RECIPE REMIX

CS 30700
Design Document

Team 32:
Michael Li, Ryan Leonard, Khoa Raisr, Andree Kolliegbo, Anh Nguyen, Tim Chou

INDEX:

Purpose:

In the fast-paced world of today, many individuals, especially busy ones such as college students, find themselves trapped in a cycle of consuming unhealthy fast food or expensive takeout, primarily due to constraints of time and budget. Many desire to cook healthy meals at home, but the challenge of lacking certain ingredients or not knowing what to prepare with what's available becomes a deterrent. Our project offers a platform for users to receive customized recipe suggestions based on the ingredients they have and dietary limitations, with an added emphasis on customization and community contribution.

Numerous recipe platforms, such as SuperCook, MyFridgeFood, and BigOven, have ventured into ingredient-based recipe recommendations. While they have this main feature integrated into their functionality, their feature restricts users to predefined ingredient selections, limiting the versatility and spontaneity that real-life cooking often demands. Dietary restrictions are also not accounted for in these applications. Moreover, these platforms lack a contemporary, user-friendly interface and the integration of a robust community feature.

We aim to create an application that stands out with an intuitive modern UI, expansive ingredient flexibility, engaging community features, customizable dietary preferences, and a commitment to cost-effective, time-efficient recipes tailored to today's diverse culinary needs.

Functional Requirements:

<u>User Authentication & Profile</u>

1. As a user, I want to register for an account on Recipe Remix using email.
2. As a user, I would like to be able to register for a Recipe Remix account using Facebook.
3. As a user, I would like to be able to register for a Recipe Remix account using Google.
4. As a user, I want to log into my Recipe Remix account.
5. As a user, I want to be sent a verification email when I log in for security purposes.
6. As a user, I want to be sent a link to verify my email after I register for an account
7. As a user, if I forget my password, I want to reset it.
8. As a user, I want to personalize my profile with a nickname and a biography description.
9. As a user, I would like to choose my profile picture from my gallery or take a profile picture from my camera.
10. As a user, I would like to edit or clear my profile information.
11. As a user, I would like to delete my account.

12. As a user, I want to toggle between light and dark mode.
13. As a user, I want to manage my dietary restrictions and preferences in my profile.
14. As a user, I want to manage my favorite cuisines in my profile.
15. As a user, I want to view my past ingredient searches.
16. As a user, I want to create customizable recipe collections that I can save to my profile with the option of making it private or public (if time allows).

Ingredient Input & Recipe Suggestions

17. As a user, I want to drag and drop ingredient icons into a virtual cauldron to perform the "recipe remix" (if time allows).
18. As a user, I would like to download a PDF of a recipe (if time allows).
19. As a user, I want to "remix all" ingredients in my pantry at once.
20. As a user, I want to select certain pantry items to remix.
21. As a user, I want recipe suggestions when I remix the inputted ingredients.
22. As a user, I want to view the full details of a recipe suggestion.
23. As a user, I want to adjust recipes based on my available ingredients.
24. As a user, I want to view which ingredients I am missing for certain recipes.
25. As a user, I want to tap on a missing ingredient and get more info about it (price, stores).
26. As a user, I want to delete my saved recipes.
27. As a user, I want to save my remixed recipes to my profile.
28. As a user, I want to be able to edit my saved recipes.
29. As a user, I would like to search for ingredients and add them to my pantry.
30. As a user, I want to filter my search results by diet, food group, allergies, dairy, gluten.
31. As a user, I want to see suggested ingredients auto-populate when I am searching.
32. As a user, I would like to be able to search for a recipe from the saved recipes.
33. As a user, I would like to be able to filter recipes by serving size and prep time

Community Features

34. As a user, I want to share my recipes with the community.
35. As a user, I want to rate other users' recipes with stars
36. As a user, I want to be able to comment with predetermined comments, gifs, and emojis
37. As a user, I want to search for recipes based on popularity, ratings, and recentness.
38. As a user, I want to be able to share my recipes to other social media platforms.
39. As a user, I want to be able to follow other users.
40. As a user, I want to be able to view, edit, and block other users from my following list
41. As a user, I want to be able to rate my remixed recipe.
42. As a user, I want to specify the privacy level of my pantry and recipes.
43. As a user, I would like to upload a video or pictures of the recipe I made

44. As a user, I would like to be able to view recipes by [hash]tags
45. As a user, I would like to have animated reactions to posts, like “Boiler Up!”, (if time allows).
46. As a user, I would like to add a story feature and highlights feature (15 second clips or pictures if time allows)

<u>Shopping Integration</u>

47. As a user, I want to generate a shopping list based on a selected recipe.
48. As a user, I want to edit or delete my shopping list
49. As a user, I want to categorize the items on my list by department.
50. As a user, I want to be able to share my location within the platform
51. As a user, I want to set the duration for which my location will be shared.
52. As a user, I want to receive location-sharing requests and approve or decline them.
53. As a user, I want to see a list of local grocery stores based on my location using maps
54. As a user, I want suggestions of where to buy certain items on my list
55. As a user, I want to be able to specify local grocery stores I want to order from in ‘My Stores’
56. As a user, I want to view a map showing nearby grocery stores based on my current location.
57. As a user, I want to filter grocery stores by criteria such as ratings, distance, and store type.
58. As a user, I want to get detailed information about a selected grocery store (e.g., address, hours, contact info).
59. As a user, I want to get directions to a selected grocery store.
60. As a user, I want the platform to provide external links to the websites of selected grocery stores for online ordering.
61. As a user, I want to track the status of my online orders, including delivery or pickup information.
62. As a user, I want to receive personalized recommendations and deals from my preferred stores (if time allows).
63. As a user, I want to set a default store for quick access.

<u>Miscellaneous</u>

64. As a user, I want to set reminders for when I would like to eat throughout the day
65. As a user, I want to be able to report bugs.

Non-Functional Requirements:

Architecture and Performance:
We will be using the MERN stack for our project. Our frontend will be written in React.js. The backend will be hosted on a MongoDB NoSQL server and will use Node.JS and Express to mediate between the frontend and backend. Authentication will be implement with MongoDB as well. We will access Recipe database APIs in order to retrieve search results for users who want to search for recipes and ingredients.

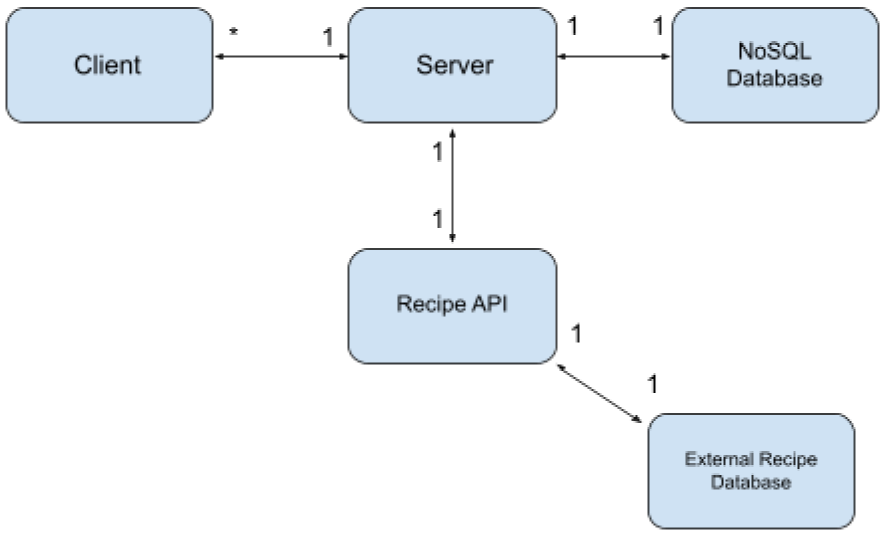
Security:
To ensure secure profiles, users will be required to meet password requirements. It must be a length with at least 12 characters, one special character, and one number. The username must meet a length of at least 8 characters and must be unique for every user in the database. We will also be storing the recipes of each user using MongoDB database in order to keep each user's recipes secure and separate. Users will only be allowed access to the recipes of other users which have been made public, so permissions will have to be specified. Recipe Remix will not facilitate direct communication between users, thus eliminating the hassle of message encryption.

Usability:
This platform is intended to be user friendly and used by a variety of people: parents, college students, ect. The app is supposed to be user friendly, support exciting graphics with the recipe remixing, allowing for dragging and dropping of pantry items, and easily accessible filters to customize recipes.

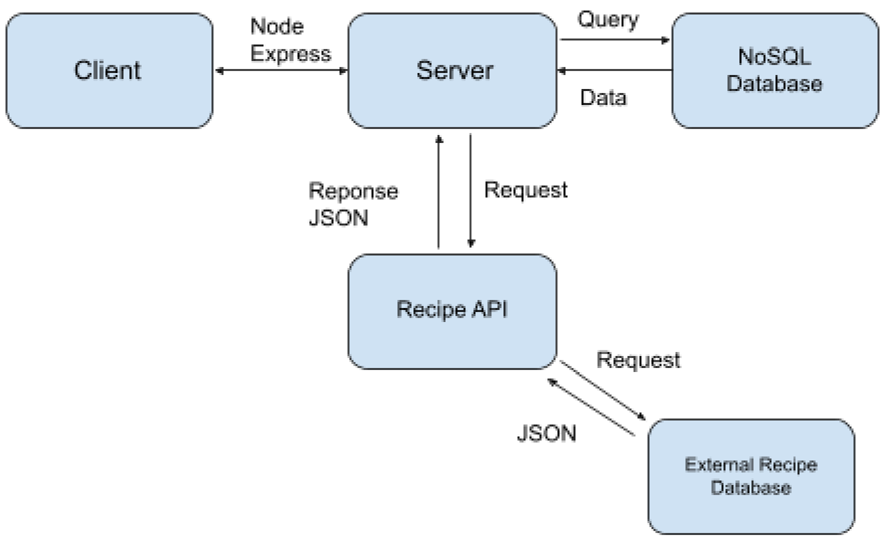
Hosting/Deployment:
Recipe Remix will leverage MongoDB for its hosting and database needs. MongoDB, a leading NoSQL database, offers flexibility with its schema-less architecture, making it an ideal choice for applications that require rapid iterations and adaptability.

Design Outline:

High Level Overview:



Detailed Overview



Recipe Remix will deploy a system that will allow the client to communicate with the server via the express web framework. The server will be a node js web server and will use mongoose to interact with the NoSQL database. The web server will also communicate with an external recipe database via an API which will send requests for various recipes depending on user input and receive those recipes in the form of JSON which will be parsed and converted into objects that are saved to our NoSQL database.

Design Issues:

Functional:

1. How to let users create/edit their own recipes?
    a. Option 1: copy recipe from database into new recipe
    b. Option 2: edit recipe from database

Choice: Option 1

Justification: We went with option one thinking that users would have more control over the recipe if they were to simply load it into a new object. This would also expand the database to include the new user recipes within a streamlined format.

2. What information is required for creating an account?
    a. Option 1 - Name, Email, Password
    b. Option 2 - Name, Username, Email, Password
    c. Option 3 - Name, Username, Email, Password, Phone Number

Choice: Option 2

Justification: We decided to go with a username system, so that users can find their friends through a chosen, unique username in our system (implemented if time allows) which is more efficient than running into many users with the same Name field. We will not require the user's phone number as we do not plan to make any calls, send any texts or enable two-factor authentication.

3. How would we implement the shopping list feature?
    a. Option 1: A shopping cart with local prices
    b. Option 2: Adding to a list of desired items within the pantry
    c. Option 3: Links to external grocery websites

Choice: Option 2

Justification: Ultimately, option 2 was the one that went best with what we feel like we can achieve within this timeframe. Giving users the ability to add items into their cart using either a button or drag and drop items if time allows. We would rather focus on implementation of other features rather than focusing on integrating prices and grocery API's within the website, which may not even be available.

4. What if a user tries to add an ingredient that isn't food?
    a. Option 1: Check the 'new ingredient' object against our current list of ingredients,
    b. Option 2: Display a preset list of ingredients that users can filter based on broader categories

Choice: Option 2

Justification: We chose Option 2 because it provides a more user-friendly and error-preventing approach. Displaying a preset list of ingredients that users can filter based on broader categories helps maintain data integrity and user experience.

Non-Functional:

1. What platform is most appropriate for our database and login services?
    a. Option 1: MongoDB
    b. Option 2: Google Firebase

Choice: Option 1

Justification: We opted for MongoDB for our database and login services because of its dynamic schema flexibility and ability in handling large datasets. MongoDB's built-in authentication mechanisms ensure secure user registration and login processes.

2. How would the AI we use interact with our recipe database and API?
    a. Option 1: Filter recipes from the database based on our client pantry items
    b. Option 2: Provide suggestions for ingredients to complete recipes
    c. Option 3: Analyze user behavior to offer personalized recipe recommendations

Choice: Option 1

Justification: We opted for Option 1 as it aligns with our primary goal of assisting users in finding recipes based on their available ingredients. By having the AI filter recipes from the database according to the contents of the user's pantry, we can provide immediate and relevant recipe suggestions, enhancing the user experience.

3. How do we plan on finding specific recipes based on the list of ingredients the user has imputed into the cauldron?
    a. Option 1: Implement a search algorithm that matches input ingredients with recipe ingredients.

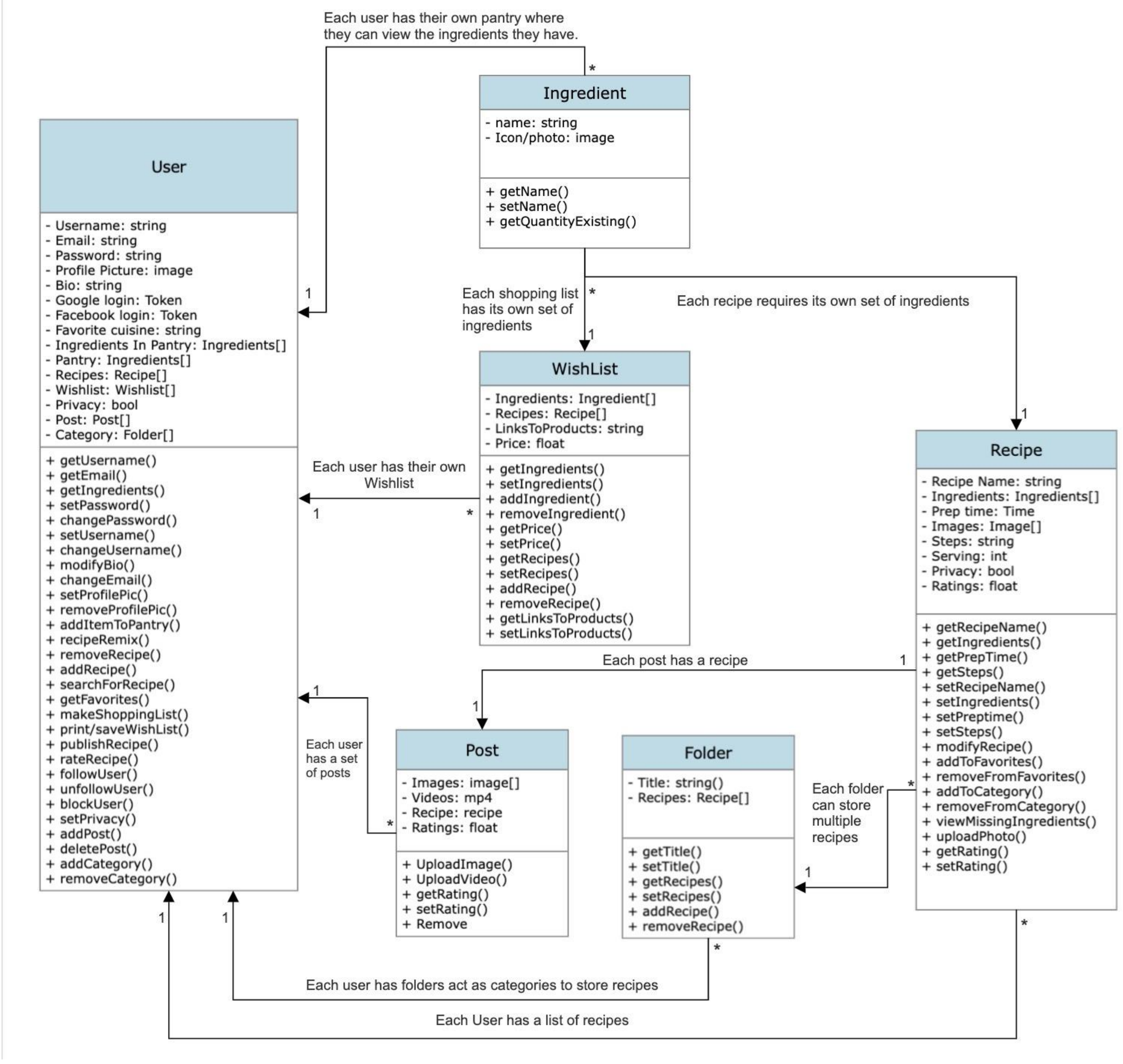
b. Option 2: Utilize machine learning to predict and recommend recipes based on user preferences.

Choice: Option 1

Justification: We selected Option 1 as it offers a practical and efficient way to find recipes based on user input. By implementing a search algorithm that matches input ingredients with those in our database, we can quickly present users with a list of recipes that best match their available ingredients, ensuring a seamless and user-friendly experience.

Design Details:

Class Design:



Description of Classes and Interaction between Classes

Based on the stories we created to produce our application’s features, the above classes were created. We have also ensured that they are designed modularly and independently, thus having their own attributes

1. **User**
    a. A user object is created when someone signs up with their email or Google account or Facebook account.
    b. Each user will have a username, password, and email for login purposes.
    c. The user can choose to fill out profile pictures as well as bio fields in their profile page.
    d. At any time, the user can modify any details on their profile page.
    e. Once signed up, the user will be able to select ingredients to add to their virtual pantry. This process can be done later.
    f. Then, the user is taken to the home page where they can perform a recipe remix with the ingredients from their pantry, search for a recipe, view their pantry.
    g. The user will also be able to navigate to the community page where they can share their recipes with other users, as well as rate other users’ recipes.
    h. The user will also be able to modify their wishlist at any time.

2. **Recipe**
    a. A recipe object is created when a user performs a recipe remix, or creates their own recipe.
    b. A list of recipes will be “pulled” from our database once a user performs a recipe remix.
    c. For a recipe that the user create, they would have to input:
        i. A name
        ii. A list of ingredients
        iii. The number of steps
        iv. Preparation time
        v. The amount of servings
        vi. Images (optional)
        vii. A tag if they want to categorize that recipe
    d. When a user creates a recipe or decides to modify a recipe from the remix, it will be added to that user’s recipes list.
    e. The user can modify any part of the recipe or totally delete it at any time.

3. **Ingredients**
    a. An Ingredient object will be created once a user inputs the recipe into their pantry or for their recipes.
    b. For an Ingredient, the user has to input:
        i. Name
        ii. Image
    c. The user can add an Ingredient or modify the old one, resulting in changes in their pantry.

4. **Folder**
    a. A Folder object will be created when a user clicks a new category button.
    b. For a Folder, the user will input:
        i. The name of the category
        ii. A list of recipes that belong to that category.
    c. The user can add new recipes to the folder
    d. The user can remove recipes from the folder
    e. The user can change the name of the folder
    f. The user can delete the folder.

5. **Wishlist**
    a. A WishList object is created when a user clicks add to Wishlist button.
    b. For a Wishlist, the user will input:
        i. A list of ingredients or an ingredient.
        ii. A list of recipes or a recipe.
    c. The user can add new recipes to the Wishlist.
    d. The user can remove a recipe or the whole recipe list from the wishlist.
    e. The user can add an ingredient to the wishlist.
    f. The user can remove one ingredient or the whole ingredients list from the wishlist.
    g. The user can click the link to products to be redirected to a grocery store website.

6. **Post**
    a. A Post object will be created when the user decides to post their recipes.
    b. For a Post, the user will input:
        i. Images or videos
        ii. Recipe
        iii. Caption

c. The user can Upload a video and/or a list of pictures along with a recipe at any time.
d. The user can delete the post at any time.

Sequence of Events Overview

The sequence diagrams detail the interactions between the user, client application, backend server, and the database. The process starts when a user creates an account or logs in. The client application handles this data and sends the request to the backend server. The server then creates a query to get or save the user's details from the database.

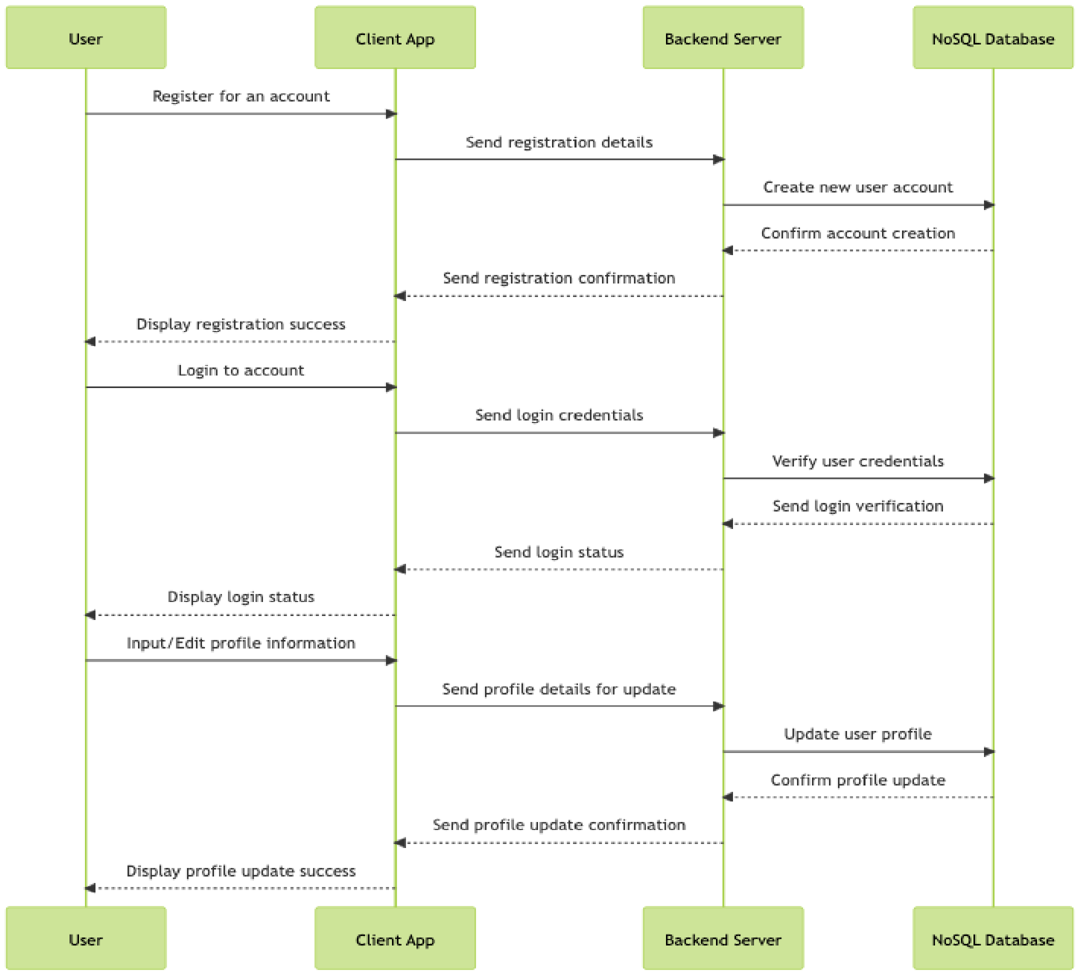
Once logged in, the user can create or edit folders, add or change recipes, add ingredients to their pantry, interact with community features, and create wishlists. Every action taken by the user is first processed by the client application. This information is then sent to the backend server, which makes a query to add new data or update existing data in the database.

If the client application needs to get information, it sends a request to the server. The server then asks the database for the data and sends it back to the client application. After each successful action, a confirmation message is shown to the user.

Sequence Diagram for Inserting and Updating User Information:

Upon launching the Recipe Remix app, users are presented with two options: either to register for a new Recipe Remix account or to sign in using existing credentials. If a user opts to register, the client application captures the necessary details and forwards a request to the backend server to store this new user data. The server, in turn, communicates with the database to save these details.

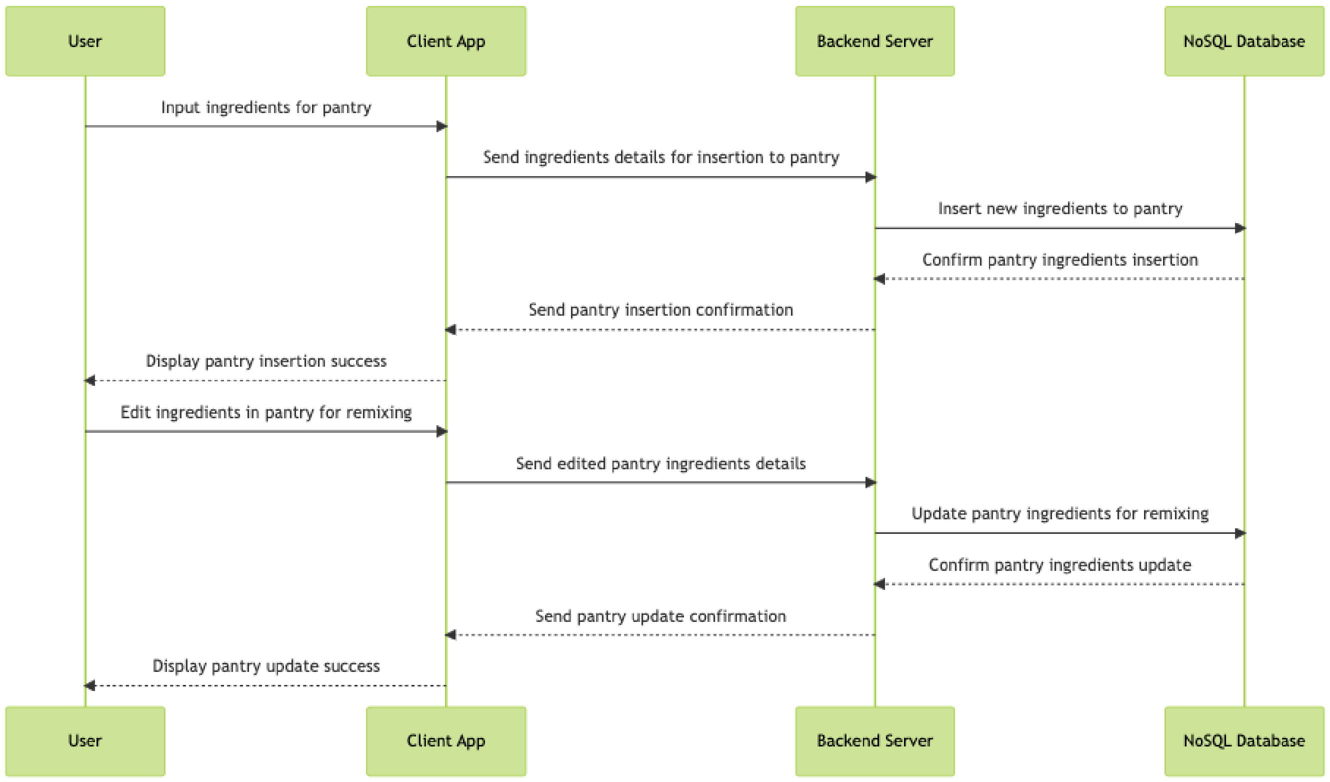
Post-registration, users have the flexibility to edit their profiles. Should a user decide to make changes to their profile, the client application processes these edits and sends an update request to the backend server. The server then interacts with the database to modify the existing user data as per the new inputs. Once the database successfully updates the information, it sends a confirmation back to the server, which is then relayed to the client application, notifying the user of the successful update.



Sequence Diagram for Inserting and Updating Pantry of Ingredients:

When a user accesses the Recipe Remix app and navigates to the pantry section, they have the option to add new ingredients to their virtual pantry or update existing ones. If the user decides to add a new ingredient, the client application captures the ingredient details and sends a request to the backend server to store this new ingredient data. The server then communicates with the database to save these details in the user's pantry. As users continue to use the application, they might need to make adjustments to their pantry, such as updating the quantity of an ingredient or removing an item.

When such changes are made, the client application processes the updates and sends a request to the backend server to modify the pantry data accordingly. The server then interacts with the database to update the pantry ingredients based on the user's modifications. Once the updates are successfully processed, the database sends a confirmation back to the server. This confirmation is then relayed to the client application, informing the user that their pantry has been successfully updated.

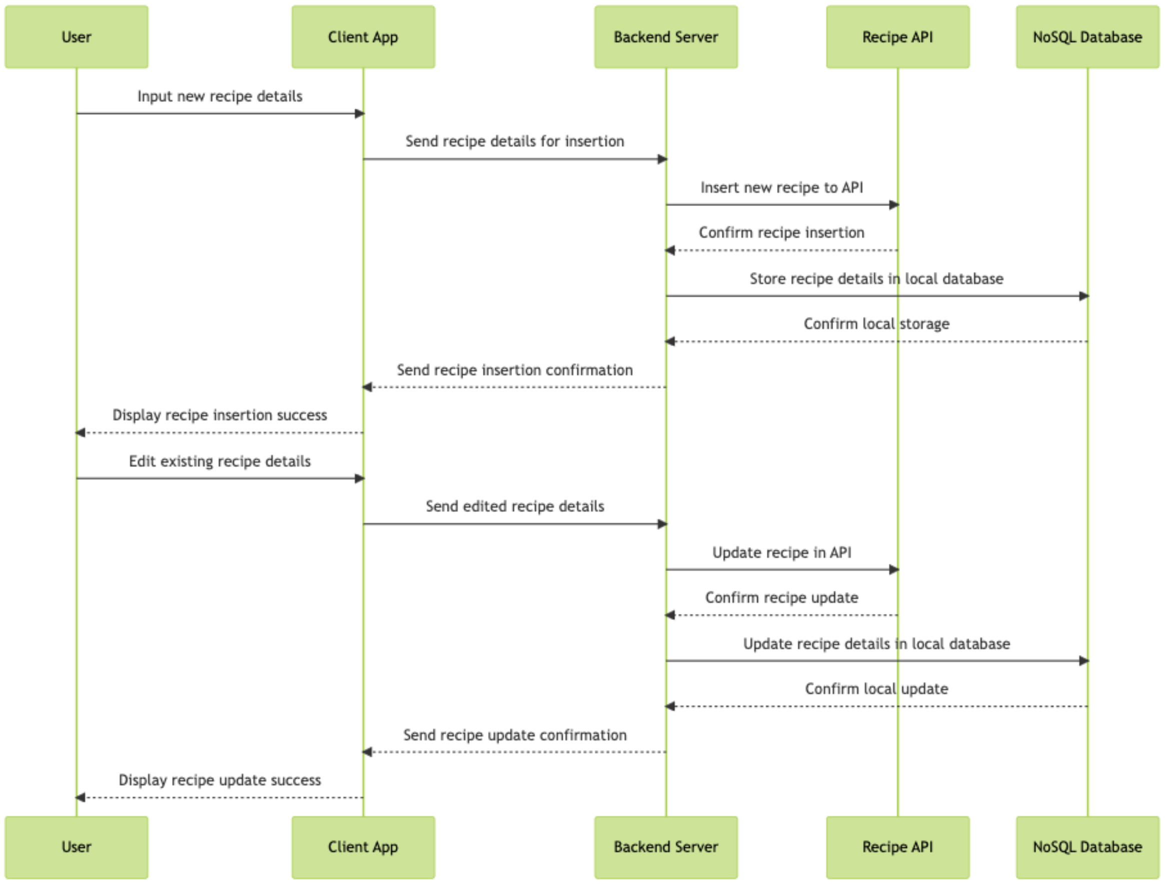

User
Client App
Backend Server
NoSQL Database
Input ingredients for pantry
Send ingredients details for insertion to pantry
Insert new ingredients to pantry
Confirm pantry ingredients insertion
Send pantry insertion confirmation
Display pantry insertion success
Edit ingredients in pantry for remixing
Send edited pantry ingredients details
Update pantry ingredients for remixing
Confirm pantry ingredients update
Send pantry update confirmation
Display pantry update success
User
Client App
Backend Server
NoSQL Database

Sequence Diagram for Inserting and Updating Recipes:

Upon submitting the new recipe or changes, the client application captures this data and sends a request to the backend server to either store the new recipe or update the existing one. If Recipe Remix leverages the recipe API or another external recipe database, the server communicates with this external API to fetch or update the recipe details. This ensures that the Recipe Remix database remains synchronized with the external recipe source.

Once the server processes the request and communicates with the external API (if applicable), it then interacts with the internal database to save or update the recipe details. After the database successfully stores or modifies the recipe, it sends a confirmation back to the server.

The server, upon receiving this confirmation, processes any necessary logic (like updating user contribution points or notifying followers of a new recipe) and then sends a success message back to the client application. The user is then informed that their recipe has been successfully added or updated, and they can view or share it within the Recipe Remix community.

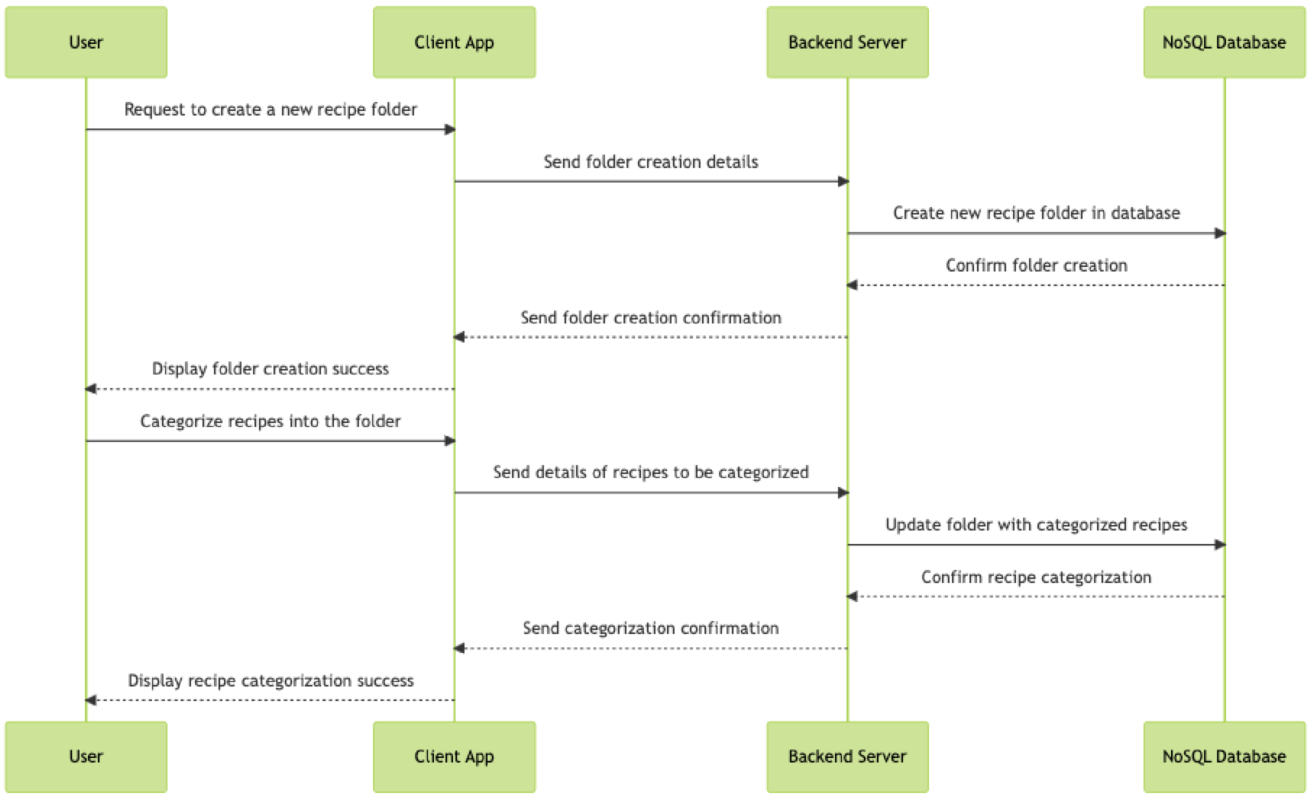

User
Client App
Backend Server
Recipe API
NoSQL Database
Input new recipe details
Send recipe details for insertion
Insert new recipe to API
Confirm recipe insertion
Store recipe details in local database
Confirm local storage
Send recipe insertion confirmation
Display recipe insertion success
Edit existing recipe details
Send edited recipe details
Update recipe in API
Confirm recipe update
Update recipe details in local database
Confirm local update
Send recipe update confirmation
Display recipe update success
User
Client App
Backend Server
Recipe API
NoSQL Database

Sequence Diagram for Inserting and Updating Folders:

When users wish to organize their recipes, they might opt to create or modify folders within the Recipe Remix app. This organizational feature allows users to categorize their recipes based on cuisine, occasion, dietary preference, or any other criteria they deem fit. To initiate the process, users access the folder management interface within the application. Here, they can choose to create a new folder, providing it with a name and optional description, or select an existing folder to update its details.

Upon confirming their choice, the client captures the folder details and sends a request to the backend server. The server then communicates with the database to either insert the new folder or modify the existing one. After successful processing, the database sends a confirmation to the server.

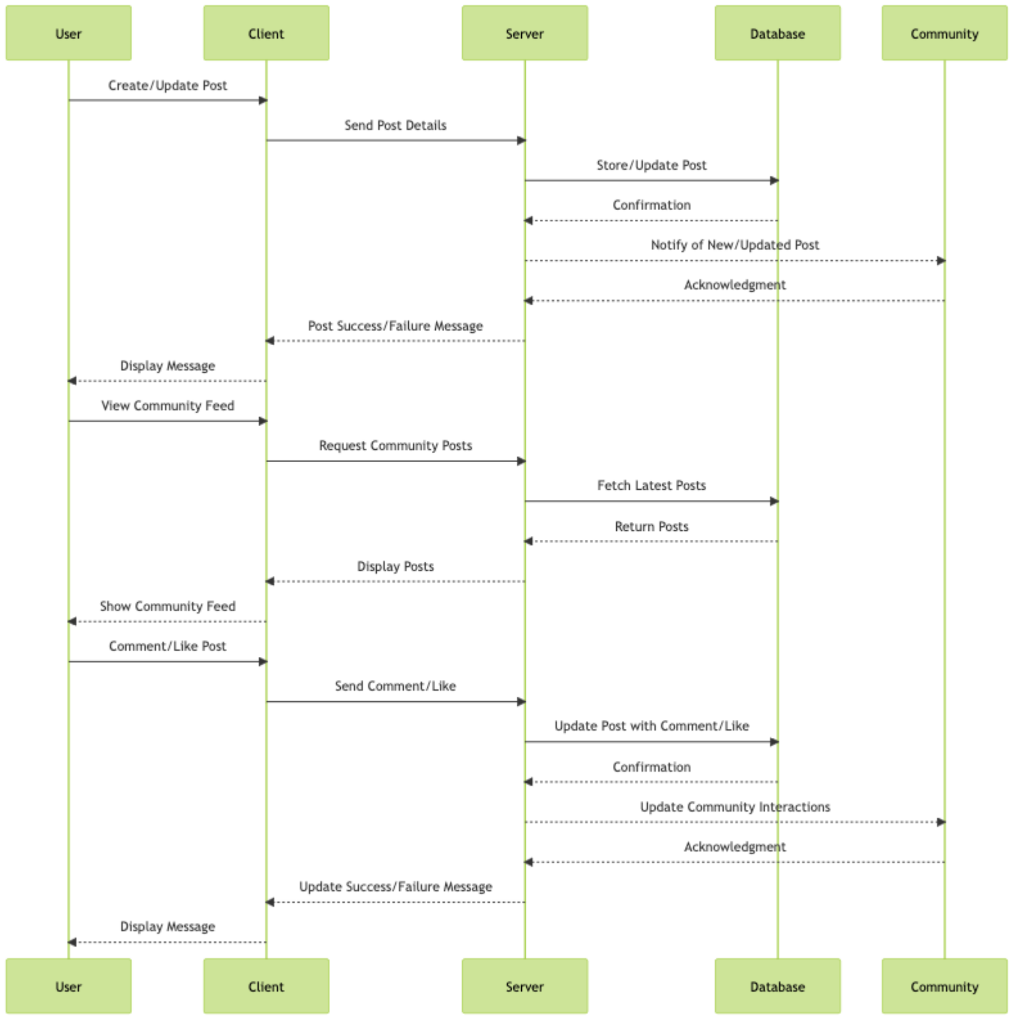
The server, after any additional processing, relays a success message back to the client. The user is then informed of their successful folder creation or update, allowing for a streamlined recipe organization experience.


User
Client App
Backend Server
NoSQL Database
Request to create a new recipe folder
Send folder creation details
Create new recipe folder in database
Confirm folder creation
Send folder creation confirmation
Display folder creation success
Categorize recipes into the folder
Send details of recipes to be categorized
Update folder with categorized recipes
Confirm recipe categorization
Send categorization confirmation
Display recipe categorization success
User
Client App
Backend Server
NoSQL Database

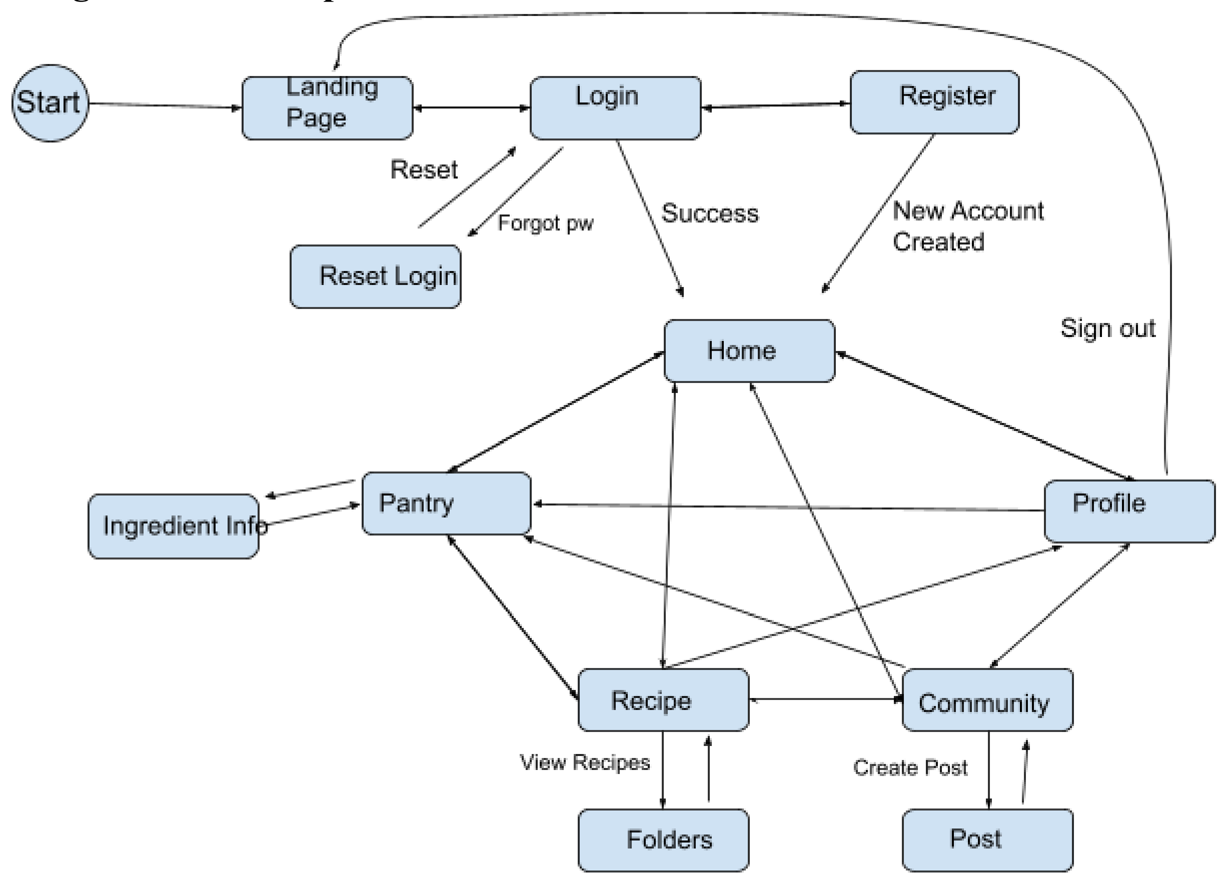
Sequence Diagram for Community Features:

In the Recipe Remix app, users can create or edit posts. After inputting details, the client sends this data to the server. The server, in turn, notifies the community module about the post, ensuring followers or interested users are informed. The server then communicates with the database to save or update the post.

Once stored, the database confirms the action to the server. The server processes any additional logic, like updating user points or sending notifications, and then relays a success message to the client. The user is then notified of their successful post addition or update, ready for community engagement.

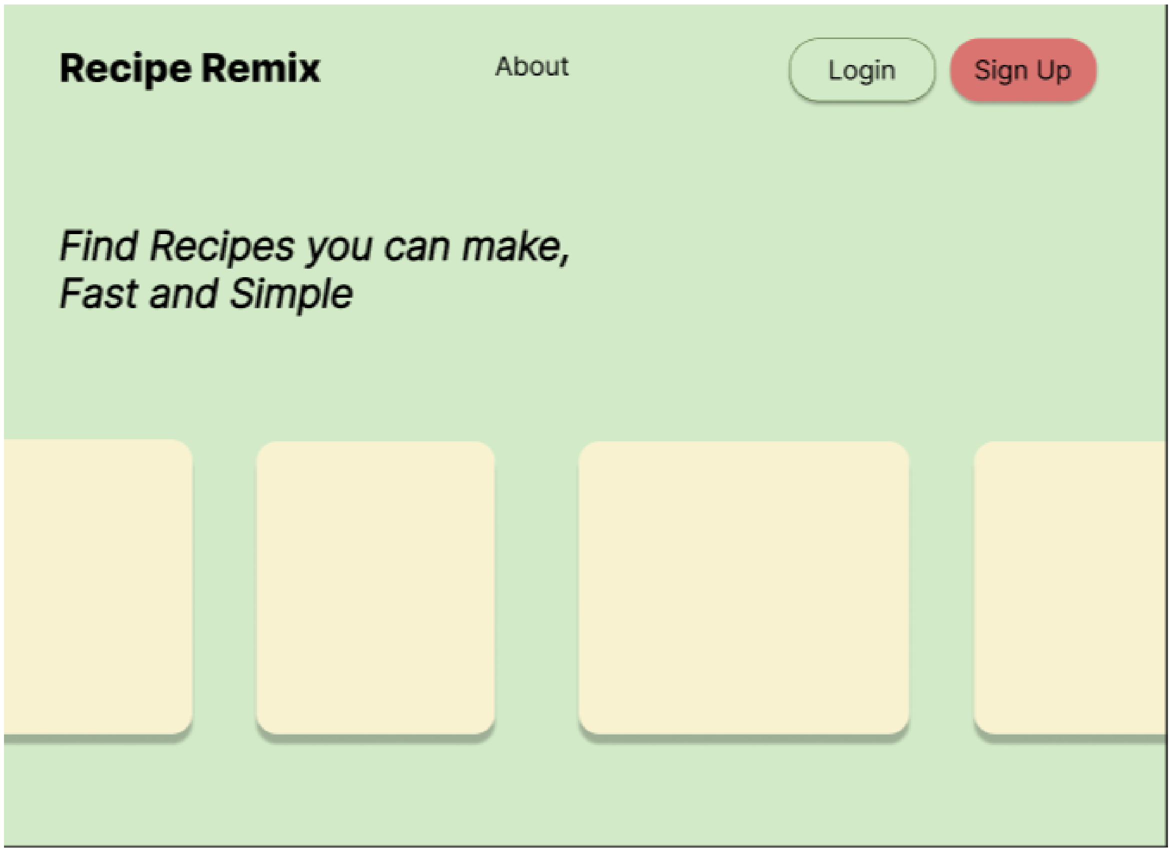


Navigation Flow Map:

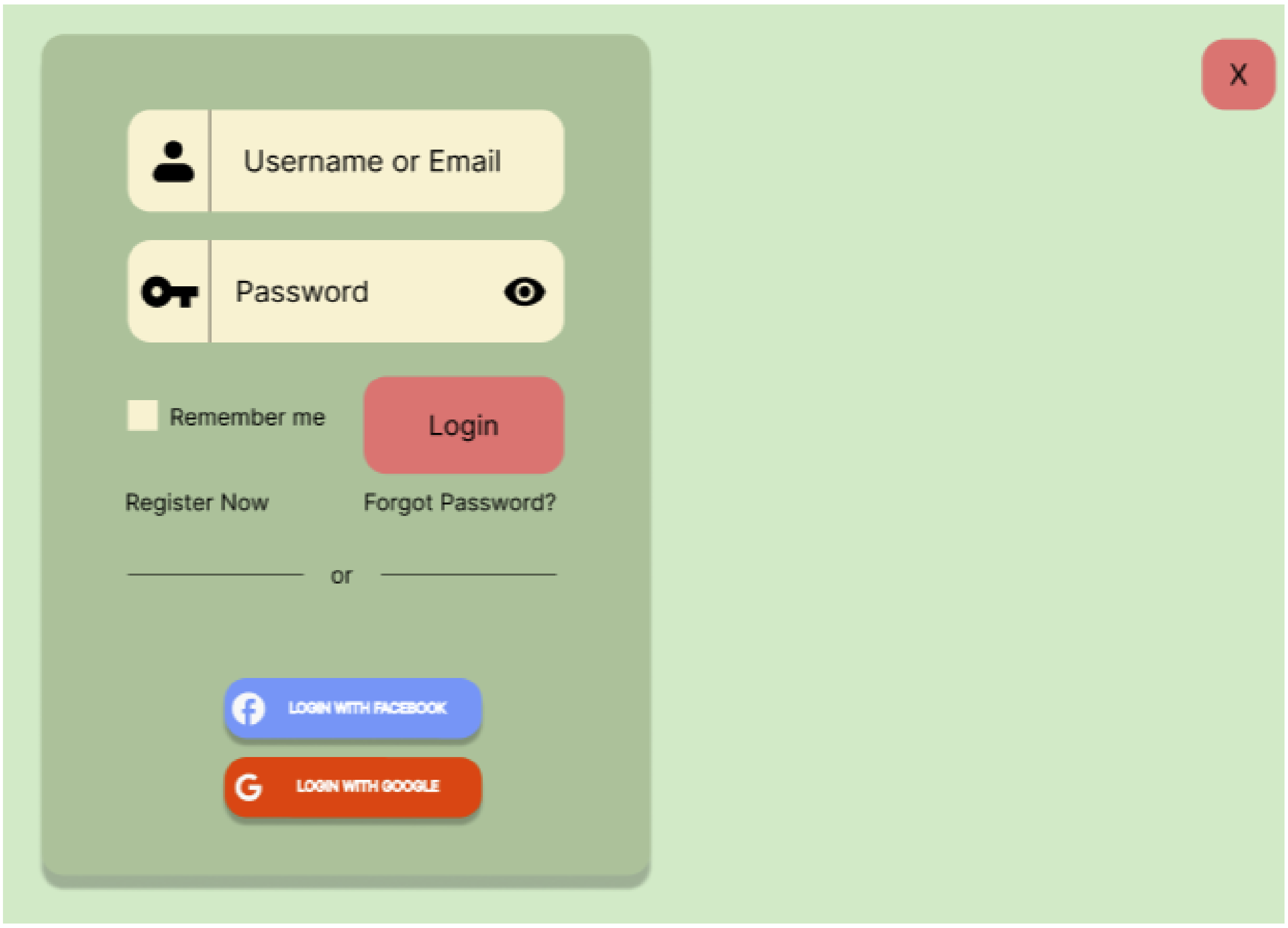


UI Mockups:

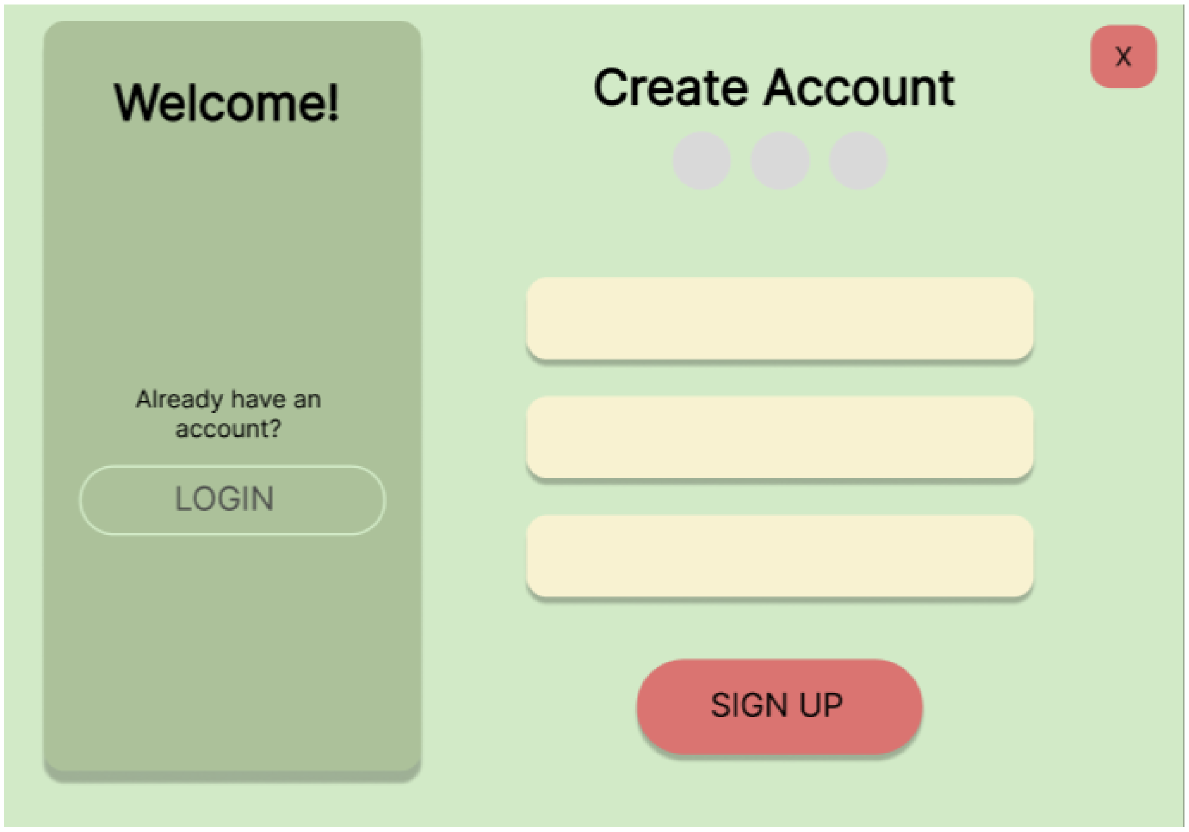
Landing Page:



Login Page:


X
Username or Email
Password
Remember me
Login
Register Now
Forgot Password?
or
LOGIN WITH FACEBOOK
LOGIN WITH GOOGLE

Register Page:


Welcome!
Already have an account?
LOGIN
Create Account
X
SIGN UP