**DESIGN RATIONALE**

# **MINI LIBRARY MANAGEMENT SYSTEM – DESIGN RATIONALE**

# **DATA STRUCTURES USED**

**1. Dictionary**

* **Used for**: Storing book records
* **Why**: Dictionaries allow fast lookup using unique keys (ISBN numbers). Each book is stored as a key-value pair, where the key is the ISBN and the value is another dictionary containing details like title, author, genre, and copy counts.
* **Benefits**:
  + Access time for retrieving book data
  + Easy to update or delete specific books
  + Clear structure for nested data

**2. List**

* **Used for**: Storing member records
* **Why**: Lists are ideal for storing multiple member entries, each represented as a dictionary. They allow flexible iteration and dynamic addition/removal of members.
* **Benefits**:
  + Simple to loop through for searches and updates
  + Supports dynamic growth as new members are added
  + Easy to filter or modify specific entries

**3. Tuple**

* **Used for**: Defining valid book genres
* **Why**: Tuples are immutable, making them perfect for fixed categories like genres. This prevents accidental changes and ensures consistent validation.
* **Benefits**:
  + Safe and unchangeable
  + Efficient for membership checks (if genre in GENRES)
  + Helps enforce data integrity

# **DESIGN DECISIONS**

* **Modular Functions**: Each operation (add, update, delete, borrow, return) is handled by a separate function for clarity and reusability.
* **Validation Logic**: Functions include checks for duplicate entries, borrowing limits, and genre correctness.
* **Scalability**: The system can be extended to include file storage, GUI, or database integration in future versions.

# **SUMMARY**

This design balances simplicity and functionality using Python’s built-in data structures. Dictionaries provide fast access, lists offer flexibility, and tuples enforce rules. Together, they form a clean and efficient foundation for managing books and members in a small library system.