Deliverable 2: Part 1

UML Diagram

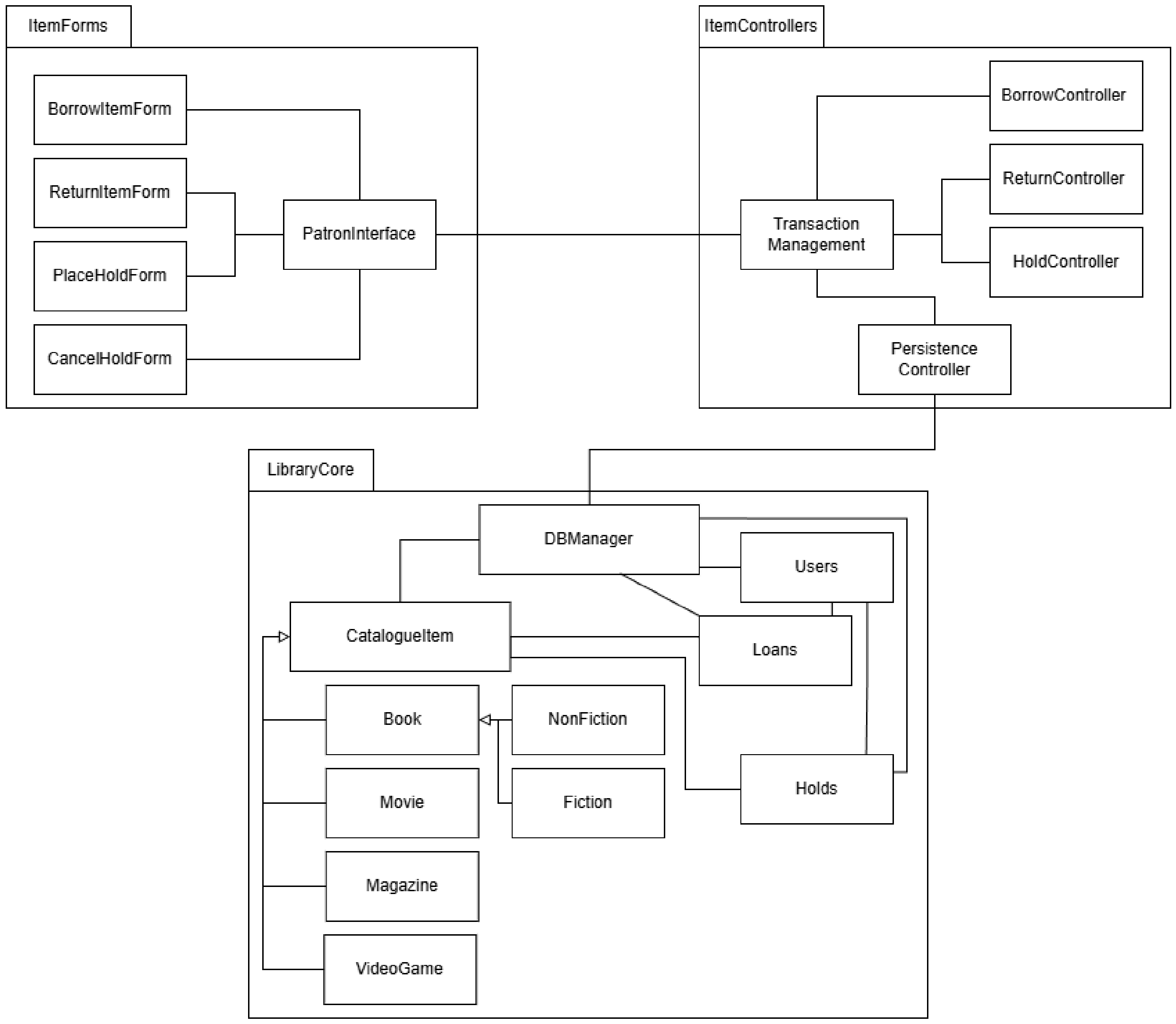

Figure 1: HinLIBS System Decomposition Showing Subsystem Organization and Dependencies

Justification

This subsystem design was chosen based on the use cases: Return Item, Borrow Item, Place Item on Hold, and Cancel Hold on Item. Based on these use cases, we agreed on the design shown in Figure 1, as it minimizes coupling and increases cohesion within a subsystem. In this design, we chose to have one form per use case, containing all boundary objects, to isolate UI changes within each form and avoid requiring changes to the controllers and databases.

The TransactionManagement object acts as a subsystem interface for the Item controller subsystem, preventing the UI from needing to know the inner workings of the item controller classes and connecting the UI to the LibraryCore through the PersistenceController by delegating work to that object to handle calls to the DB Manager. With this design, the Controllers are responsible for calling data access objects for storage-related operations rather than the UI. This design fits within the (Entity-Boundary-Control) design, as each subsystem contains the majority of one of the three class types.

In addition to low coupling, this design also exhibits high cohesion, as seen in LibraryCore and ItemForms. In the Library core, the top-level classes are related to each other through aggregation, providing strong bookkeeping across multiple relationship types. In the UI, these forms are cohesive UI objects, and with the form classes focused solely on presenting their respective data, each concern is in its own place.