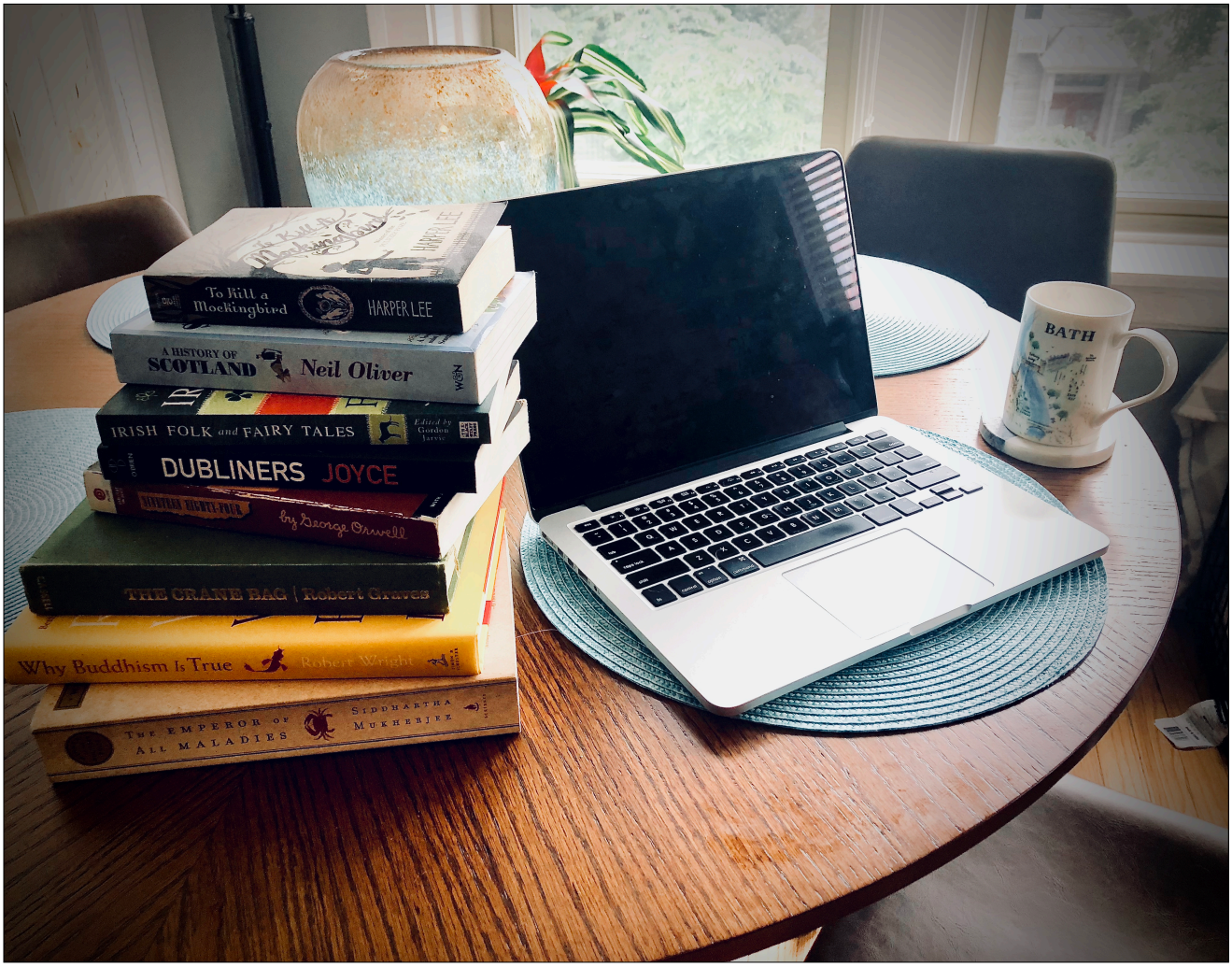


Book Note Book

CSC 452 Object Oriented Enterprise Computing

Ben Huff

November 11, 2018

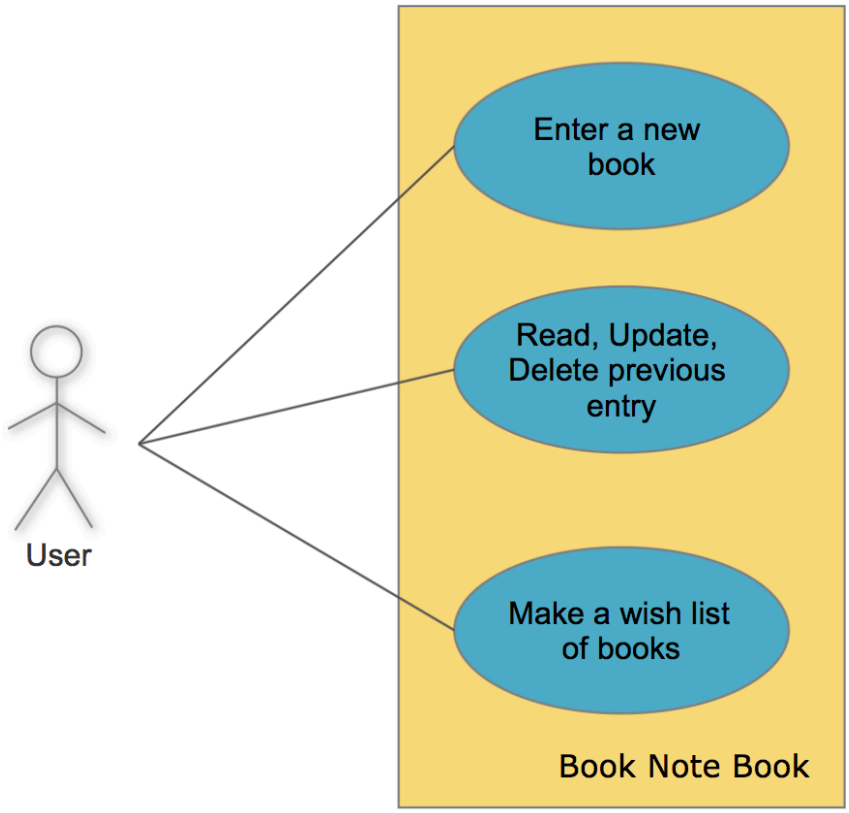
CONTENTS

Overview

An application that lets a user track what books they have read.

Requirements

Use Case



Description of problem

If you read frequently, it can be difficult to remember all the books you've consumed. Remembering how you felt about a particular book, or when you read it, ends up being an approximation that gets less precise as the years go by.

The reader, or user, needs a web interface where they can log all of this information. Amazon's GoodReads essentially solves this problem, but the interface is busy and confusing to navigate.

In this stripped-down version of GoodReads, the user will log on to the website using their credentials. The landing page will give the user three options: (1) Enter a book, (2) View previously entered books, and (3) Make a wish list of books.

When the user chooses to enter a new book, they'll input the particular details of the book, e.g., title, author, genre. Next, they'll rate the book on a scale of 1 to 5 and enter a review that can be up to three paragraphs.

When the user chooses to view previously entered books, a list will appear. They will have the option of viewing the details of a particular book.

When the user chooses to make a wish list, they will simply enter the title, author, genre, and ISBN. If, at a later time, the user reads a book that's on their wish list, they can click a button to give it a date read, a rating, and a review. The book will then be removed from their wish list and added to this list of books read.

The user can also accrue badges for the books they've read. If the user reads a book from a certain category, like classics, and that book is on the list of the 100 greatest classics, then they'll earn a badge for that book. There are lists for other areas as well, like history and politics.

Design

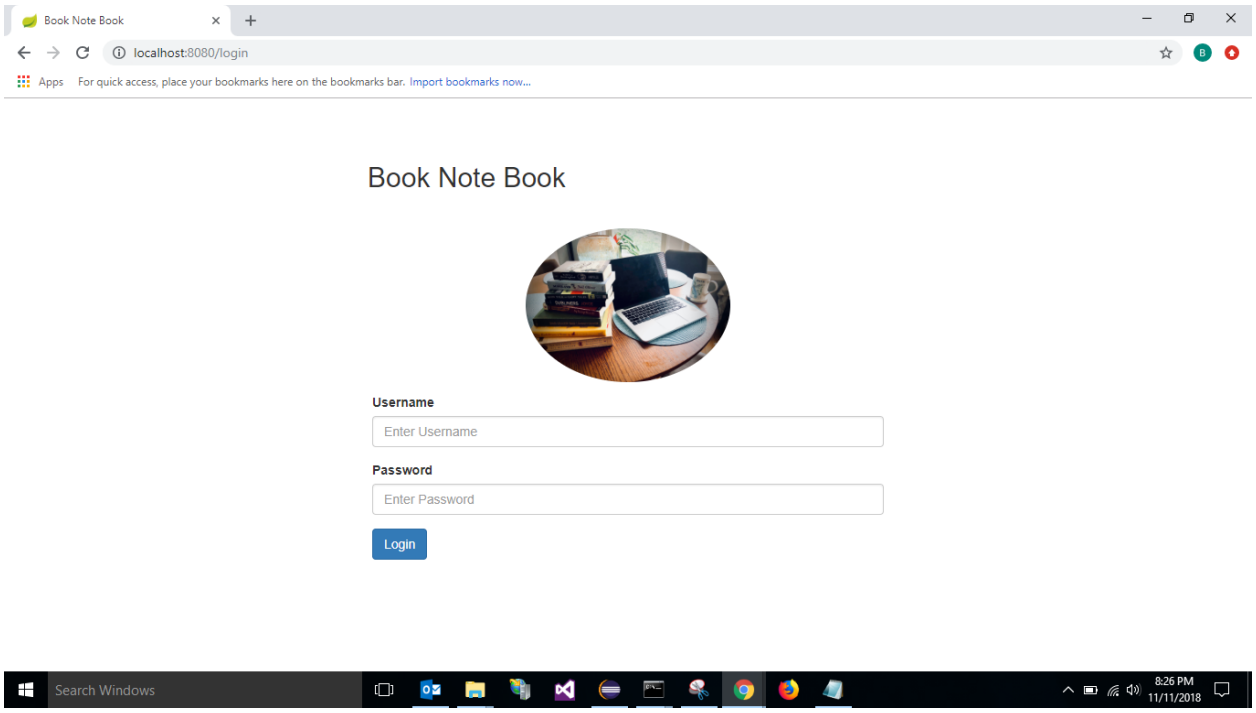
My application uses an in-memory JDBC database. It also uses a few design patterns such as Factory and Singleton. For the front end, it uses HTML, CSS, and some jQuery.

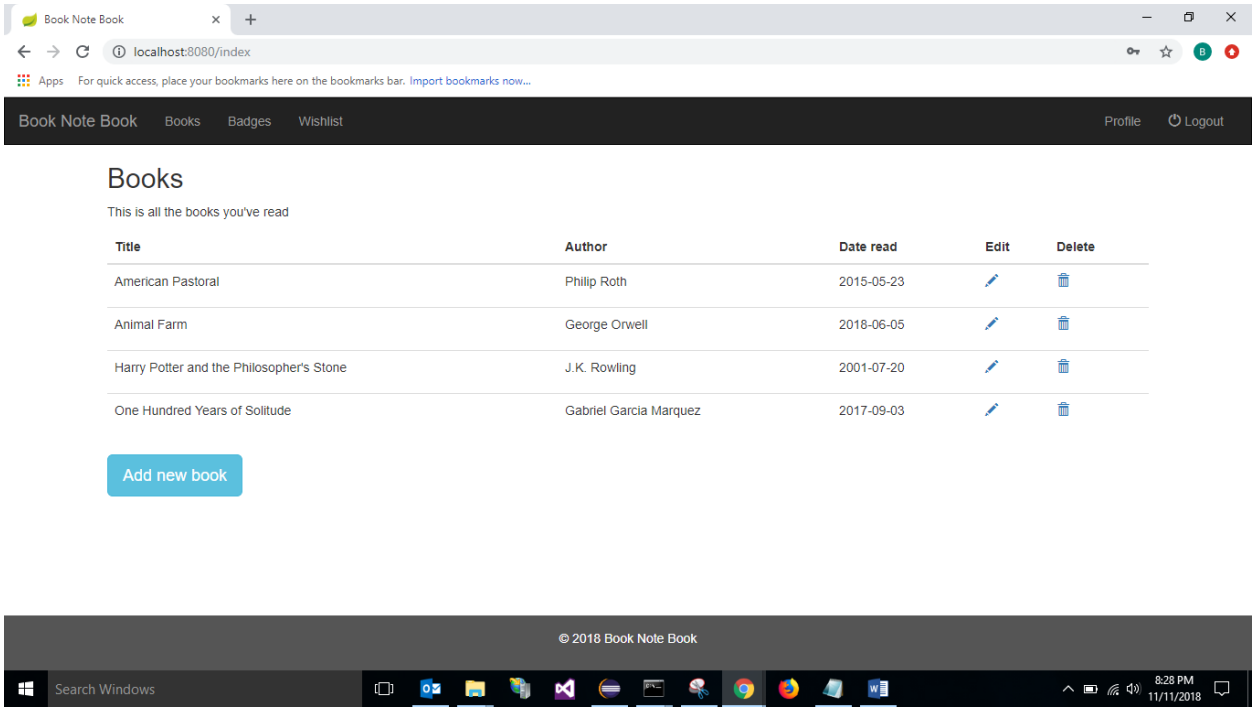
Discussion of how your design met the requirements

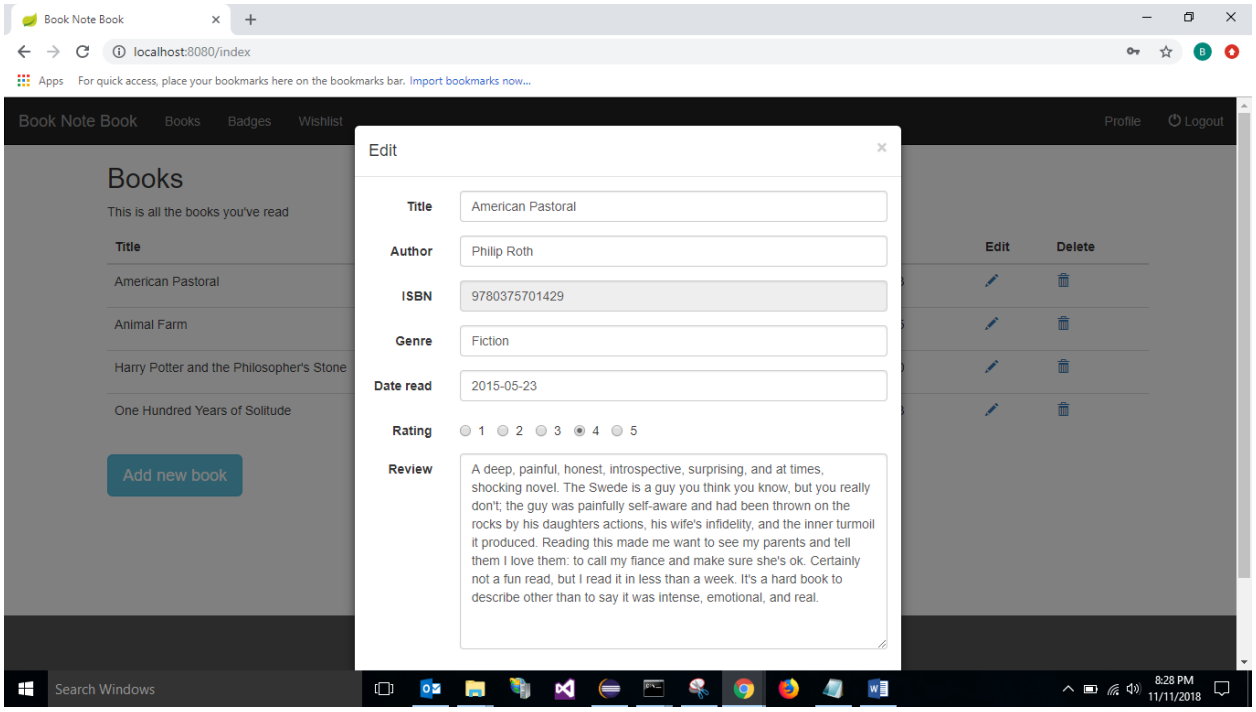
My design met the requirements precisely. My design lets user keep track of the books they read in a clear and easy-to-use manner. For example, the user can add a book to their wish list via a pop up—once they've read that book, they can quickly add it to the list of books they've read via a separate pop up which has some of the book's data filled in for them.

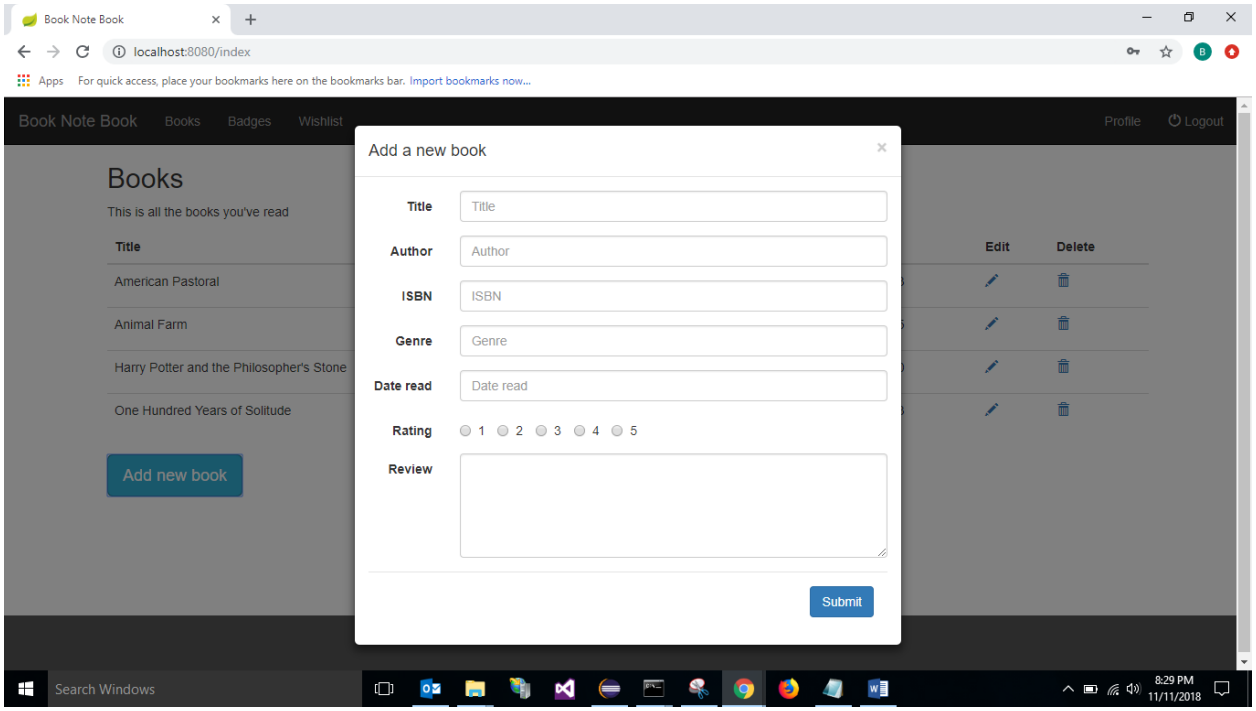
I'm particularly proud of the application's simplicity. It has just what it needs in terms of buttons and links—no more, no less. Each link has an intuitive icon (e.g., a pencil for editing, a check mark for finishing).

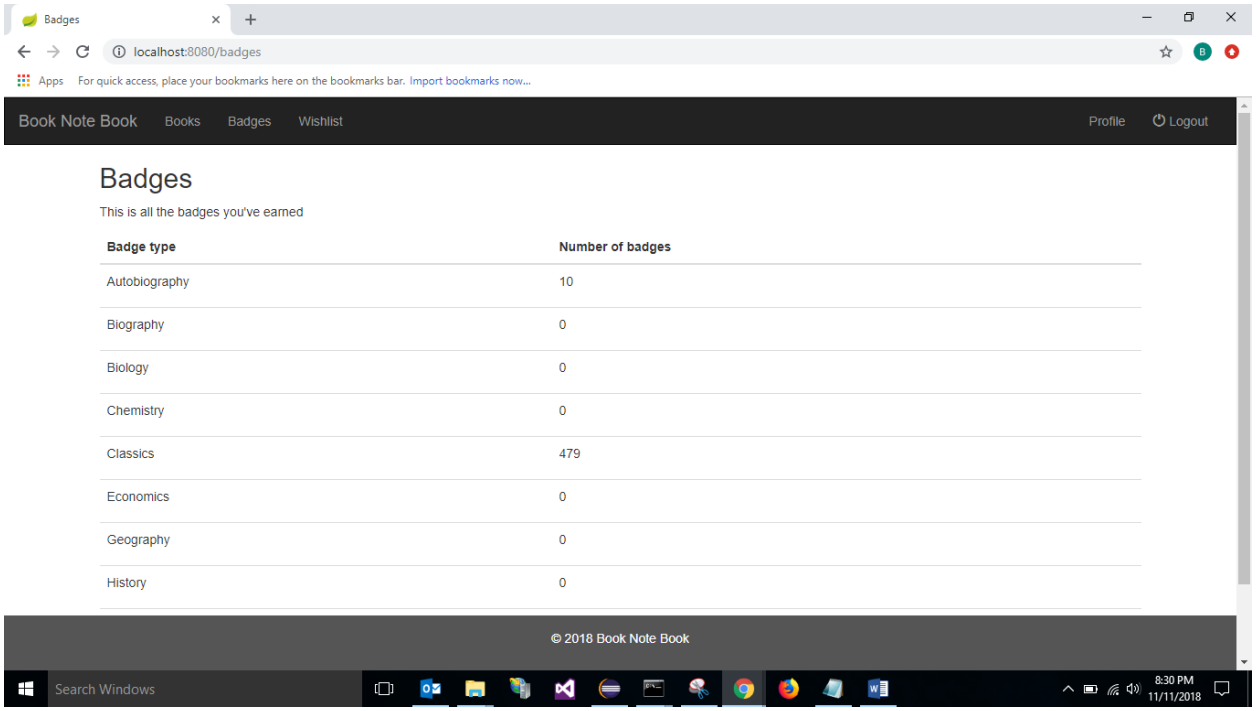
Mockup

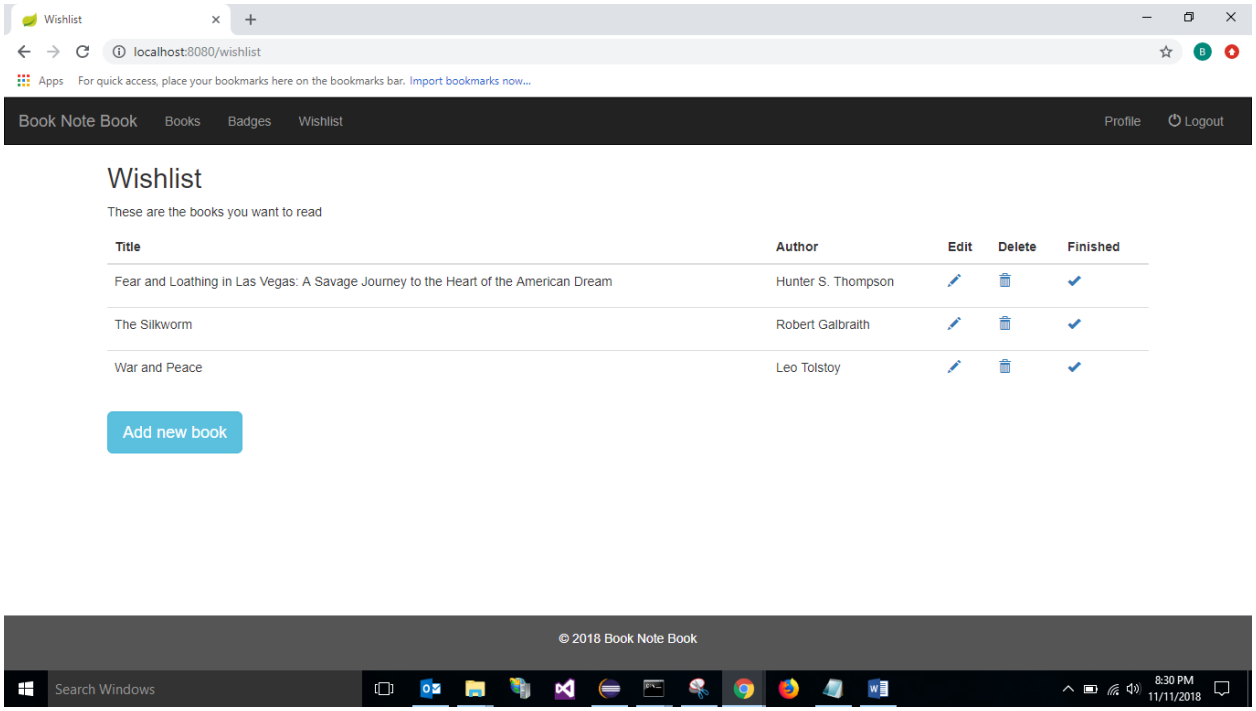


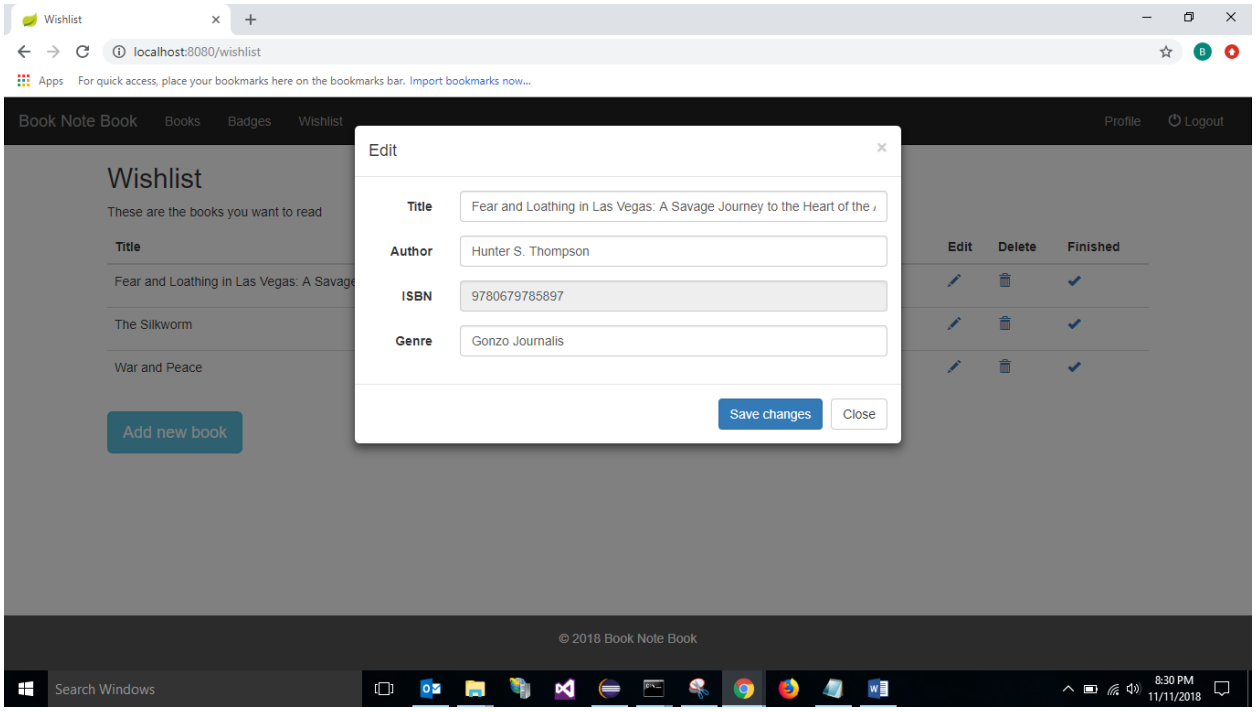

Book Note Book
localhost:8080/index
Apps
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Book Note Book
Books
Badges
Wishlist
Profile
Logout
Books
This is all the books you've read
Title
Author
Date read
Edit
Delete
American Pastoral
Philip Roth
2015-05-23
Animal Farm
George Orwell
2018-06-05
Harry Potter and the Philosopher's Stone
J.K. Rowling
2001-07-20
One Hundred Years of Solitude
Gabriel Garcia Marquez
2017-09-03
Add new book
© 2018 Book Note Book
Search Windows
8:28 PM
11/11/2018

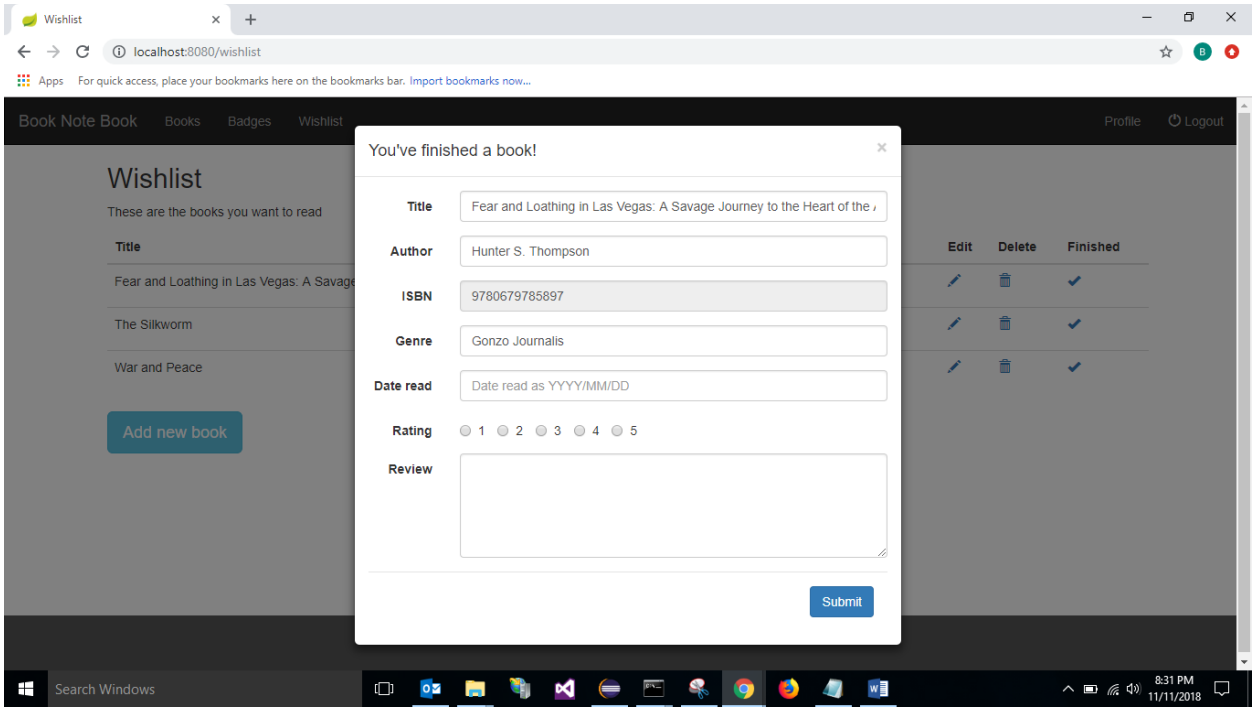

Book Note Book
localhost:8080/index
Apps
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Book Note Book
Books
Badges
Wishlist
Profile
Logout
Books
This is all the books you've read
Title
Edit
Delete
American Pastoral
Animal Farm
Harry Potter and the Philosopher's Stone
One Hundred Years of Solitude
Add new book
Edit
Title
American Pastoral
Author
Philip Roth
ISBN
9780375701429
Genre
Fiction
Date read
2015-05-23
Rating
1
2
3
4
5
Review
A deep, painful, honest, introspective, surprising, and at times, shocking novel. The Swede is a guy you think you know, but you really don't; the guy was painfully self-aware and had been thrown on the rocks by his daughters actions, his wife's infidelity, and the inner turmoil it produced. Reading this made me want to see my parents and tell them I love them: to call my fiance and make sure she's ok. Certainly not a fun read, but I read it in less than a week. It's a hard book to describe other than to say it was intense, emotional, and real.
Search Windows
8:28 PM
11/11/2018

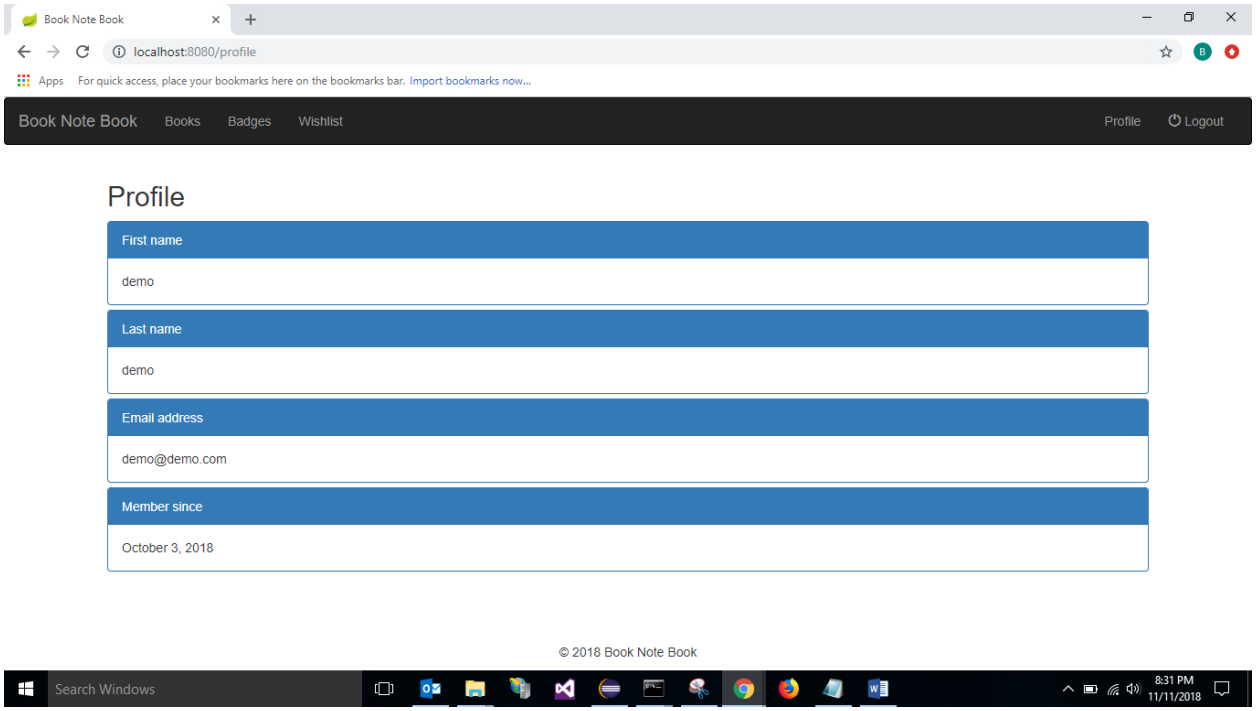

Book Note Book
localhost:8080/index
Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Book Note Book Books Badges Wishlist Profile Logout
Books
This is all the books you've read
Title
Edit
Delete
American Pastoral
Animal Farm
Harry Potter and the Philosopher's Stone
One Hundred Years of Solitude
Add new book
Add a new book
Title
Author
ISBN
Genre
Date read
Rating
1 2 3 4 5
Review
Submit
Search Windows
8:29 PM
11/11/2018


Badges
localhost:8080/badges
Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Book Note Book Books Badges Wishlist Profile Logout
Badges
This is all the badges you've earned
Badge type
Number of badges
Autobiography 10
Biography 0
Biology 0
Chemistry 0
Classics 479
Economics 0
Geography 0
History 0
© 2018 Book Note Book
Search Windows
8:30 PM
11/11/2018


Wishlist
localhost:8080/wishlist
Apps
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Book Note Book
Books
Badges
Wishlist
Profile
Logout
Wishlist
These are the books you want to read
Title
Author
Edit
Delete
Finished
Fear and Loathing in Las Vegas: A Savage Journey to the Heart of the American Dream
Hunter S. Thompson
The Silkworm
Robert Galbraith
War and Peace
Leo Tolstoy
Add new book
© 2018 Book Note Book
Search Windows
8:30 PM
11/11/2018


Wishlist
localhost:8080/wishlist
Apps
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Book Note Book
Books
Badges
Wishlist
Profile
Logout
Wishlist
These are the books you want to read
Title
Edit
Delete
Finished
Fear and Loathing in Las Vegas: A Savage
The Silkworm
War and Peace
Add new book
Edit
Title
Fear and Loathing in Las Vegas: A Savage Journey to the Heart of the A
Author
Hunter S. Thompson
ISBN
9780679785897
Genre
Gonzo Journalis
Save changes
Close
© 2018 Book Note Book
Search Windows
8:30 PM
11/11/2018


Wishlist
localhost:8080/wishlist
Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Book Note Book Books Badges Wishlist Profile Logout
Wishlist
These are the books you want to read
Title Edit Delete Finished
Fear and Loathing in Las Vegas: A Savage
The Silkworm
War and Peace
Add new book
You've finished a book!
Title Fear and Loathing in Las Vegas: A Savage Journey to the Heart of the A
Author Hunter S. Thompson
ISBN 9780679785897
Genre Gonzo Journalis
Date read Date read as YYYY/MM/DD
Rating 1 2 3 4 5
Review
Submit
Search Windows
8:31 PM
11/11/2018


Book Note Book
localhost:8080/profile
Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Book Note Book Books Badges Wishlist Profile Logout
Profile
First name
demo
Last name
demo
Email address
demo@demo.com
Member since
October 3, 2018
© 2018 Book Note Book
Search Windows
8:31 PM
11/11/2018

Discussion of lessons learned

The project you start with is most likely not going to be the project you end with. To expand the scope of my project (and to give the NoSQL database something to do) I plan on incorporating badges for users reading certain types of books (e.g., if you read a book on history, you get a history badge).

As of **milestone 3**, I have successfully included a NoSQL database, specifically MongoDB. It took longer than expected to get it to work. As the code currently runs, it's not updating the badges correctly—it'll make a document in MongoDB for a user, but it's not incrementing values like it should. I had this functionality working earlier, but when I tried to separate my code out into packages, it stopped working. I didn't have enough time to get it to work before the deadline.

As of **milestone 4**, I have my front end code connected with my backend databases, both NoSQL and SQL. I've learned that using my current JDBC approach to everything involves a lot of coding and covering for edge cases to ensure that there's no errors. As the code currently stands, it won't update books correctly.

As of **the final submission**, I have all of the essential components of the application completed. The user can do CRUD operations for both their list of books read and their wishlist, as well as view the number of badges they have.

I learned a few things while working on this project.

First, take time to name items well, whether it's variables, functions, or classes. Naming something generic like 'myButton' means that you'll have to come back and change it later. If you don't, it'll be hard to make changes to your code later because names don't have any explanatory meaning.

Second, when you start trying to connect your frontend code with your backend code, you realize that a lot of the code you wrote for the backend isn't useful. The BookService class became massive, with many methods that weren't being used. I had to go through and delete most of them.

Decision log

Milestone 2:

I decided to go with the in-memory database and hard-coding SQL queries. I'm familiar with this approach and it seemed the most straightforward to me. I plan on implementing a NoSQL database, perhaps MongoDB, but I didn't have time to get this done before the milestone's due date.

I tried to write generic methods to access the data via BookService. For example, readTable() where you pass in the table you want to read and the specific user's ID. I took a similar approach with updateTable() and deleteFrom().

In the code, I define a schema for two tables: WishList and RecordedBooks. The schemas are the following:

CREATE TABLE WishList (

```
        USER_ID int NOT NULL,
        TITLE varchar(250)  NOT NULL,
        AUTHOR varchar(150) NOT NULL,
        GENRE varchar(100),
        ISBN varchar(50)

        PRIMARY KEY (USER_ID, TITLE, AUTHOR)
);
CREATE TABLE RecordedBooks (
        USER_ID int NOT NULL,
        TITLE varchar(250)  NOT NULL,
        AUTHOR varchar(150) NOT NULL,
        GENRE varchar(100),
        ISBN varchar(50),
        DATE_READ Date,
        RATING int

        PRIMARY KEY (USER_ID, TITLE, AUTHOR)
);
```

Milestone 3:

The feedback I received from milestone 2 said I needed to add more tables. So, I added a new feature to the project: badges. I looked around the internet and found lists of books from different areas such as history, chemistry, and philosophy that are said to be must-reads. If a user adds a new book that is from one of these lists, they get a badge for that topic (the badges are maintained in a NoSQL/MongoDB). The code reads .txt files and inserts the data into tables—each .txt file maps to its own class and table. I utilized two design patterns to achieve the badge functionality: Factory and Singleton.

Part of the feedback from milestone 2 was to document my database implementation. I'm using JDBC with an in-memory approach (i.e., every time the application starts, it starts over again with only seed data).

Initially, I was going to try and code all of the front end without Bootstrap. After a fair amount of trial and error, I decided to use Bootstrap and it looks basic but slick.

I also separated my code into different packages, e.g., service, book, controller.

Milestone 4:

The code has been updated to integrate with the databases. A user can add a book to their read books list, and they can add a book to their wish list. As the code currently stands, there are errors when you try and edit a read book's values (e.g., the rating you gave it). This will be fixed before the final submission.

Final Submission:

I decided that the user shouldn't be able to change the ISBN of a book they've entered, whether it's on the reviewed books list or the wish list. The reason for this is because 'update' operations became more complex if you let the one truly unique aspect of a book be changed.

I decided to have a users table. For demo purposes, I wanted the user to be able to walk through a session using the application. I know this isn't the most ideal way to store users, but I think it enables proof of concept.

I decided to use some extra CSS, not just Bootstrap. The reason for this was that Bootstrap wasn't able to accommodate every aspect of the way that I wanted the application to look. In addition, I had to incorporate some jQuery to get all of the modal popups to behave correctly.