Team Number: 011 - 6

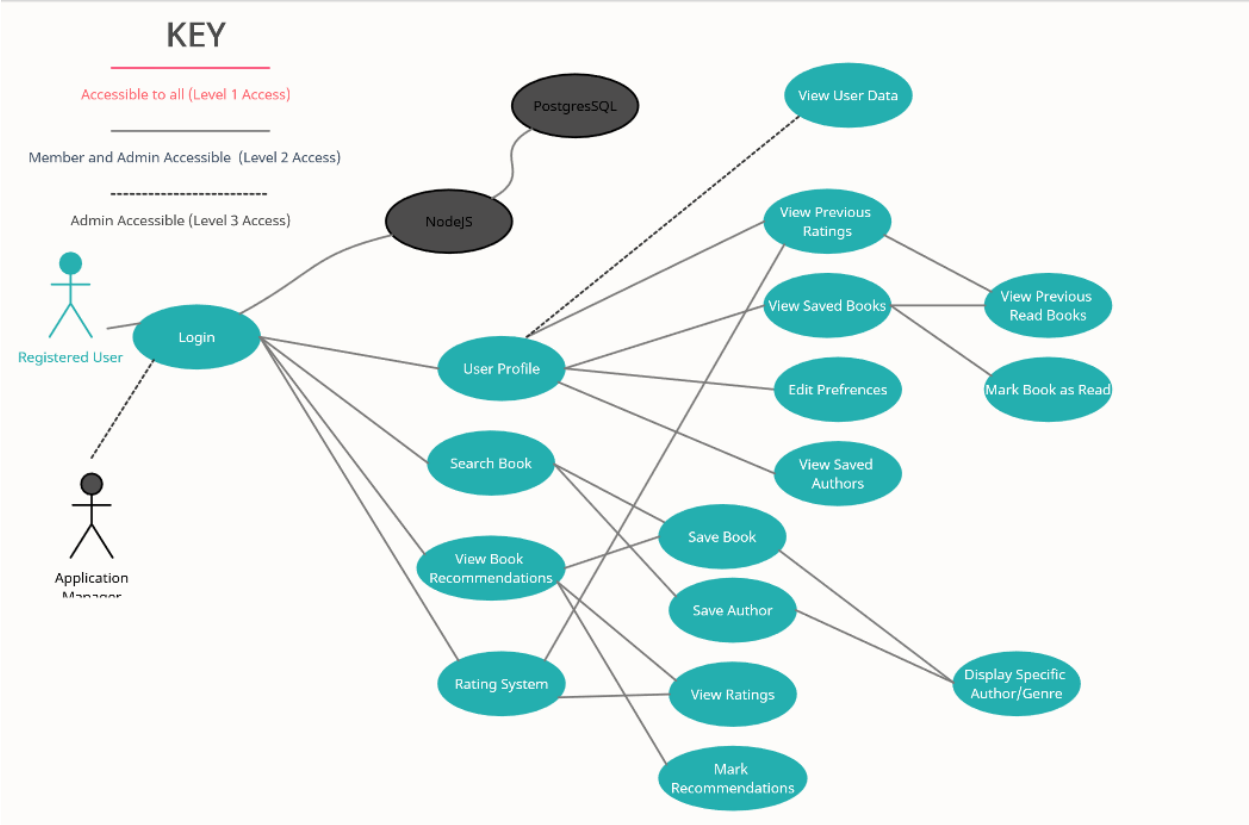
Team Name: JAAAC

Features Completed:

- Front End: All pages (login, registration, profile, recommendations, and search) have a basic HTML structure and standard style that will be used across the website. We believe each page to
- Front End: Each HTML page links to the other correctly (so users can navigate to all the pages)
- Front End: The search bar auto-populates from a list
- Integration Layer: The server.js file connects to the books database and can complete a post request that allows the registration page to add a user to the users database
- Back End: The users database has been initialized
- Back End: The books database has been initialized (contents loaded from a CSV file found on Kaggle)

Architectural Diagram:

https://app.creately.com/diagram/6AmYhNEDesV/edit



Working Parts (during demo):

Demo: Thursday, March 10 at 7 PM

- Each page links to the others (ex: the login page connects to the recommendations page, the profile page is accessible from the recommendations page, etc.)

Issues Faced During Development/Demo:

- Registration Page
    - On the registration page, users must confirm the password for their account. We had trouble checking that the passwords met all the requirements (character length, special characters, both password and confirm password match, etc.) but will refer to the Javascript lab.
    - Books might have multiple genres, so we struggled with how to store them in a database. We decided we will append each genre to an array, then store the list into the database.
- Search Page
    - There are thousands of books in our dataset, so cannot use a list to store all the titles and information. We must find a more efficient way to handle it.
- Recommendations Page
    - During development, we debated which algorithm we should use to suggest books to users. We decided that we will filter by genre, then show books in descending order by average rating.
- Home Page
    - The books and images on the current home page are hardcoded. We must figure out how to use Node.js to pull the books/images/information from the database.

Suggestions Offered:

- Registration Page
    - Refer to the Javascript 6 lab for the confirm password section
    - When calling the API, refer to two of the inputs (username and password) instead of all three (username, password, and confirm password). Do validation in the front-end - disable the registration button until the passwords match
    - Store user's preferred genres as an array in the database
- Search Page
    - Use the keyword LIKE on the database - this way, users will not have to type an exact match but will still see results that are close
- Home Page
    - Solidify a simple way of deciding which books will be shown
    - Make clear what genre each book falls under and keep them separated from one another
- Database
    - Use Python to generate one final CSV, then download it and upload it to SQL. This way, we won't have to use JOINs

    - Don’t incorporate Python into the HTML pages - Python doesn’t store the data
- Node.js
    - The Postman app helps call these APIs so we don’t have to go into Docker and debug it, but just debug normally if learning so many new technologies gets to be overwhelming
- Now that our HTML structure is done, merge all the code and start working on new features on different branches
- The front-end developers should write down everything they will need from the database so the back-end will know better how to structure it and what information should be included

Feedback from TA:

Great work!
The frontend has been developed well and the team is working on enhancing a few features and adding validations.
The lab exercises were leveraged and the feedback given during the standup meetings so far, have been taken into account.
They have acquired an external books dataset for their website which they will bulk-upload to their database.The team got their DB schema reviewed and is making progress with the setup.
They will now be working with NodeJS for their backend development.

Individual Contributions:

Cody Aker: I wrote the HTML code for the suggestions and the search pages and a javascript function for the search page to search through a list and auto fill

Al Haddad: I wrote the HTML code for the Login and Registration page, and helped set up the docker compose file. I also designed the nextPage logo for the website. Lastly, I made some minor tweaks to the architectural diagram to reflect the backend changes for login.

Anna Rahn: I wrote the basic HTML code for the profile page. I then found and cleaned the books.csv/google_books_dataset.csv files, which we will be using for our books database. I initialized both datasets in SQL and created/wrote the server.js file, which has one post request for the registration page. I then wrote our Project Milestone 3 document.

Abigail Sullivan: