Design Rationale – Mini Library Management System

Overview
This Mini Library Management System is designed using core Python data structures to manage books, members, and genres. The goal is to make the system simple, fast, and easy to modify.
Data Structure Choices
1. **Dictionary (books)**
The dictionary is used to store books using the ISBN as a unique key. This allows fast lookup, updating, and deletion of book details. Each book record stores title, author, genre, and number of copies.

2. **List of Dictionaries (members)**
Members are stored as a list because it allows flexible addition and iteration. Each member is represented as a dictionary containing their ID, name, email, and borrowed books. Lists are ideal since member order is not important, but the ability to search and update is.

3. **Tuple (genres)**
A tuple is used for genres since it contains a fixed, unchangeable set of allowed categories (e.g., Fiction, Sci-Fi). Using a tuple ensures genre data cannot be accidentally modified during program execution.

Functions and Operations
The system provides basic CRUD (Create, Read, Update, Delete) functionality, along with borrowing and returning operations:
- add_book, add_member: Ensure uniqueness and data validation.
- update_book, update_member: Allow editing details.
- delete_book, delete_member: Prevent deletion when borrowed.
- borrow_book, return_book: Manage book availability and borrowing limits.

Design Principles
The design emphasizes:
- Simplicity: Clear, readable functions that do one job.
- Modularity: Each operation is separated into its own function.
- Data Integrity: Validations ensure no duplicate or invalid data.

Conclusion
By combining dictionaries, lists, and tuples, this system balances flexibility, speed, and structure — making it efficient for small-scale library management in Python.