

Boilerbazaar

CS30700

Design Document

Team 30

Dung “Ryan” Doan

Xavier Huu Pham

Jeffrey Yian Wang

Shicheng Fang

Michio L Sekiguchi

Index

Purpose	3
Functional Requirements	3
Non-Functional Requirements	5
Design Outline	7
Components	7
High-Level Overview	8
Sequence of Events Overview	8
Activity/State Diagram.....	10
Design Issues	11
Functional Issues.....	11
Non-Functional Issues	13
Design Details	15
Class Design	15
Description of Classes and Interaction between Classes.....	15
Sequence Diagram	18
Use Case Diagram	21
UI Mockup	22

Purpose

The use of textbooks plays an important role during a student's years at Purdue University. When purchasing a textbook, it might be hard to find a website that sells your textbook, and the price of a textbook could be very expensive. With the high price point of new textbooks, students often spend time searching for better deals, either through different vendors or individual sellers. It also might be hard to find the specific textbook for your class due to the number of different versions there are.

The purpose of this project is to develop a textbook marketplace web application for Purdue students. BoilerBazaar will scrape the web for textbooks that the student searches for and will list the different websites that are selling the textbook with the cheapest option highlighted. Moreover, it is also a marketplace where students can buy and sell their own new/used textbooks. The buyers and sellers can also give a rating to each other depending on the buying/selling experience. All in all, this web application helps students find textbooks they need fast and at an affordable price point.

Functional Requirements

1. User account

As a user,

- a. I would like to be able to login through Purdue authentication.
- b. I would like to be able to see past messages.
- c. I would like to be able to view my profile.
- d. I would like to be able to edit my profile.
- e. I would like to be able to view the profile of sellers.
- f. I would like to change between light and dark mode.

2. Students can search for textbooks

As a student,

- a. I would like to search for a textbook based on author, ISBN, or title.
- b. I would like to be able to sort textbooks by price, popularity, ratings, etc.

- c. I would like to be able to see a list of websites that are selling the textbook that I searched for with the cheapest price highlighted.
 - d. I would like to be able to search for used textbooks from other students.
 - e. I would like to be able to easily access the websites selling the textbooks.
3. Students can buy textbooks from other students
- As a buyer,
- a. I would like to be able to easily message the seller.
 - b. I would like to scroll through images that a seller is trying to sell.
 - c. I would like to be able to click on seller to see seller profile when trying to buy a textbook from them.
 - d. I would like to be able to filter through the student sold textbooks by course, edition, etc.
 - e. I would like to share a seller's listing.
 - f. I would like to be able to see other textbooks that are related to the current listed textbook.
 - g. I would like to be able to give seller ratings based on interaction and ease of purchase.
4. Students can sell textbooks to other students
- As a seller,
- a. I would like to be able to list textbooks for sale.
 - b. I would like to be able to add/remove pictures of what I am selling.
 - c. I would like to be able to list the title, price, condition, and description of the textbook.
 - d. I would like to be able to edit or delete my textbook listings after submitting them.
 - e. I would like to be able to set a meetup preference (public, in front of house, etc.).
 - f. I would like to be able to edit the status of a transaction for a book I am selling (available, pending, completed).
 - g. I would like to give reviews about my buyers.

- h. I would like to be able to view reviews from my buyers.

Non-Functional Requirements

1. Performance

As a developer,

- a. I would like to send and receive messages within 5 seconds.
- b. I would like to search and list textbooks within 10 seconds.
- c. I would like to be able to host 1 percent of Purdue's student population at any time (~4160 students).
- d. I would like the application to run smoothly without crashing.

2. Server

As a developer,

- a. I would like the server to communicate with the front and two databases quickly and efficiently.
- b. I would like the server to be able to store users' data in a database.

3. Appearance

As a developer,

- a. I would like the application to be aesthetically pleasing.

4. Security

As a developer,

- a. I would like user account to be secure and available only to the specific user.
- b. I would like to notify users of basic etiquette and security precautions when interacting with other users through the messaging system (no harassment, discrimination; passwords should not be given to anyone).
- c. I would like to ensure that the use of users' data is non-malicious and transparent.

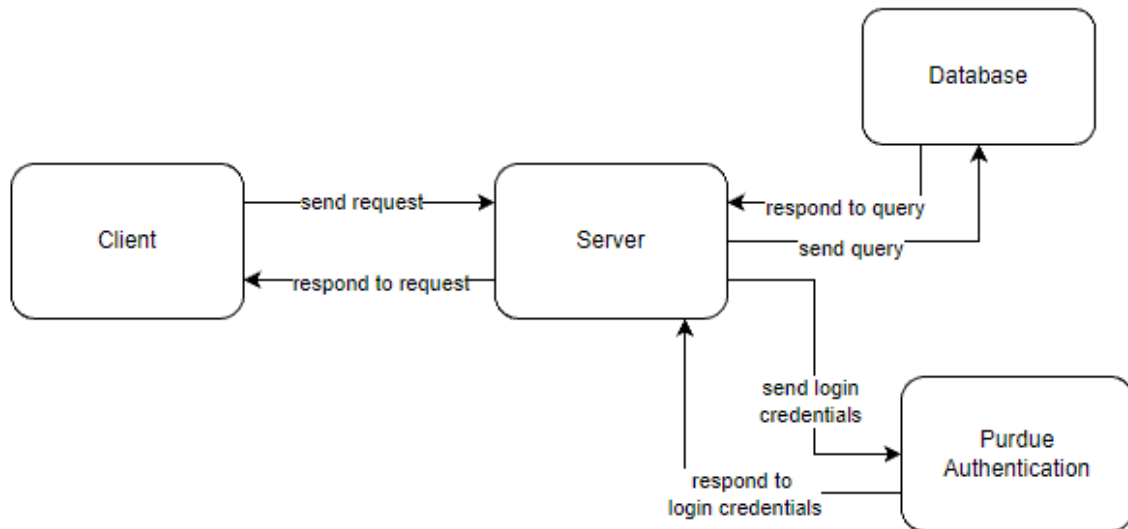
5. Usability

As a developer,

- a. I would like the interface to be clean and easy to navigate.
- b. I would like the application to be able to run on web browsers.

Design Outline

Components



1. Client (webpage):

- Displays information to the user.
- Allows user to send requests to the server to perform multiple functions (messages, search, view, etc.)

2. Server:

- Middleman between client and database.
- Query and returns user requests.
- Updates the database per the user's request.
- Also includes web scraper.

3. Database:

- Stores data critical to the client and server's function.
- Includes (but not limited to): textbooks, listings, messages, ratings, etc.

4. Purdue Authentication:

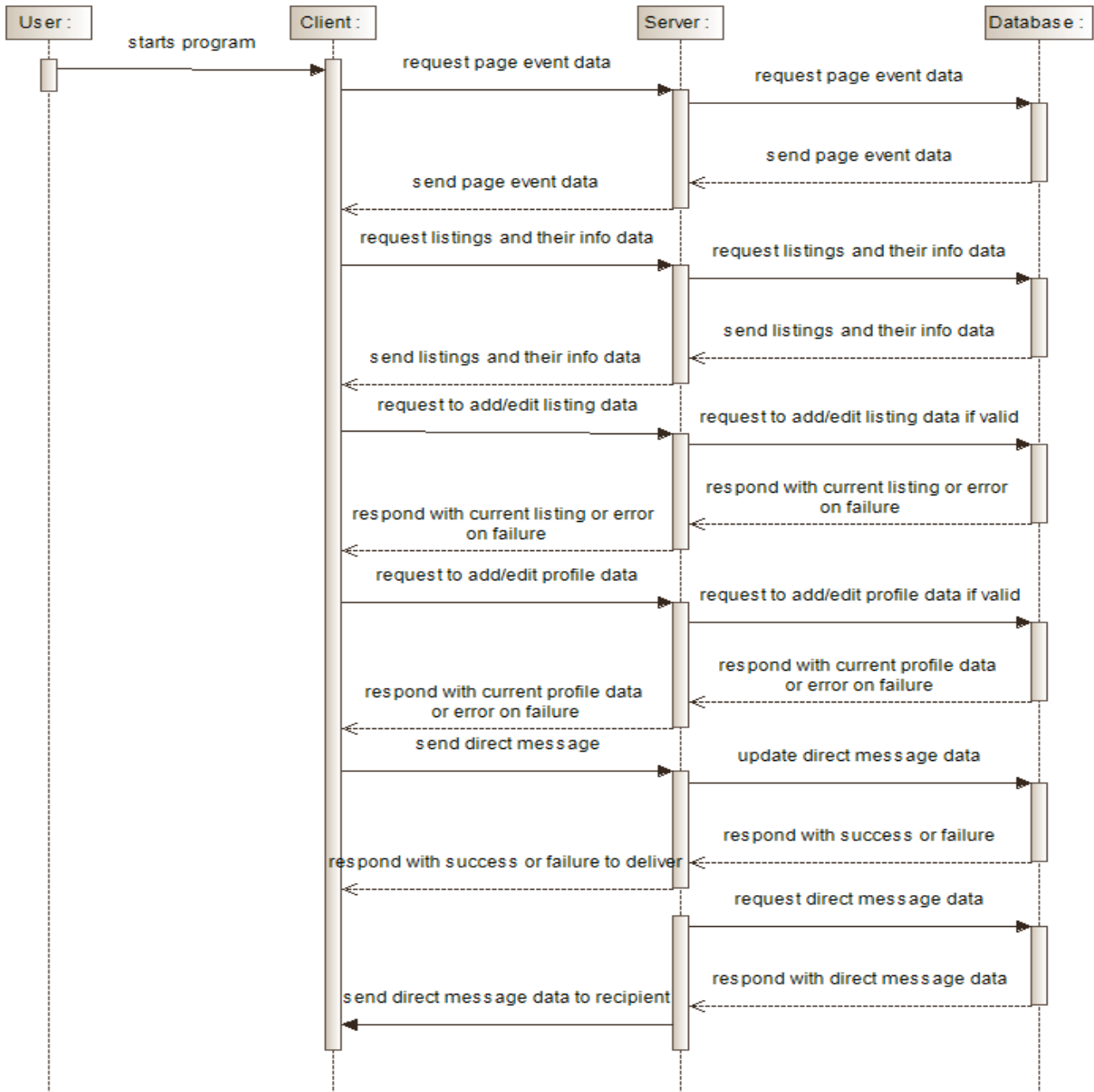
- Authenticates login credentials for Purdue students.

High-Level Overview

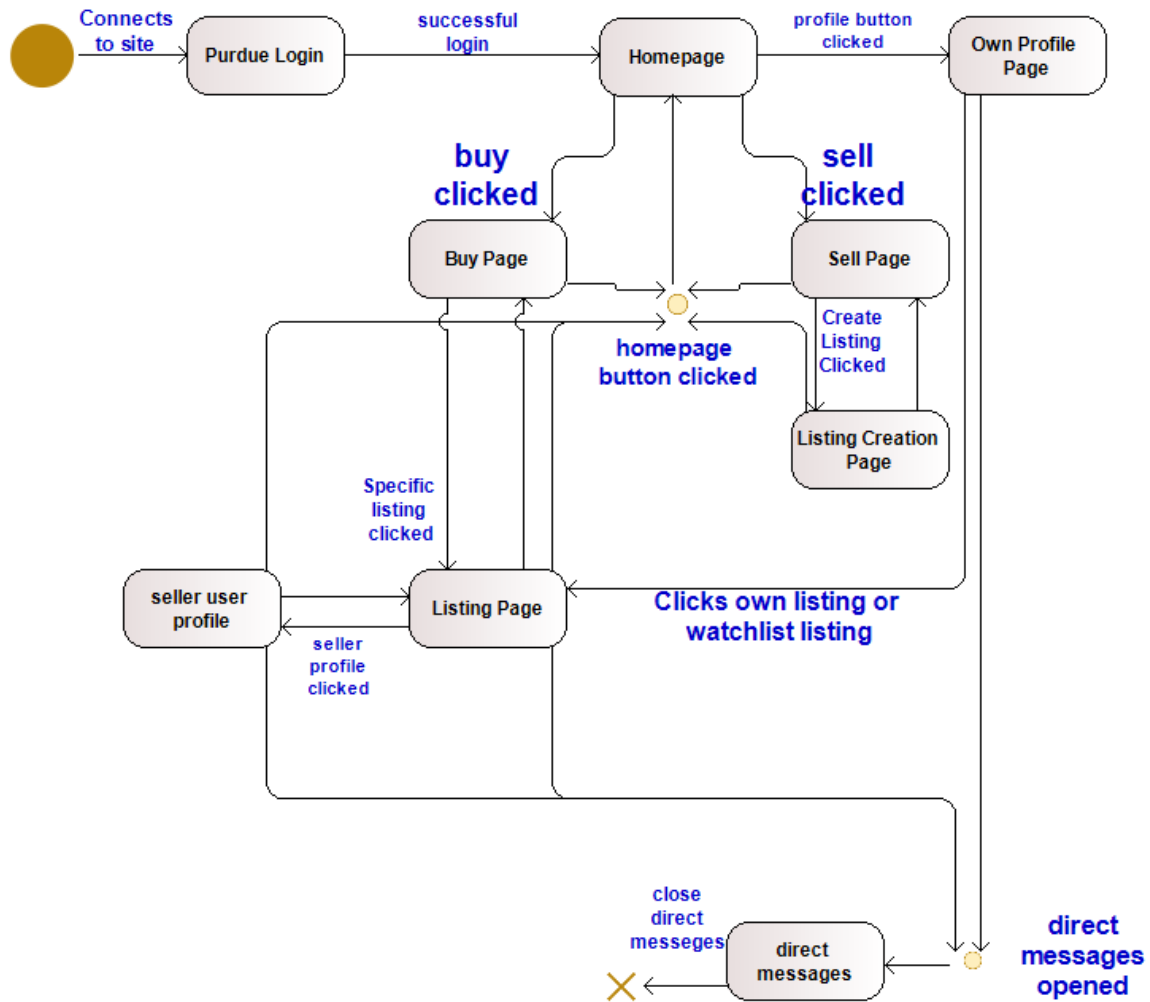
We use a client-server architecture. The client is a webpage where users can view data from and send requests to the server. The server performs the checks needed to ensure the user is authorized to make the request. If successful, the server interacts with the database, querying user requests and/or making changes to data such as book listings, ratings, messages, etc. If needed, the server returns the result from the query to the client, where the result is made visible to the user.

Sequence of Events Overview

In the diagram below, it shows how the client, server, and database interacts. After the user logs in, the client will send a request to the server and the server will send a query to the database to retrieve data. After the user has logged in, there will be other functions that follow the same actions such as listing textbooks, edit profile, etc.



Activity/State Diagram



Design Issues

Functional Issues

1. How does user's login to use our service?
 - a. No login
 - b. Login using username/email, password
 - c. Purdue Authentication

Choice: C

Justification: We only want users who attend Purdue to be able to access this web application. When buying a used textbook, there will be filters that only correlate to Purdue courses.

2. How do we compute the ratings of the users?
 - a. Average all ratings.
 - b. Average ratings but have a buyer rating and seller rating for each user.
 - c. Have good (+1) or bad (-1) ratings that add up to a final score.

Choice: A

Justification: Averaging all the ratings should simplify the process for other users to determine if someone is reputable when buying from them or selling to them. Typically, a user who is rated unfavorably with selling will also be unfavorable when it comes to buying. Averaging all ratings also tells the user more about another user than +1 and -1 ratings that get summed up would.

3. How do we authenticate user's requests?
 - a. Client POSTS username and password with every request
 - b. Use a session cookie that contains a username with expiration handled by the client
 - c. Use a session cookie that contains a UID with expiration handled by the server

Choice: C

Justification: By requiring requests to contain a UID, we can ensure that requests related to an account are coming from the account's actual owner. Using a cookie that includes the username would make it easy for malicious individuals to bypass account security. We should have the cookie expiration handled by the server as security should be as centralized as possible to prevent tampering. Furthermore, as we are handling logins through ITaP, having a client send requests with a username and password would not make sense.

4. How do buyers and sellers communicate with each other?
 - a. Email
 - b. Our messaging system (direct messages/ DMs)
 - c. Leaving a comment

Choice: B

Justification: Allowing direct messages through our platform makes it easier for users to message each other with the press of a button rather than having to go to their email provider to compose and send a message. It also allows users to keep track of messages that may get lost in an email inbox otherwise. Leaving a comment may be problematic in terms of privacy as unwanted users may interfere with the transaction.

5. How do we protect users from online harassment and discrimination?
 - a. No need, everybody on the Internet is nice.
 - b. Have a report system and someone to review the request.
 - c. Have a block function to prevent interaction with another user.

Choice: C

Justification: There aren't sufficient resources for a system to review reports. Allowing users to block disruptive individuals is the best option to protect the user but is also feasible with the scope of this project.

Non-Functional Issues

1. Which backend language/framework should we use?
 - a. Python
 - b. PHP
 - c. C#
 - d. Java

Choice: A

Justification: Python is an easy to learn language that is also easy to read with its concise, no semicolons, indentation over brackets syntax. We expect this to expedite the time for developers to familiarize themselves with the backend.

2. Which front-end language/framework should we use?
 - a. React
 - b. Raw HTML, CSS + JavaScript, jQuery
 - c. Angular
 - d. Vue

Choice: A

Justification: React is flexible and it can build quality user interfaces. It also automatically updates the UI based on modifications and updates within the component.

3. Which database should we use?
 - a. MySQL
 - b. PostgreSQL
 - c. MongoDB
 - d. Oracle XE

Choice: C

Justification: ACID compliance is not required, so NoSQL is preferred with its capability of horizontal scaling.

4. Which webscraper should we use in case an API isn't available?
- a. BeautifulSoup
 - b. Selenium
 - c. LXML
 - d. Mechanical Soup
 - e. Provided API

Choice: A and E

Justification: If API isn't available, BeautifulSoup can easily parse through HTML and it is easy to use and learn. It is also best for small and simple tasks. If an API is available, we will just use the provided API so that it is easier to obtain data.

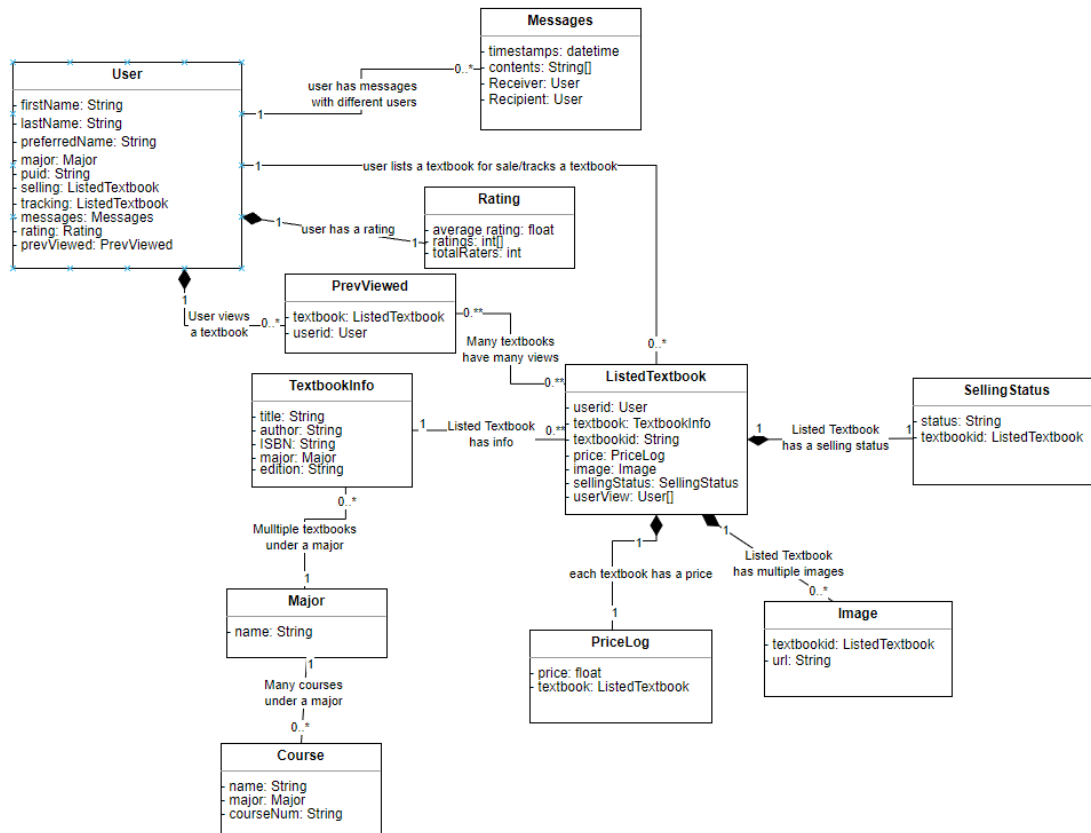
5. Which APIs should we use? (Choose multiple)
- a. GraphQL
 - b. REST

Choice: B

Justification: The schema doesn't have convoluted relationships, so REST is sufficient. Also, there might be times where we will need to make complex queries and REST is a better choice. Rest is simpler and therefore easier to implement, and our site doesn't need the fine-grained precision and complex requests needed by GraphQL.

Design Details

Class Design



Description of Classes and Interaction between Classes

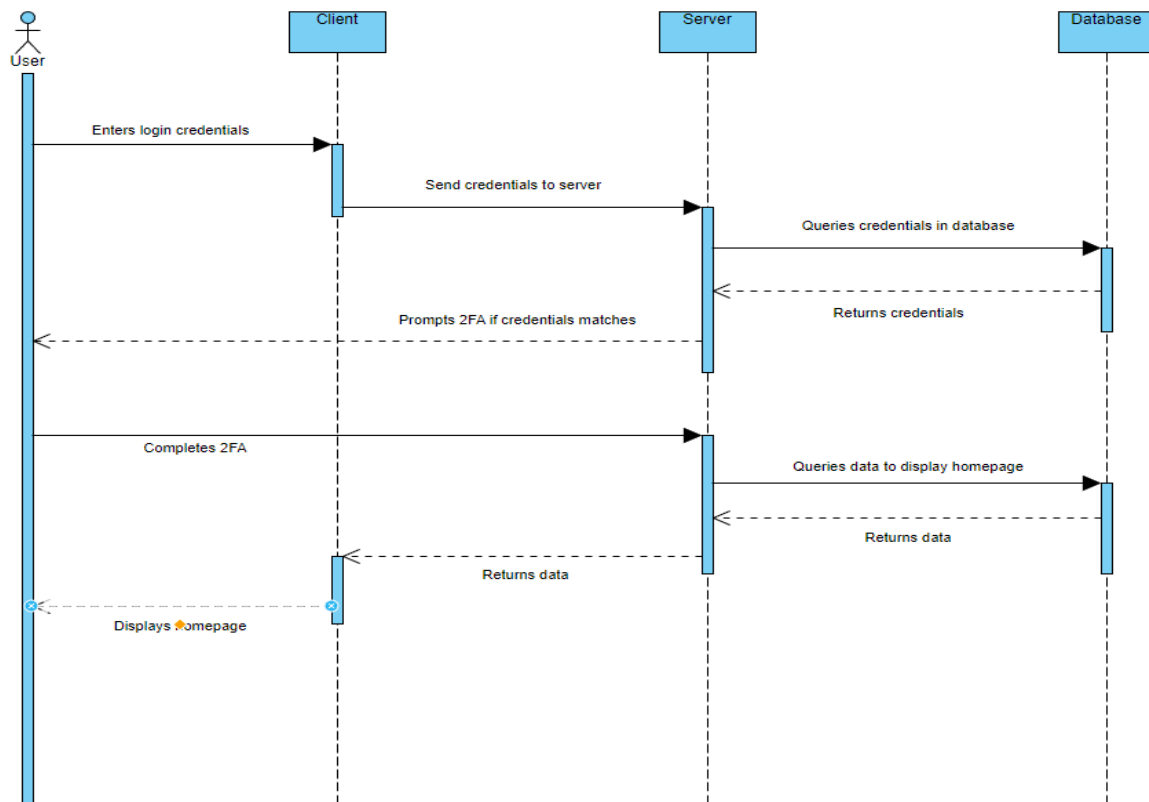
- User
 - User class is created when someone signs into the website for the first time.
 - Each user will log in using their Purdue login through ITaP.
 - To access almost all of the site's functionality, logging in is required.
 - Each user will be assigned a unique user id that may be based on a preexisting id from ITaP.
 - Each user will have information associated with it: first name, last name, preferred name, major, and preferred meeting times.
 - Each user can edit their own preferred name, major, and preferred meeting times.
 - Each user will have a rating associated with their account.
 - Each user will have their purchases and sales tracked.

- Each user can create textbooks, seller listings, messages, and ratings.
- Each user can block another user to prevent the other user from messaging them or viewing their listings.
- TextbookInfo
 - Textbook classes will be created when a user adds a new textbook to the database by creating a new seller listing or adding a textbook manually.
 - Each textbook will have information associated with it: title, author(s), ISBN, major, and edition.
- ListedTextbook
 - Created when user lists a textbook for sale.
 - Each user will have 0 to many listed textbooks that they want to sell.
 - Each listed textbook will have the textbook info, image of the textbook, price that they want to sell it for.
- PrevViewed
 - PrevViewed class will be created when a user clicks on a textbook.
 - Each user will have their own PrevViewed class.
 - This will contain the three most recent textbooks that the user has viewed.
 - PrevViewed will be used to display a user's previously viewed textbooks on the home page.
- Messages
 - Messages class is created when a user starts a conversation with another user.
 - Each messages class includes two users and their messages.
 - Each message includes a timestamp, sender, receiver, and contents.
- Rating

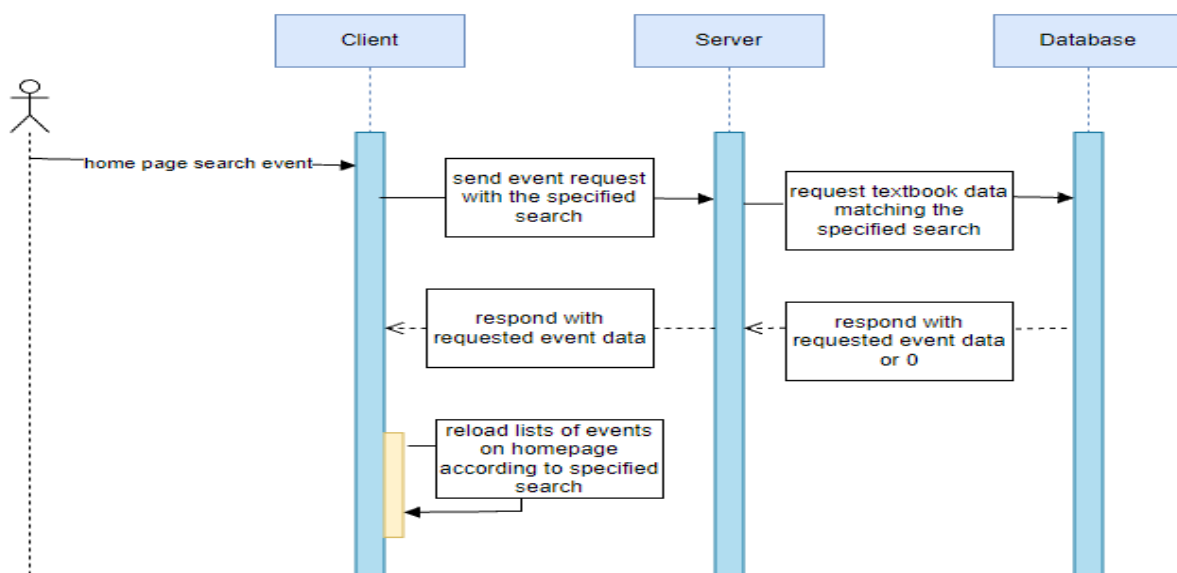
- After a transaction, users can create a rating of the other user in the transaction.
- Ratings are on a scale from 1-5 stars.
- Image
 - Image class is created when user uploads an image of a textbook that they want to sell.
 - Each image class includes image url and the id of the textbook with the image.
- Major
 - Each major class will have a name of the major.
 - A major will have many courses and textbooks.
- Course
 - Each course will have name, major that it falls under, and the course number that it is a part of.
- PriceLog
 - Created when user lists a textbook for sale.
 - It contains the price of the textbook.
- SellingStatus
 - Created when user lists a textbook for sale.
 - It contains the status of the textbook which default is “not sold”.
 - It also contains the textbookid of the listed textbook.

Sequence Diagram

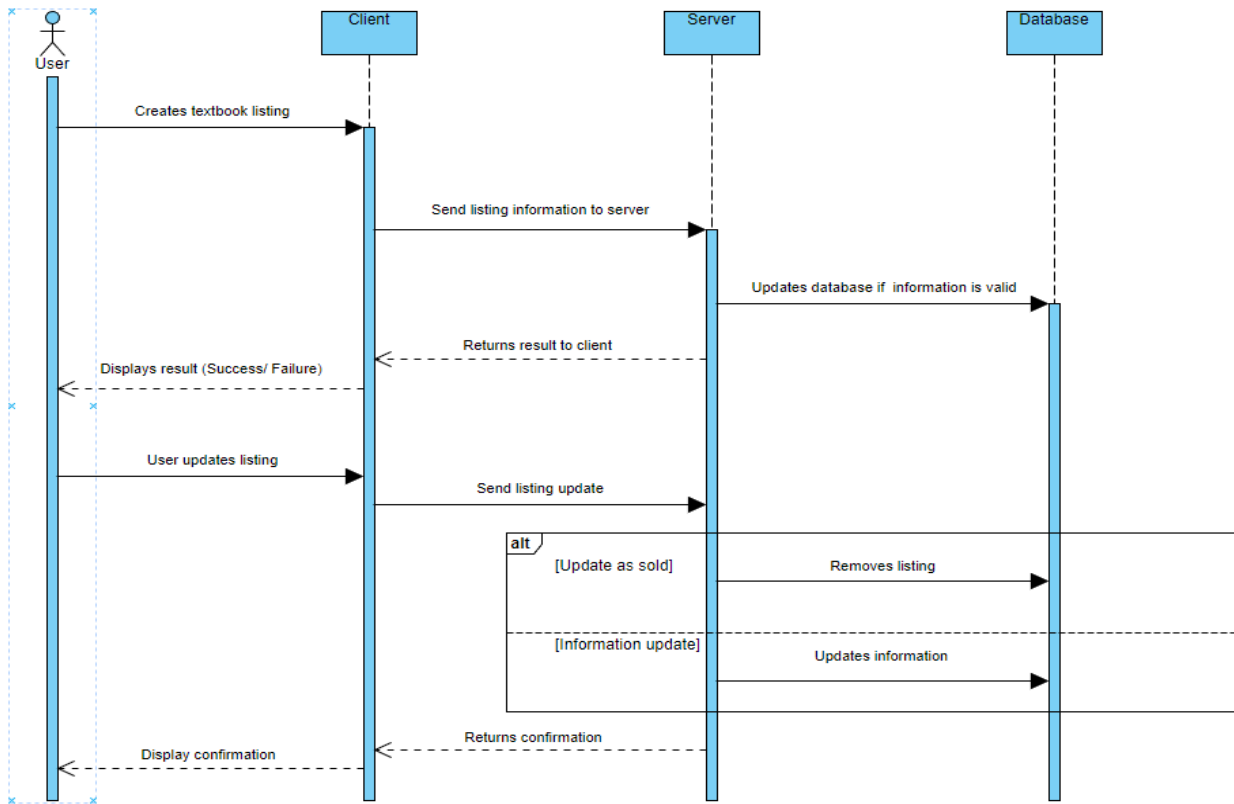
1. Sequence of events when user logs in



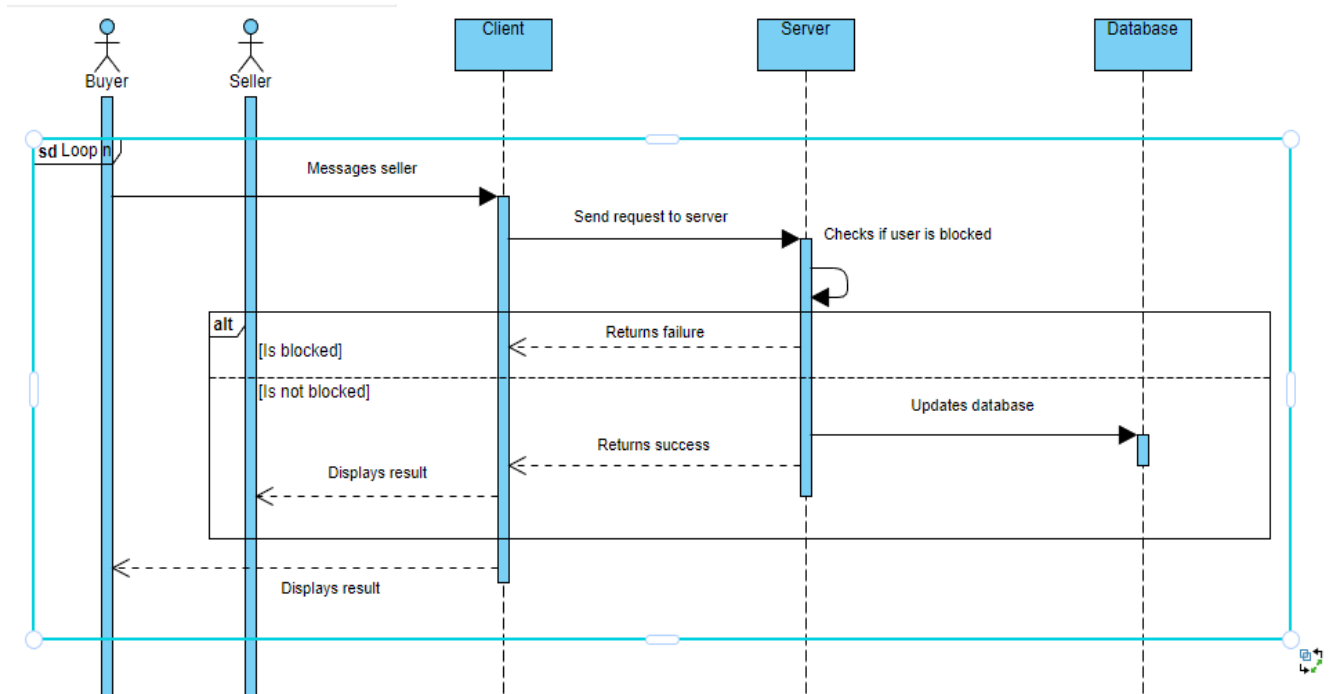
2. Sequence of events when user uses home page search.



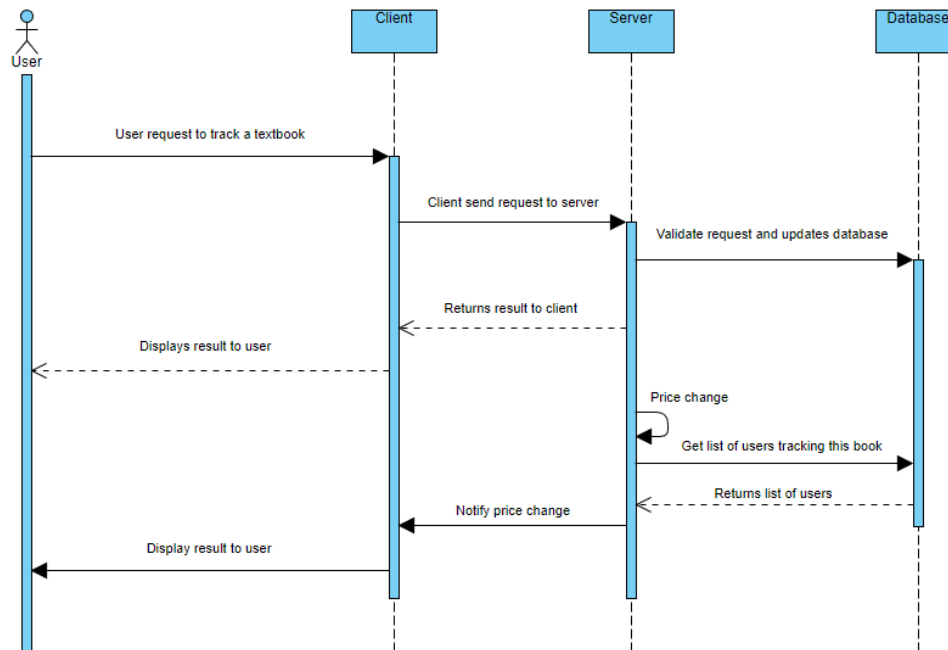
3. Sequence of events when user creates textbook listing.



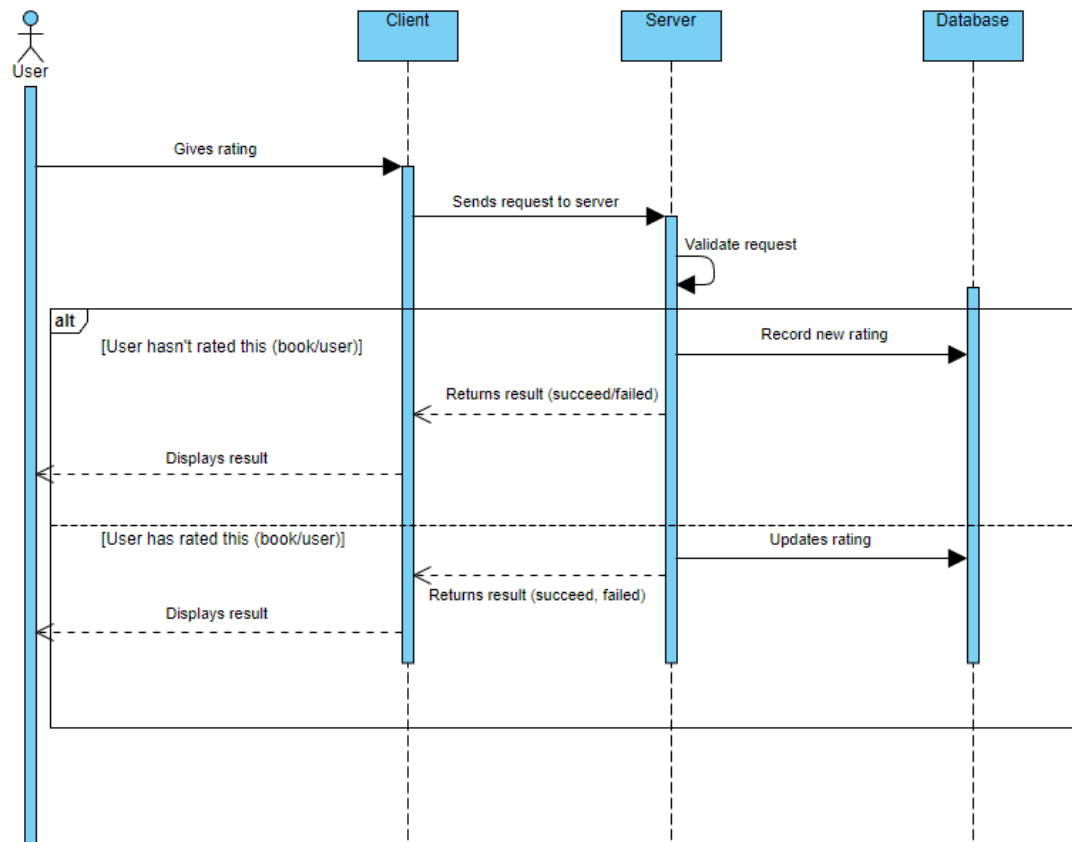
4. Sequence of events when buyer messages seller.



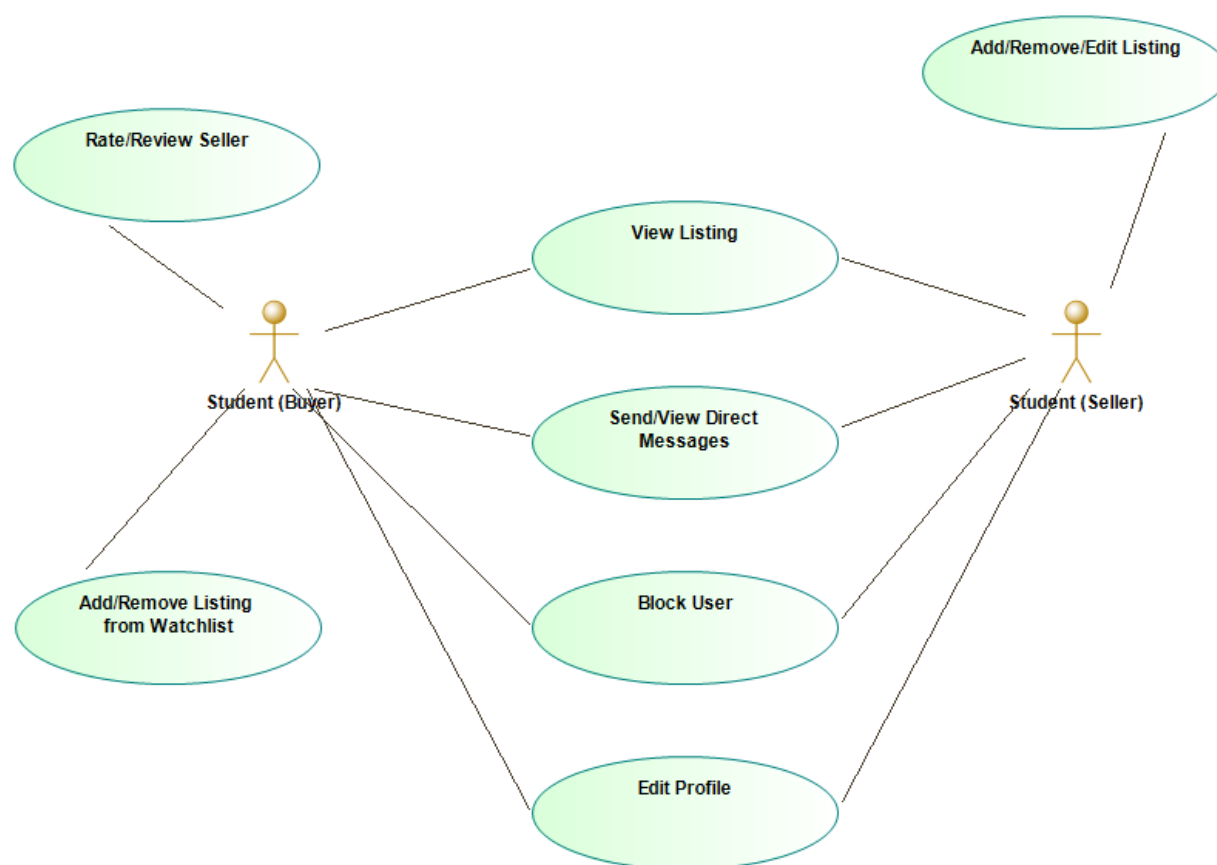
5. Sequence of events when user tracks a textbook.



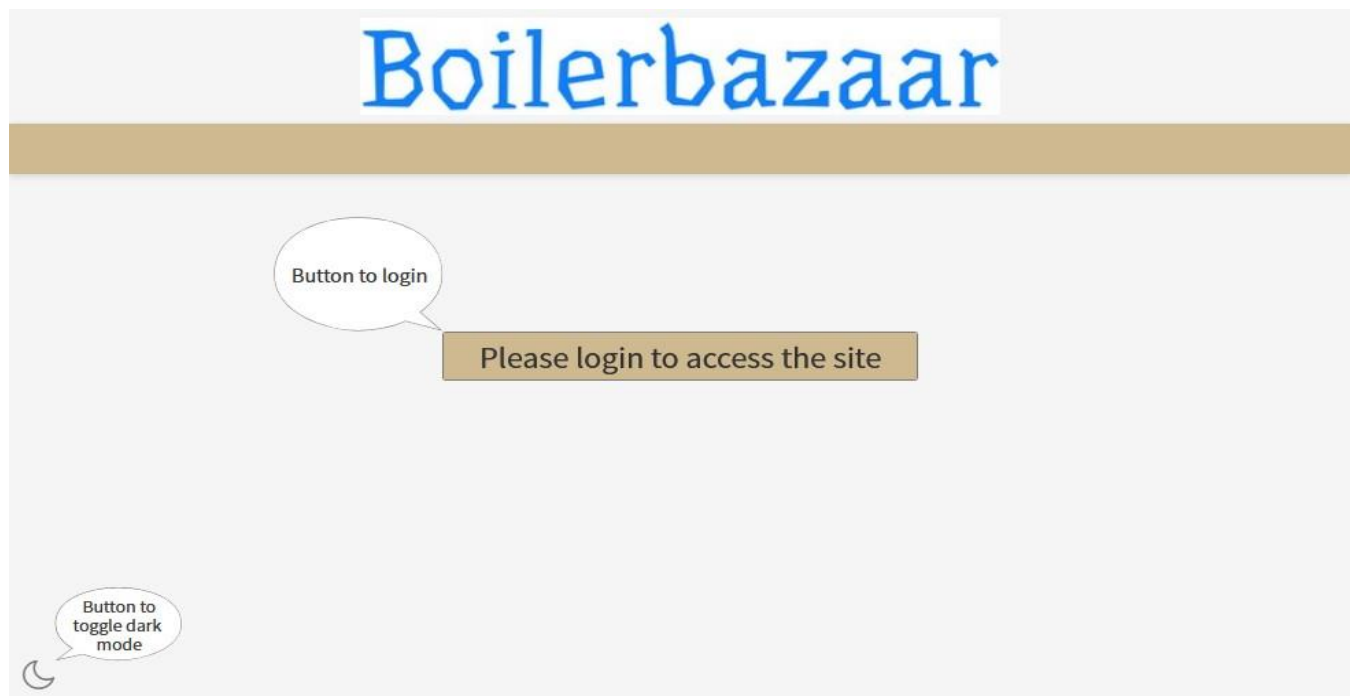
6. Sequence of events when user gives a rating.



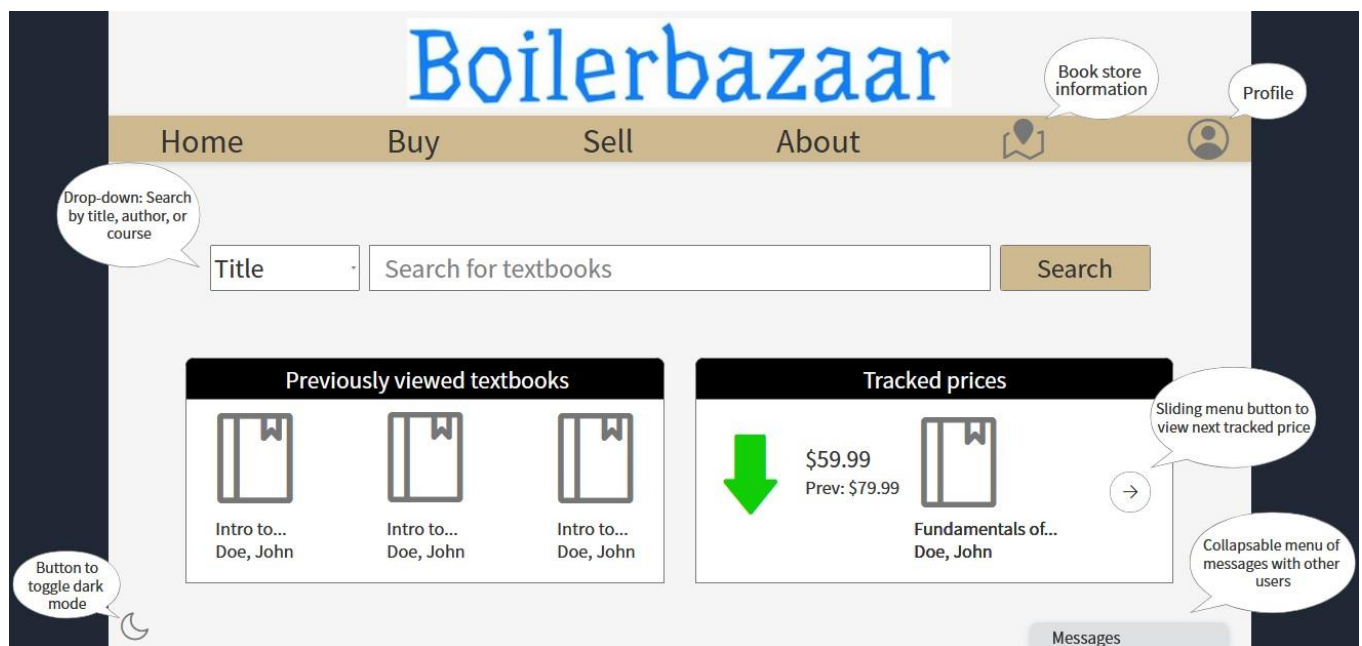
Use Case Diagram



UI Mockup



1. This is the login page. You will click the button and it will take you to the login site. The site will be like any other Purdue sites that require you to login through Duo authentication.



- This is the home page that will appear after the user logs in. The search button in the middle allows the user to search for a textbook based on title, author, or ISBN.

Frame 1

Title	<input type="text" value="Search for textbooks"/>	Search
-------	---	--------

RESULT

Frame 2

Image	Programming in C		
	Author	Year: YYYY	Edition: X

Frame 3

Image	Programming in C		
	Author	Year: YYYY	Edition: X

Frame 4

Image	Programming in C		
	Author	Year: YYYY	Edition: X

- This is the page that appears after the user clicks search from the home page. Clicking on a book will provide the user with information about prices of the book from other websites.




4. This is the buy page where the user will see lists of textbooks that are for sale from other Purdue students. Clicking on a listing will provide the user with more detailed page of the listing.



5. This is the page that appears when you click on a textbook that you want to buy. This provides additional information about the listing and allows the user to message the seller.

Sell A Book



* Book:

* ISBN:

* Author:

* Edition:

* Price:

Anything you want to share:

6. This is the sell page where users can sell their new/used textbooks to other Purdue students. There will be certain information about the textbook that will be required to fill out to list it on the market.



7. This is the profile page where users can verify and edit information about their profile. This includes their preferred name, major, and preferred meeting location. An average of their past ratings and a count of their past transactions are also displayed.



8. This is the Purdue Book Store Information page where users can view information related to the Purdue Book Store including locations and hours.