COMP 730 + 830 Final Software Project Update 2

Team Members:

Charishma Bandari and Ben Heney

Past Project Progress:

Modified the User.java class to manage user information and authentication.

Implemented attributes like first name, last name, date of birth, address, phone number, library card number, issue type, email, and zip code in User.java.

Developed methods in User.java for user registration, information update, and retrieval.

Implemented functionalities in BookCatalog.java for adding, editing, deleting books, and retrieving book details.

Integrated the book catalog management system into the project structure.

Reviewed and updated code comments, documentation, and codebase documentation in both User.java and BookCatalog.java for clarity and ease of maintenance.

Ensured proper code structure and organization in both User.java and BookCatalog.java for readability and maintainability.

Developed and implemented the AppSettings and UserSettings classes, as well as the Database class and an installation script.

Past Team Member Activities:

Charishma Bandari:

Finalize and optimize the user authentication features in User.java, ensuring robust security measures and smooth user experience.

Conduct comprehensive testing of user registration, information update, and retrieval methods in User.java, addressing any bugs or issues promptly.

Reviewed and updated code comments, documentation, and codebase documentation in User.java

Ben Heney:

Began designing the Patron Kiosk home page GUI and implementing the Media, Book, and Search classes.

To-Do Items :

Maintain a field in the Book table to track the availability status of each book, such as "Available," "Checked Out," or "On Hold." Update the availability status based on check-in/check-out transactions and user actions. Ensure that user authentication functionalities are fully implemented and tested, including registration, login, password management, and session handling. Focus on enhancing user authentication by adding account details management functionalities, such as profile editing and password change. will continue to enhance the user authentication functionalities and improve data security. Develop a login system with secure authentication mechanisms such as hashing passwords. Create a password reset feature with email verification for added security. Complete the implementation of the AppSettings, UserSettings, and Database classes to support the application's backend infrastructure. Test and validate the installation script (install.sh) to ensure smooth deployment and configuration of the application. Complete the implementation of CRUD operations for book management, allowing users to add, edit, and delete books from the catalog. Develop advanced search and filter functionalities to Implement advanced search and filter options allowing users to refine book searches based on criteria like author, genre, publication year, ISBN, and availability status. Provide filters to sort and categorize search results for easier navigation. Provide filters to sort and categorize search results for easier navigation and improved user experience. Add account details management functionalities such as email change or recovery options.

Difficulties:

Validating the installation script (install.sh) and ensuring smooth deployment and configuration of the application on various environments was challenging. Ensuring that the installation process is user-friendly and error-free is crucial for a positive user experience.

On several occasions, Git indicated 'nothing to commit' even though we had made significant modifications to the codebase. Delays in pushing changes led to synchronization issues among team members, especially when working on interconnected modules. Additionally, the 'nothing to commit' situations caused confusion and required extra effort to ensure that all changes were properly committed and reflected in the version history, affecting our overall productivity and timeline.

Making Media an extensible class is proving challenging. Each implementation will have a unique list of searchable terms. For example, Book will have an ISBN, Title, Author, and Publisher. Film will have a Title, Director, Cast, and Producer. One approach is to create a simple

SearchableField class that provides a uniform interface for retrieving the field title and value. The Media class would maintain an array of SearchableFields; however, this approach is extremely inefficient as the field title cannot be static, but is repeated in every instance. Furthermore, this approach lacks the compile-time security that an Enum provides. Figuring out how to use an Enum to support this class has been difficult.