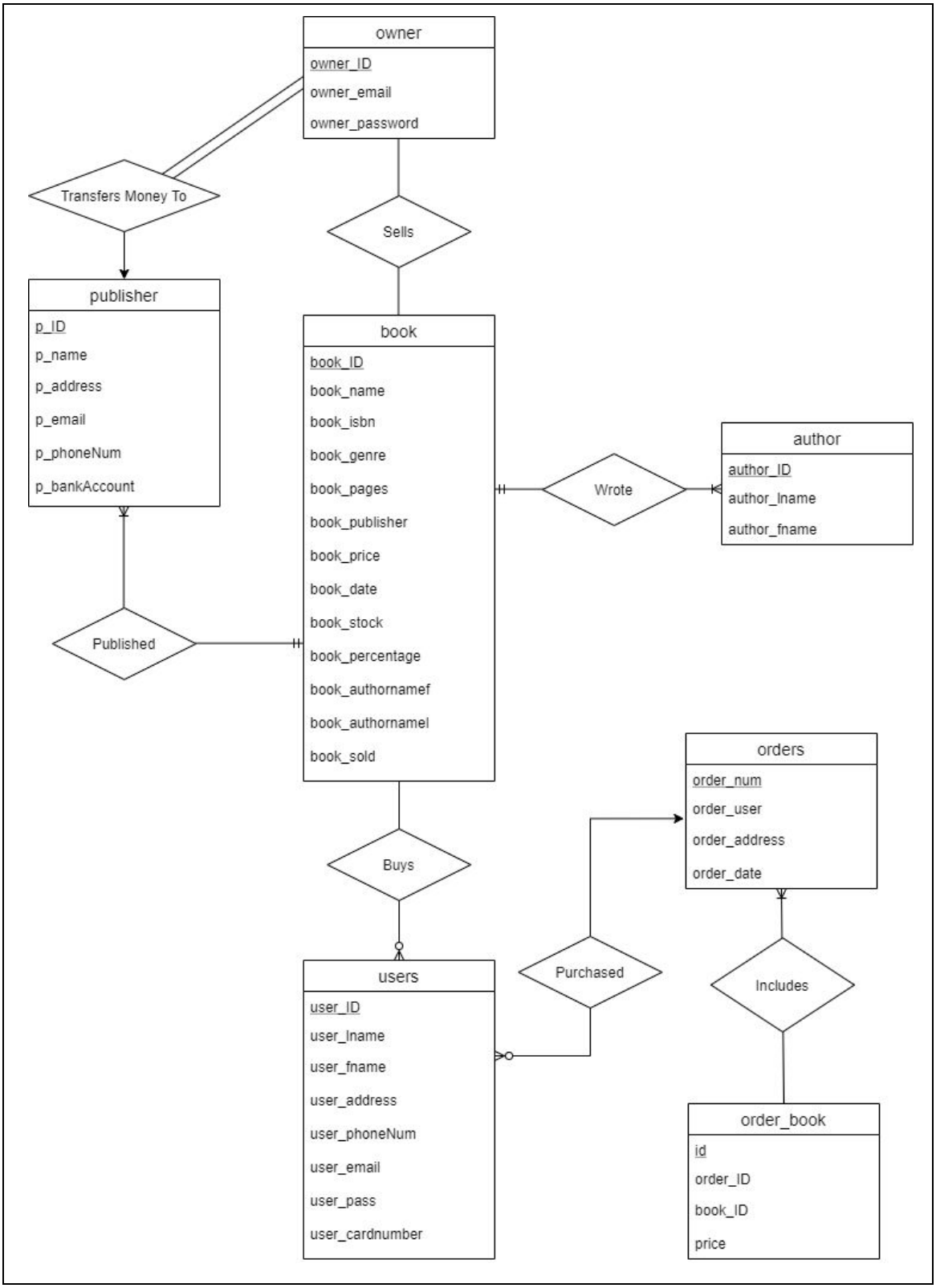
COMP3005B Project - Book Store

Alyssa Chow
101103472
Prof. Ahmed El-Roby
COMP3005B
April 14, 2020

Table of Contents

2.1 Conceptual Design



The bookstore database contains tables for the shop owner, author, book, publisher, user, orders, and individual book in the order.

The book shop owner has an ID (primary key), email, and password. They are related to the book table by selling them and are weakly related to the publisher table for the access required for transfering percentages to them according to the sales.

Next, each book has an ID (primary key), title, ISBN, genre, number of pages, publisher, price, date published, amount in stock, the publisher's royalty percentage, and the author. The book table is related to the publisher table by being published by them. The assumption is that every book has one publisher. The book table is also related to the author table as they were written by an author and it is assumed that there is only one author per book. It is also related to the users table because the books are bought/browsed by users.

As previously mentioned, there is an author table. Each author has an ID (primary key), last name, and first name. This table is connected to the book table and it is assumed that each author must have written at least one book.

Another table that was mentioned is the users table. Every user has an ID (primary key), last name, first name, address, phone number, email, password, and card number for billing. This table is related to the orders table through a relationship of purchasing a book. It is assumed that a user can order nothing, or many things.

The orders table has an order number (primary key), the order's user, the shipping address, and the date it was ordered. This table is also related to the order_book table as it includes the items from the latter. Each order must have at least one book.

Lastly, the order_book table denotes the individual books that are in an order. Each order_book has an ID (primary key), the order ID it comes from, the book's ID, and the price. The last three are foreign keys coming from other tables.

2.2 Reduction to Relation Schemas

author(author_ID, author_lname, author_fname)

book(book_ID, book_name, book_isbn, book_genre, book_publisher, book_price, book_date, book_stock, book_percentage, book_authorname)

publisher(p_ID, p_name, p_address, p_email, p_phoneNum, p_bankAccount)

owner(owner_ID, owner_email, owner_password, owner_lname, owner_fname)

users(user_ID, user_lname, user_fname, user_address, user_phoneNum, user_email, user_pass, user_cardnumber)

orders(order_num, order_user, order_address, order_date)

order_book(id, order_ID, book_ID, order_date)

2.3 Normalization of Relation Schemas

I believe they are already in normal form in 2.2.

author(author_ID, author_lname, author_fname)

book(book_ID, book_name, book_isbn, book_genre, book_publisher, book_price, book_date, book_stock, book_percentage, book_authorname)

publisher(p_ID, p_name, p_address, p_email, p_phoneNum, p_bankAccount)

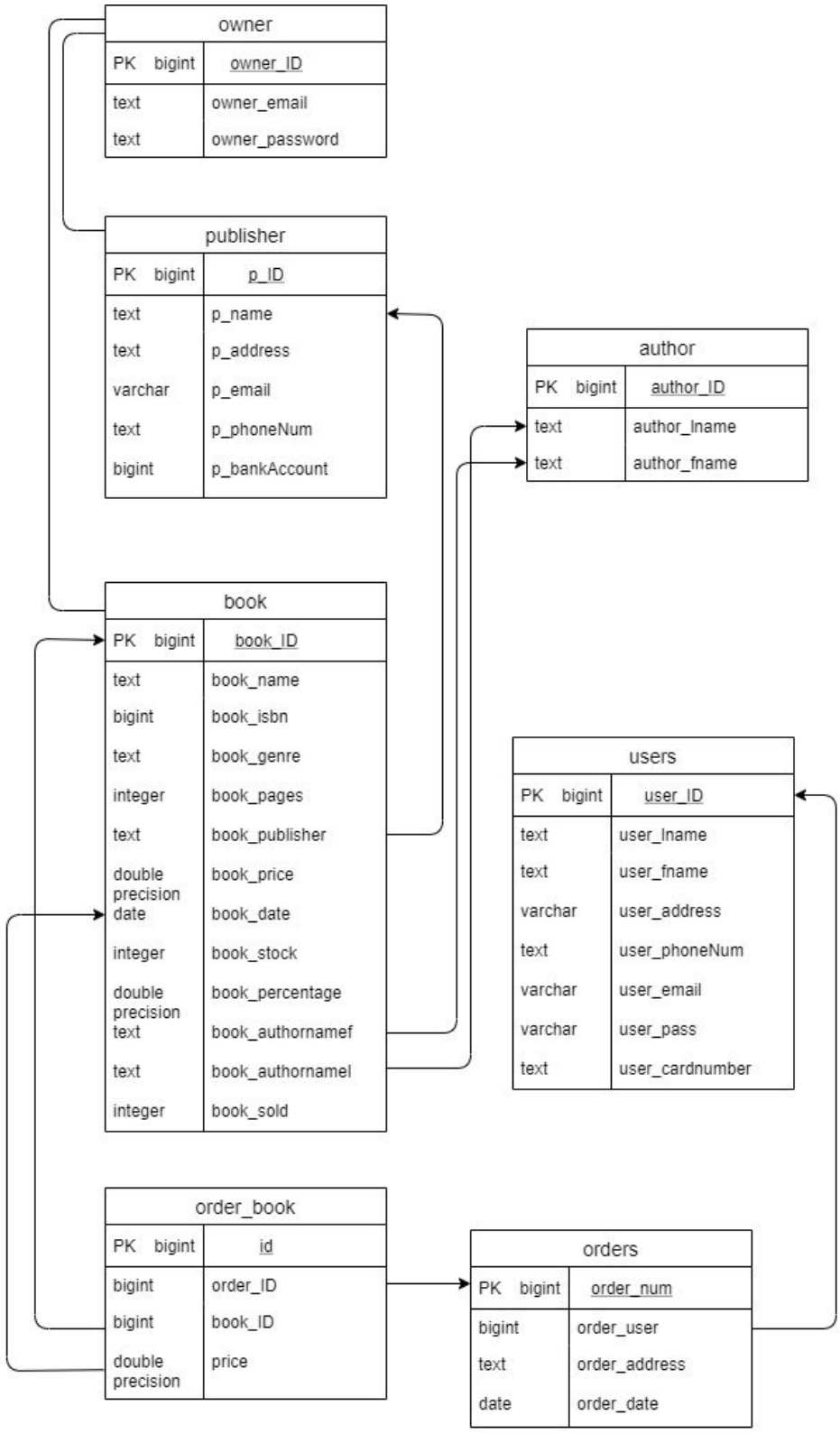
owner(owner_ID, owner_email, owner_password, owner_lname, owner_fname)

users(user_ID, user_lname, user_fname, user_address, user_phoneNum, user_email, user_pass, user_cardnumber)

orders(order_num, order_user, order_address, order_date)

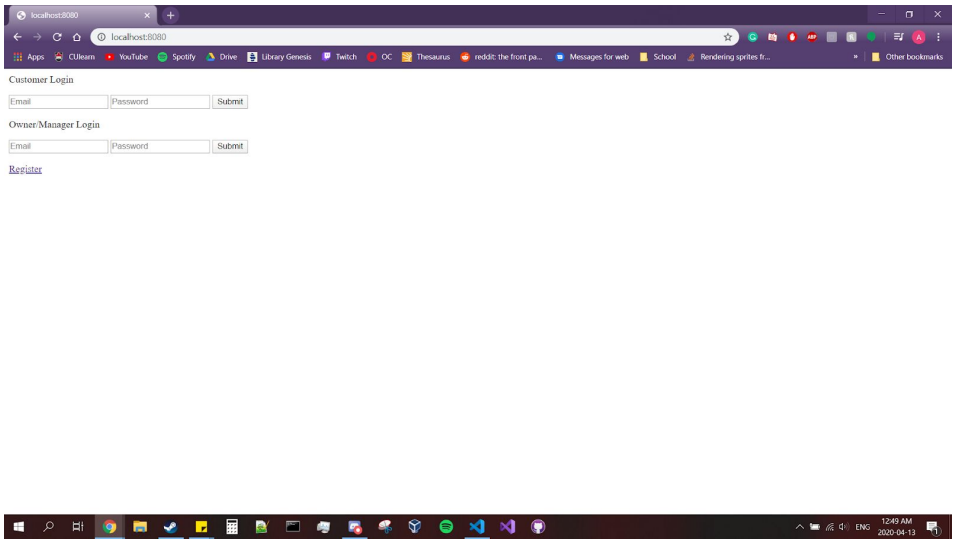
order_book(id, order_ID, book_ID, order_date)

2.4 Database Schema Diagram



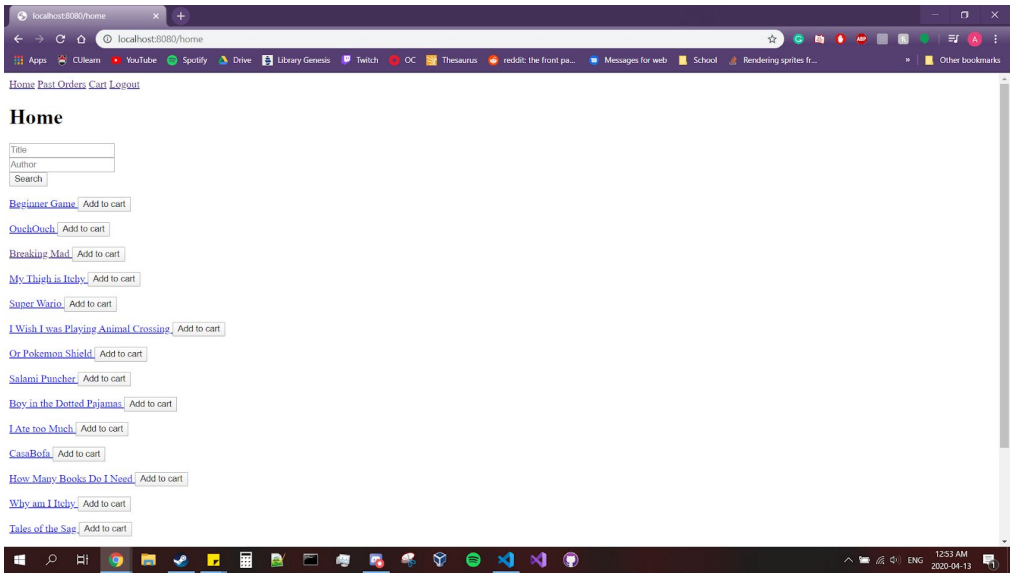
2.5 Implementation

The implementation for this was made with JavaScript, ejs, html, node.JS, and postgreSQL. It is a web based application (localhost:8080) as well. The application starts at a login screen where you can either login as a customer or as a manager, or register to become a customer. Should you choose to register, you will be taken to a registration route where if you fill out the registration form correctly, the user will be created and added to the users table. Once it is on the database, it is ready for use.

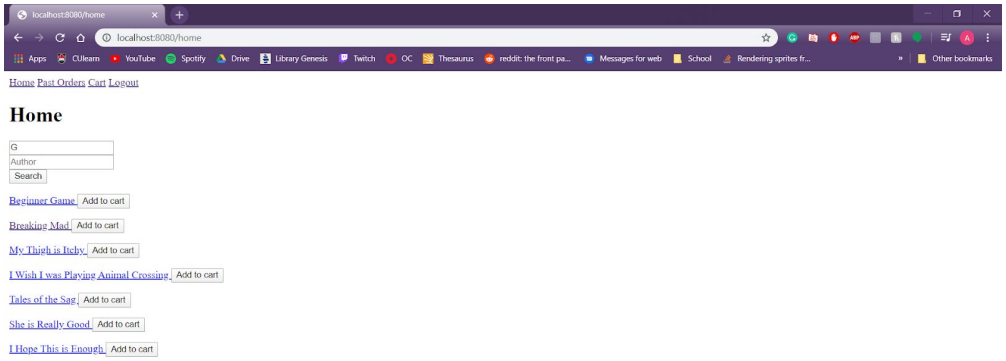


The flow of the application now depends on how you wish to proceed. We will currently follow the route should you login as a user and not an owner.

After you log in, you will be taken to the home page which displays all the books in the store and two text boxes. The text boxes are used for searching within the library by title or author. Each book has an ‘Add to cart’ button next to it. The header of the page is a list of links that will lead you to various routes.



(all books listed)

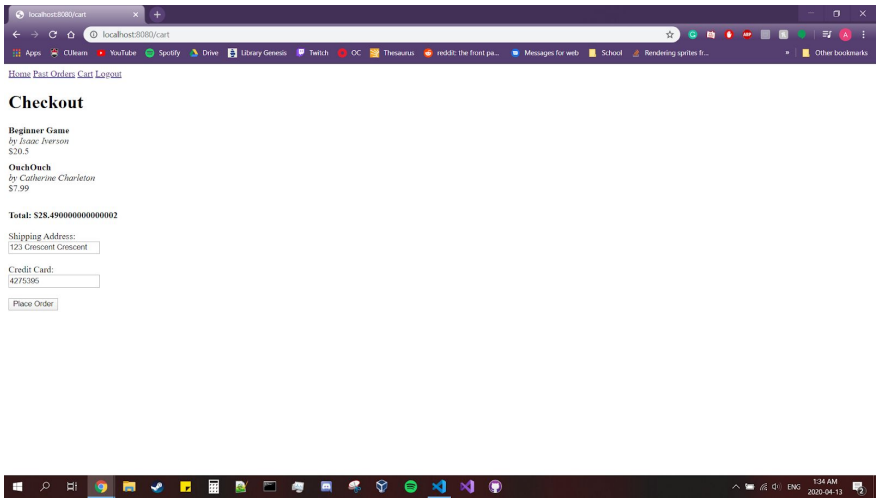



(using the search)

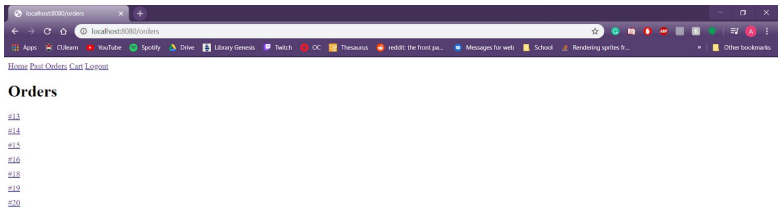
Then, if you click on the book title, it will route to a page where the book's information is displayed. (It isn't pretty, but it is there.)

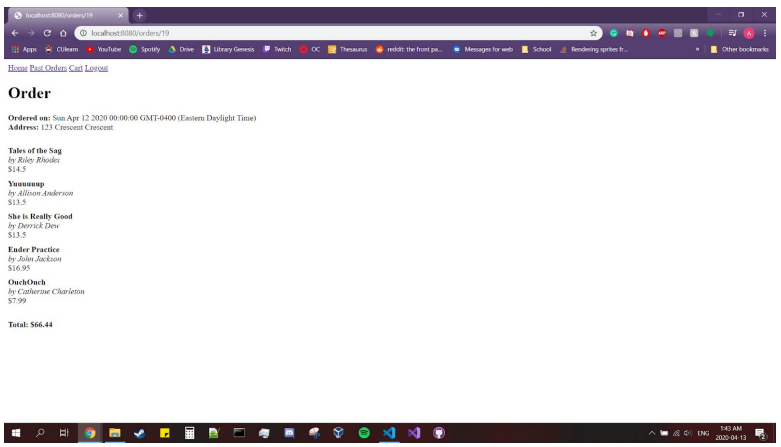
To purchase the book, you return to the previous page and click the button. From there it will be added to your cart which you can access in the header. Multiple books can be put in the cart. Once the book is added to the cart, the stock amount is decreased by one in the database. Once the book reaches a stock below 10, it automatically gets restocked to 20. This is all done behind the scenes with queries within the code.

The cart page lists the book's title, author, and price. Underneath, the user can enter the shipping address and the credit card number. These are auto filled with the information the user has in their registration, but can be changed.



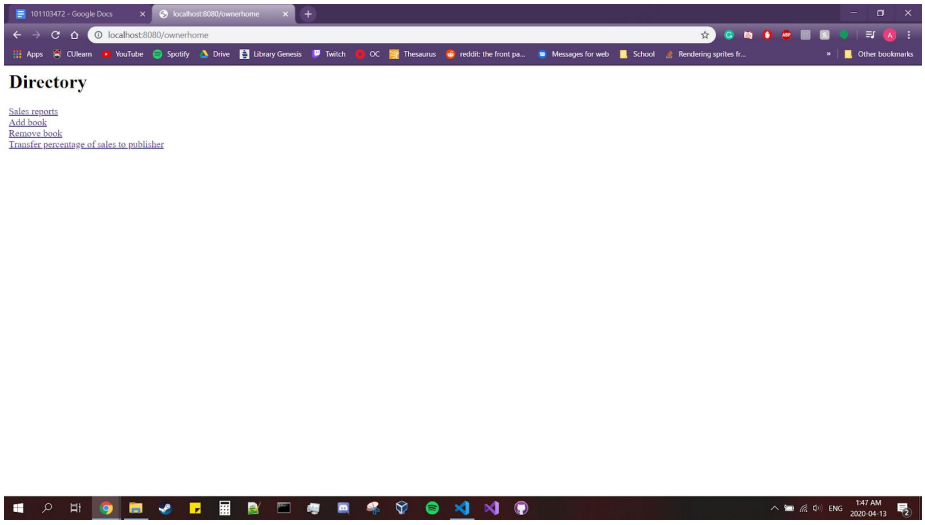
Once the order is placed, the user is sent to the tracking order page of this order they just made. This page can also be accessed through the ‘Past Orders’ link in the header. This link leads you to the list of orders made by this specific user, each with their own order number. When you click the specific order, you will see the list of books in the order and their title, author, and prices. It will also list the shipping address and the date it was ordered.



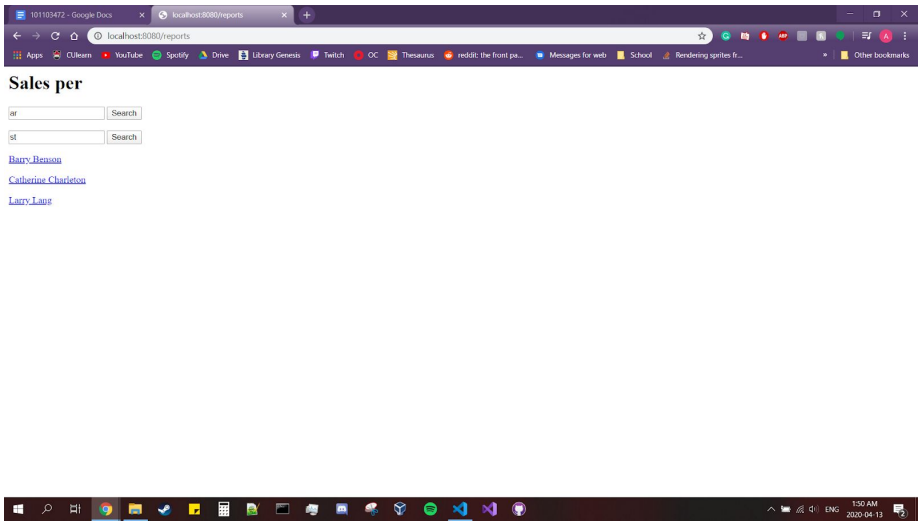


Beyond that, the user can then logout and that is all with the user's portion.

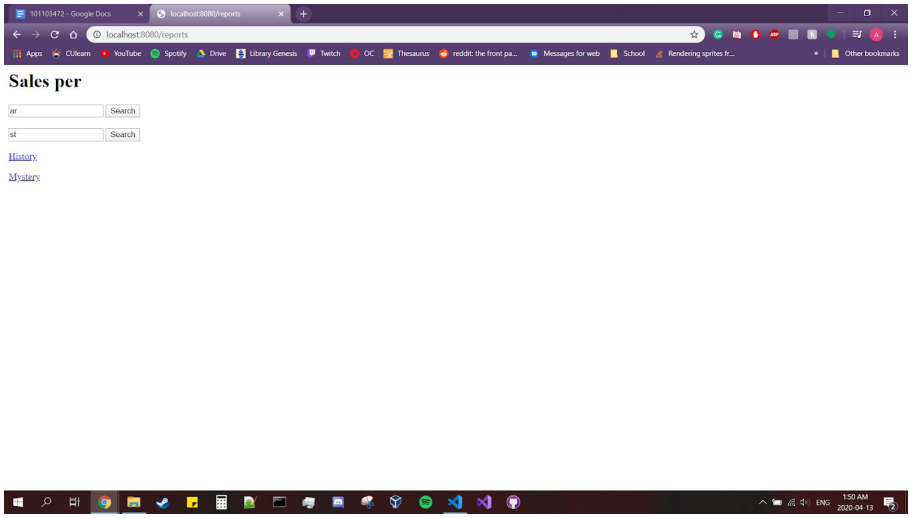
Switching over to the manager's/owner's side, you login with one of the owner accounts. Once logged in, the directory is displayed. In the directory there are links to the sales reports, book adding, book removing, and transferring the royalties to the publishers.



Let's start with the sales report section. When clicking that link, you are routed to two text boxes where you search for a genre or an author. From there you can click on one of the results and it will display the sales for that genre/author.

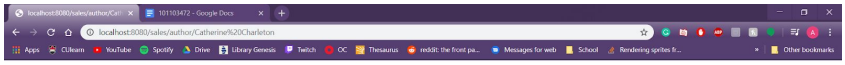


(sales by author search)

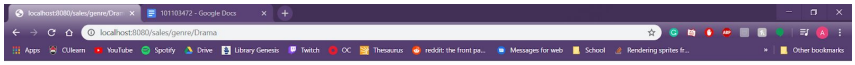


(sales by genre search)

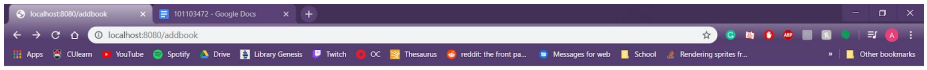
When you click one of the results for the author, you will see their name and their sales amount.



Or if you clicked on a genre, you will see the genre and their sales amount.



Now, let's go back so we can see the book addition page. Here, you can fill in the form required to add a book into the database's book table. Once filled in, you will be redirected to a notice saying if the book could be added or not (based on the credentials you entered.) As a notice, the book must have a publisher and author that are already in the database, date must be entered in the form of yyyy-mm-dd, the percentage must be entered as a decimal, and all information should be starting with a capital letter. In the code there are queries that perform behind the scenes and will update the database when needed.



Add a book

Title
Genre
Number of pages
Publisher
Price
Date Published
Royalties percentage
Author First Name
Author Last Name
Add book

Add a book

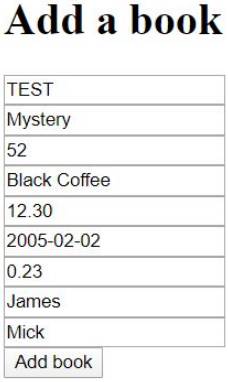
TEST
Mystery
52
yoyo
12.30
2005-02-02
0.23
James
Mick
Add book

(invalid because the publisher does not exist)

Add a book

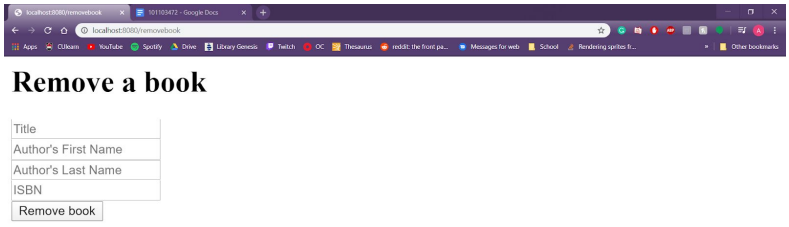
TEST
Mystery
52
Black Coffee
12.30
2005-02-02
0.23
Mick
Mick
Add book

(invalid because the author does not exist)

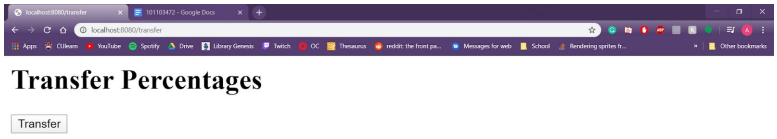


(valid)

Moving on to removing a book, similar queries are used to ensure that the book that is being deleted exists. If yes, the book will be deleted from the database, and if not, it won't be. Either way, the user will get a message saying the results. The page consists of text boxes where you fill in what is required to delete the book. The user needs to enter the title, author name, and isbn, in order to delete.

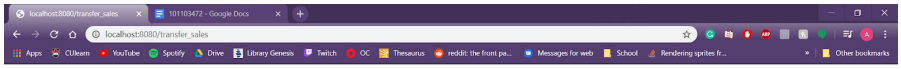


Lastly, the manager can transfer the royalties to the publishers.

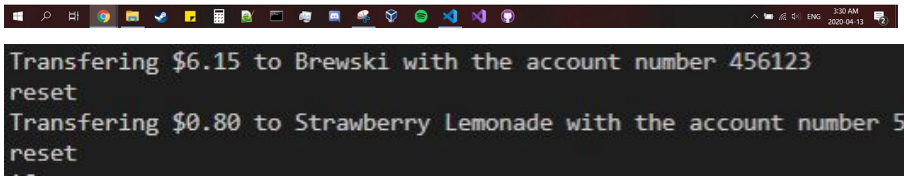


When the button is clicked, the money is "transferred" by searching through the books for those who have an amount sold that is not 0, multiply that amount by the percentage and cost, and

use the book's publisher name to find the publisher within its own table and get the bank number. Unfortunately, I am not a good enough coder to display the confirmation on screen, but it is displayed in the terminal.

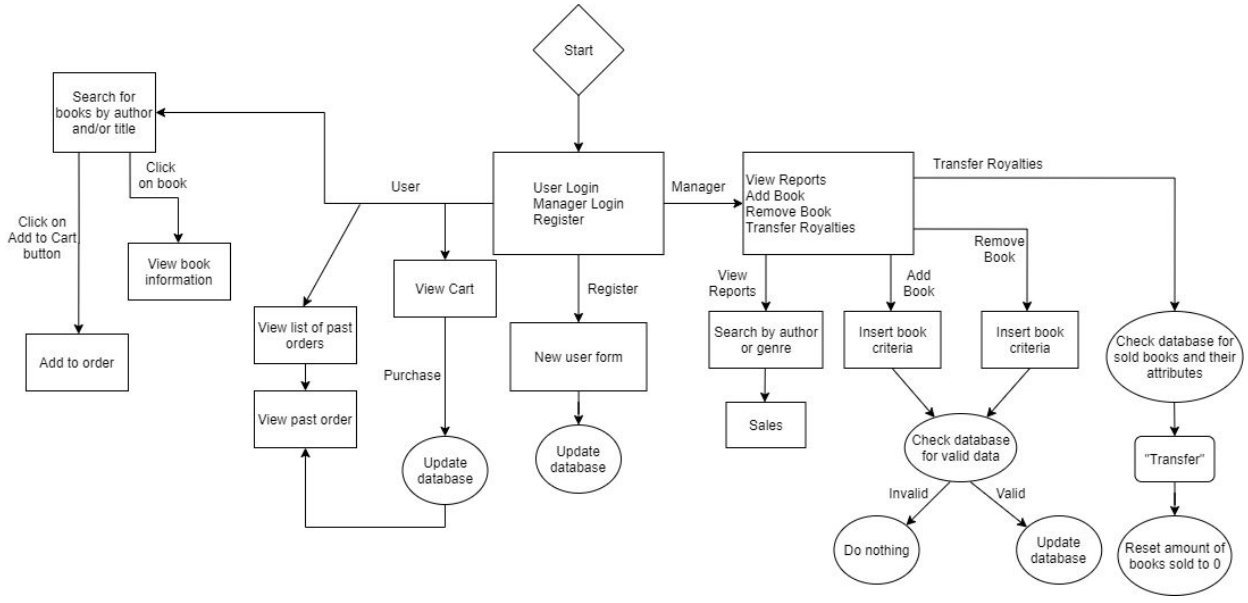


Look at terminal/console.



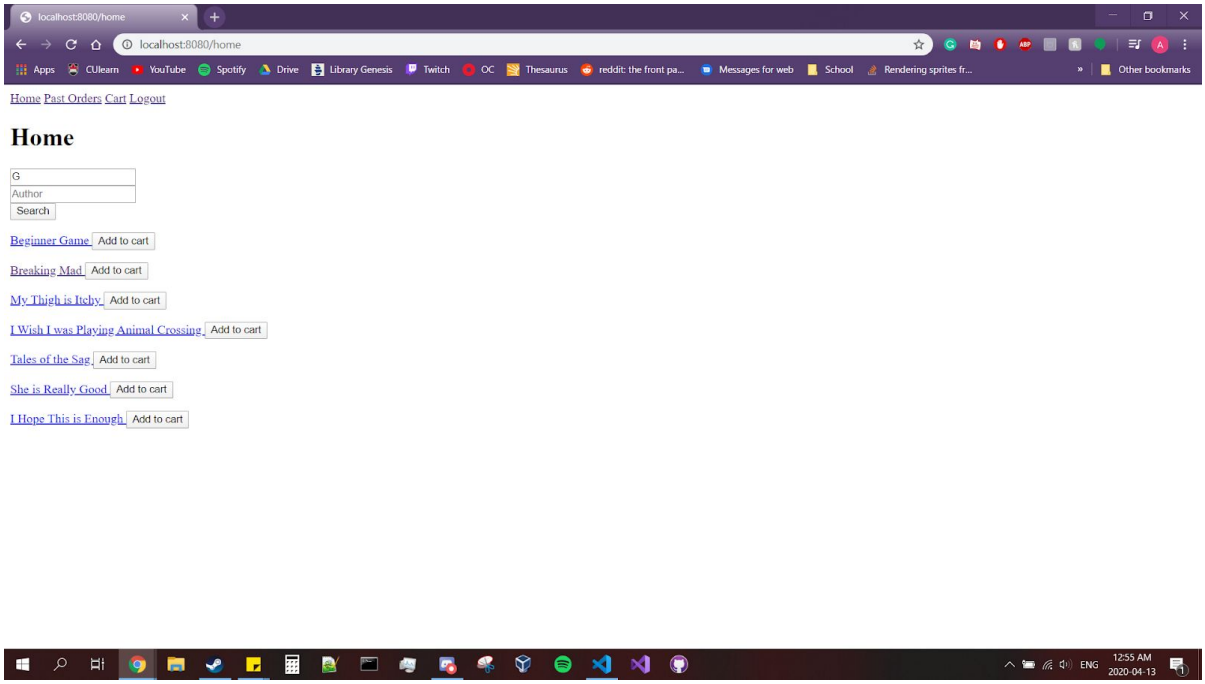
Once the royalties are transferred, the amount sold of these books get reset back to 0 for later transfers.

This brings us to the end of the implementation. Below is a brief visual map of how the implementation works. Rectangles represent coding and application, ovals represent database, rounded rectangles represent a hidden feature.



2.6 Bonus Features

Bonus features that have been implemented are the ability to search by letter. This means that anything that has the letter ‘G’ as an example, will be displayed when being searched for.
Ex:



Another potential bonus feature is the ability to register new users. I don’t recall the project document saying that this must be implemented, so I assume it is a bonus feature.

2.7 GitHub Repository

github.com/alyssachow