users can search up keywords, which will be retrieved from frontend and will be parsed and saved according to the context of the posts such as post ID, user ID, title, body, etc. The keywords are essentially categories, such as places like California or Spain, and restaurants like McDonald's. With the database, there will be a table that lists all public blog posts contents that are separated by specific parts of the content. Each new blog post that a registered user posts as public posts, they will be stored based on the specific category that they choose, and they can choose multiple categories for one post.

## High Level APIs and Main Algorithms

Some of our APIs will include a user login authentication function, a new user authentication function, a user information storage function, a list of blogs storage function, a search bar function, and an API function for updating passwords. The user login authentication function is a function that retrieves the login information by intaking the information that are inputted in the user interface login page then the API function authenticates the information by comparing the information stored in MySQL database to make sure the username and password matches. The key is the username and the password is the value.

**New User Authentication :**
The new user authentication function is a function that will utilize the open authorization feature to enable Google API and/or Facebook API that will be used to authenticate a new user trying to register for an account. New users will be able to create an account by connecting to either their Google account or Facebook account by routing the page to Google or Facebook API and allowing them to login. Username and email are collected from their Google/Facebook account.

**User Information Storage :**
The user information storage function is a function that stores user information, such as name, username, password, and contact, after registering for an account into MySQL database. User will input information on the register for an account page, the function will store the information into variables and insert into MySQL database.

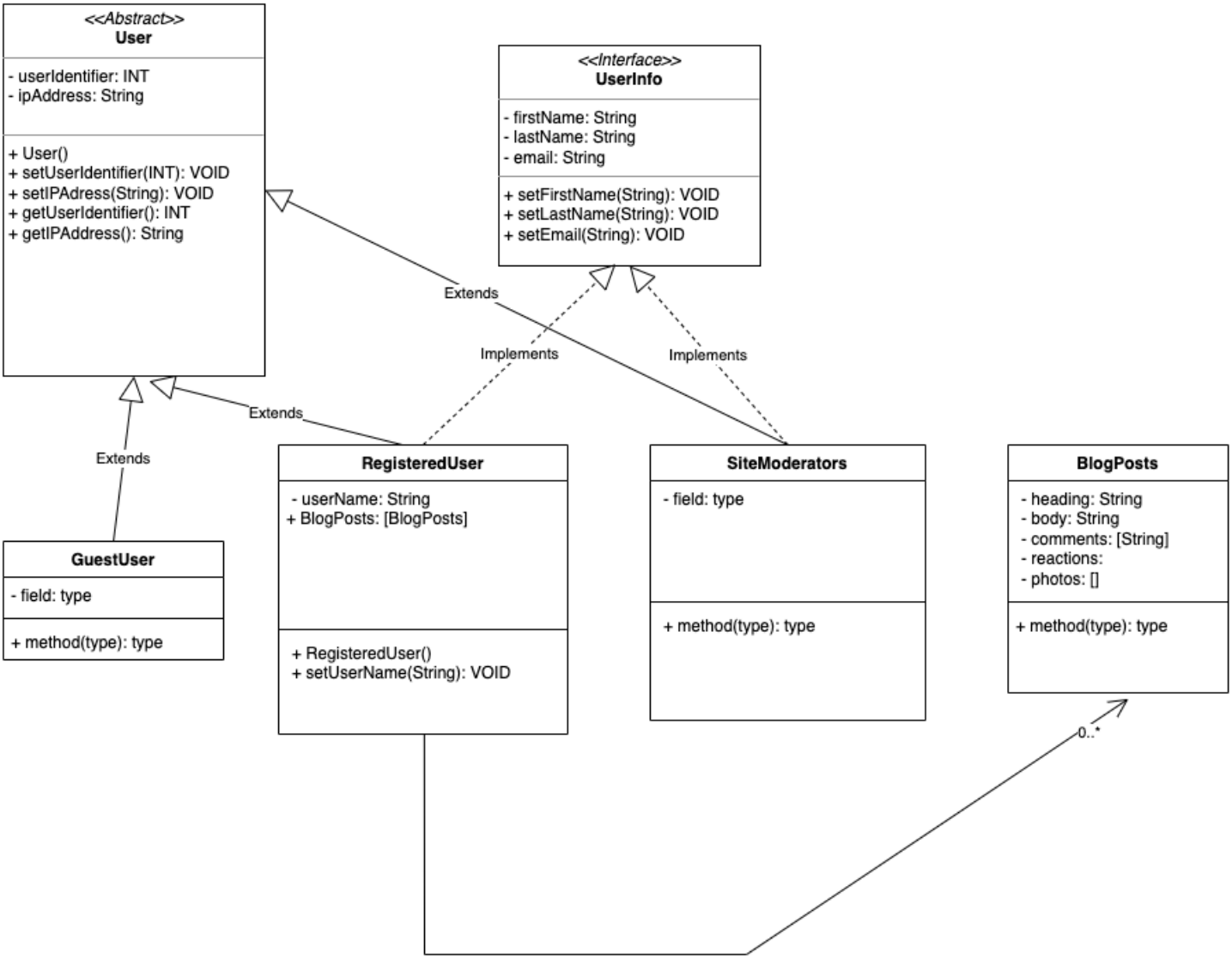
**Blog Storage :**
The list of blogs storage function is a function that stores each new blog that is posted into MySQL database. When a registered user creates a new blog post, a new page is created which essentially is a sublink to the user's page. The links to the blog post will be stored in a string array. There would be multiple hashmaps of string arrays that will store the blog links based on keywords.

**Search Function :**
The search bar function is a function that populates a list of data from MySQL database based on the specific keyword input from the user in frontend. When a specific keyword is searched in the search bar, the function will use the key to search through MySQL database and return an array of links to blog posts relating to the keyword searched.

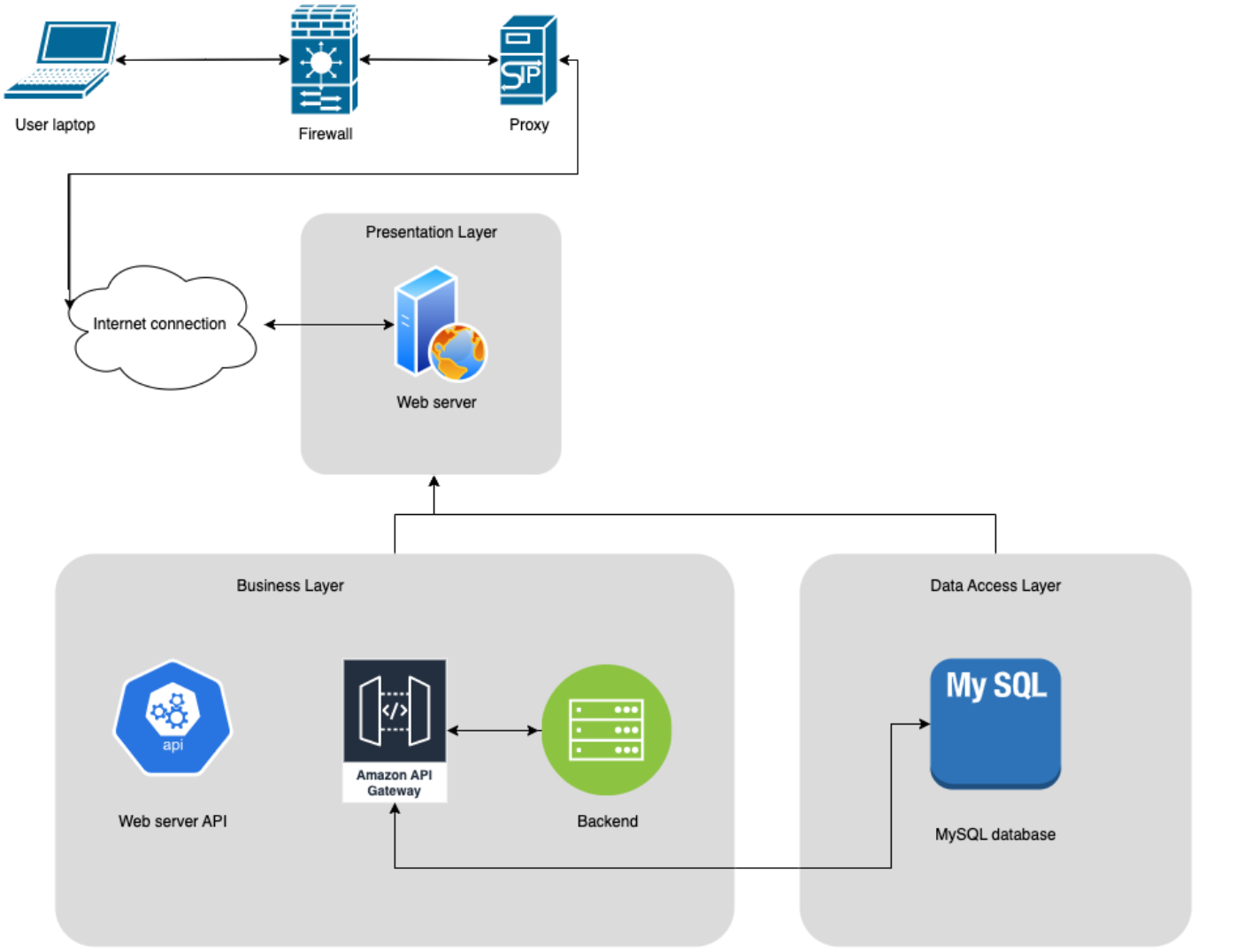
**Updating Password :**
The API function for updating passwords is a function that will allow the registered user to change their password. When the user clicks on a button to change their password, it will direct them to a page where they need to enter their current password and the new password twice in

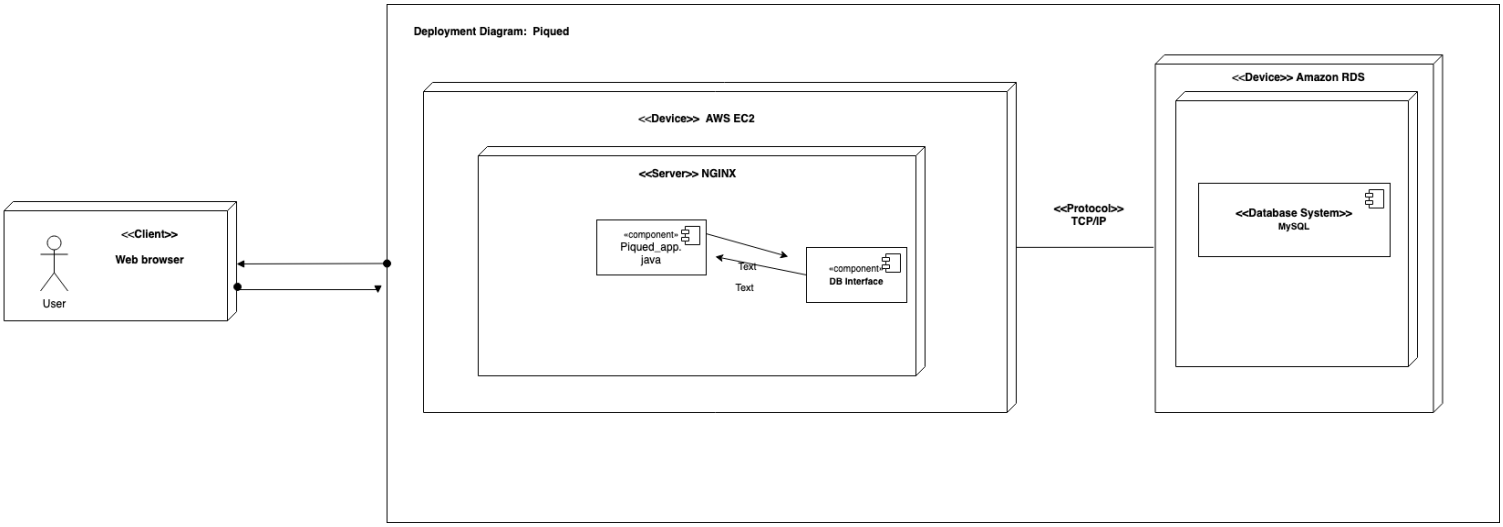
order to update it. The function will then check the current password in MySQL database by searching for the key, username, and comparing the value, password, to make sure they match. If the current password matches, it will then take the new password and store it into the database by replacing the current value, password, with the new value, new password.

# High Level UML Diagrams



# High Level Application Network and Deployment Diagrams



## Identify *actual* key risks for your project at this time

1. **Skill risks:** There is a risk that we may not have enough experience with high-level APIs, which could hinder our ability to develop certain features.

   **Solution:** To address this, we plan to conduct extensive research and learn more about the relevant APIs. We will also consider reaching out to experts in the field for guidance and advice.

2. **Schedule risks:** There is a risk that we may not be able to complete the project on time, given our commitments and available resources.

   **Solution:** To mitigate this risk, we will prioritize tasks based on their importance and assign them to team members with the appropriate skills and experience. We will also regularly monitor progress and adjust our plan as needed to ensure that we stay on track.

3. **Technical risks:** There is a risk of encountering technical issues related to our use of AWS, which could impact our ability to deploy and maintain the project.

   **Solution:** We will conduct extensive testing and research to identify and address any technical issues that arise. We will also consider consulting with experts in the field for guidance and advice.

4. **Teamwork risks:** There is a risk of issues related to collaboration, participation, task assignment, and meeting attendance, which could impact our ability to work effectively as a team.

   **Solution:** We will establish clear communication channels and expectations, resolve conflicts quickly and proactively, and hold each other accountable for meeting commitments and attending meetings.

5. **Legal/content risks:** There is a risk of encountering copyright issues related to the content and software we use in the project.

   **Solution:** We will ensure that all content and software used in the project are properly licensed and free of any copyright infringement.

# Project management

For Milestone 2, we used Trello as our primary tool for task management. Tasks were allocated based on team members' skills and represented by cards that could be moved between different lists on the board. Trello allowed us to track progress in real-time and identify bottlenecks or issues that needed addressing.

We also had frequent team meetings on zoom and in-person. Used Discord for communication and to discuss progress and challenges. For future milestones, we plan to continue using Trello, creating a new board for each milestone and adding lists for the different stages of the project.

Our approach to project management will be based on effective communication, clear task allocation, and regular progress tracking to ensure timely completion with high quality. We will continue to have frequent team meetings and use Discord as our primary communication channel, but explore other tools if needed for specific requirements.

## Detailed list of contributions

| Name | Role | Task Contribution | Scores |
|---|---|---|---|
| Andy Shi | Frontend Lead | Write data definitions. Lead group effort to decide on color palette and typefaces. Created mockups and user flows storyboards on paper. Worked on frontend homepage code and created UI components. Collaborated with the backend lead, github master, and database master to code backend and connect database with backend and frontend for vertical SW prototype. | 9 |
| Leo Saeteurn | Backend Lead | Prioritized functional requirements, high level database architecture and organization, high level APIs and main algorithms, high level UML diagrams, high level application network diagram, collaboration on logo, collaboration on UI for color and font themes. Collaborated with the frontend lead, github master, and database master to code backend and connect database with backend and frontend for vertical SW prototype. | 9 |
| Nishit Pachchigar | Github Master | Researched APIs, how to code backend to connect frontend and MySQL database. Collaborated with the backend lead, frontend lead, and database master to code backend and connect database with backend and frontend for vertical SW prototype. Worked on AWS and Trello project management. Lead the structure of GitHub branches and folders. | 9 |
| Joshua Hayes | Database Master | Prioritized functional requirements, high level database architecture and organization, deployment diagram. Created database from diagram for backend virtual prototype. Collaborated with the backend lead, github master, and frontend lead to code backend and connect database with | 8 |

| | | | |
|---|---|---|---|
| | | backend and frontend for vertical SW prototype. | |
| Gautami Kollolu Srinivasa | Document Editor | Identified actual key risks and solutions for the project. Wrote project management and data definitions. Collaboration on UI for color and font themes. Proofread the complete milestone 2 document to meet the requirement. Collaborated with the front end team for the vertical SW prototype. | 5 |