READr - A Book Tracking Application

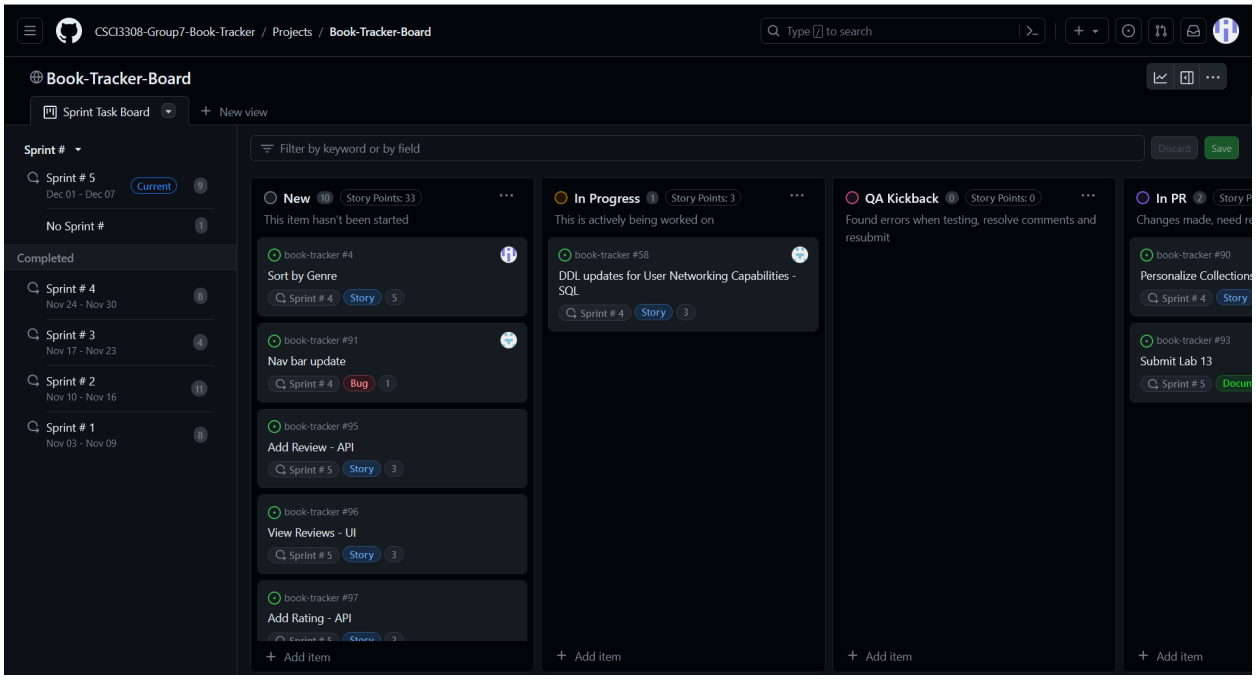
Contributors:

- Ibrahim Aldulaijan
- Beth Belay
- Elizabeth Cutting
- Lucas Fiedler
- Rayaan Lodhi

Description:

READr is an interactive app designed to improve the experience of finding your next good book. With READr, you can create a personalized account and use it in whichever way you see fit. All passwords are safely stored, and users can be confident in this application's security. Within your individual account, you can search for books and see numerous results related to your search. You can see how popular books are by reading the reviews left by other users, and you can leave your own reviews to let other users know how much you liked or disliked a book. Reviews are text based, so you can be as detailed as you would like. You can also see the date that reviews were left on, which can help you decide on each review's relevance to your current reading journey. All of the books that you read will be stored in your personal Collections page, which tracks all of your books that you add from the explore page. These books will be unique to your profile, so you can see your taste develop and improve over time. READr also has a friend feature, so you can follow others and see what books they like to read.

Project Board: https://github.com/orgs/CSCI3308-Group7-Book-Tracker/projects/1



Video:
Here is a link to the demonstration: https://youtu.be/u9vGG9OqxDc

VCS: https://github.com/CSCI3308-Group7-Book-Tracker
We used github to manage our project board, which we used to track user stories and stick with our agile methodology.

Contributions:
Ibrahim Aldulaijan: Contributed to stylization and bug fixing, as well as revising and developing the Index file.

Beth Belay: I also worked on the front end of the collections page and made sure that the cards were displaying neatly with the right books and information. I worked on the API calls with Liz to make sure that collection page was up and running. I helped make the add book button on the modal that would help with adding the book from the explore page to the collection page.

Elizabeth Cutting: I worked with the external API, and set up the explore page to receive this call, which included both frontend and backend pieces. I also worked with Beth to create both the backend and the frontend of the collections page. I contributed to a few bug fixes along the way and worked on the project report.

Lucas Fiedler: Worked on the Profile page stats and friends feature and Explore page review feature. Contributed to front and back end with a focus on front end formatting and user experience.

Rayaan Lodhi: Largely contributed to the reviews feature of the Explore page, revised/overhauled the Index/database to support changes, developed the friends system, implemented the Welcome page, designed the project board. Implemented test cases and modals.

Use Case Diagram:
- Looking up a book's details
- Shows user the books they have read in a bookshelf
- Seeing how many pages is in a book
- Track how many pages read
- Search for certain books by title
- Search for certain books by author
- Can search for a magazine as well
- Returns results in order of the relevance of search terms (this is the default) Shows descriptions of book

- Shows book ISBN
- Shows book year published
- Show a book's reviews
- Show review date
- Allow a user to leave a review
- Allow a user to follow another
- Show's another users stats to their followers

Test Results:
We focused our test cases on the login and user accounts aspect of our project, since we deemed this to be most important due to user security and creating proper session variables. We also used sinon to be able to preserve the user-free state of the database, which was challenging but enhanced our confidence in our code. We had seven total test cases:

Use Case 1: The user can log into a preexisting account:

Test Case 1: ✓ Positive : Successful Login (1348ms)
This test case checks if the user can log into an account. For this test case, we use the admin account because we always insert this account when starting up our database. This requires functionality of the password encryption and decryption and database issue.

Test Case 2: ✓ Negative : User Does Not Exist (133ms)
This test case also requires functionality of the login system, but this checks if the username does not exist. This is one of the first checks that we must perform when a user attempts to login.

Use Case 2: Usernames that are too long or invalid are rejected:

Test Case 3: ✓ Negative : Username Too Long (249ms)

This test ensures that the user is properly following the user policy. The username has a fixed maximum length, and the proper error message must be sent if the user does not comply with these guidelines.

Use Case 3: A new user can register for an account:

Test Case 4: ✓ Positive : Successful Register (254ms)
This positive test case is designed to see if a user's data is properly stored in the database after they submit the registration form. It utilizes a SQL query to ensure this is done correctly.

Test Case 5: ✓ Negative : User Already Exists (223ms)
The negative test for this use case ensures that if the user already exists, they cannot create another account with the same username, because this would be problematic for our system, both within the login and register features. Thus, we check if the user is already stored in the database, and we prompt the user to sign in instead of creating a duplicate account.

Use Case 4: A user will be asked to retry if their password is incorrect:

Test Case 6: ✓ Negative : Incorrect Password (244ms)
The password will need to be hashed and compared to the encrypted password stored in the database. We are using the bcrypt library to accomplish this, and we check this by entering an incorrect password and ensuring that the user is prompted to retry, as opposed to being given access to their account.

Deployment:

We deployed our project with Azure, using a free service virtual machine. We took it down to save on hours, but here is a screenshot. Link to Website(not running):

http://recitation-11-team-07.eastus.cloudapp.azure.com:3000/