CS 307 Design Document

Project Name: Capsule

Team 12

- Nathan Schneider
- Kevin Jones
- Evan Zimmerman
- PJ Henwood

Index

Purpose

Problem Statement

Our project is a time capsule mobile application. An app that users can download and then receive monthly snapshots (called capsules) of their recent memories; from pictures to music, this immersive experience allows users to reminisce on the positive aspects of their month. The magic of our application is its automation: unlike other apps where the user is forced to spend the time creating their own monthly snapshot, our app gets downloaded and forgotten about... until the end of the month, when the user is surprised with a captivating, personalized visual and acoustic masterpiece!

As the world around us continues to move at an increasingly fast pace, time seems to fly by and good memories fall to the backs of our minds. This is the problem we hope to solve. People don't have enough time to scroll through all their history of the past month and organize it to find their favorite moments. Capsule automates this process so that users can focus on their other day to day tasks and still enjoy a recap of their best memories.

We also seek to create a platform where users are able to share their Capsules with their friends and reminisce on good memories together. By allowing users to comment and react to their friend's Capsules, we provide a platform in which a community wishing to be reminded of their best moments from the month can come together and positively impact one another. Our app combines automation and ease with an uplifting social platform, revolutionizing how people share past experiences together.

Functional Requirements

Login/Sign up

1. As a user, I would like to be able to log in/sign up via my Google account, so that the log in/sign up process goes faster.
2. As a user, I would like to be able to create a username during registration, so that I can identify myself on the platform.
3. As a user, I would like to be able to pick a profile picture during registration, so that I can better identify myself on the platform.
4. As a user, I would like to be able to have the option to link/not link my spotify account during registration, so that music can be part of my capsule account.
5. As a user, I would like to be able to have the option to link/not link my instagram account during registration, so that recent instagram photos can be part of my capsule account.
6. As a user, I would like to be able to delete my account, so that I can remove my information from the platform.

Profile, Friends, and History

7. As a user, I would like there to be a profile page unique to my account, so that I have a central hub for all account information.
8. As a user, I would like to be able to toggle between light and dark mode so that I can view the app differently depending on my preference.
9. As a user, I would like to be able to link my spotify, link my instagram, and change my profile picture, on the ‘my profile’ page, so that I can update any information linked to my account.
10. As a user, I would like there to be a friends page with all information relevant to friends quickly accessible, so that I can complete any actions related to friends.
11. As a user, I would like to be able to invite friends by username, so that we can see each other's capsules.
12. As a user, I would like to be able to view who has requested me as a friend in the friends page, so that I can visualize who wants to be my friend.
13. As a user, I would like to be able to accept or reject a friend request in the friends page, so that I can manage who can view my capsule.
14. As a user, I would like to be able to remove a friend that I previously accepted, so that I can choose to no longer see another person’s capsule.
15. As a user, I would like there to be a history page containing a scrollable list of past capsules, so that I can be reminded of past months.
16. As a user, I would like to have a history page where I am able to see previous capsules, so that I can reflect on previous months.
17. As a user, I would like to be able to click on a snapshot to enlarge it, so that I can view more options regarding the snapshot.
18. As a user, I would like to be able to share my snapshot as an image to another app, so that I can easily send my snapshot to other people.
19. As a user, I would like to be able to edit my snapshot (options for edit page described later), so that I can further customize my snapshot.

Story Board (sharing snapshots)

20. As a user, I would like to have a Story Board page, so that I can see my friend’s snapshots.
21. As a user, I would like to be able to comment on my friend’s snapshot after I enlarge it, so that I can leave my opinion about their snapshot.
22. As a user, I would like to be able to react to my friend’s snapshot after I enlarge it, so that I can emotionally connect with my friend.
23. As a user, I would like the Story Board to be reset every month, so that I can only view the most recent capsules.

Main Page

24. As a user, I would like to be able to see a countdown until I get my monthly snapshot if it is not time for it to be displayed, so that I know when to expect my new capsule.
25. As a user, I would like to be able to view my snapshot at the end of the month, so that I can reminisce on the past month.
26. As a user, I would like my snapshot to have my most played song of the month, so that I have another reminder of the month.
27. As a user, I would like a portion of the snapshot's song to play out loud anytime I am looking at a snapshot so that I can audibly connect with the capsule
28. As a user, I would like Time Capsule to intelligently choose photos for my snapshot, such as landscapes and people smiling, so that I have quality photos for my capsule.
29. As a user, I would like my snapshot to use my Instagram photos, if I posted that month, so that I can relive the experiences in my capsule.
30. As a user, I would like my snapshot to not contain identical pictures, so that I have a variety of photos on my snapshot.
31. As a user, I would like my snapshot to pick default photos / songs if I have not taken enough pictures / linked my spotify account , so that I always have enough information for the capsule.
32. As a user, I would like my snapshot to contain a variable amount of photos, so that it allows more flexibility in the photo picking process.
33. As a user, I would like my snapshot to contain a quote from my song of the month, so that I am reminded of the lyrics.
34. As a user, I would like to be able to share my snapshot to the Story Board, so that my friends can see it.

Editing

35. As a user, I would like to be able to view an edit page where I can access all features regarding the editing of a snapshot, so that I can have all editing features in one spot.
36. As a user, I would like to be able to click on a photo to change the photo in the edit page, so that I can replace the photo if I do not like it.
37. As a user, I would like to be able to click on the song to change my song in the edit page, so that I can choose another song I would rather have in the snapshot.
38. As a user, I would like to be able to edit the quote on the snapshot, so that I can add my own thoughts to the snapshot.

39. As a user, I would like to be able to change the number of photos used in the snapshot, so that I can have greater control over the photos shown in the snapshot.
40. As a user, I would like to be able to save my edited snapshot, so that it automatically updates my Story Board and history

Yearly Recap (Moments)

41. As a user, I would like to be able to insert brief descriptions of moments that occur throughout the year, so that I can remember them later on.
42. As a user, I would like to be able to see how many moments I have entered throughout the year, so that I have a visual reminder of how many moments I have already entered.
43. As a user, I would like to receive all the moments I sent at the end of the year, so that I can have a recap of all my favorite memories from the year.
44. As a user, I would like all user moments to have a timestamp of when they were submitted, so that I can better picture what was happening at that time.

Notifications

45. As a user, I would like to receive a notification when my monthly snapshot is released so that I know when it has become available
46. As a user I would like to receive a notification when someone accepts my friend request or invites me to be their friend so that I am up-to-date with friend requests
47. As a user I would like to receive a notification when someone reacts or comments on my snapshot so that I am aware of the reactions / comments
48. As a user I would like to receive a notification when a friend shares their snapshot to the Story Board so that I know and can go look at it

Non Functional Requirements

Architecture

We plan to develop an application with a completely separate frontend and backend. Using this structure will allow for better compatibility while developing both simultaneously. The frontend will consist of the user interface while the backend will consist of the application server and database.

The frontend will be written in JavaScript using the React Native framework. This will allow us to have an extremely portable application that can be used cross-platform across mobile devices. For testing and demo purposes we intend to use Expo Go,

which will allow us to see our app on our phones without requiring any of us to have an Apple computer.

The backend will be a Node.js server hosted on an Amazon Web Service (AWS) EC2 instance. We will use API calls to this server to access information that will be used on the frontend.

The database will be MongoDB. Here we will store user info such as snapshots, friends, and usernames. Using MongoDB's flexible schema we can have varied ways to store information making MongoDB a good choice for flexible development

Security

Security is crucial to protect our client's snapshots and login data. Using Google's authentication services with ReactNative we will ensure only users with correct login data may request their data from the MongoDB database. We may allow users to create their username/password information, but for now, we plan on just using G-Auth because it provides its own security. We will also be routing every request for information through our API, which will require authentication.

Usability

Since the app has such a wide range of potential users, the interface must be extremely intuitive. We intend to design it in such a way that the user should have to do as little work as possible. This means that after the initial setup (linking other accounts/apps to get data) the user interface will be relatively simple. The bulk of the app will focus on receiving and modifying the monthly snapshot, which will be easy for users to understand. The app frontend will also be accessible on different types/sizes of mobile devices and will run smoothly because we used a modern stack that can support requests times up to 500 ms.

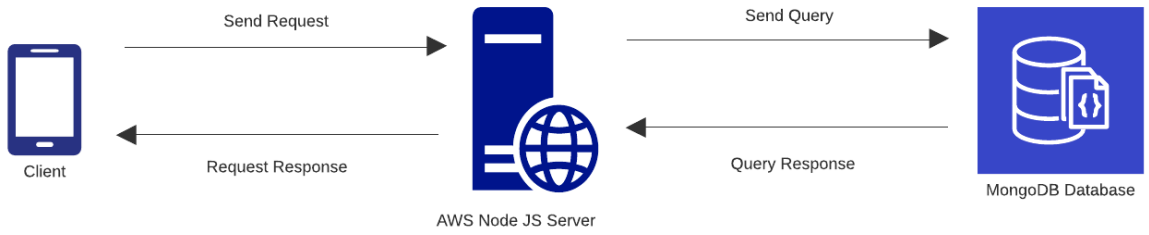
Hosting/Deployment

Our frontend and backend are developed separately, which means that their deployment will be in different places as well. Our backend will be hosted on an AWS EC2 instance for the semester and beyond, we chose AWS because it's a reliable service that we can have running 24 hours a day. The frontend will be deployed for testing on Expo Go, and we intend to explore other options for deployment to the public (potentially GitHub pages or Netlify).

Design Outline

High Level Overview

Capsule will be an IOS app where users can link external social media and content related apps and receive a time capsule / recap at the end of the month. This capsule will contain important information from each app that month. For example, top spotify songs and recent photos from a vacation. The application will be a client server relationship where the server stores and retrieves data from the MongoDB database, mainly being user information like usernames, capsule history, friends, etc. The server will also communicate with the client when processing the capsule from all the user's connected apps.

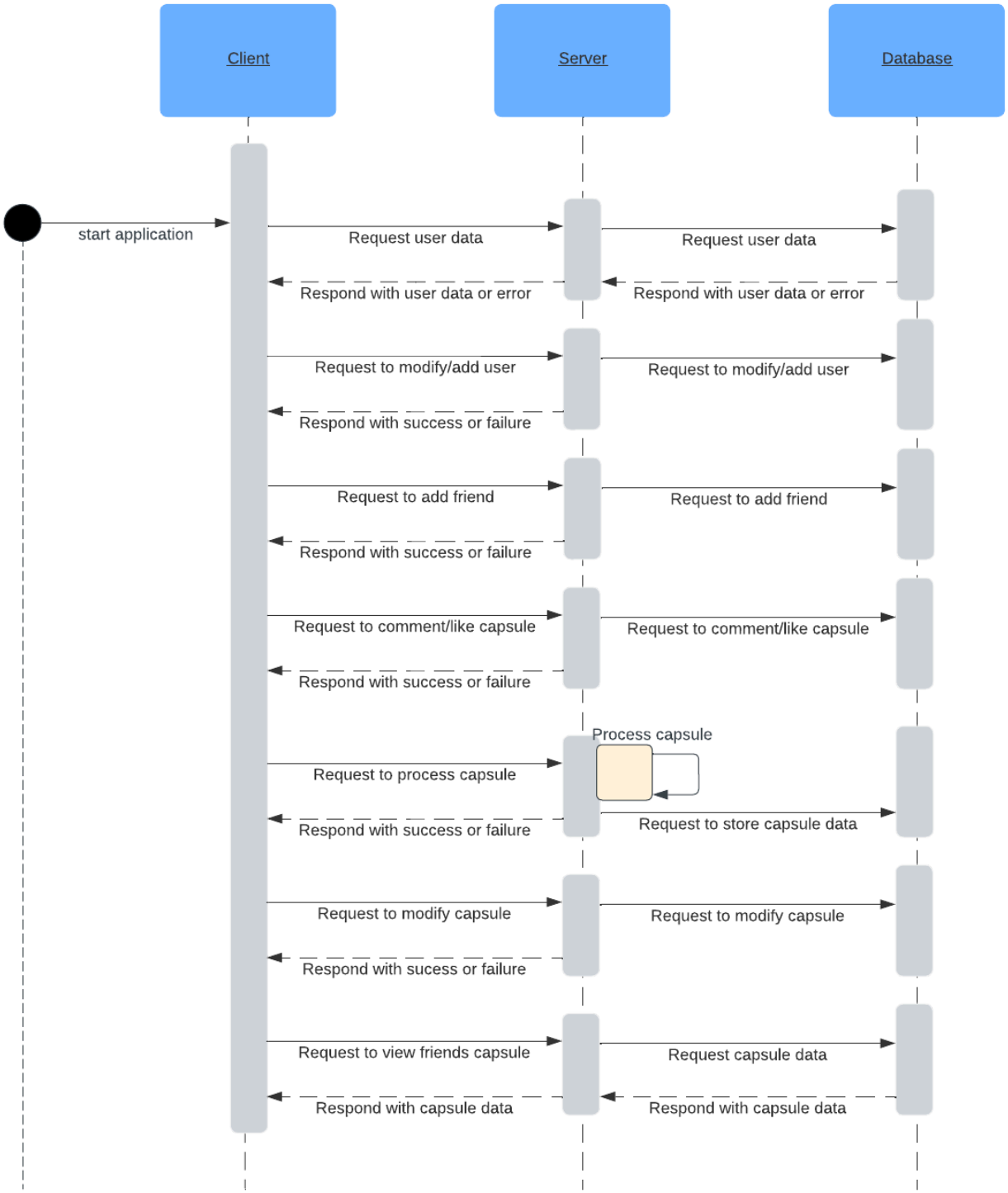


1. Client
    a. The location of the frontend for our app that users will interact with.
    b. The client will send data to the server's API to be stored or processed.
    c. The client will receive data from the server containing processed data or information stored in the Database.
2. Server
    a. Server will receive data from the client containing information to be stored or processed.
    b. Server will validate requests to process data from the client and send it back
    c. Server will validate requests and send data to the database
3. Database
    a. Database will store information for users, such as profile information and previous capsules.
    b. Database will accept queries from the server and send information back.

Sequence of Events Overview

This sequence diagram demonstrates the communication between the client, server, and database. Upon opening the app, the client sends a request to the server which sends a query to the database for user information. After the client receives this

information, other actions performed will send additional data to the server to be stored or processed. Depending on what data/request is sent, the server will either process data or make more queries for information from the database. This cycle is repeated indefinitely for whatever actions are taken on the client side.



Design Issues

Functional Issues:

1. How will the user sign up for an account?
    - Option 1: Username and password only
    - Option 2: Username, password, and email address
    - Option 3: Sign In with a Google Account and username only

    Choice: Option 3

    Justification: Security was the prime factor that led to us choosing to sign in with Google as our method of creating accounts. If users were to link a custom password to their account, we would have to securely store the passwords in a database and take on risk if a data breach occurred. By signing in with Google, the user's account is only at risk if their Google account is breached. Although users are now restricted to having a Google account, we are confident that the large majority of our target audience has a Google account. Another reason we chose this option, is that signing up with a Google account is considerably faster and easier for the user. This way the user doesn't have to worry about creating a unique password and writing it down.

2. How should the images be chosen for the snapshot each month?
    - Option 1: Google Vision API to find relevant photos
    - Option 2: Create our own AI algorithm to find relevant photos
    - Option 3: Randomly choose photos from the past month

    Choice: Option 1

    Justification: A large part of the draw of our app is the automation. Therefore, we have to find a way to choose the best photos to help the user reminisce over the last month. We initially thought randomly choosing photos would be a good way to get an accurate recap of the month, but we quickly realized this had several problems. What if the user had duplicate photos or took several photos of the same object? Additionally, what if the user took a picture of just their homework. With this in mind, choosing randomly was not an option. Our next thought was to design our own AI. However, there are already great image scanning AI's already out there so we did not want to reinvent the wheel. This is why we eventually decided to use Google Visions API to find relevant photos as it efficiently and effectively scans photos giving quick descriptors, which we can use to find group photos, landscapes, and portraits to provide the best photos of the months.

Non Functional Issues:

1. What database should we use to store our app's information?

- Option 1: MongoDB
- Option 2: MySQL
- Option 3: Oracle XE
- Option 4: Redis

Choice: Option 1

Justification: MongoDB uses a flexible schema model. This means that documents in a collection do not need to have the same fields or data types by default. Since our project will be constantly evolving with dynamic data structures, this model gives unrivaled flexibility in the development process. In addition, MongoDB is designed with scaling in mind. This means that the database could quickly be modified such that it could handle large amounts of traffic in the event that our app has substantial growth.

2. What frontend language / framework should we use?

- Option 1: Flutter
- Option 2: Swift and Kotlin
- Option 3: React Native

Choice: Option 3

Justification: One reason we chose React Native as our mobile app development framework is that React Native utilizes native components for rendering UI elements which gives the feel of a native application. This offers increased flexibility in the UI design. Also, React Native is unique because it can create mobile apps that work on both iOS and Android without having to rewrite the code. This makes for much faster development and allows us to focus on learning a single JavaScript framework over learning both Kotlin and Swift. We want our app to work on both iOS and Android so using Kotlin or Swift was not an option. In addition, React Native has a thriving community with numerous guides and available libraries. In the past, our group has also had limited experience with React and JavaScript which allows us to have a basic understanding of the environment and we can use guides online to learn what we struggle with. React Native has a very similar structure to React and is built on JavaScript so our prior experience will carry over to React Native.

3. What language/framework should we use for our backend?

- Option 1: Django

- Option 2: Node.js
- Option 3: Express

Choice: Option 2

Justification: Node.js is built on JavaScript and therefore allows for full-stack JavaScript development, allowing us to use the same language for both the front-end and back-end. This allows our group to focus on mastering a single language. Also, similar tools and libraries can be used in the development stack. For example, the Node Package Manager (NPM) will manage dependencies for the Node.js server and our React Native mobile app. Finally, some members of our team are familiar with using Node.js servers which will help with the development process.

4. What server hosting service should we use?

- Option 1: AWS
- Option 2: Azure
- Option 3: Google Cloud

Choice: Option 1

Justification: We chose to use AWS as our hosting service. AWS compared to Azure and Google Cloud is much better for smaller general development, for its scalability and flexibility. Compared to Azure, AWS is known for its open source development. Furthermore, AWS is easy to use and flexible whereas Azure while having more complexity comes at the cost of ease of use. Next compared to Google Cloud, AWS is very similar, but when you turn to see what is more widely used in industry it is AWS by far. After digging deeper we found this was because of how universally reliable AWS hosting is, and how easy it is to integrate into any project.

5. What version of React Native should be used?

- Option 1: React Native CLI
- Option 2: Expo

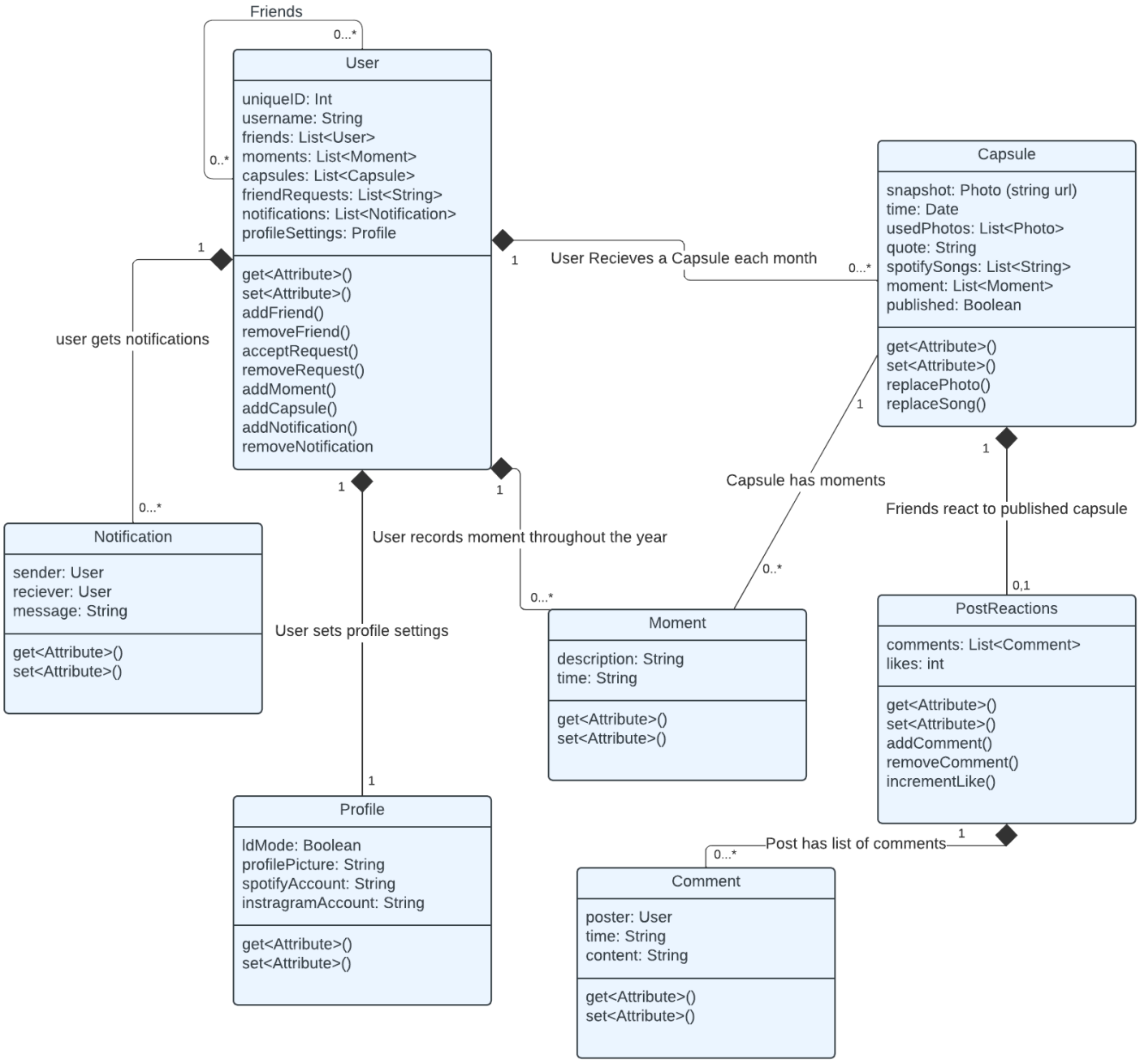
Choice: Option 2

Justification: React Native CLI has a steep learning curve that requires lots of initial setup and configuration. Expo is extremely user friendly and allows for quick setup and deployment. With the Expo Client App, we can preview our application on our phone and see updates made in real time. This allows us to quickly test additional features and visualize our app. Also, Expo offers a large set of dev tools that allow for

easier debugging and optimization of our application. Overall, Expo is much more user friendly for beginners seeking to learn React Native.

Design Details

Class Diagram



Descriptions of Classes and Interaction between Classes

These classes outline the different objects that will be used in the application. Each class has associated variables which outline what is included in the object.

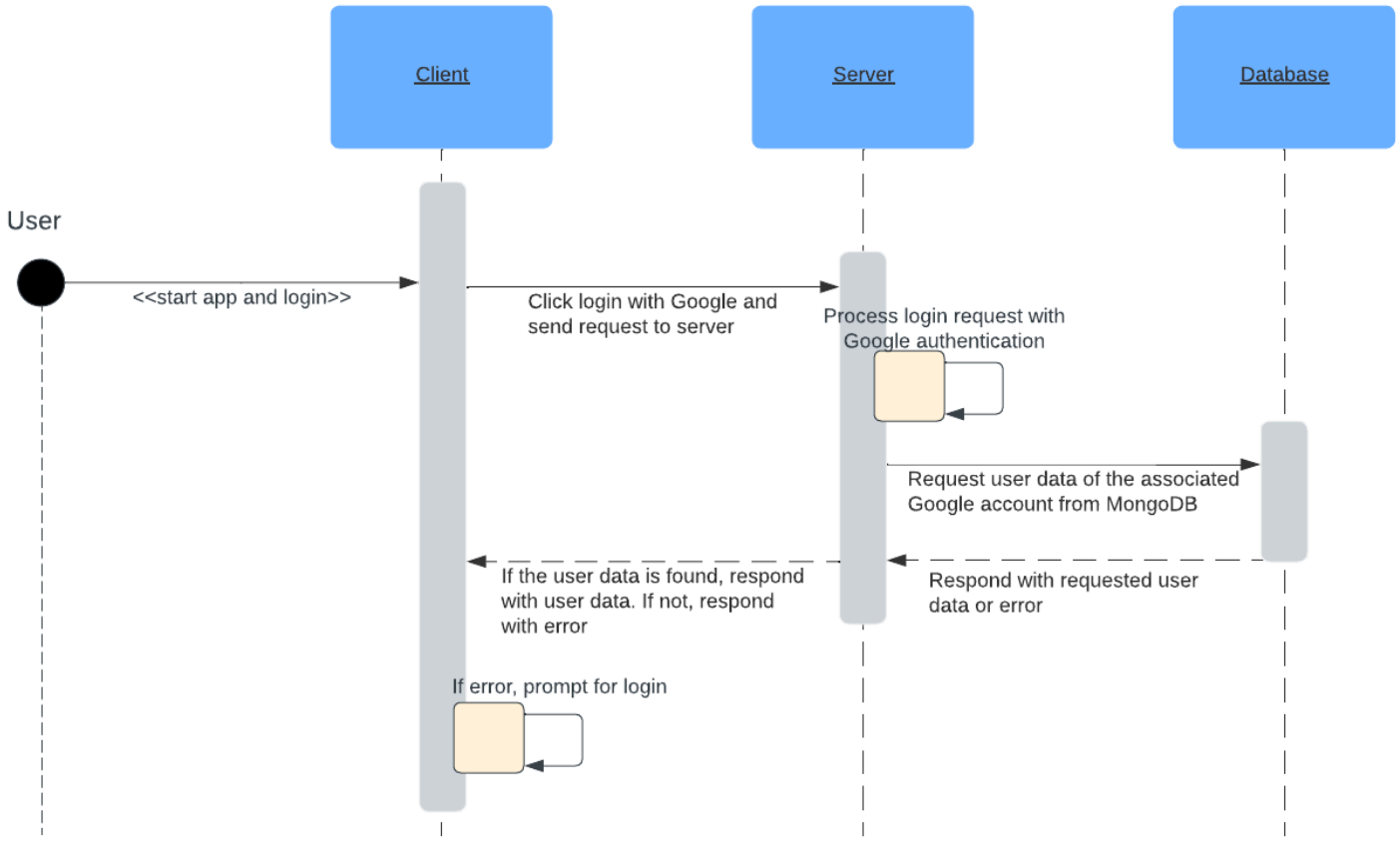
- **User**
    - A user is created when someone enters a username and signs in with their google account.
    - Each user is assigned a unique ID that can be used to retrieve data from the MongoDB database.
    - Each user has a list of friends that is a list of other users. Users will be able to see their friends published capsules and can comment/like those posts.
    - Each user has a list of moments. Users can record random moments from throughout the year and then receive all the moments at the end of the year and a couple at the end of each month.
    - Each user has a list of their previously generated capsules. These will be displayed on the history page and on a friend's dashboard.
    - Each user has a list of friend requests that are sent by other users. They can be accepted to add a user to the friends list.
    - Each user has a list of notifications that are sent in response to other users.
    - Each user has a single Profile object so that the profile settings screen does not need to access the entire user and can solely use relevant information.
- **Capsule**
    - Capsules are created automatically at the conclusion of each month. They are then sent to the user if they would like to modify/share the capsule.
    - Each capsule contains the snapshot that was generated for the month. This will be a url string to a google cloud database where the photo is stored.
    - Each capsule contains the time when the capsule was originally created. This will only be a month and a year.
    - Each capsule contains a list of photos (same string url) that were used to generate the snapshot. In the editing page, these can be switched out and moved around by the user.
    - Each capsule has a quote that will originally be filled with a verse from their top spotify song.
    - Each capsule has a list called spotifySongs that contains the songs displayed on the snapshot.
    - Each capsule has a list of moments that can be displayed on the snapshot. Users can modify how many they would like visible on the snapshot.
    - Each capsule has a boolean value ‘published’ that indicates whether that capsule is visible to friends.

- **Profile**
    - Profile object is created with the creation of each user object
    - Profile settings include light dark mode, profile picture, spotify account, and instagram account.
    - When on the profile settings page, all data corresponding to profile settings is visible and able to be changed by the user.
- **Moment**
    - On the main dashboard screen, users are able to type and submit moments as they occur throughout the year. This creates a moment object that is added to a list in the user object.
    - Each moment has a description and time to describe the moment.
    - Some moments are included in the monthly capsule. At the end of the year, all moments entered that year are given to the user.
- **PostReactions**
    - When the capsule is published, the PostReactions object is created for the capsule.
    - Each PostReactions object has a list of comments and the number of likes given to the post.
    - These post reactions are shown when a user enlarges a snapshot and sees more details associated with it.
- **Comment**
    - Comments are created when another user posts a comment on another user's published capsule.
    - Each comment has a poster, time, and content variable to store data on the comment.
    - Comments are shown after a user expands a capsule and views the post reactions.
- **Notification**
    - A notification object is created when a capsule is released for the month, a friend shares their capsule, a friend request is received, a friend request is accepted, and when a friend reacts to the user's post.
    - This object will be used for sending notifications to related users.
    - Each notification has a sender, receiver, and message.

Sequence Diagrams

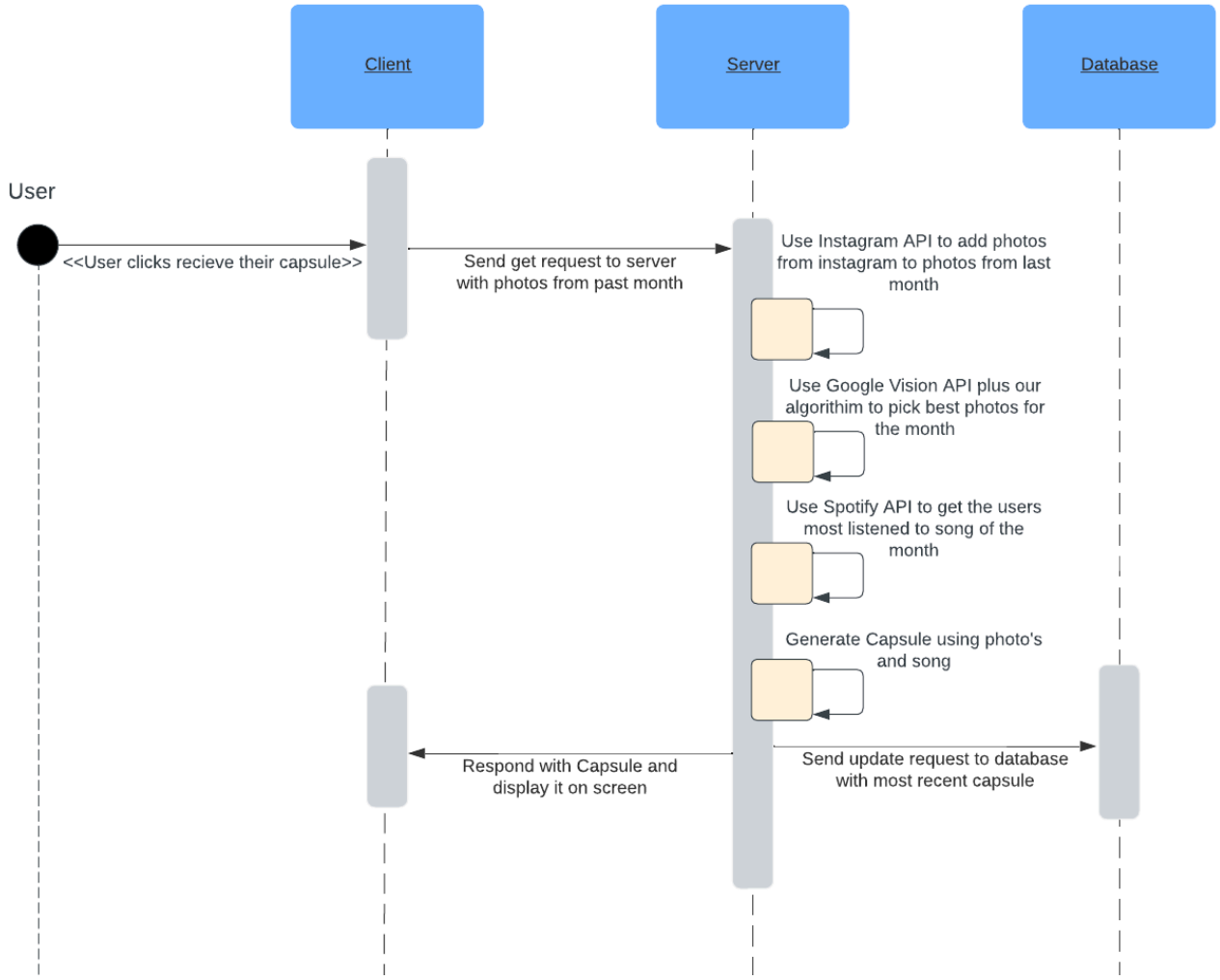
Sequence of Events when Users login

When a user attempts to log in to their account, they only have to click the “sign in with Google'' button and select their Google account. The request is then sent to the server which confirms the identity of the user. A request is then sent to the database to retrieve any data correlated with the user. If a user is found, the user is logged in to their account. If the user is not found in the database, they are asked to sign in again or create an account.



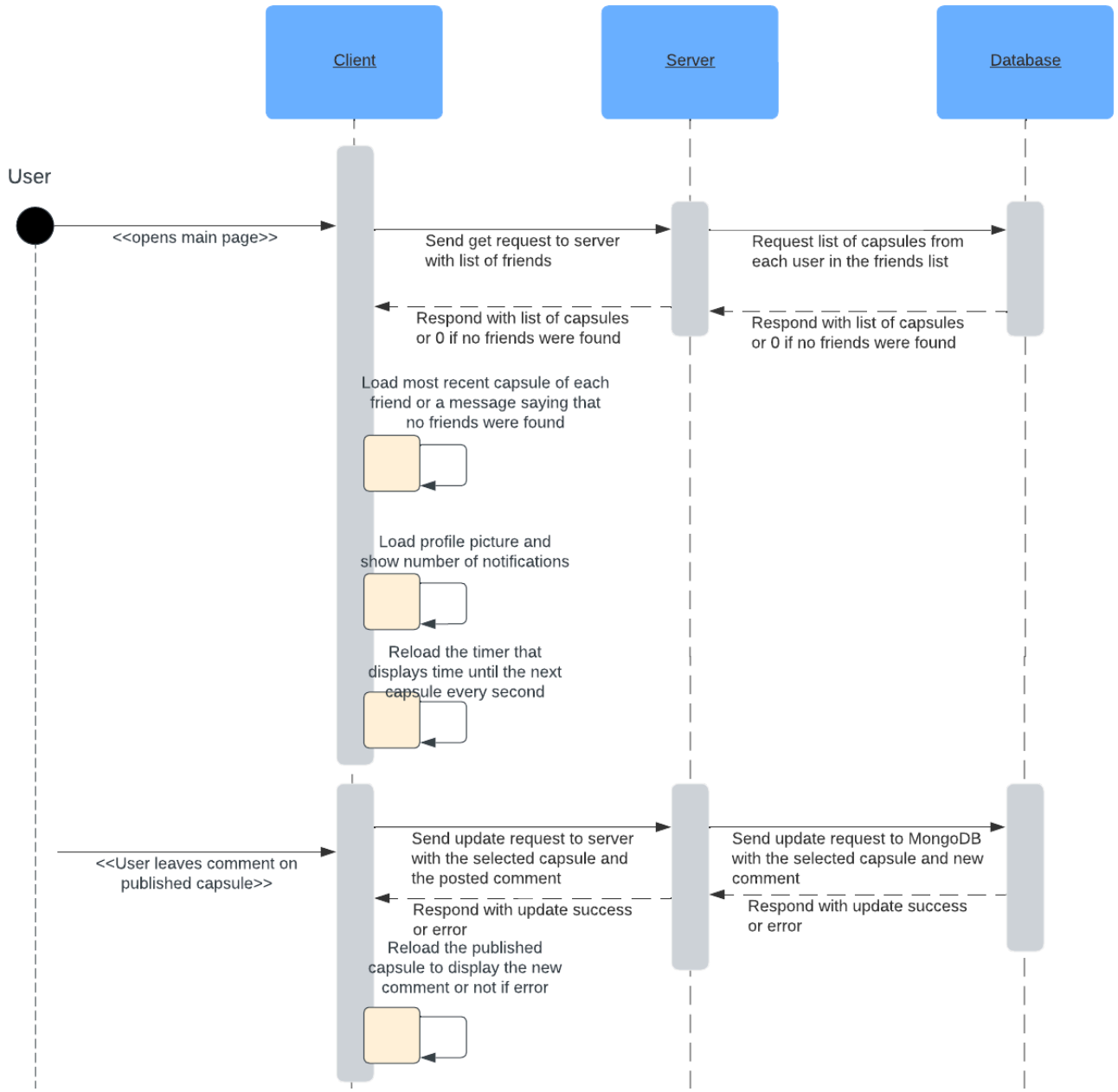
Sequence of Events when user clicks to receive their Capsule

When the timer on our app hit's 0 the user will be prompted to receive their capsule. When they accept the client will send a request to the server with all the data it needs to create the capsule. Then using Instagram, Spotify, and Google Vision's API we will create the capsule, store it in the server and send it back to the client to be displayed.



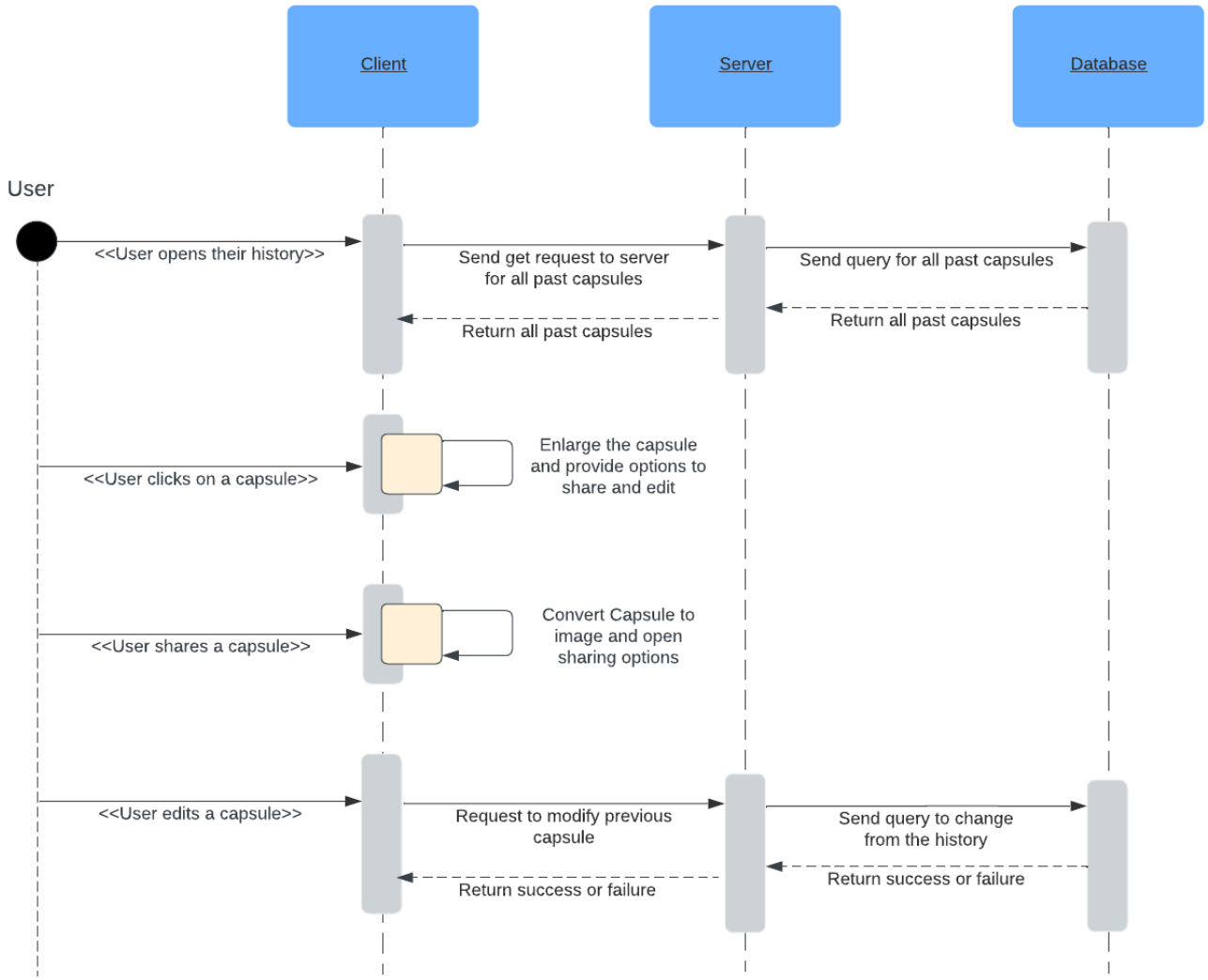
Sequence of Events when the Main Page is generated

After logging in to an account, the user is immediately taken to the main dashboard. The server sends a list of the current user's friends to the server. Get requests are then sent to the MongoDB database to retrieve all capsules published by those users. The most recent capsule of the first 20 friends is then displayed on the dashboard and more will be displayed when the user scrolls to the bottom of the list. In addition, when leaving a comment on another user's capsule, the information is first sent to the MongoDB database with an update request and then will be displayed on the user's dashboard.



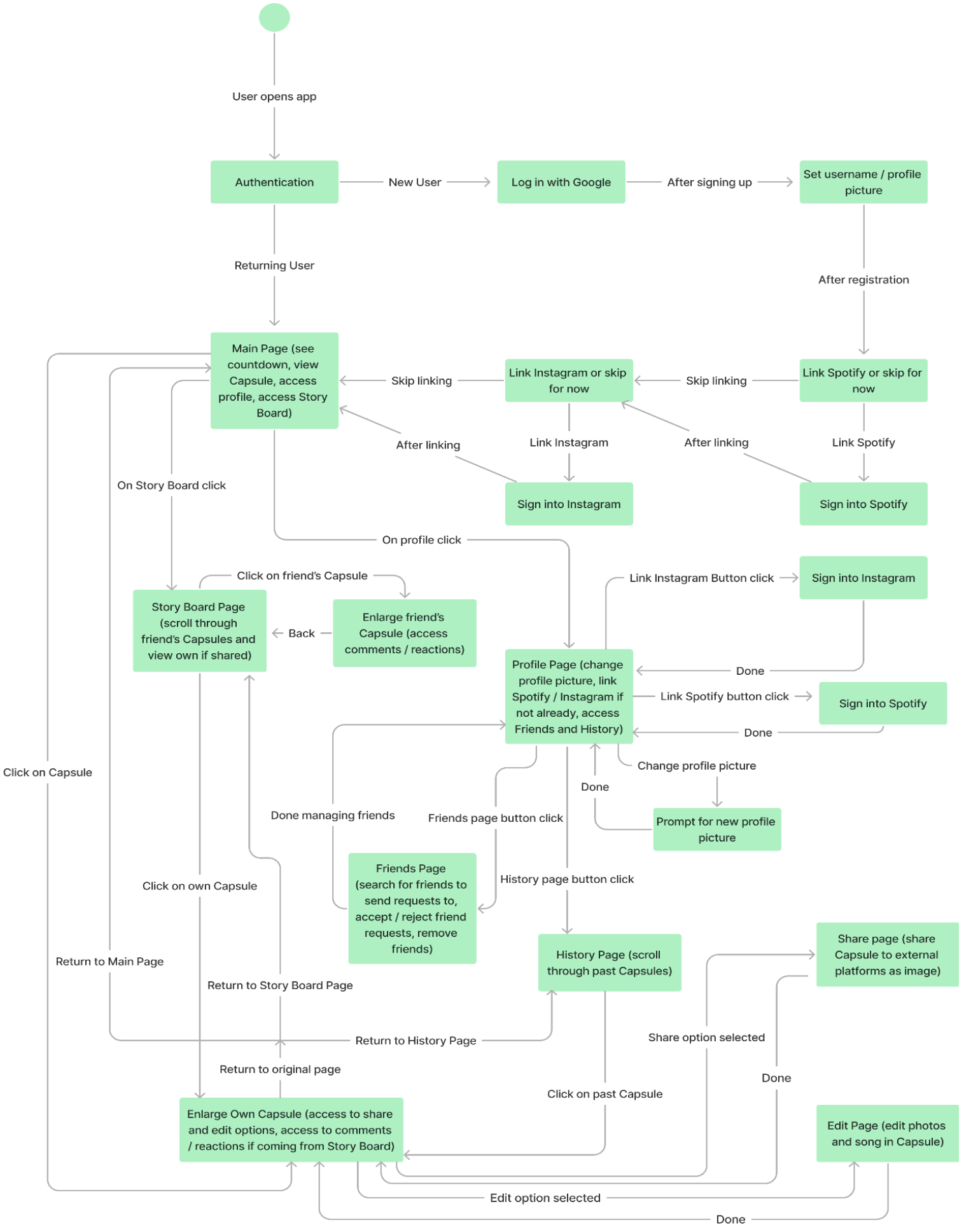
Sequence of Events when History Page is generated

When the user opens the history page, all past Capsules are requested from the database through the server. These Capsules are displayed in the client in a scrollable list. The user can then click on an individual to enlarge it and gain access to sharing and editing options. Sharing and enlarging is handled solely by the client. If the user edits the Capsule, the client will send a request to the database through the server to update the Capsule in the database.



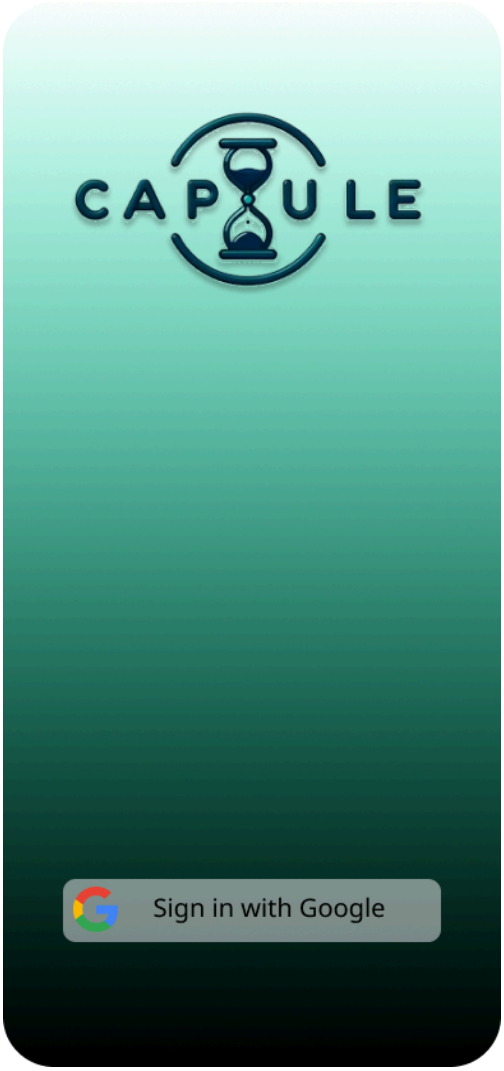
Navigation Flow Chart

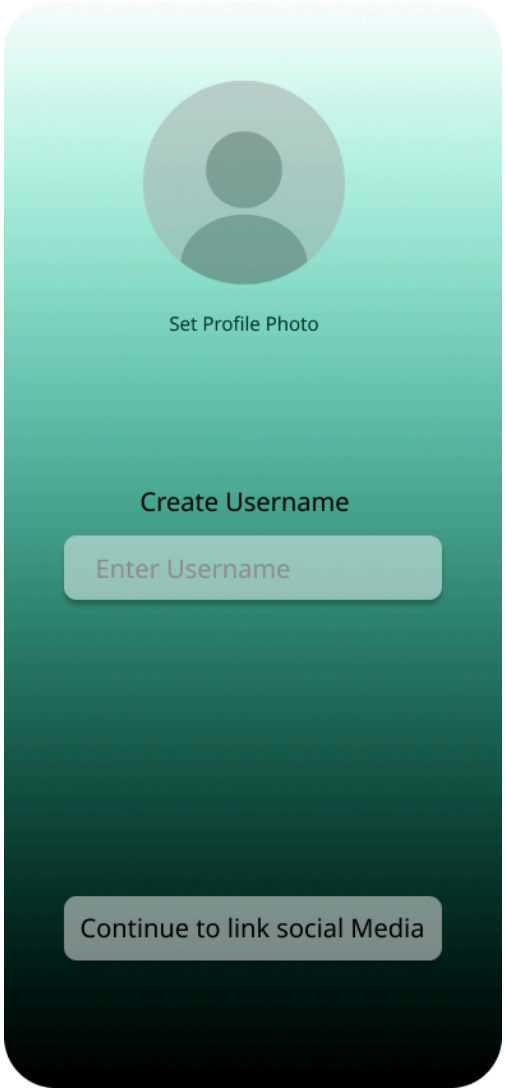
When designing our navigation schema, we wanted to prioritize a user-friendly environment. The user begins by logging in with Google Authenticator, and if it is their first time using the application, they will be prompted to enter a username and select a profile picture. From there the new user will have the option to link Instagram and Spotify or skip and link later. After the login / registration process, all users will be directed to the main page. From here, all features of the application are only a couple clicks away. The user can navigate to the Story Board page and profile page from the main page, and to the friend page and history page from the profile. Between these five easy- to-access pages, all functionality relating to a user's profile, friends, and Capsules can be easily carried out. The diagram on the next page represents the navigation layout of our application and further visualizations can be found in the UI Mockup.


User opens app
Authentication
New User
Log in with Google
After signing up
Set username / profile picture
Returning User
After registration
Main Page (see countdown, view Capsule, access profile, access Story Board)
Skip linking
Link Instagram or skip for now
Skip linking
Link Spotify or skip for now
After linking
Link Instagram
After linking
Link Spotify
Sign into Instagram
Sign into Spotify
On Story Board click
On profile click
Click on friend's Capsule
Story Board Page (scroll through friend's Capsules and view own if shared)
Back
Enlarge friend's Capsule (access comments / reactions)
Link Instagram Button click
Sign into Instagram
Profile Page (change profile picture, link Spotify / Instagram if not already, access Friends and History)
Done
Link Spotify button click
Sign into Spotify
Done
Change profile picture
Done
Prompt for new profile picture
Done managing friends
Friends page button click
Click on Capsule
Click on own Capsule
Friends Page (search for friends to send requests to, accept / reject friend requests, remove friends)
History page button click
History Page (scroll through past Capsules)
Share page (share Capsule to external platforms as image)
Return to Main Page
Return to Story Board Page
Return to History Page
Share option selected
Done
Return to original page
Click on past Capsule
Enlarge Own Capsule (access to share and edit options, access to comments / reactions if coming from Story Board)
Edit Page (edit photos and song in Capsule)
Edit option selected
Done

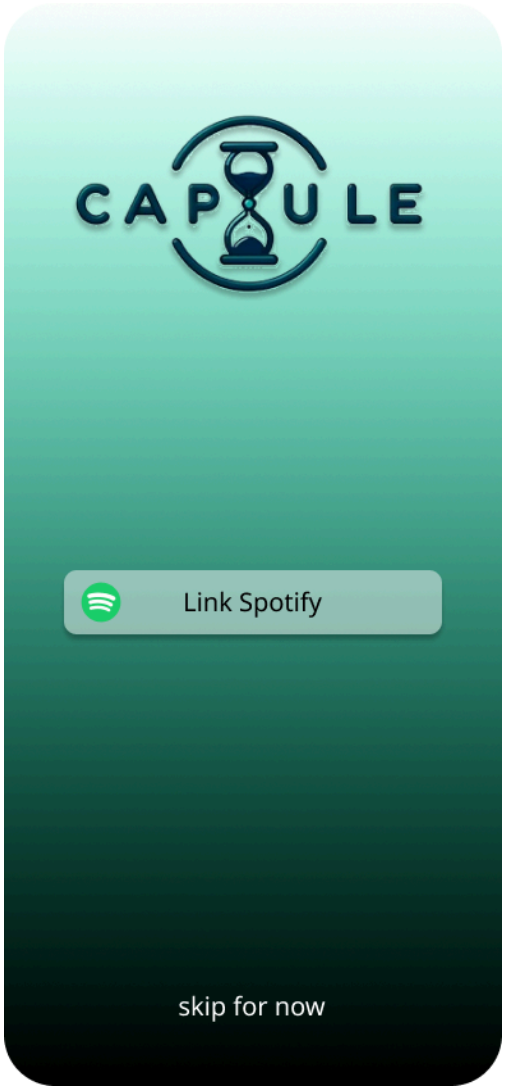
UI Mockup

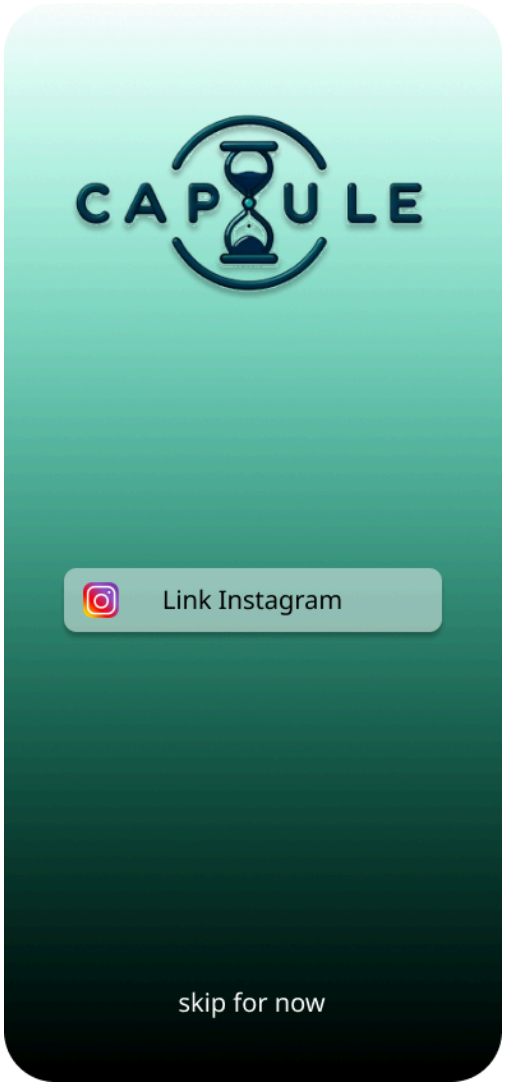
When a new user opens the application, they will be directed to a screen prompting them to sign in with google. After they go through the login process with google authenticator, they will be prompted to enter a username and select a profile picture.



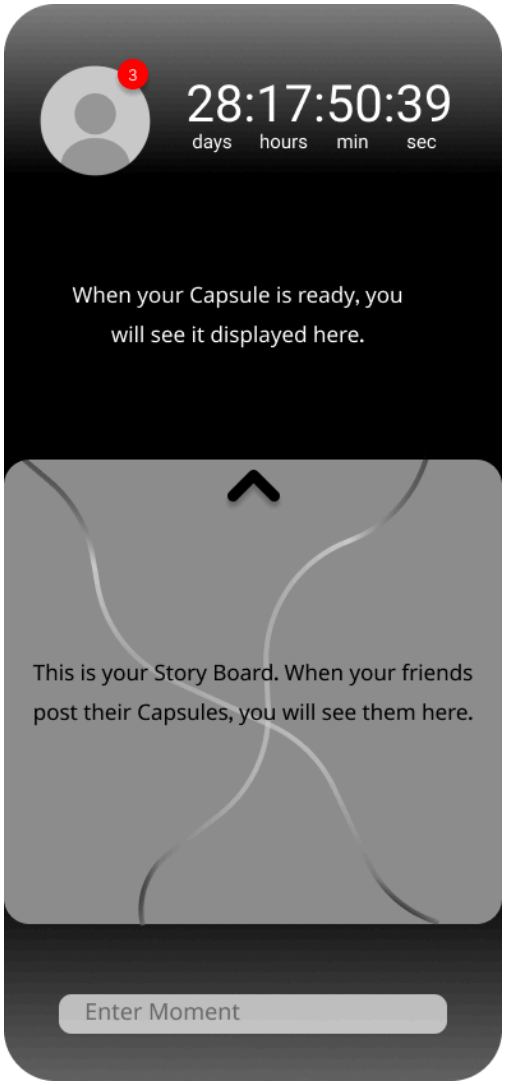


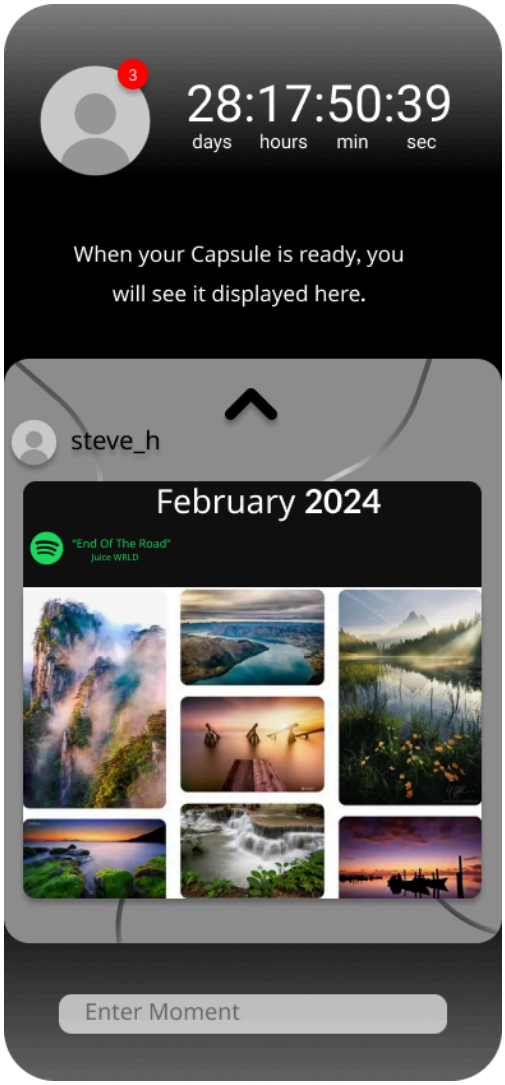
After registering, the user will have the option to connect their spotify and instagram accounts or wait and do it later.



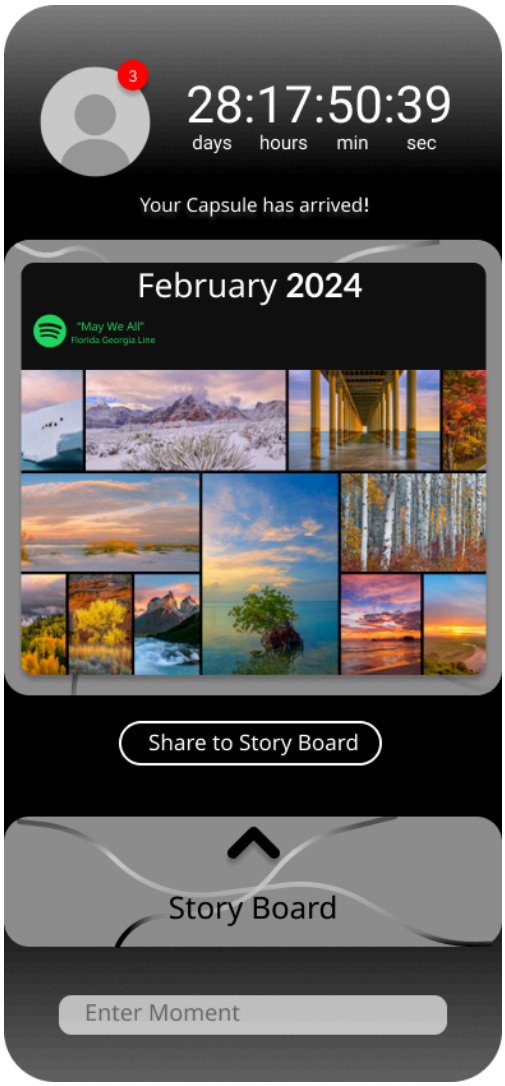


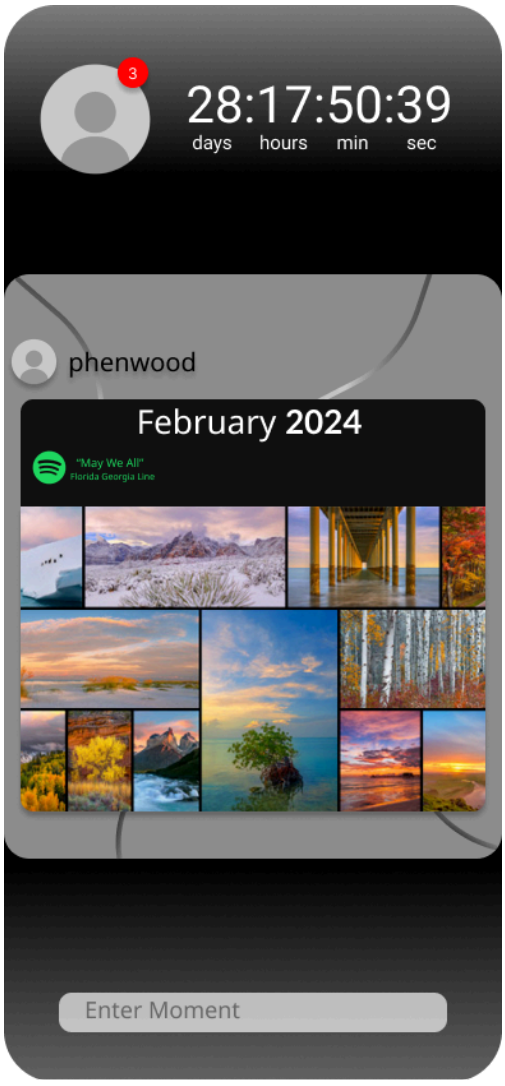
After the user registers for the first time, and anytime they enter the app after initial registration, they will be taken to the main page. If this is their first month using the app, there will not be a current Capsule displayed. If they have not added any friends, or no friends have shared a capsule, the Story Board section will be blank (left). If they do have friends that have shared capsules, the Story Board section will contain the shared capsules (right).



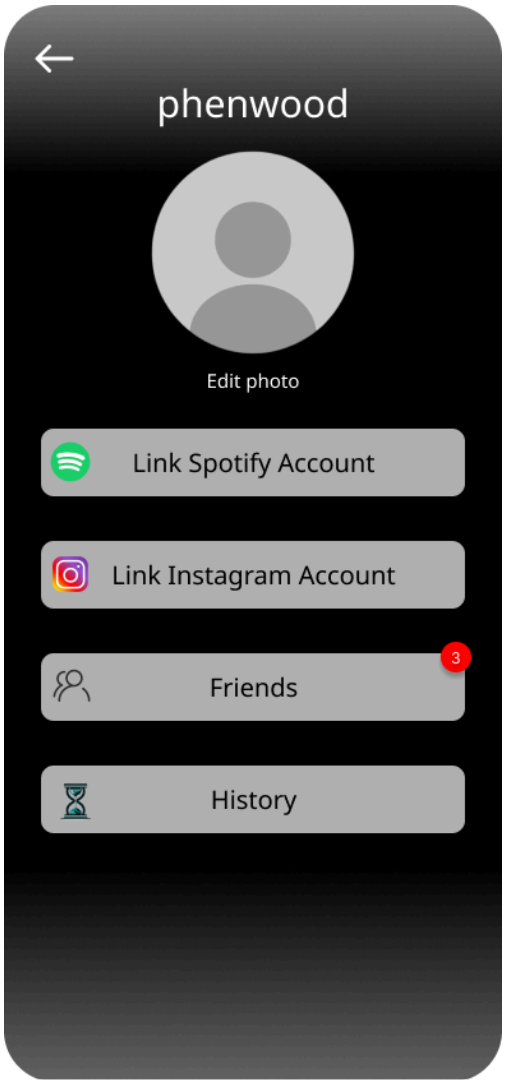


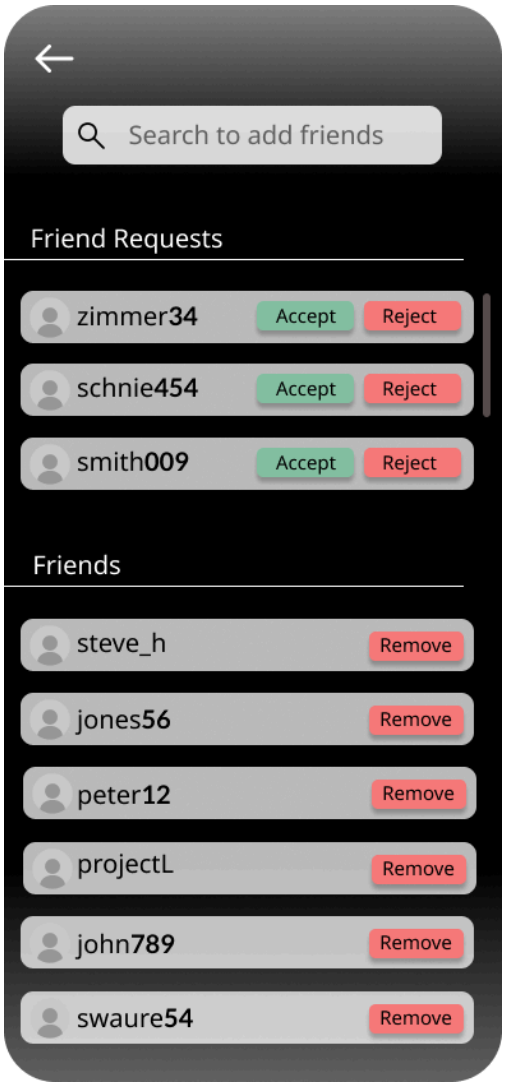
If the user has used the app for at least a month, at least one Capsule will have been created for them. If they have just received their capsule and not shared it to the Story Board yet, or if they have decided not to share it to the Story Board, their monthly Capsule will appear above the Story Board (left). If they have added their Capsule to the Story Board, it will appear as the the top Capsule in the s as an indication that it was successfully added (right).



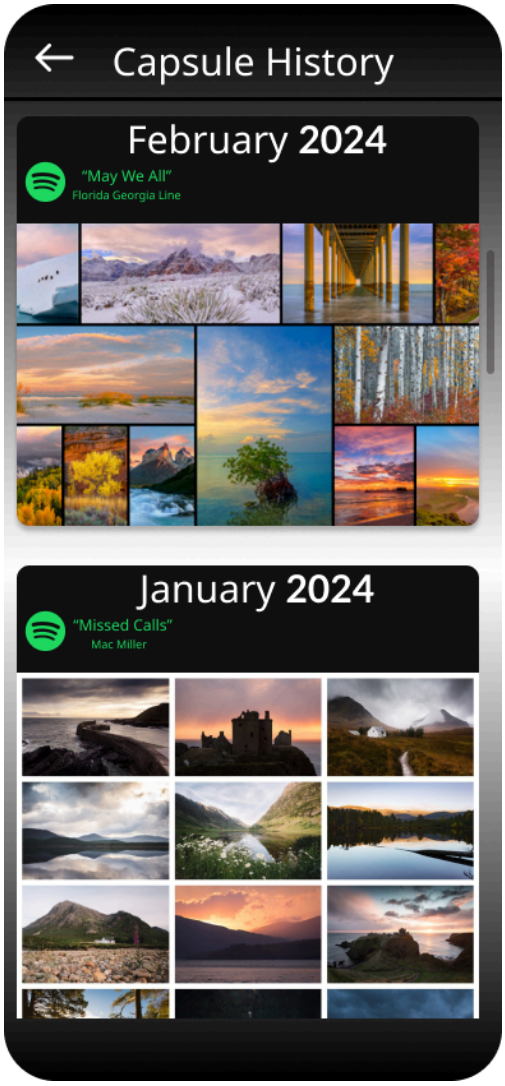


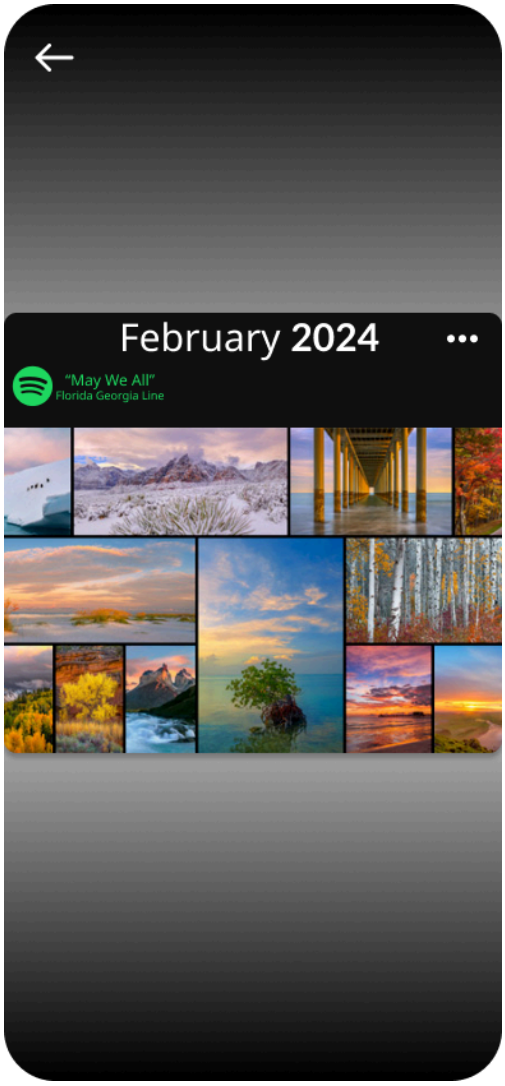
If the user presses on their profile image in the top left corner they will be directed to the profile page (left) where they can link their spotify and instagram accounts if they haven't already, and view their friends and past Capsules. In the friends page (right) users can accept friend requests and search for other users to add as friends.



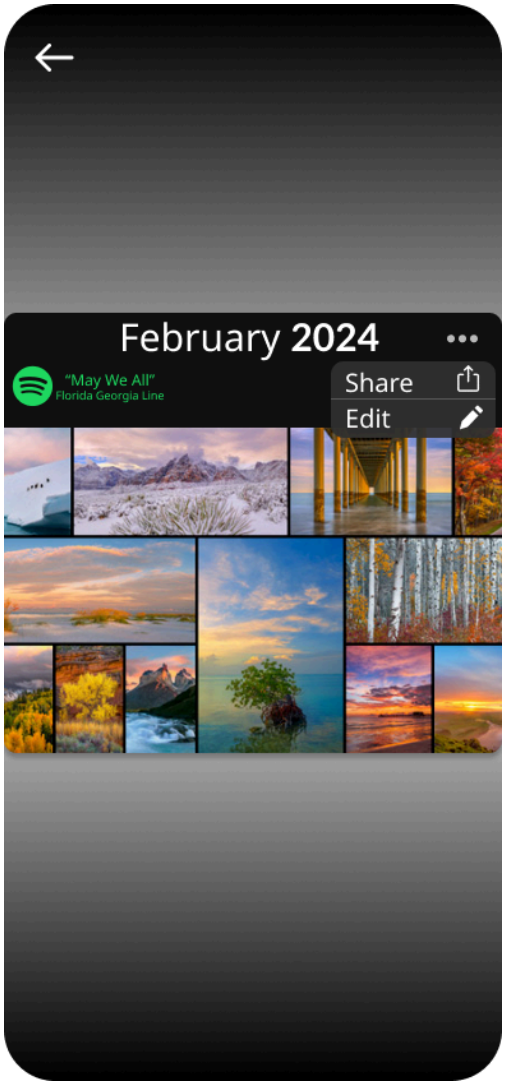


The history page (left) displays past Capsules that the user can scroll through. Clicking on a Capsule will enlarge it (right).

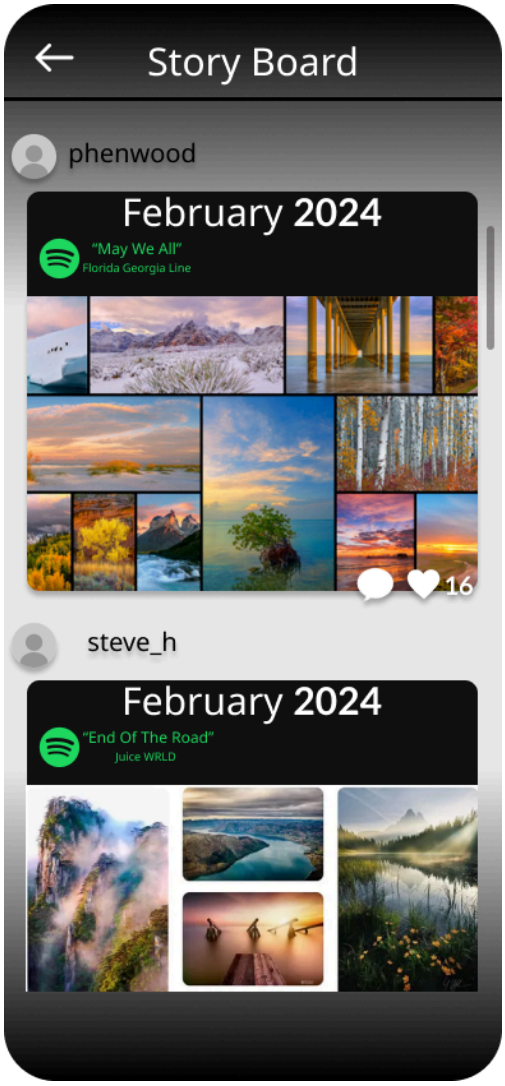


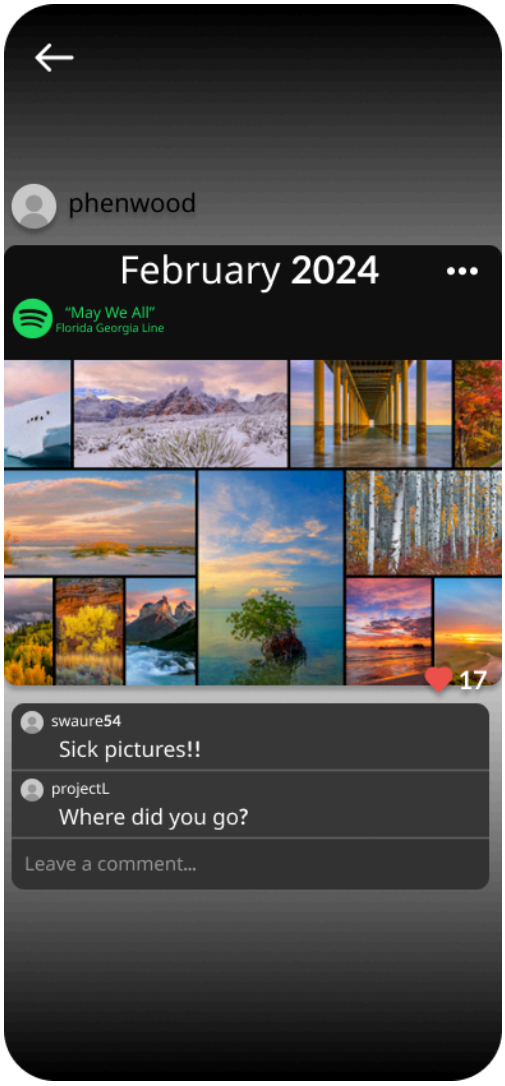


If the user presses the three dots on the enlarged Capsule, editing and sharing options will come up (left). If the user selects the edit option they will be redirected to the edit screen (right) where they can swap out pictures and the song, and their changes will be updated in their history and Story Board.

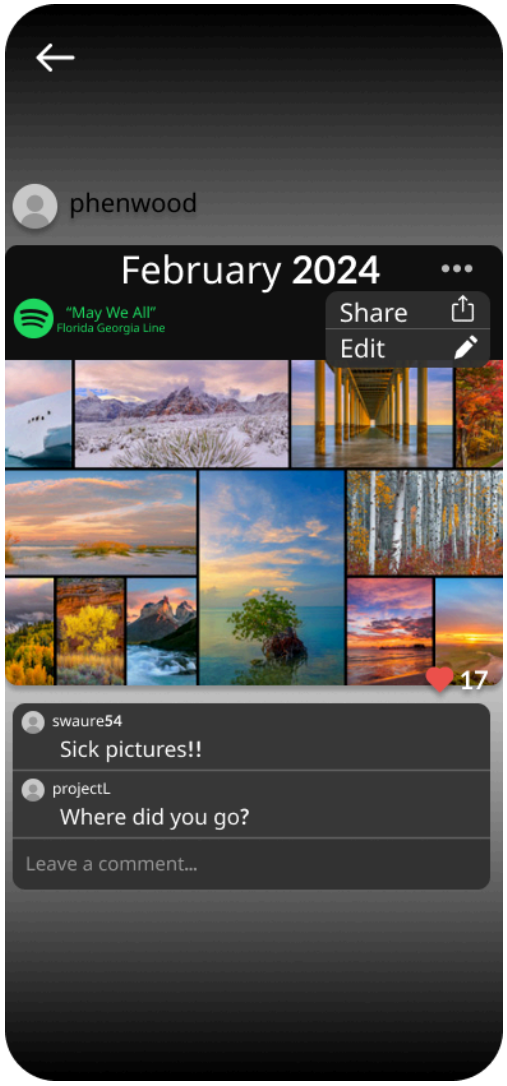


If the user has shared their Capsule to the Story Board and they click on the Story Board, their Capsule will be the first one displayed (left). If they then click on their own Capsule, they can comment and react to it (right).





After the user has enlarged their own image, they can click the three dots for sharing and editing options (left) just as in the history page.



If the user has not shared their Capsule to the Story Board and they click on the Story Board, one of the user's friend's Capsules will be the first one displayed (left). If they then click on a friend's Capsule, they can comment and react to it (right).