Jordan Fernandes

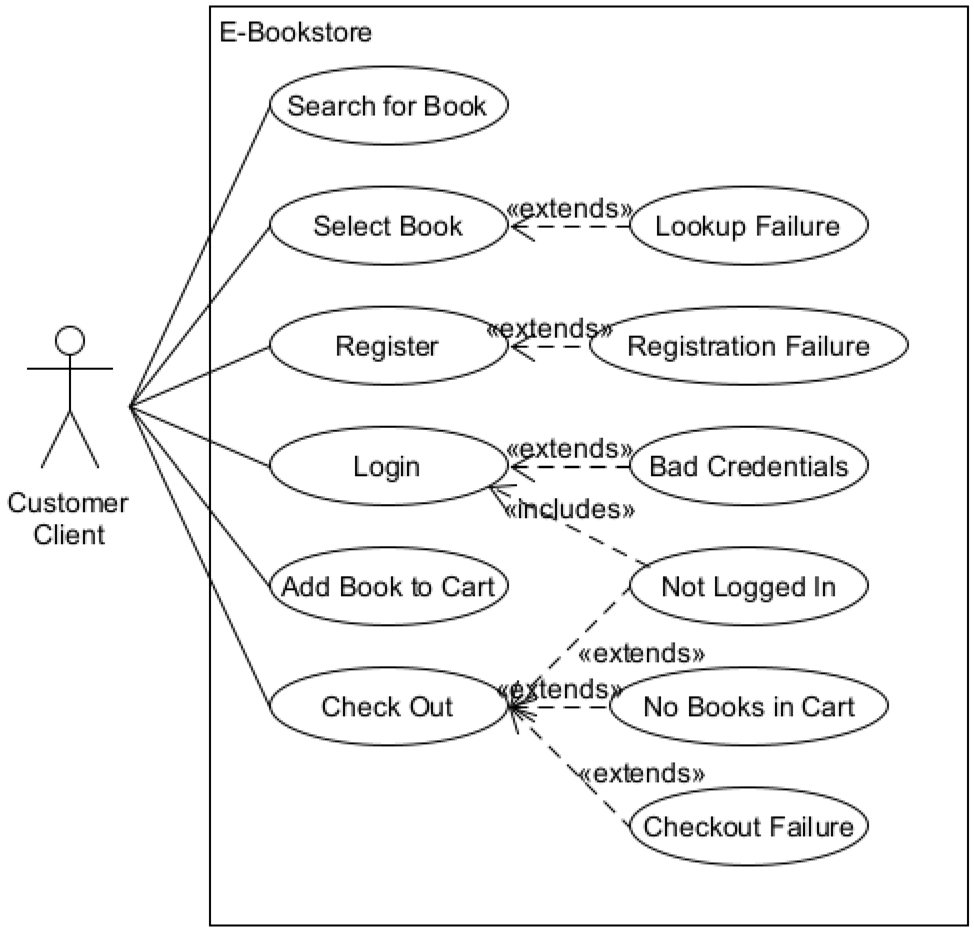
2/13/22

SSW-344 A

I pledge my honor that I have abided by the Stevens Honor System.

HW 2: Modeling E-Bookstore

Use-Case Diagram



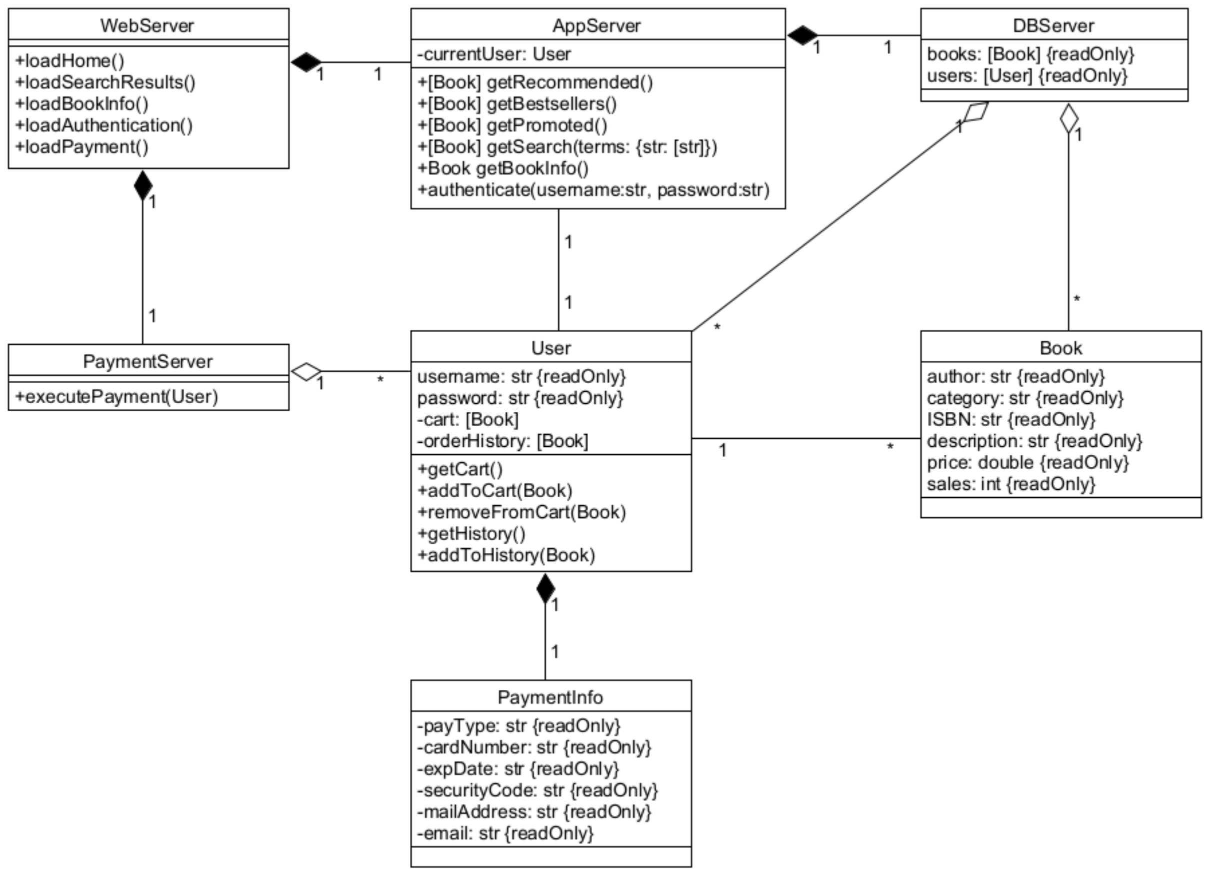
For the use-case diagram, I identified six major customer features. They can:

- Look up books (i.e., search function)
- Select a book (i.e., bring up its information page)
- Register as a new customer
- Login to the site
- Add a book to the cart
- Check out (i.e., purchase books)

In this system, I identified the customer client as the *only* outside agent. Note that, naturally, this system has some other features, such as loading the home page with bestsellers and promotions. However, these are internal functions, and are not direct interactions from the customer client.

I also identified a few exceptional cases as <<extends>> relationships, including user edge cases (i.e., not being logged in, trying to check out with no books, etc.) and system error states (i.e., failure to load a requested book's info page). For the "Not Logged In" case, I also <<included>> the login use case; the user would be prompted to do so immediately after this error.

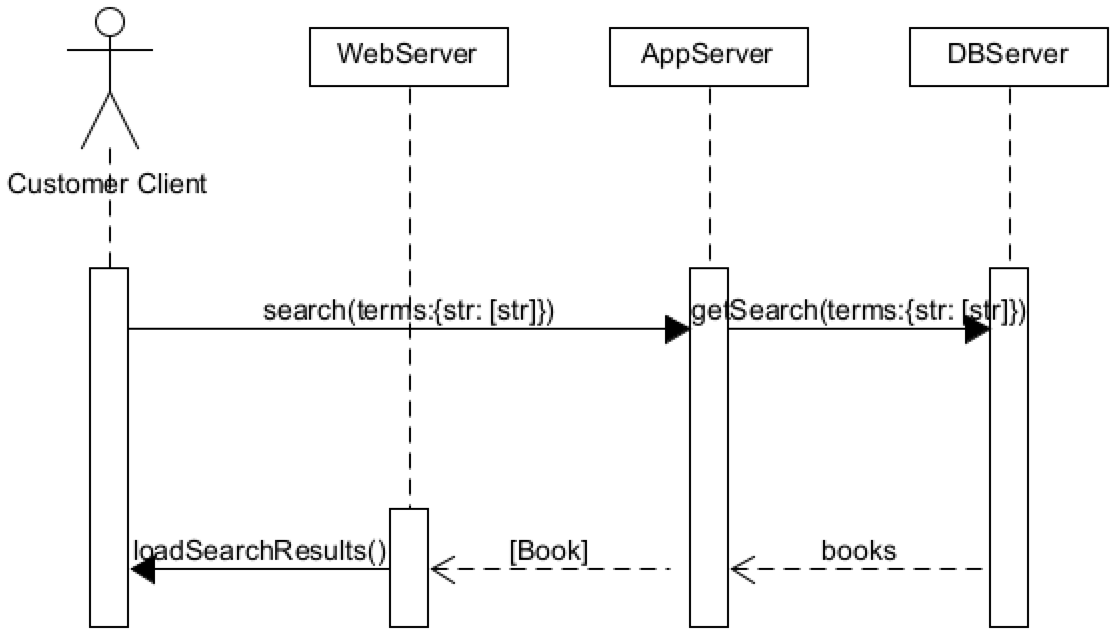
Class Diagram



- The WebServer contains page-loading functions, and is *composed* of one PaymentServer and one AppServer.
- The AppServer stores the currently logged-in user and contains various functions of the website; it is *composed* of a DBServer, and it is associated 1-to-1 with a User.
- The DBServer contains read-only lists of books and users in the system; thus, it aggregates a number of each into itself.

- A User contains various user-specific functions and information. It is associated with a number of books (in cart and history) and is composed of a paymentInfo object by a 1:1 relation.
- PaymentInfo is simply a read-only class containing the user's payment info.
- The PaymentServer aggregates a list of users; it can then simply call executePayment with an input user to automatically execute a payment using the user's cart and their paymentInfo object.
- A Book is another read-only class containing information on a book. Books are contained in both the DBServer and in Users; these are passed throughout the entire program for it to work together.

Sequence Diagram



This diagram represents the "Search for Book" use case. Here, the Customer Client uses the "search function": i.e., he places search terms into the site search bar and presses "Search". (This can happen anywhere via the on-page GUI.) These terms are passed to the AppServer object, which then does the "dirty work" of looking through the DBServer to find books that apply to the terms.

(Note: Terms are organized as a string dictionary; for instance, "category": "horror" could be a possible key-value pair. In other words, the search function is intended as a catch-all to find category, keyword, author, and/or ISBN searches, depending on the user's input.)

By looking at the DBServer’s book list, the AppServer grabs the relevant sub-list and passes it to the WebServer. Finally, using this data, the WebServer renders the search results page and passes it back to the Customer Client.