**Overview**

**The Online Library System**

Our system allows the users to create their personal user accounts, search for books, check them out and return them. There is also functionality for the librarian to have control over the books being checked out, the books in the library and the transaction information when a particular book is checked out.

The construction of our system begins with the Person class (superclass). It obtains two instance variables, Username and Password, which are both strings. Within the class, it contains multiple methods so it gives the other subclasses who extend to the superclass access to call upon that method. It contains three unique search() methods with distinct method variables. Each having to do to look up a book, author, etc. Its purpose is to have universal methods for other subclasses to call. The Student (subclass) only motive is to check and pay the balance due by the overdue books. The Librarian (subclass) only function is to provide methods that add, update and delete book information. Other roles are being supplemented by the main driver. The Book (subclass) contains information about the book: Author, Title, ISBN, Unique code, and Availability. It's only intention is to display the data of the book. The Transition (subclass) depends on the Book and Student classes. Transaction is like an example of a receipt; when a book is being checked out by a student. Where information such as Transition code, Transition date, Due date, Return date, and Book status is being recorded. The student as a user of the system is able to check out books, search for books and return the books. However, the Student class itself doesn’t contain these methods. They’re implemented through the main driver class Login. Login calls on Transaction class and creates objects of Student and Book classes.

Our system implements 3 CSV files to maintain the database of users, books and transactions. To keep track of the books included in each transaction, we have added a separate field to the Books.csv file that records the status of each individual copy in the library. Doing so, allowed us to increase the efficiency of the look-up between the transaction and the book. We have also implemented unique ID codes for each copy of the book so that we can keep track of each physical copy in our library. This was done to ensure that we can easily keep track of the inventory in the library. Additionally, you will notice that using our Interface, you can only create a profile as a Student and not the Librarian. This is was done for security reasons. The librarian is essentially an administrator of our system, so those credentials will have to be allocated through the university.

We have included the individual test cases for each class with our source code. We have tested functions of individual classes to ensure that they run with no bugs. However, a big part of our programs success relies on the user experience. Our main driver, the Login class, is responsible for all the user interaction. It’s a series of print statements to the console that rely on the underlying functions from other classes. As previously stated, the underlying functions have been tested separately in the included test suite. To test the actual driver, we had 10 fellow students (outside of our class) that aren’t familiar with the project scope, use the program. All 10 have conclusively stated that the UI was easy to follow, and they had no problem understanding how to perform the tasks in hand.

**The UML class diagrams**

