General responsibility assignment of library checkout app

* Creator
  + The object creation responsibility within this project is assigned to the Library-Checkout-domain submodule. Within this module each object entity that can exist within this project is defined with all values private. The ability to read, update, create or delete any object or property of these object is private to that object and defined in this submodule only.
* Low Coupling
  + Within this application, each submodule can act totally independent from each other submodule that concurrently runs at deployment. The Library-Checkout-UI submodule runs as its own server with its own dependencies as well. Each module has its own dependency managements, maven or node, and is capable of running and building in isolation.
  + Despite each submodule running in isolation, each one has a relatively similar dependency tree. All of the submodules that rely on maven use the same spring, hibernate, and JPA frameworks. This allows us to set a parent module with its own dependencies that we only need to update once to create a dependency cascade through all the maven submodules. Even with low coupling between submodules, we can maintain reusability through a single source of dependency management and updates.
* Controller
  + The Library-Checkout-API submodule is the responsibility assignment for all controller methods. Each entity, Librarian, Student or Book, has its controller defined and routed here. This submodule is strictly the controller though, it only receives data and passes it to the correct data interpreter source elsewhere in the application. No database inserts occur here, only caching.
  + Each controller is also defined in isolation only to its entity. All Book routes are in a controller that is strictly for book CRUD operations, no other entities are managed here, thus ensuring low coupling as well.
* High Cohesion
  + Within our three primary entities, Librarian, Student, and Book, we have ensured nearly zero overlap between class assignments and responsibilities. While Librarian and Student share similar non primary key properties, such as name and email, the assignment of these entities in the application has no overlap and can be non-unique. Each one of these entities assumes the minimum responsibility, all the methods available to each of these classes pertains only to that class itself, no global methods are used in any.
* Information Expert
  + In this application, we have ensured that all application data is stored and retrieved in one place, the Library-Checkout-data submodule. This submodule is responsible for all data handling throughout the application. All controller routes utilize one of the repository helper methods to cache the request body that was passed. When a new page is loaded that requires this data, again the repository is accessed for the entity. Any searching for an entity by Id, deleting by Id, creating, or updating will occur in the repository once the request has been routed through, he controller. All requests are cached before DB insert to increase performance.