**Introduction**

The Mini Library Management System is a Python-based project designed to simplify the management of books and library members. The system uses core Python data structures such as dictionaries, lists, and tuples to handle various operations efficiently. It enables users to perform basic CRUD (Create, Read, Update, Delete) operations while supporting additional functionalities like borrowing and returning books. This project provides an opportunity to apply programming concepts such as data organization, modular coding, and function implementation to simulate a small-scale library system. It also emphasizes problem-solving, logical structuring, and testing through assert statements to ensure functionality and reliability. The primary objective of this system is to implement core CRUD (Create, Read, Update, Delete) operations alongside specialized borrowing and returning functionalities. By leveraging dictionaries for books and lists for members, the system ensures fast lookups and efficient data management. The use of tuples for genre validation ensures data integrity by preventing unauthorized modifications to the predefined genre categories.

**Design Rationale (Humanized Explanation on Use of Dictionary, List, and Tuple)**

When designing the Library Management System, I chose to use dictionary, list, and tuple because each one plays a special role in keeping the system organized, fast, and reliable just like how a librarian uses different tools for different tasks.

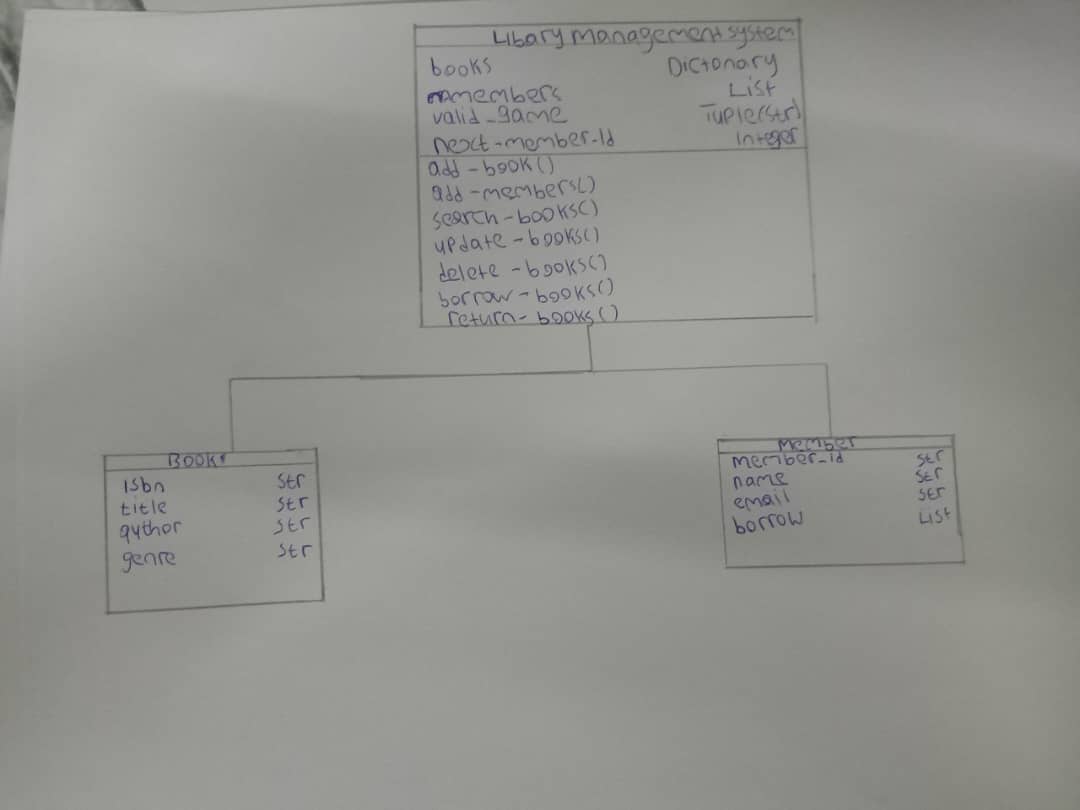
The dictionary is like a smart filing cabinet. Each book has its own unique ID, and with that ID, the system can quickly find all the book’s details -+-its title, author, and whether it’s available or borrowed. This saves time and prevents confusion, just like a librarian instantly finding a book by its code.

The list works more like a record book that keeps track of things that change often such as new books added, members joining, or books being borrowed. It’s flexible, allowing new items to be written in or removed easily whenever the library updates its records.

The tuple, on the other hand, is like a notice board where important rules or schedules are posted. These don’t change—like library opening hours or book categories so using a tuple ensures that such information stays safe and unaltered.

In short, the dictionary helps with quick searching, the list handles everyday updates, and the tuple protects fixed information. Together, they make the Library Management System more human-friendly, organized, and dependable just the way a good library should be.

**UML DIAGRAM**



**Conclusion**

In conclusion, the Mini Library Management System demonstrates how fundamental Python data structures and functions can be combined to create an effective management tool. By utilizing dictionaries for books, lists for members, and tuples for genres, the system ensures easy data access and manipulation. The implementation of CRUD operations, borrowing, and returning processes further showcases the practical application of programming logic in solving real-world problems. This project not only strengthens programming and debugging skills but also lays a strong foundation for developing larger and more advanced systems in the future. Through comprehensive testing and validation, the system proves capable of handling essential library operations including book cataloging, member registration, borrowing transactions with copy limitations, and book returns. The modular design of the functions ensures maintainability and scalability for future enhancements.

This project reinforces fundamental programming concepts including data structure selection, function design, error handling, and systematic testing—all critical skills for software development. The system provides a solid foundation that could be extended with additional features such as due date tracking, fine calculations, or database integration for persistent storage.