# **Library Management System Documentation**

This project outlines a basic **Library Management System** created using **Python**.  
The system helps manage library resources by allowing users to **add**, **borrow**, **return**, and **search** for books easily.  
It is divided into three primary Python scripts:

* **operators.py** – contains all class definitions,
* **test.py** – used for function and method testing, and
* **demo.py** – demonstrates the system’s full functionality.

This project adopts **Object-Oriented Programming (OOP)** principles to ensure a clean, modular, and maintainable design.

## **2. Objectives**

* Showcase the application of OOP principles in Python.
* Develop a practical system for monitoring book availability.
* Enable smooth borrowing and returning of books.
* Implement efficient book search and management features.
* Maintain a well-structured and easily testable codebase.

## **3. System Design**

### ****3.1 Class Structure****

#### ****a) Class: Book****

This class defines an individual book within the system.

**Attributes:**

* **title:** The name of the book.
* **author:** The person who wrote the book.
* **isbn:** A unique 13-digit identifier assigned to each book.
* **total\_copies:** The total number of copies stored in the library.
* **available\_copies:** The number of copies that can currently be borrowed.

**Methods:**

* **get\_book\_info():** Displays complete information about a book.
* **borrow\_book():** Decreases available copies when a book is borrowed.
* **return\_book():** Increases available copies when a book is returned.
* **Setters:** Used to update and validate book data (e.g., title, author, ISBN).

#### ****b) Class: Library****

This class represents the main library system and manages all books.

**Attributes:**

* **name:** The name of the library.
* **books:** A dictionary storing all book objects indexed by their ISBN numbers.

**Methods:**

* **add\_book(book):** Adds a new book to the collection.
* **remove\_book(isbn):** Deletes a book from the library using its ISBN.
* **find\_book\_by\_title(title):** Searches for a book based on its title.
* **list\_available\_books():** Displays all books currently available for borrowing.
* **borrow\_book(isbn):** Allows borrowing of a book using its ISBN.
* **return\_book(isbn):** Allows returning of a borrowed book.

## **4. Implementation**

The program is organized into three separate Python files:

* **operators.py** – Contains the Book and Library class implementations.
* **test.py** – Handles the testing of individual methods and features.
* **demo.py** – Runs the complete demonstration of system functionalities.

## **5. Code Workflow**

1. Execute the **demo.py** file.
2. Initialize book objects and a library instance.
3. Add books to the library collection.
4. Display all available books.
5. Borrow books and observe updated copies.
6. Return borrowed books to restore inventory.
7. Search for books using their titles.
8. Show the final list of all available books.

## **6. Testing and Results**

| **Test Case** | **Description** | **Expected Result** | **Status** |
| --- | --- | --- | --- |
| Add Book | Add a new book to the library | Book is added and visible in the list | ✅ Passed |
| Borrow Book | Borrow a book that is available | Available copies decrease | ✅ Passed |
| Borrow Book (no copies left) | Try borrowing when no copies remain | Displays “No copies available” | ✅ Passed |
| Return Book | Return a borrowed book | Available copies increase | ✅ Passed |
| Search Book | Search for a book by title | Displays correct book details | ✅ Passed |

## **7. Conclusion**

The **Library Management System** successfully applies OOP principles to create a simple yet effective program for managing library books.  
It supports essential operations such as adding, borrowing, returning, and searching for books.  
Future enhancements may include:

* Adding user registration and login functionality.
* Connecting to a database for data persistence.
* Developing a graphical user interface (GUI).
* Saving book records permanently for future sessions.

## **8. References**

* [Python Official Documentation](https://docs.python.org/3/)
* [W3Schools Python Tutorial](https://www.w3schools.com/python/)
* [GeeksforGeeks – OOP Concepts in Python](https://www.geeksforgeeks.org/)