

BookNet

Design Document

Team 13

Matt Hedge

Vitalii Kovtounenko

Glen Eder

Brandon Wu

Index

1. Purpose	
a. Functional Requirements	4
b. Non Functional Requirements	6
2. Design Outline	
a. High-level Overview of Components	7
b. Sequence of Events	8
3. Design Issues	
a. Functional Issues	10
b. Non Functional Issues	13
4. Design Details	
a. Class Diagram	15
b. Class Description	16
c. Sequence of Events Diagrams	18
d. Activity Diagram	22
e. UI Mockup	23

Purpose

Many students find themselves buying textbooks and using them for only a single semester. Having the right book is an important part of doing well in the classes, but the cost associated with buying new books each semester can be daunting. There are already many sources to buy the book - the university bookstore, facebook, craigslist, and amazon are some of the larger retailers where students can buy books, but there is demand within the student community for a tool that will allow students to save money by trading books with other students rather than buying.

We believe that there should be a place where students can come together to trade their books with other students - helping themselves and fellow students in the process. We're creating that place with BookNet. Using BookNet, students can trade their old textbooks with their peers in order to obtain the necessary textbooks for their next class. By allowing users to trade with one another, we've taken out the high cost and given the opportunity to connect with other students in the community. Not only can you connect with other students through trading, there's also an opportunity to form study groups with students trading for similar books or taking the same courses. While we believe BookNet is great for most students, there are times when students may need to buy a book from a retailer because there are no user books of a certain title left. In this situation, BookNet also offers a lowest price solution by finding the cheapest and nearest source of the required book.

Our purpose at BookNet is to help college students on a budget by connecting students with one another for trades and allowing students the opportunity to connect through study groups.

Functional Requirements

1. User Account

As a user,

- a. I would like to create an account on BookNet
- b. I would like to delete my account
- c. I would like to log in to my account.
- d. I would like to log out of my account
- e. I would like to add to my user profile.
- f. I would like to add a profile picture
- g. I would like to edit a profile picture
- h. I would like to add a bio to my profile
- i. I would like to edit my profile info
- j. I would like to set and change my account password

2. Book Searching

As a user,

- a. I would like to input what book(s) I need
- b. I would like to search for a book by its title
- c. I would like to search for a book by its author
- d. I would like to search for a book by its ISBN
- e. I would like to search for books for sale
- f. I would like to search for books for donation

3. Book Trading

As a user,

- a. I would like to input what book(s) I have available for trade
- b. I would like to see a history of my previous book trades
- c. I would like to see with whom I previously traded
- d. I would like to be matched with other users that have the book I need
- e. I would like to specify trading location
- f. I would like to accept a trade
- g. I would like to reject a trade
- h. I would like to be able to cancel a trade
- i. I would like to see who my potential trading partners are

4. Book Purchasing

As a user,

- a. I would like to purchase a book at the cheapest price if trading is not an option
- b. I would like to know where the nearest bookstore with the book I am looking for is located

5. Rating

As a user.

- a. I would like to input the condition of the book I am giving away
- b. I would like to be able to rate the condition of the book I received in a trade
- c. I would like to be able to see other users' book condition ratings

6. Notifications

As a user.

- a. I would like to receive a notification on BookNet's website when a trade is available
- b. I would like to receive an email notification when a trade is available

7. Study Groups (If time allows)

As a student.

- a. I would like to be able to create a study group for one of my classes
- b. I would like to enter study groups with students taking the same classes
- c. I would like to chat with other members of my study group

Non Functional Requirements

1. Server Requirements

As a developer,

- a. I want the server to handle real time server client communication
- b. I want the server to properly store data in the database

2. Performance

As a developer,

- a. I want the web app to support 10,000 users
- b. I want the web app to run without hanging or crashing
- c. I want the webpage to load within 3 seconds

3. Appearance

As a developer,

- a. I want the user interface to be responsive
- b. I want the user interface to be attractive and intuitive

4. Security

As a developer,

- a. I want the user's private information to be encrypted between the database and the UI
- b. I want users to only have their university account linked to their profile
- c. I want trade and user chats to be encrypted

5. Usability

As a developer,

- a. I would like the application to be intuitive to use
- b. I would like the application to run on all web browsers

Design Outline

High Level Overview of Components

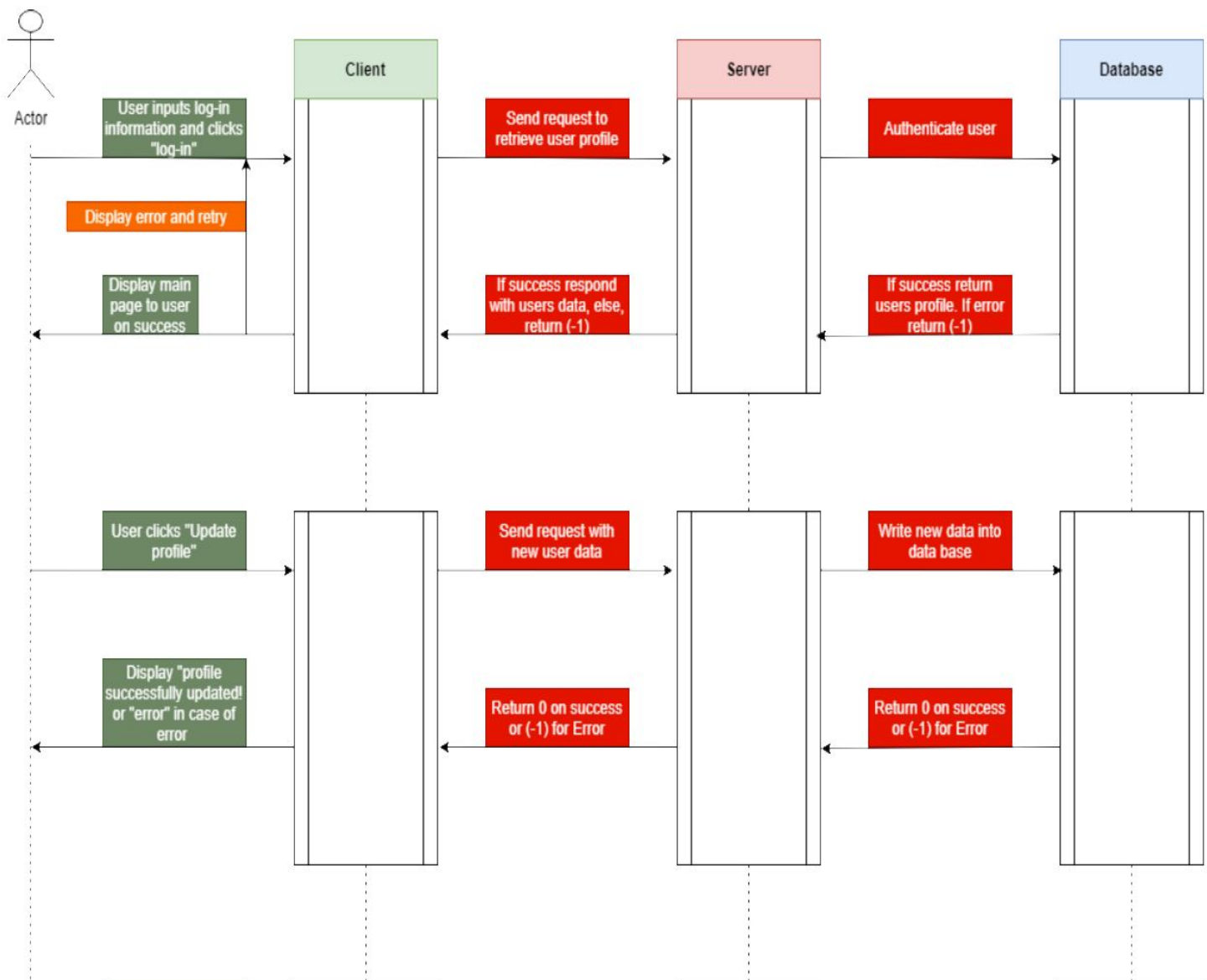
This project will be a web application that allows users to connect with each other and trade books that they are in need of. This application will use the client-server model where one server handles matching users based on client needs. The server will handle client requests, database calls, and server-to-client messages.

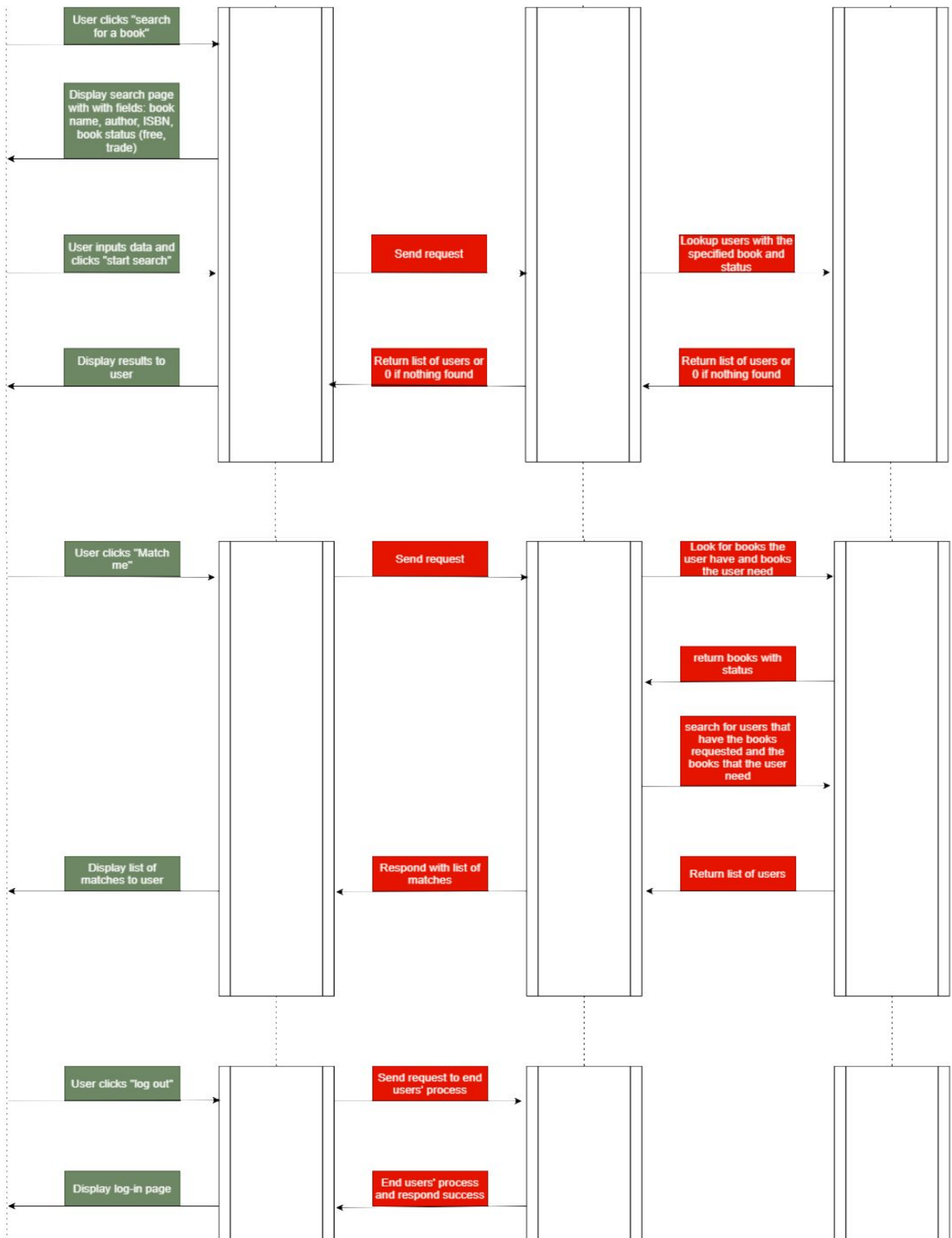


1. Client
 - a. Client will provide an interface for users.
 - b. Client will send requests to the Firebase server.
 - c. Clients will interpret responses from the server and display necessary information.
2. Server
 - a. Receives and verifies requests from the client.
 - b. Handle necessary database calls to gather requested information.
 - c. Return information in JSON format to client.
3. Database
 - a. Database will store all data for BookNet.
 - b. Standard data in the database will be users' info, the books they have, and the books they are in need of.
 - c. Database will respond to the calls from server by sending requested information to the server
 - d. Database will ensure no loss of data gathered from users.

Sequence of Events Overview

The sequence diagram shows the typical interactions between the client, server, and database. The sequence begins with the user accessing the web application. The user will input login information which then the client will make a request to the server to login. Upon verification, the server will query the database for user information that will be sent back to the user. After being logged in, the client will make requests to the server for actions such as: updating user information, inputting books, checking for matches, and sending messages. The server will send queries to our database to get the relevant information required for each request from the client. The database will return up to date data to the server which will then be sent out to the client.





Design Issues

Functional Issues:

1) What information do we need to create an account?

- **Option 1:** First name, last name, university, academic email, password
- **Option 2:** First name, last name, address, phone number, university, major, academic email, password.
- **Option 3:** First name, last name, user name, phone number, university, major, academic email, password.

Choice: Option 1 - First name, last name, university, academic email, password.

Justification: We would like it to be as simple as possible for the user to establish an account. Therefore, we ask only for necessary information that will help us to identify the user and keep his account secure. We are using the academic email in order to prevent people who are not college students from creating an account, and to prevent the creation of multiple accounts by the same user.

2) What Information do we need for signing into an account?

- **Option 1:** Academic email, password
- **Option 2:** User name, password
- **Option 3:** User name, academic email, password

Choice: Option 1 - Academic email, password

Justification: For logging into an account we need to identify the user. We will use a password to increase the level of security and authentication. The academic email will serve as another layer of authentication. This will allow us to verify a user, and use the users' email to change or recover the password if forgotten. Since the academic email is unique for every student, it is not necessary for the user to specify a username which will make it more complicated for the user.

3) How will the user rate the quality of the book?

- **Option 1:** Star rating; the user will rate the quality of the book using the star rating model. 1 star is the lowest quality and 5 stars is the best quality
- **Option 2:** Condition description rating; the user will specify the condition of the book. The rating will be based on very poor, poor, fair, good, like new. "Very poor" is the lowest and "like new" is the highest. Additionally the user will be able to add a comment on the condition.

Choice: Option 2

Justification: A condition description rating will be less ambiguous than a star rating. It will be more accurate and close to reality. A user will be able to describe their book in much better detail and the comments allow a user to even say something like "the first page is missing." This should help to increase user satisfaction with the books they are receiving.

4) What should be done if no matches are found?

- **Option 1:** Message letting user know that no trade is available
- **Option 2:** Direct user to the web page of the retailer with the cheapest copy
- **Option 3:** Give user directions to a local bookstore with a copy available

Choice: Option 2 and 3

Justification: We want to make the entire process of obtaining textbooks easier for users. This means that even if there is no trade available, we want to give the user a way to obtain the book they need. Therefore, we have decided to incorporate options 2 and 3 depending on the demand of the user. If the user chooses to buy online, they will be directed to the webpage of the trusted online retailer with the cheapest price on the requisite book. If they choose to buy locally, directions to the local retailer with a copy available will be given.

5) What should the trading process look like?

- **Option 1:** Discussion forum where users can post books they are looking to get or looking to trade away
- **Option 2:** Matching system that pairs users with a good potential trade together and notifies each user of their partner's email
- **Option 3:** Option 2 AND real time chat between the pair of users

Choice: Option 3

Justification: A discussion forum would simply become disorganized and chaotic with an increased number of users. Therefore, using a matching system that pairs users together

efficiently is important. We also wanted to simplify the process of setting up a trade meeting so a real time chat service between traders will be helpful. Users can converse with each other about where and when to meet and about further details regarding the condition of the book.

6) How should users be matched up for trading?

- **Option 1:** BookNet assigns user pairs
- **Option 2:** Users receive a list of potential partners and they can choose who to trade with

Choice: Option 2

Justification: In order to give the user freedom to trade with who they want, we will be using option 2. This gives the user more confidence that they are receiving the book in the condition they want. This is because they will be able to read the book condition comments that other users have written for the books they are looking to trade.

Non Functional Issues:**1) What platform should we use to host our web application?**

- **Option 1:** Amazon Web Services
- **Option 2:** GitHub Pages
- **Option 3:** Heroku

Choice: Heroku

Justification: We chose Heroku because we need to implement a dynamic website. In order to implement a dynamic website we need a hosting service that will support server-side functionality. AWS and Heroku both support Node.js for server side development. However, Heroku gives limited free services that are suitable for our project. On the other hand, AWS is a paid service and GitHub pages only supports development of static web pages that don't require server-side functionality.

2) What language should we use for our frontend?

- **Option 1:** HTML + CSS
- **Option 2:** AngularJS
- **Option 3:** Javascript (ReactJS)

Choice: ReactJS

Justification: React has many benefits on the development side that we aim to take advantage of. Chief among these advantages is code reusability. React uses components that can be copied and placed anywhere, which makes sharing code between developers easier and makes having to redevelop code for similar functions redundant. This is especially important as BookNet incorporates many similar functions such as trading, buying and donating. Another crucial advantage of React is its speed. ReactJS uses a virtual DOM (Document Object Model) system that detects when data changes and the UI needs to be reloaded. This greatly improves speed and will provide a smooth and responsive user experience.

3) What language should we use for our backend?

- **Option 1:** Java
- **Option 2:** Python
- **Option 3:** Javascript

Choice: Javascript (specifically NodeJS)

Justification: Most IDEs support NodeJS development and plugins, giving us the freedom to choose our platform. NodeJS is one of the most popular server-side technologies, which makes it easier to find information if needed. It also has the power of full stack Javascript development, providing us with many free tools, code sharing and reuse, efficiency and performance. Finally, due to its Javascript base, it also meshes with our choice of frontend in ReactJS.

4) What database should we use?

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

Choice: Firebase

Justification: BookNet depends heavily on a fast and secure database since we need to store and retrieve information for the functions of trading, buying and donating. Firebase offers many useful functions such as user authentication. It also meshes well with our plan to use ReactJS for the client side, and being a fast real-time database enables us to handle logging books from all our users. Firebase is also extremely fast to set up which will be crucial for BookNet's development timetable.

5) How should we implement our chat service?

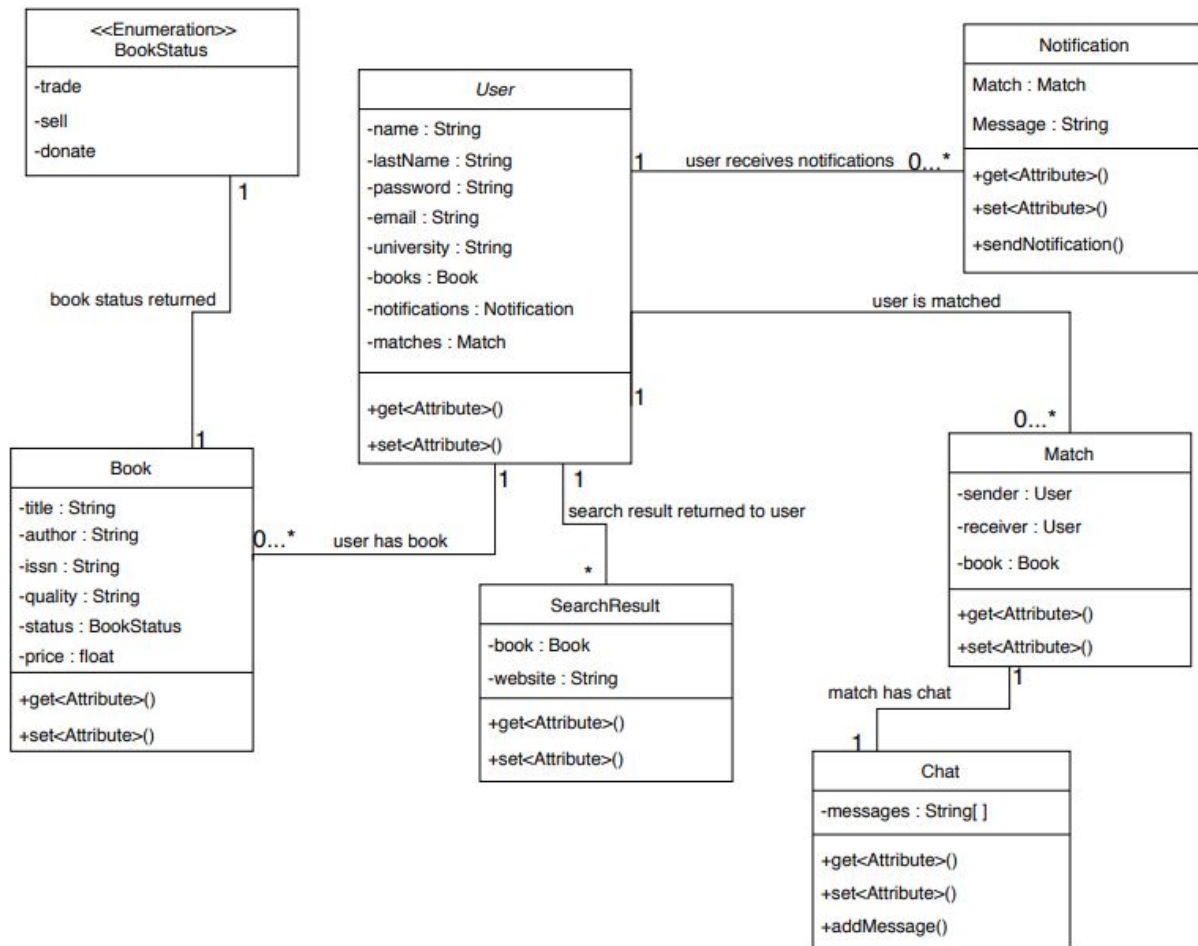
- **Option 1:** FireChat
- **Option 2:** Socket.IO

Choice: FireChat

Justification: We had originally thought about using Socket.IO due to its relative simplicity, allowing us to focus more on the book-related aspects of the project. However, as user count increases, the load on the server it is running on will quickly become too heavy. Since we aim to support a multitude of users, FireChat's stability with increased user count is a great advantage. It also becomes easier to implement as we gain increased familiarity with FireBase, our group's choice of database.

Design Details:

Data Class Level Design



Description of Classes and Interactions Between Classes

Classes were designed by determining the objects that would be needed within our web application. The descriptions of individual classes and interactions with connected classes is shown below.

User:

- A user object is created when someone registers on our web application
- Each user will receive a unique userID upon registering
- Each user will have their name, last name, password, email, and university linked to their unique userID
- Each user will authenticate identity using their password before receiving access to their homepage
- Each user will have books listed with their account that can be current books for trade and current books needed
- Each user will receive notifications sent when a trade match is found
- Each user will be matched to other users based on tradeable books and books needed

Book:

- A book object is created when a user begins to input their books available for trade
- Each book object will have an author, issn, quality, and status available for other users to see
- Each book will have an enumeration book status, showing what the user would like to do with the book
- Each book will have a price based on lowest-cost book retailer prices that will only be used if a certain book is not available through BookNet or not available at a high enough quality

Book Status:

- A book status is created when a user creates a book object
- Each book status will show whether the book is listed by the user as tradeable, sellable, or donatable

Search Result:

- A search result is created when a user types into the search bar on the UI application and hits the search button
- Each search result will return to the user the name of the book searched for, whether matches were found within the BookNet database and if not, a list of lowest cost retailer prices and links to the retailer's website

Match:

- A match is created when a user known as the sender has an available book that another user known as the receiver needs and the receiver also has a book the sender needs

- Each match will be determined by an algorithm and a particular user may have 0 matches or several book matches to choose from
- Each Match, when accepted by both users, then will then open up to a chat application between the two users

Chat:

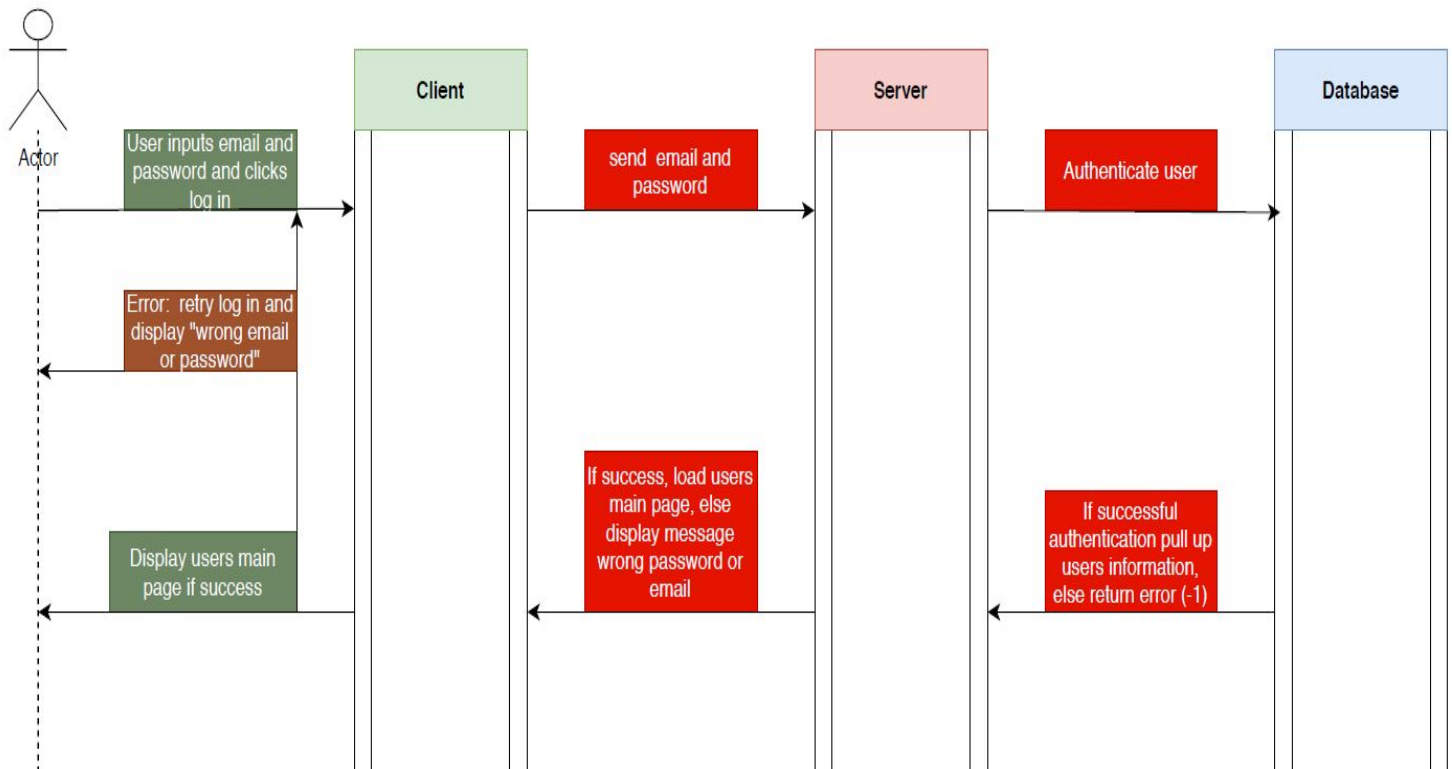
- A chat is created when two users have accepted a match with one another for a particular book
- Each chat will allow the users talk with one another to determine whether to move forward with the trade and where to meet

Notification:

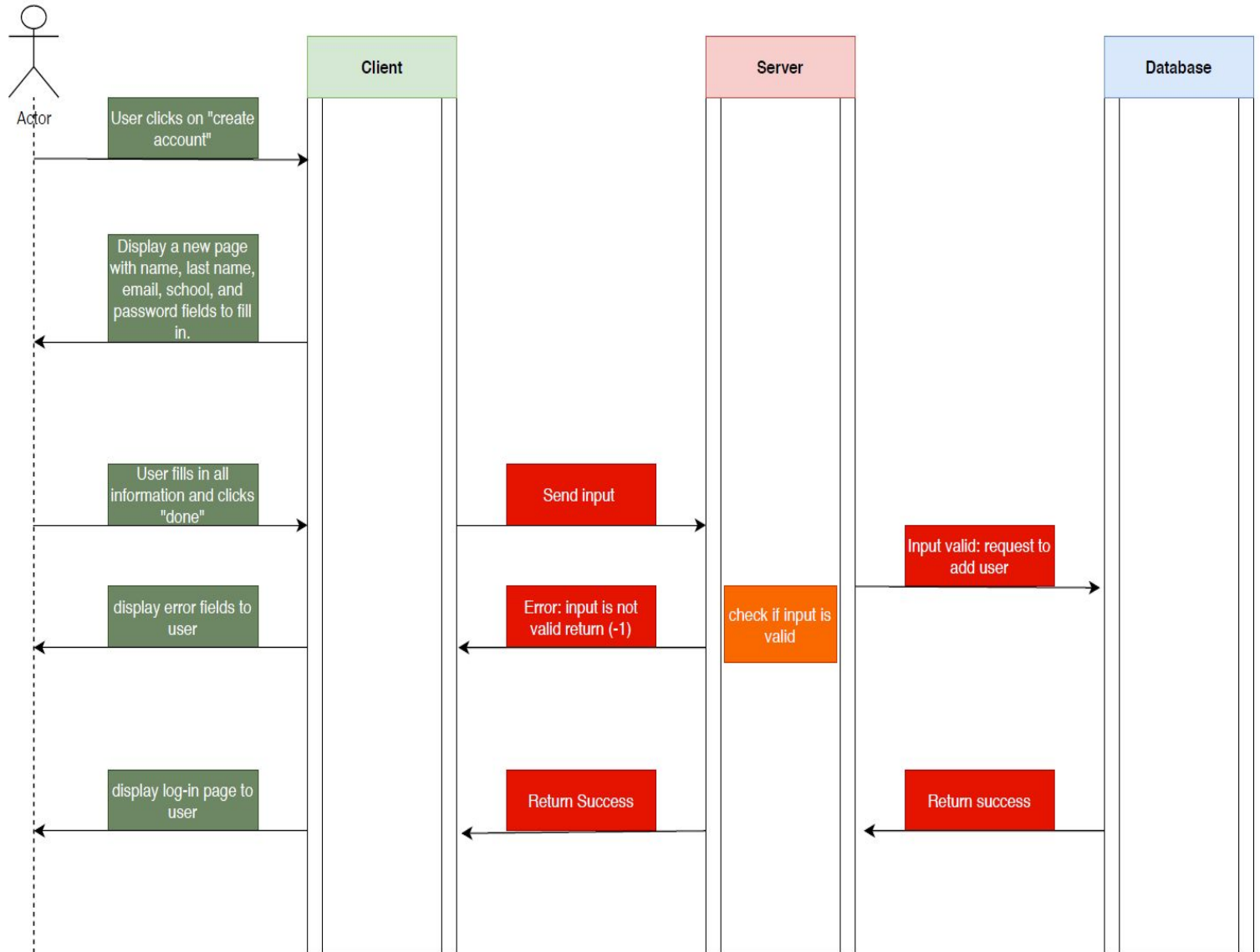
- A notification is created when two users are listed as a potential match for trading and also when both users have accepted the match
- A notification alerting the user of a potential match will give the user the option to accept or decline the match
- A notification alerting the user that a match has been accepted will prompt the the user to open up a chat with the match to begin communication

Sequence Diagram

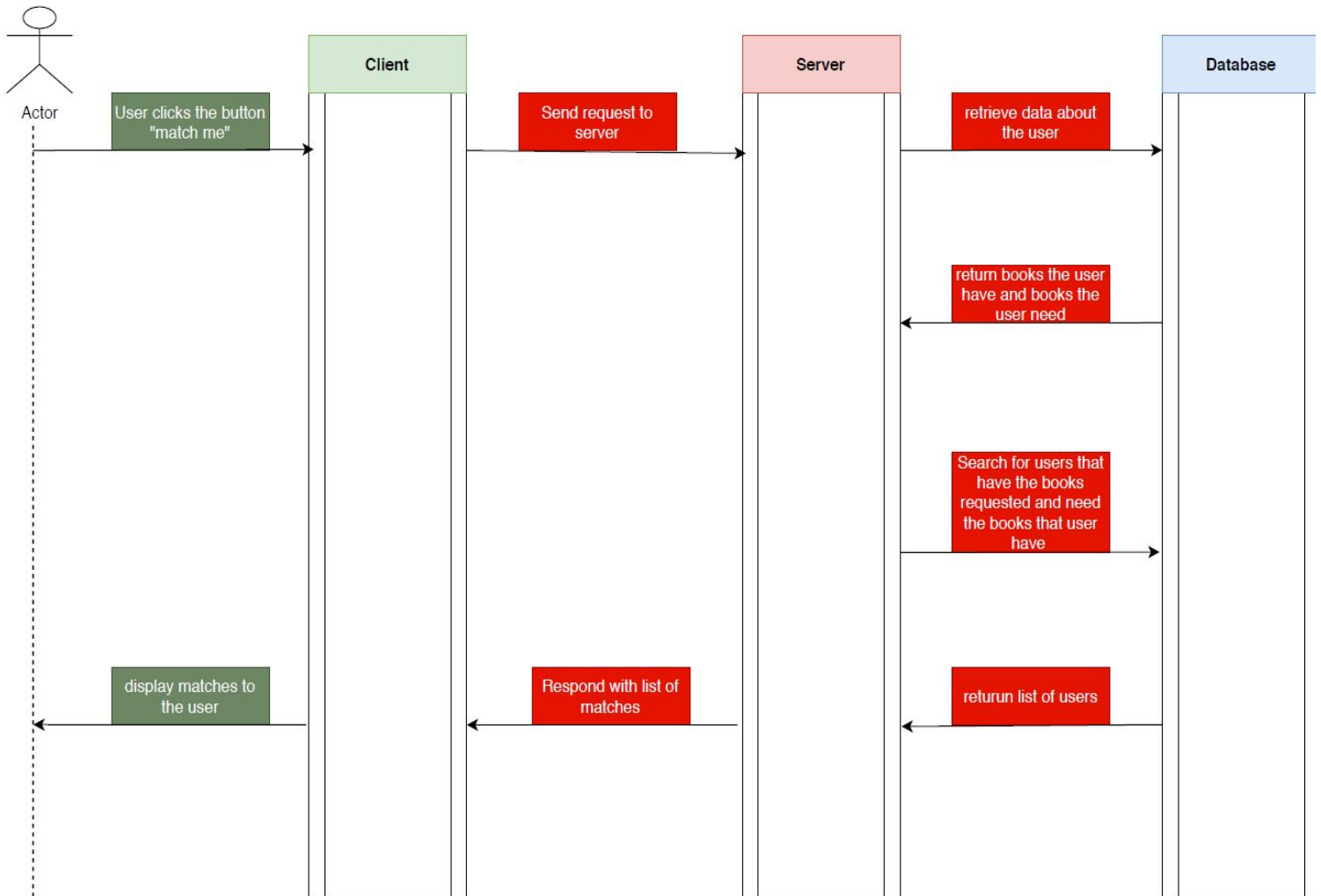
Sequence of events the users logging-in



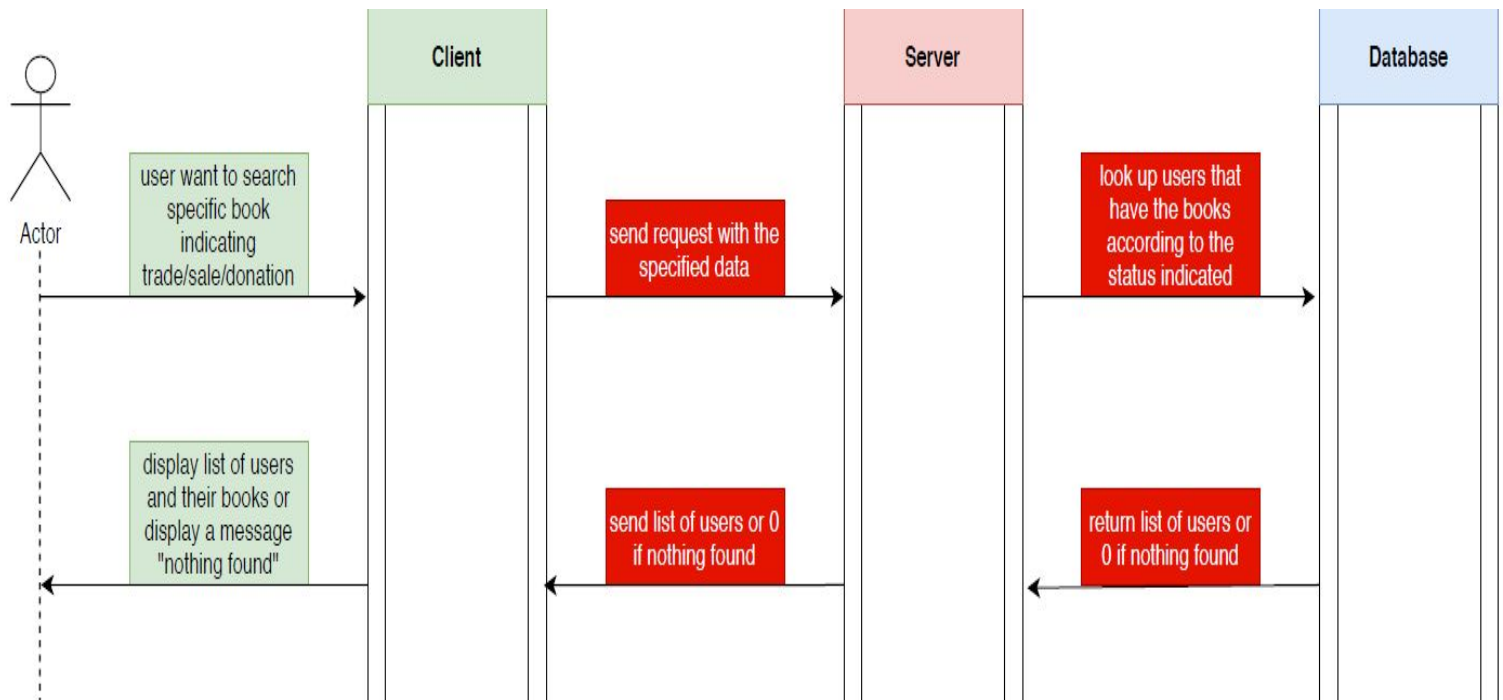
Sequence of events for creating an account



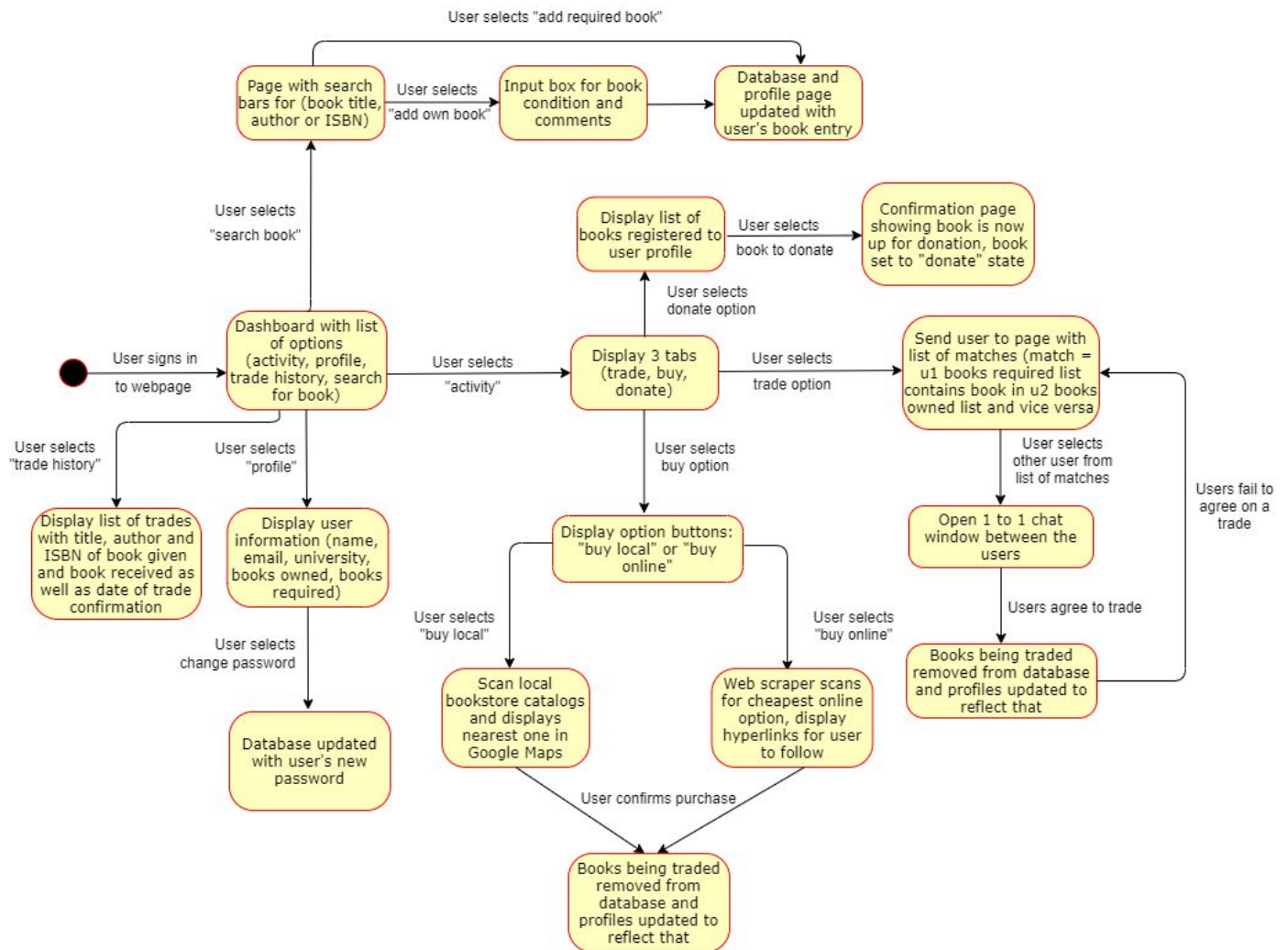
Sequence of events when a user looking for trading



Sequence of events when the users search a book

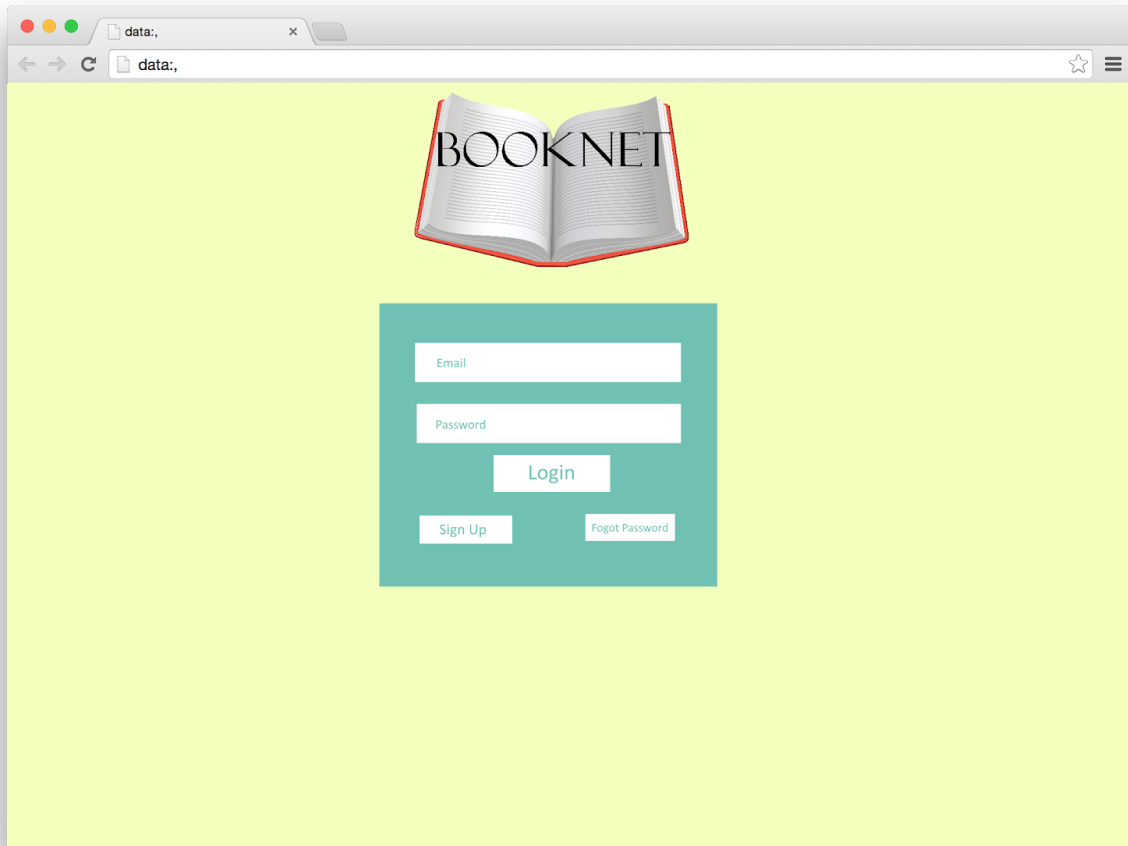


Activity/State Diagram

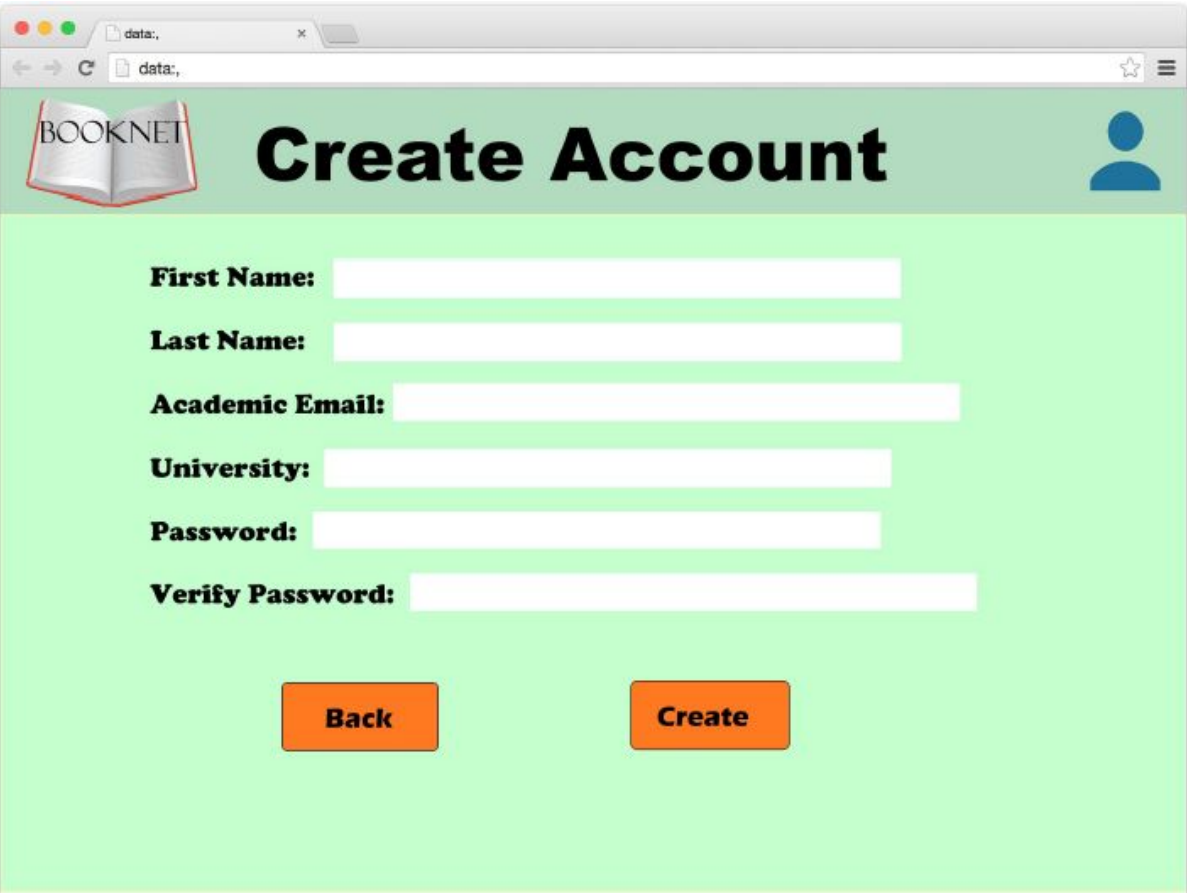


UI Mockup

The goal with BookNet's UI is to be clean, simple and easy to navigate. We plan to limit the amount of pages to load for a more seamless experience for the user.



This is the sign-in page that the user will first land on when coming to the website. From here the site requests the users email and login password. If they don't have an account they can sign up for an account, as well as look up a forgotten password.



BOOKNET

Create Account

First Name:

Last Name:

Academic Email:

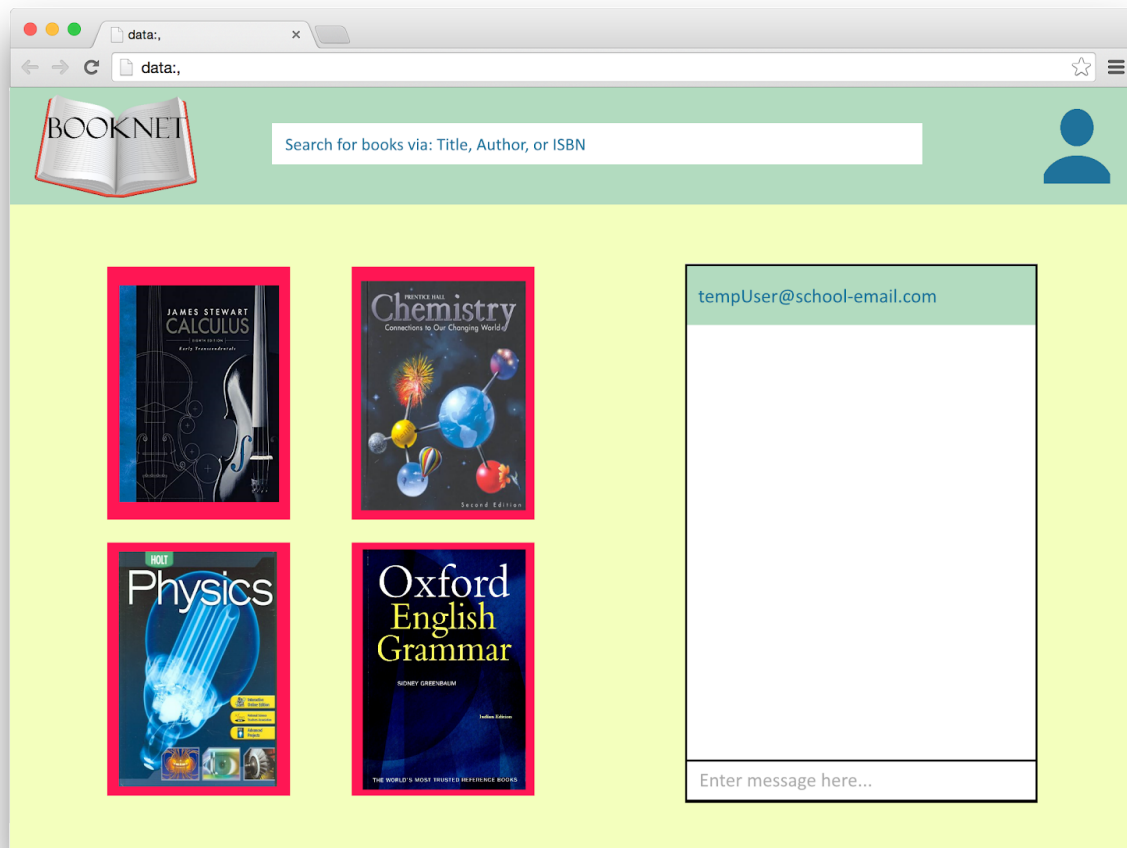
University:

Password:

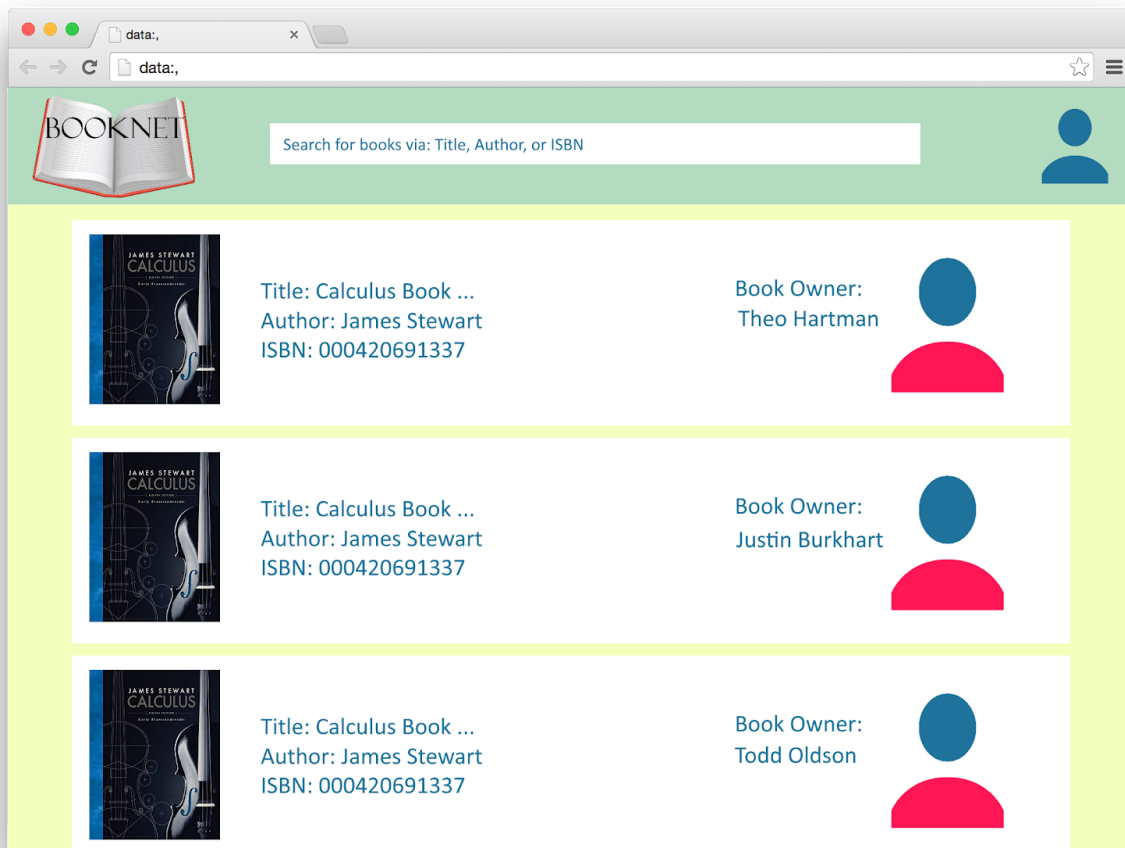
Verify Password:

Back **Create**

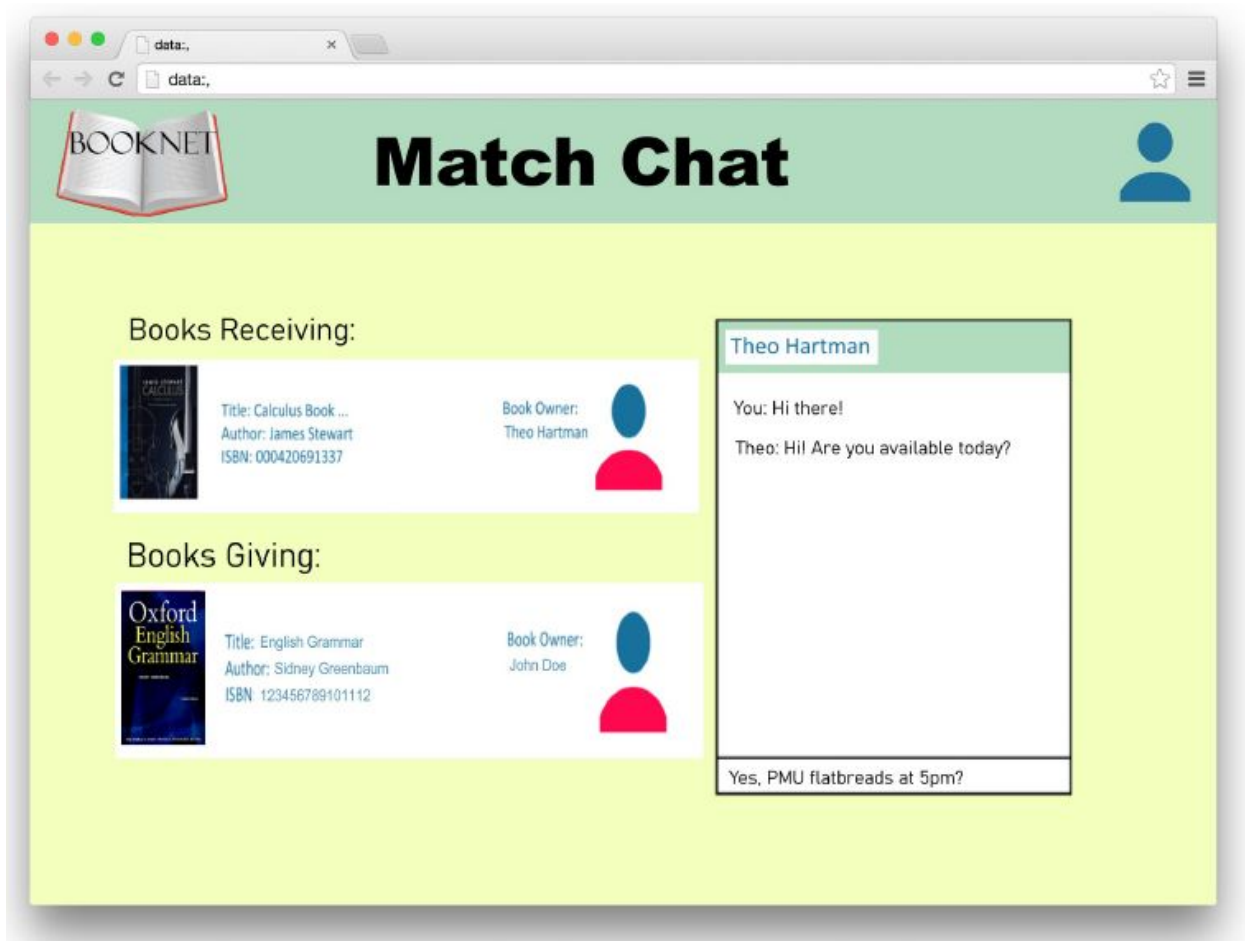
This is an example of when the user clicks the sign up button. The user will be prompted for their first name, last name, email, university and then will verify their selected password. The information provided will be verified against the database information to make sure there isn't already a user with the same information and then the account will be created.



This is an example homepage. The books on the left are the books that the user is looking for and are outlined in red to represent that there is no match for a trade yet. The background behind the book will change to green upon finding a match. To the right is a chat window that allows the user to communicate with trade partners. The profile icon in the top right will be used to access user information for editing.



This is a draft of our search result screen. The screen will display all books that match that are up for trade. The books title, author, and ISBN is displayed along with the books image to verify it is the book the user is requesting. On the right side of the result is the book owner's name and profile image.



This is a draft of our match chat page. The chat window is on the right side of the screen and this page is available when both users have matched for a trade. On the left side of the screen will be listed the books that will be given to and received from the other user. Chat allows users to set their own times and places to meet and exchange books.