****

**BookXchange** iOS App

CS 307 Design Document

September 22nd, 2023

Team 11

Aditi Patel, Harshini Manian, Elena Monroe, Pooja Mathi, Sam DeLucia-Green

Table of Contents

[**Purpose**](#_6912g5y3veoc) **3**

[Functional Requirements](#_bi56vy29tk9m) 4

[Non-Functional Requirements](#_2bjnit7sjqb1) 7

[**Design Outline**](#_krc98uz10k2q) **9**

[Design Decisions](#_w3qdgd9oqzjq) 9

[System Components](#_a75kp4mxcgvt) 9

[High Level Overview](#_w5gbs72n4e3l) 9

[Detailed Overview](#_ruws9ho0gew3) 9

[**Design Issues**](#_vc50ptvywx0o) **11**

[Functional Issues](#_3zh8kbk8zfyj) 11

[A. What information do we need to make an account?](#_pxat9v4ouauo) 11

[B. How should users upload books into their library?](#_ewuqf5e7fgig) 11

[C. How will we obtain user location information?](#_e63isck3vwj4) 11

[Non-Functional Issues](#_23mv364y461b) 12

[A. What platform will we use for frontend/UI development](#_pyc5d2srr997) 12

[B. Which programming language are we going to use for backend development](#_acp1bafop3xq) 12

[C. Which programming language are we going to use for Database Management](#_stdfini23uh1) 12

[**Design Details**](#_hgl3sywqftzc) **14**

[Class Level Design](#_m0s2wiol3859) 14

[Class Descriptions and Interactions](#_389ets4akz0e) 15

[User](#_fjxgu06e0xzp) 15

[Library](#_alkls04f0wo) 15

[Book](#_cc4hk7yqoovz) 15

[Chat](#_jgl9f6ruuwha) 16

[Message](#_sl4okw1oapjj) 16

[Notification](#_ryd76tpr4ifm) 16

[Sequence Diagrams](#_28lul8af8pyg) 17

[Sequence Diagram and Explanation for Creating an Account](#_5a8ad1m5t78d) 17

[Sequence Diagram and Explanation for Logging In](#_x7q725jzp8b9) 18

[Sequence Diagram and Explanation for Adding a Book](#_74z7b3210cos) 19

[Sequence Diagram and Explanation for Chat Sequence](#_6ej3qfvulxmd) 20

[Sequence Diagram and Explanation for Matching and Swiping](#_njbamqwmfmg7) 21

[UI Mockups](#_w1kdfj97qxje) 22

# **Purpose**

Avid readers often run into a couple of the same problems time after time. First, they can’t think of what book to read next. Second, they can’t find the book once they’ve decided on one. They don’t own it, it’s been checked out at the library, and they don’t want to pay for it at a bookstore. What to do? This is where the BookXchange app comes in. BookXchange aims to facilitate book swapping between local users.

The purpose of the BookXchange app is to create a user base of readers who face the issues explained above, suggest the most relevant books to each user, and assist users who’ve matched books with exchanging information and meeting up to trade their respective books with each other. BookXchange is a system built for iOS and Android that incorporates user information into a database and constructs an algorithm that identifies each users’ literary preferences and suggests the most relevant books that they can trade their own books for. Once users have indicated that they want to swap books, the system creates a chat function between the two users and suggests a location for them to meet.

There’s one similar app called Swappy Books that allows users to swap books with other people in their area. However, it doesn’t have an algorithm that suggests books based on the user’s previous swipes, which is the main feature of the BookXchange app. Swappy Books also shows users’ exact location regardless of whether they’ve indicated that they want to read the other user’s book, whereas BookXchange allows users to meet up in a general zone and never reveals users’ locations.

## **Functional Requirements**

User account:

1. As a user, I would like to be able to create an account on BookXchange with a unique email, unique phone number, and unique and appropriate username.
2. As a user, I would like to be able to create an account through a Google account.
3. As a user, I would like to be able to login and manage my BookXchange account.
4. As a user, I would like to be able to reset my password.
5. As a user, I would like a verification email sent to my email account after registering.
6. As a user, I would like a verification text sent to my phone number after registering.
7. As a user, I would like to be able to utilize two-factor authentication (if time allows).
8. As a user, I would like to be able to delete my account.

User profile:

1. As a user, I would like to be able to create a profile with my name and bio.
2. As a user, I would like to be able to change my profile’s name and bio.
3. As a user, I would like to be able to pick my preferred genres from the provided options when setting up my account.
4. As a user, I would like to be able to take a photo with my phone camera to use as my profile photo.
5. As a user, I would like to be able to upload a profile photo from my gallery.
6. As a developer, I would like to retain a database of users’ profiles.
7. As a developer, I would like to automatically log out a user who hasn’t been on the app for a while.
8. As a user, I would like to be able to invite someone to the app via phone number or email.
9. As a user, I would like to be able to share my profile on social media.

User library:

1. As a user, I would like to be able to manually upload a picture of my book to my library.
2. As a user, I would like to be able to manually enter the title, author, isbn, and genre of my book to my library.
3. As a user, I would like to be able to scan the UPC on my book and upload its information (if time allows) to my library.
4. As a user, I would like to be able to manually enter the ISBN number of my book and have the book information auto-populate.
5. As a user, I would like to be able to remove books from my library.
6. As a user, I would like to see a visual display of the books in my library.
7. As a user, I would like to be able to edit the information on the books in my library.
8. As a user, I would like to be able to change the status of a book to available or out for loan.

Reporting/managing users:

1. As a user, I would like to be able to report other users who are misusing the app.
2. As a moderator, I would like to be able to delete books posted by users.
3. As a moderator, I would like to be able to delete users.
4. As a user, I would like to be able to give other users a rating based on my interactions with them so that other users can see this rating when they match with that user.
5. As a user, I would like to be able to see other users’ community ratings.
6. As a user, I would like to be able to block other users so their books don’t appear on my swapping page.
7. As a user, I would like to be able to report an issue with the application to the administrators.

Profile interaction features:

1. As a user, I would like to see my most recent messages at the top of the page.
2. As a user, I would like to retain my chat history from people whose books I’ve matched with.
3. As a user, I would like to have a chat feature open up when I’ve matched with another user’s book.
4. As a user, I would like to be able to send and receive messages with users whose book I’ve matched with.
5. As a user, I would like to be able to unmatch with other users’ books.
6. As a developer, I would like to be able to suggest a nearby public location for users whose books have matched to meet up to swap.
7. As a user, I would like to receive a notification when I have matched with someone for a book, when a match has messaged me back, and when I have not been active on the app in a while.
8. As a user, I would like to disable push notifications.
9. As a user, I would like to ‘heart’ other peoples’ messages.
10. As a user, I would like to see the profiles of users whose books I’ve matched with.
11. As a user, I would like to see a visual display of all the books I’ve swapped with.

Swiping feature:

1. As a user, I would like to be able to swipe left (dislike book) and right (like book) on books on my personal feed.
2. As a developer, I would like another book to immediately pop up after the user has swiped on the previous book.
3. As a user, I would like the books on my bookshelf, books relating to my preferred genres and authors, and books whose users are in my general area to influence what types of books appear on my personal feed.
4. As a user, I would like to see an animation on the screen when I match with someone for a book (if time allows).
5. As a user, I would like to set a preferred radius of distance for other books’ locations for my personal feed.
6. As a user, I would like to be able to change my geographic location.
7. As a user, I would like to be able to reset my personal feed algorithm.

## **Non-Functional Requirements**

Response time:

Since users will be potentially swiping through many book suggestions and reaching out to other users, it’s crucial that the response time be fast. Its server-side lag should be less than 300 ms so that users have accurate, up-to-date information about what books are available. Algorithm book recommendations and swiping should have a minimal amount of lag time to make the user experience as smooth as possible. The messaging feature should also have a very short time delay so that users can exchange information in real time.

Scalability:

We plan to design the app in a way that allows seamless integration of new features further enhancing the user experience. This will occur without consequence on the functionality of other features or errors on the client-side.

Usability:

The app will be user-friendly so users can easily navigate the interface and be able to easily manage their library, swipe books, message other users, and report concerns. The iOS app will be compatible on all iOS devices and have a seamless UI.

Security:

Security is essential to the user experience on Book Exchange. The app will not be sharing locations of users and will make sure that users will have privacy in terms of their profile information. The user’s password will also be stored in a secure manner, utilizing Firebase.

# **Design Outline**

## **Design Decisions**

We decided to create an iOS app following the client-server-database model to bring users the Book Xchange app.

## **System Components**

Components: Client, Server, NoSQL Database #1 (MongoDB)

## **High Level Overview**



## **Detailed Overview**



**System Component Interactions**

Book Xchange consists of many clients that utilize JSON to transmit personal information, user data, swiping preferences, updated messages etc. to the server (Python). The server can then update the NoSQL Firebase authentication database with user information. This information is further stored under a unique user ID in a MongoDB Database that also contains book information, and per-user book recommendation data. Furthermore, the client utilizes the UPC/ISBN LookUp API to create an auto-fill experience when uploading to their library.

# **Design Issues**

## **Functional Issues**

### What information do we need to make an account?

* 1. Option 1: Username and password only
  2. Option 2: Username, password, email address
  3. Option 3: Username, password, email address, phone number

Choice: *Option 3*

Justification: To set up an account, username and password are needed to protect a user’s account. The email will be used for extra verification and also resetting password if needed. We will also use phone numbers to send SMS texts when matches occur within the app and push notifications.

### How should users upload books into their library?

* 1. Option 1: Manually inputted by users
  2. Option 2: Autofill by ISBN number using Barcode Lookup API
  3. Option 3: Scan UPC code using Barcode Lookup API

Choice: *All Options*

Justification: In order to get matched with other users using the app, users must first upload books that they own into their library. In the case users want to upload a book that is not able to be picked up by the UPC code or the ISBN number, BookXchange allows for users to manually upload their books. Additionally, for the sake of ease, users can also type in the ISBN number or scan the UPC code, so that the Barcode Lookup API can autofill the rest of the information for the book and store it in the database. Once the information is filled in, the user will be shown to a page where they can manually edit the information if needed. With the goal to give users the most use, BookXchange provides various options for uploading books.

### How will we obtain user location information?

* 1. Option 1: Continuously access location even when users go offline
  2. Option 2: Obtain location only when users go online
  3. Option 3: Users manually enter their zip code

Choice: *Option 3*

Justification: Precise location is not needed for our app since it is catered towards Purdue Students and we will be shuffling through common location suggestions on campus for a book swap meetup. Therefore, we will just have users manually enter their zip code to ensure that they are in the general target area.

## **Non-Functional Issues**

### What platform will we use for frontend/UI development

* 1. Option 1: Flutter
  2. Option 2: ReactNative
  3. Option 3: Swift

Choice: *Option 1*

Justification: Flutter has more extensive libraries than the other two options and will be more suitable for the devices that our group has access to. Flutter will also allow us to cater to android as well as ios.

### Which programming language are we going to use for backend development

* 1. Option 1: Python
  2. Option 2: Java
  3. Option 3: C

Choice: *Option 1*

Justification: Python will be the easiest language to manage data and we will be able to use pandas which will be better suited for our needs. The syntax is also clean and our code will be easier to read.

### Which programming language are we going to use for Database Management

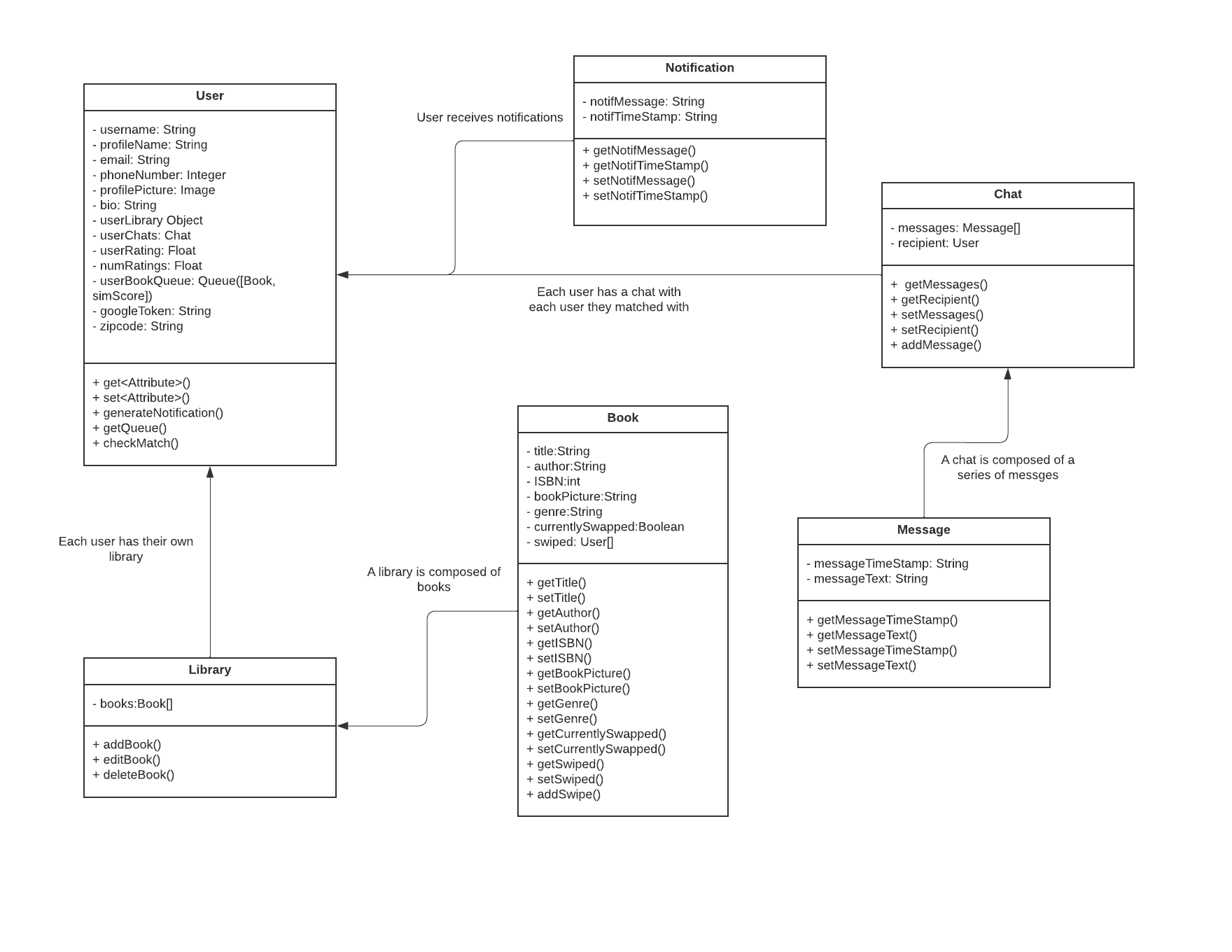
* 1. Option 1: MySQL
  2. Option 2: MongoDB
  3. Option 3: Firebase

Choice: *Option 2*

Justification: We decided MongoDB would be the best option for us because several of us are already familiar with NoSQL. Additionally, we are able to use firebase’s authentication abilities without using a firebase database. MongoDB will provide us with a place to store all of our information regarding users (except their passwords), books, libraries, and chats.

# **Design Details**

## **Class Level Design**



## 

## 

## **Class Descriptions and Interactions**

### User

* A new user object is instantiated when someone creates a new account on Book Xchange using their email, phone number, or Google Account
* Each user has a series of information prompted for them to enter upon account creation: Username, profilename, bio, zip code, and profile picture.
* The user has the ability to modify any information on their profile at any time except for their username which is unique and unchangeable.
* Once the profile is fully created, the user is able to start their library and add/remove books to it.
* The user is also able to navigate between their profile, settings, swiper, and messages tab.
* The user can also view their personal rating (dictated by other users) on their profile.

### Library

* A new library class is created when a user goes to add their first book to their account.
* The library object serves as a container for a list of book objects.
* The library object also contains a unique library ID in order to associate it with a user in the database.

### Book

* A new Book object is created when a user adds a book to their library.
* For this the user will input the ISBN, Author, Genre, picture, and rating manually or be entering the ISBN and having the information auto-complete, or be scanning the UPC barcode on the back of the book.
* This book is then added to the list of books in the library object associated with the user.
* The user can also remove/edit this book object and, therefore, change its appearance/information listed in their library.
* This book also has a list of users who have swiped right on it.

### Chat

* A new chat object is instantiated when two users match by having both swiped right on one of the books in each other's libraries.
* This chat object contains the username of the other user and a list of Messages that stores the chat history.
* The creation of a new chat also leads to a new Notification object.

### Message

* A new message object is created when a user sends a new message in a chat.
* When a message is sent it also creates a new Notification object.
* This message object contains the individual message string as well as the timestamp of when it was sent.

### Notification

* A new notification object is created when a user gets a new match or receives a new message.
* This notification object contains the notification message and the time stamp when the notification was pushed to the user.

## 

## **Sequence Diagrams**

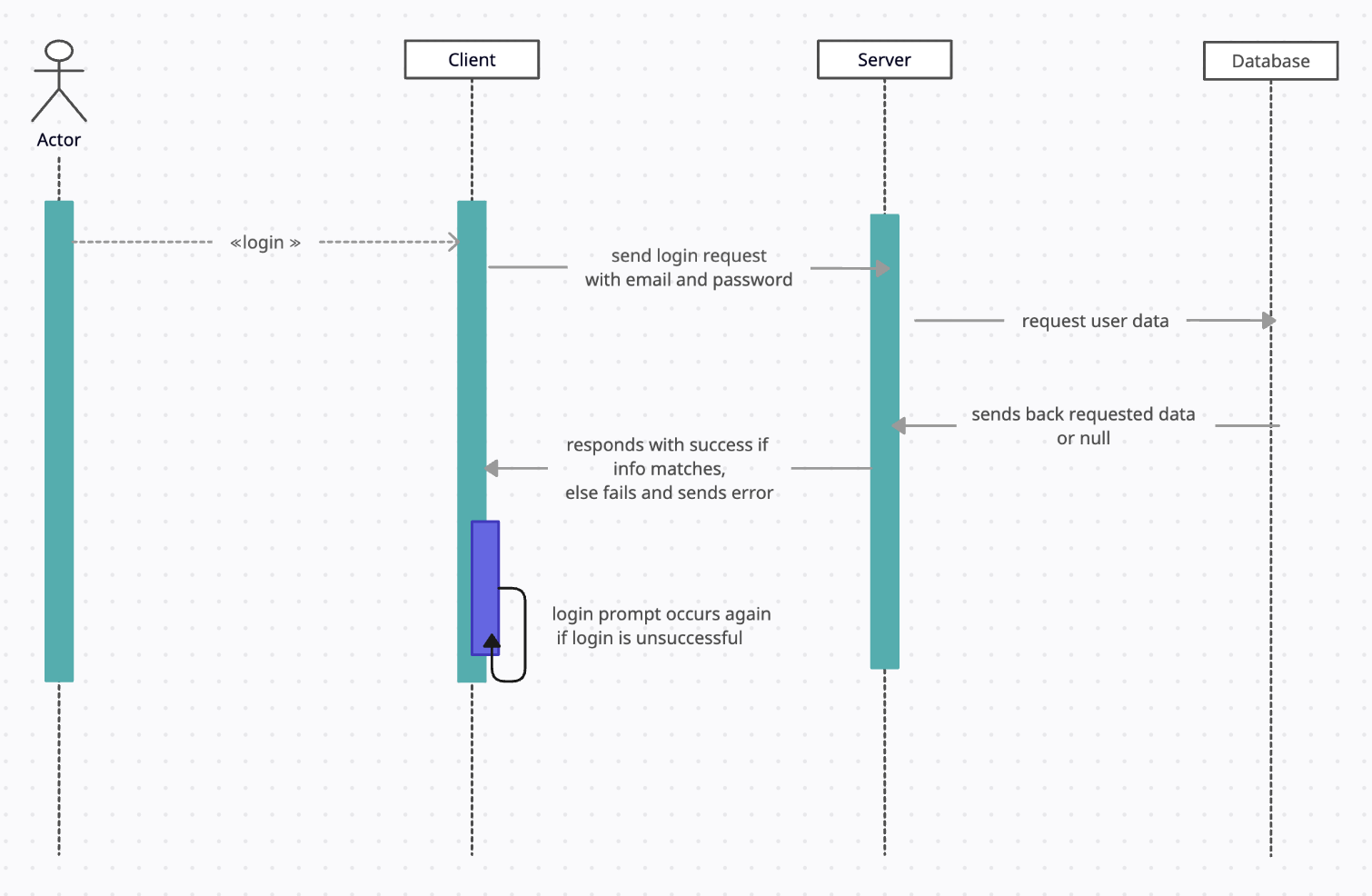
### Sequence Diagram and Explanation for Creating an Account

The below sequence diagram shows how a user would go about creating an account with Google for BookXchange. The user would attempt to create an account and would provide their Google account credentials to the app. The client sends this information to the server which in turn sends it to the database. The database then communicates to the server if the user’s credentials are valid and the creation of the account is valid. The server responds with a success to the client if the creation of their account was successful and otherwise would keep prompting the user to create their account another way, or with the accurate credentials.

### 

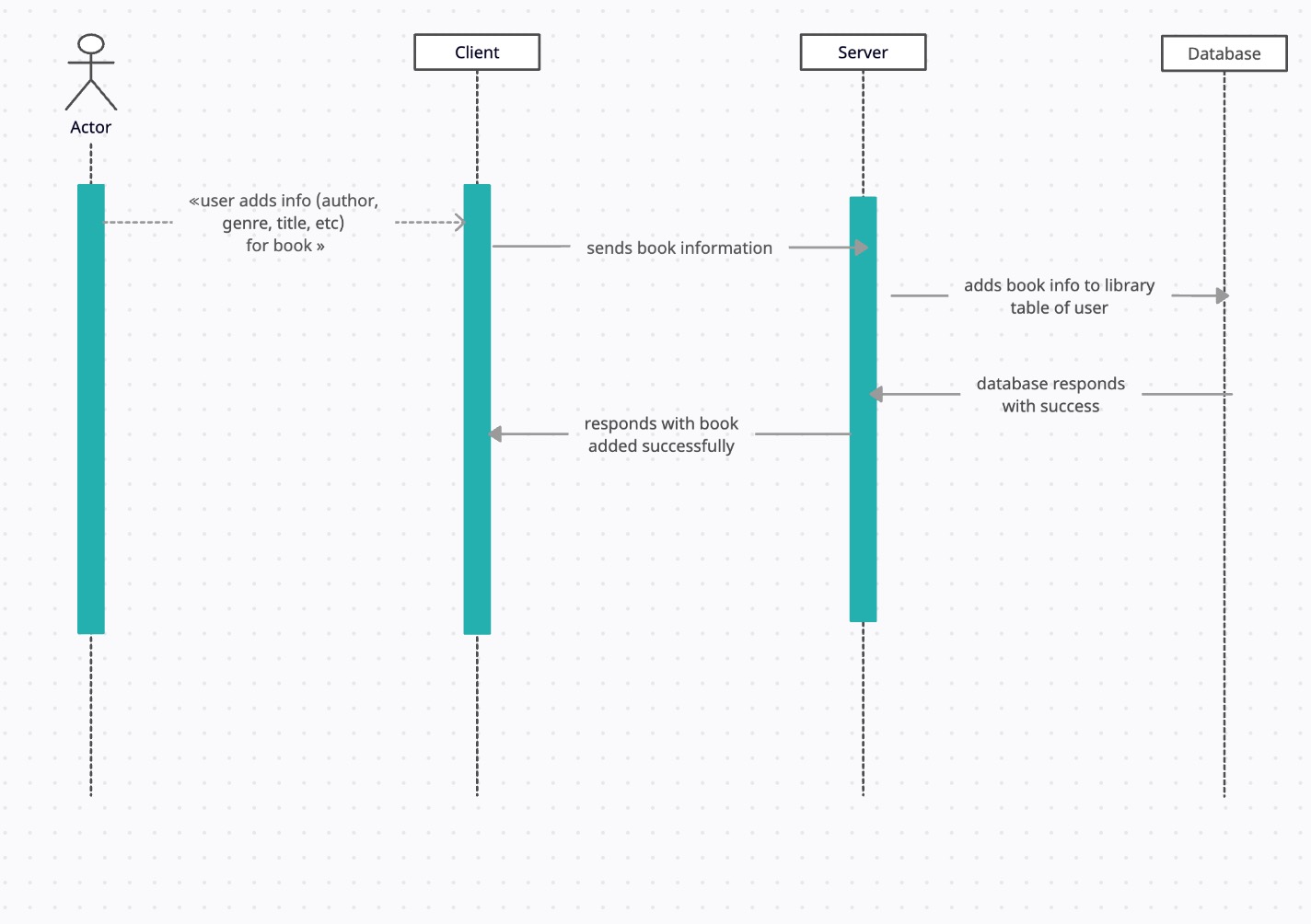
### Sequence Diagram and Explanation for Logging In

The below sequence diagram shows the process and interactions that would occur when a user goes to login to the app. First the user will be prompted to enter their login information, or login with their existing google account. The client then sends a request to the server to check if the entered user details are correct. The server, in turn, checks the information in the database where the user information is stored. The database will then respond with the user details and the server will send back to the client if the information matched or not. If the login info was correct, the user will be logged in, and if it is wrong the user will be prompted to re-enter their information. If the latter occurs, the communications will repeat as stated above.



### Sequence Diagram and Explanation for Adding a Book

The sequence diagram below goes through the process of a user adding a book into their library. The user is able to add their respective book info into their library manually, through a QR code, or ISBN. The client sends over this book information request to the server and the server communicates with the client by adding the book successfully to the user’s library tab. The server translates this information to the database by adding the credentials that the user has given into a user library table. The database then responds to the server which in turn will respond to the client with the book added successfully.

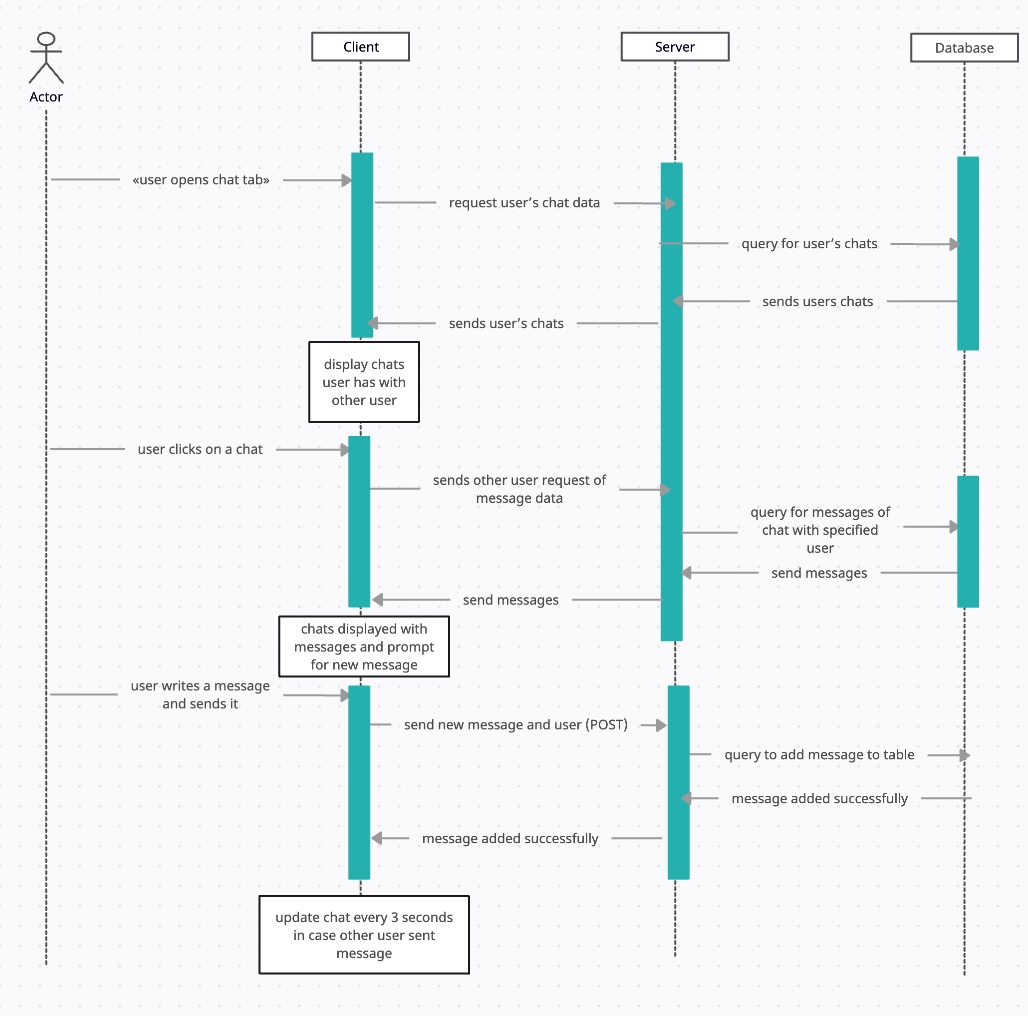


### Sequence Diagram and Explanation for Chat Sequence

The below sequence diagram shows what happens when a user interacts with another user via the chat feature. When the user goes to open their chat tab this causes the client to send a request to the server for their chat history with every user they’ve communicated with. In turn, the server references the database to get this information, and upon receiving it sends it back to the client to display them.

The diagram also shows what happens in the client-server model when a user opens a chat with an individual user. This prompts the client to send a request for the message data of that one chat history to the server, and then the server queries the information from the database. That message history, including timestamps and sender, is sent back up to the client. The user is then prompted with a message bar to engage with the other user.

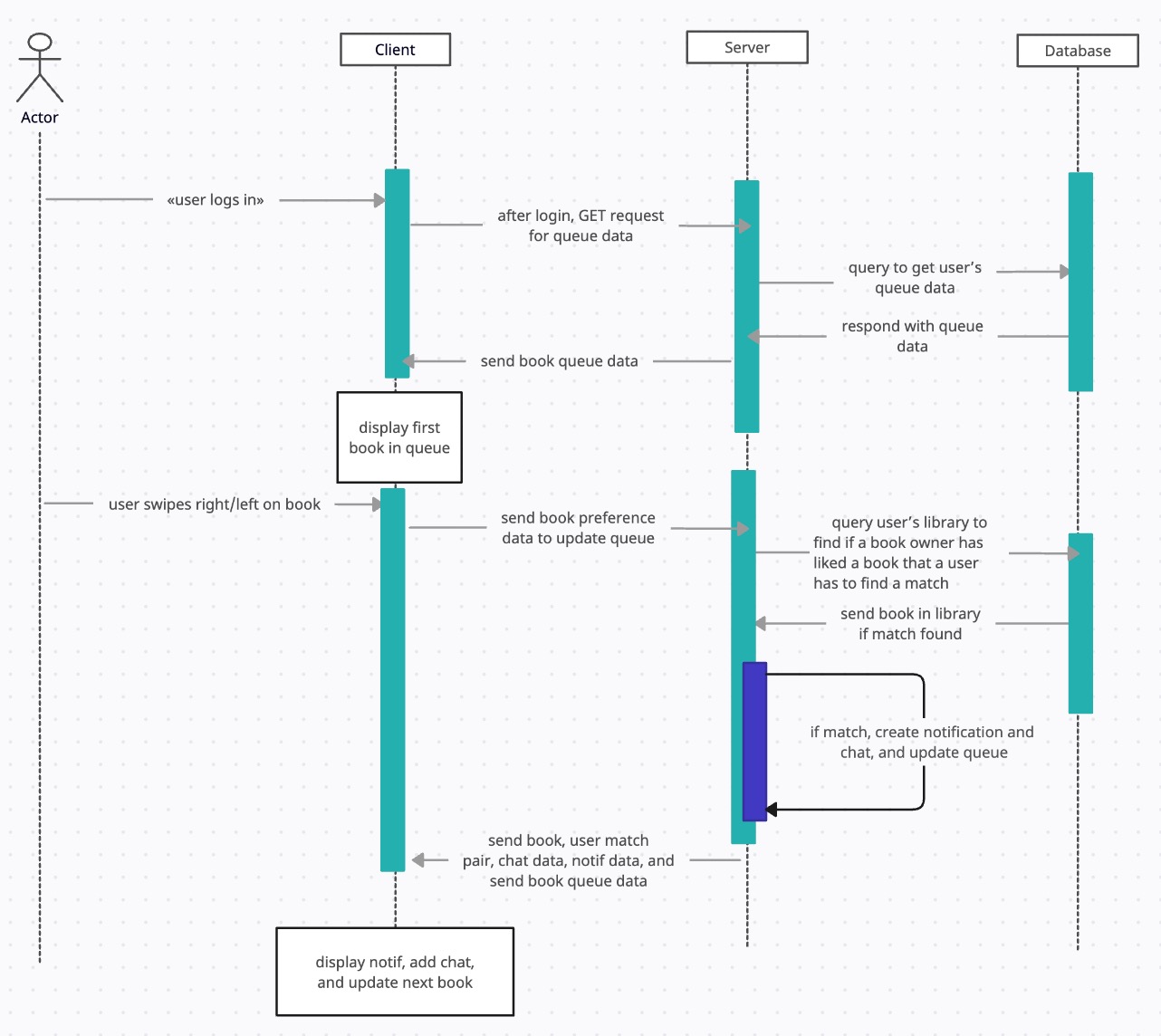
Finally, the diagram also displays what occurs when a user sends a new message to someone they’ve matched with. The client sends the message information to the server, and the server moves to add that information to the message table in the database. The database will report back success if it is added to the server, and that successful sending will also be displayed on the client. The chat for the other user is updated every 3 seconds in order to maintain up-to-date information.



### Sequence Diagram and Explanation for Matching and Swiping

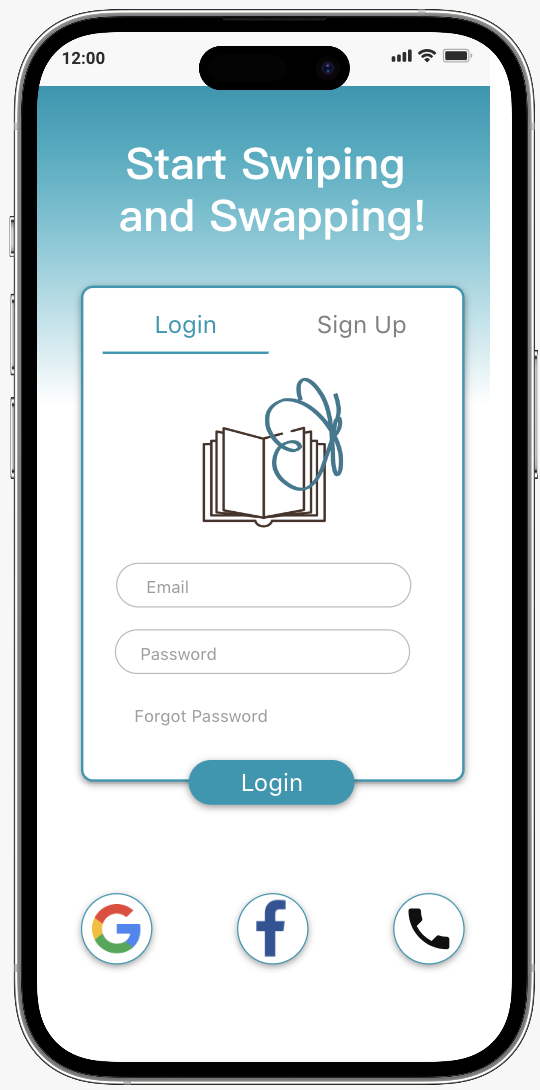
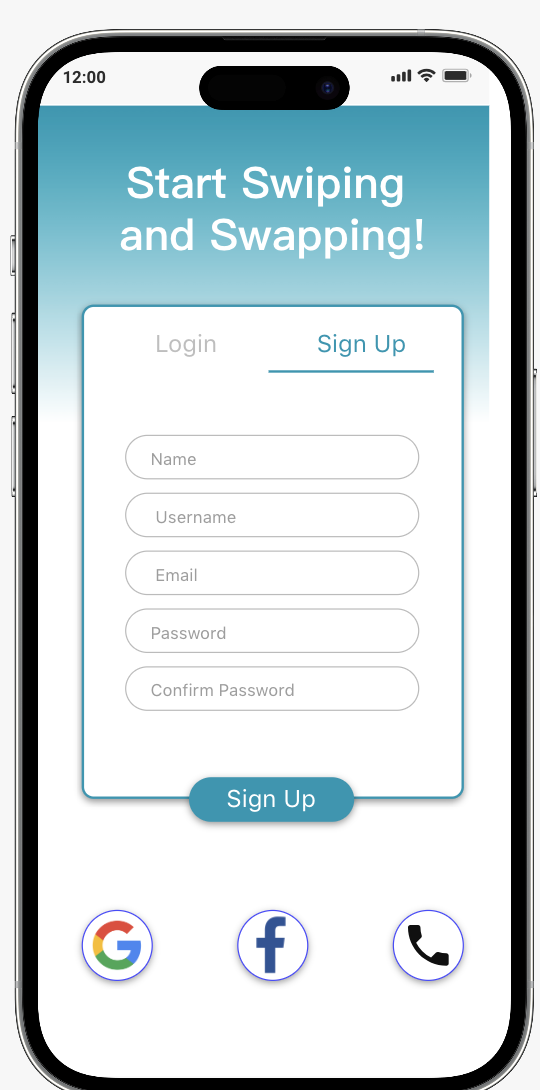
The below diagram dictates what occurs when a user logs in to the app and begins to swipe left and right on the books displayed to them. Firstly, once the user logs in the client will request the personal queue data from the server, and the server will query this information from the database. The returned data will be sent all the way back to the client and displayed for the user to begin swiping.

Once the user swipes left or right, that information is first sent from the client to the server to update the user’s queue with the new data from their swipe. If this swipe is to the right the server will then query the database to check if it caused a match. A match would occur if another user swiped right on one of the swiper’s books in their library. If that user is found to have swiped on one of the swiper’s books the match will be sent back to the server. The server will then create a new chat with the other user so the two can discuss meeting up and exchanging their books. This match is then also displayed for the client, and the next book in their queue is centered.



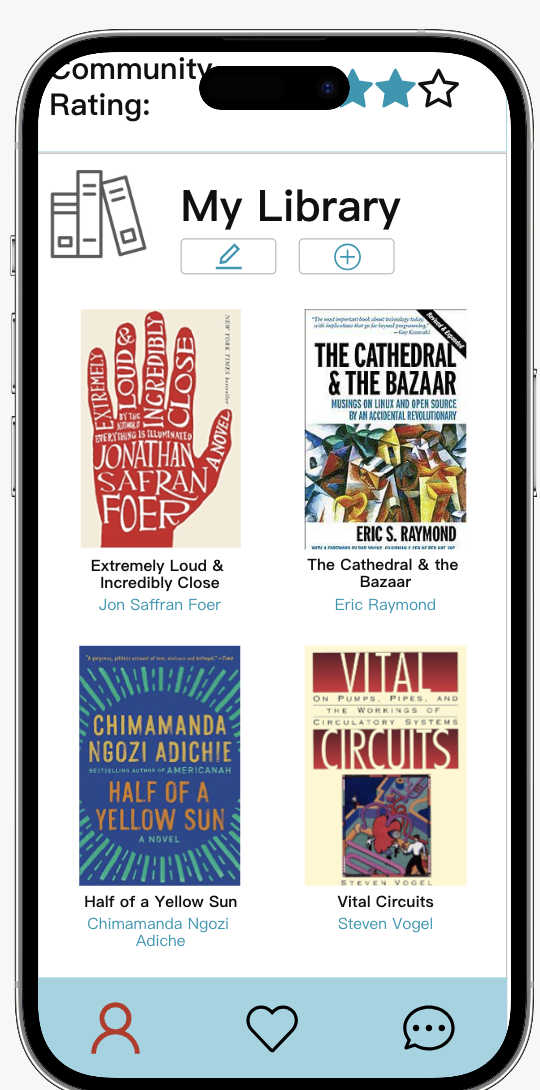
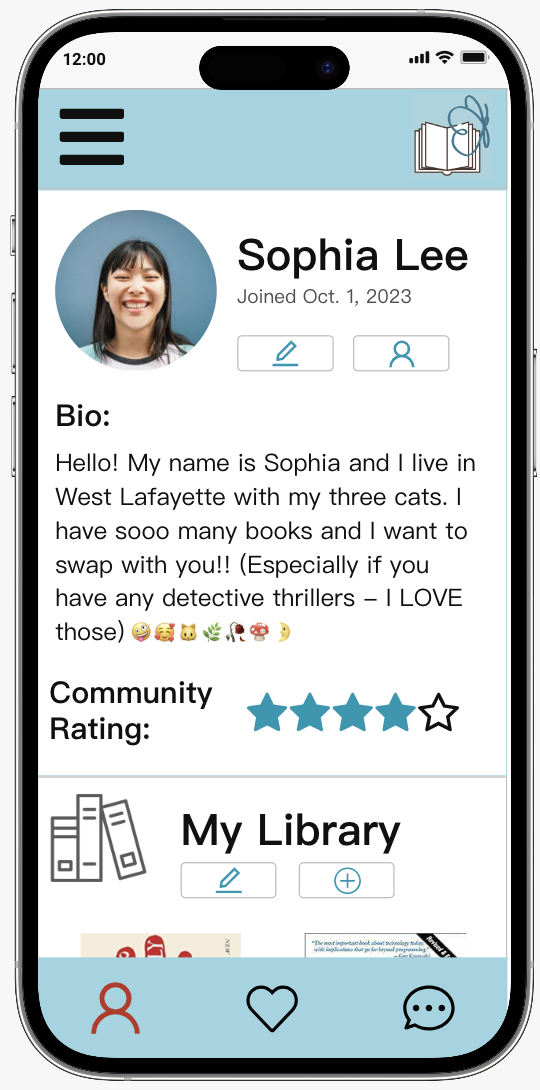
## 

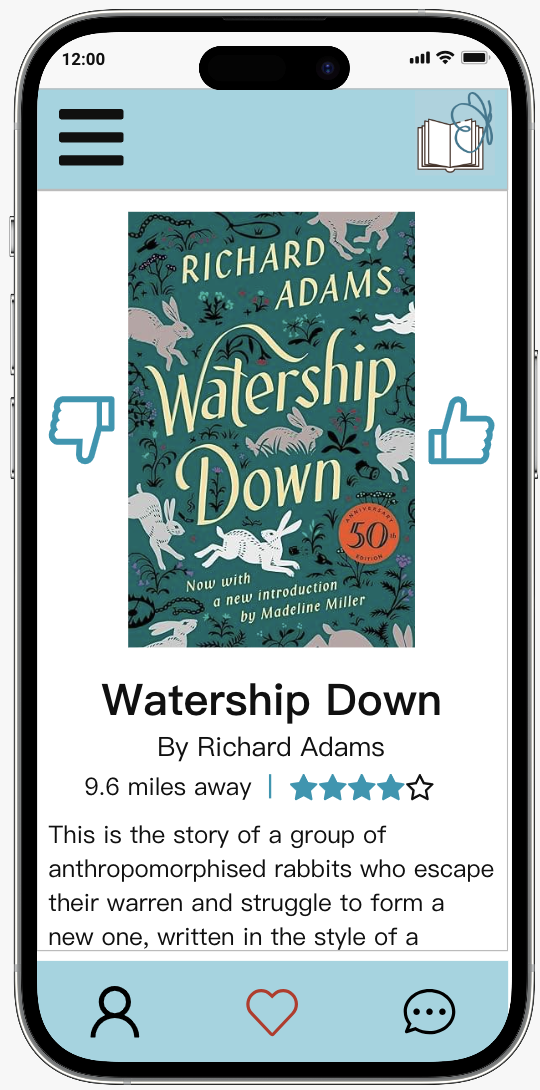
## **UI Mockups**

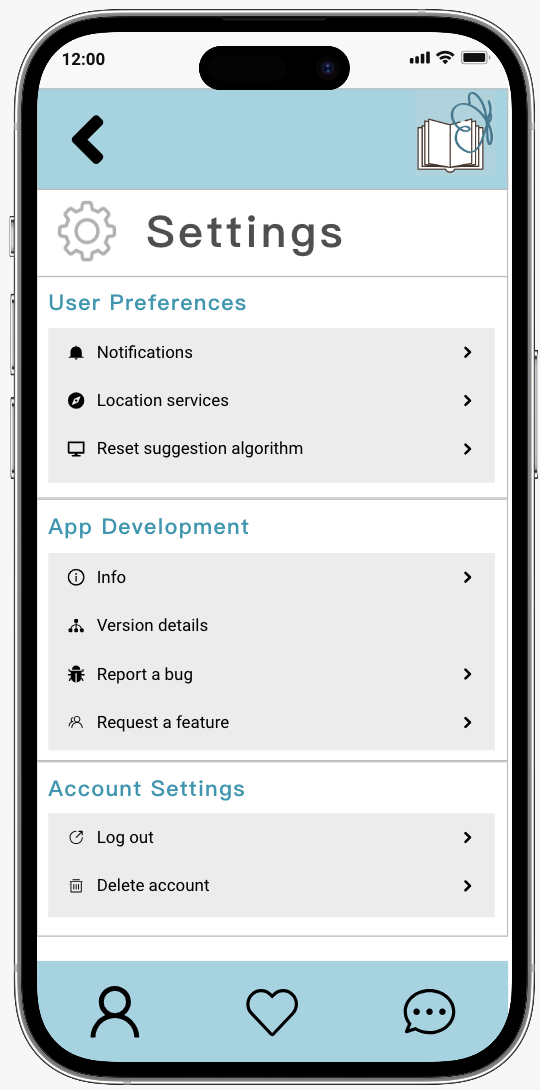
Loading Screen Login Sign Up 



User Profile User Profile (scrolled down)



Swiper Home Screen Settings



Messages Tab Chat

