

# ***Library Management System (Python)***

## **1. Project Objective :**

The objective of the Library Management System project is to develop a simple, efficient, and user-friendly program that helps in managing books within a library.

This system allows the librarian to add new books, view existing books, and maintain proper records without manual paperwork.

**Key objectives include:**

- To organize and store library book data efficiently
- To reduce manual work
- To improve accessibility of book information
- To implement Python concepts such as lists, functions, and dictionaries
- To create a beginner-friendly project demonstrating basic software development skills

## **2. Scope Requirement :**

- The scope of this project covers the essential features needed for a small library or a personal book-collection manager.
- This includes adding, storing, and displaying book records.

### **2.1 Functional Requirements**

These are the core features the system must provide:

#### **1. Add Book**

- User can enter the book title and author.
- The system stores the book with a default status "Available".

#### **2. View Books**

- Displays all books currently in the library.
- Shows title, author, and status.

### **3. Store Books**

- Books are stored in a Python list as dictionaries.
- Records remain accessible until the program exits.

## **2.2 *Non functional requirements***

These describe system quality and performance expectations:

### **1. Usability**

- Simple text-based interface
- Easy for beginners to operate

### **2. Reliability**

- Code handles empty library list safely
- No risk of system crash due to wrong input type

### **3. Efficiency**

- Uses Python's built-in data structures
- Fast operations even for large number of books

### **4. Maintainability**

- Functions are modular
- Code can be easily extended in future (e.g., search, delete, update book)

### **5. Portability**

- Works on any device with Python installed

### ***3. Technical Expectations :***

The project uses basic yet essential Python concepts:

- Python lists (to store multiple book records)
- Dictionaries (for individual book details)
- Functions (add\_book, view\_books)
- Loops (for displaying books)
- Conditional statements
- Input/Output operations
- 

### ***4. Design & Document Requirements :***

#### **System Design**

##### **1. Data Structure Design**

Library is a list:

```
library = []
```

Each book is a dictionary:

```
{"title": "", "author": "", "status": "Available"}
```

##### **2. Function Design**

```
add_book()
```

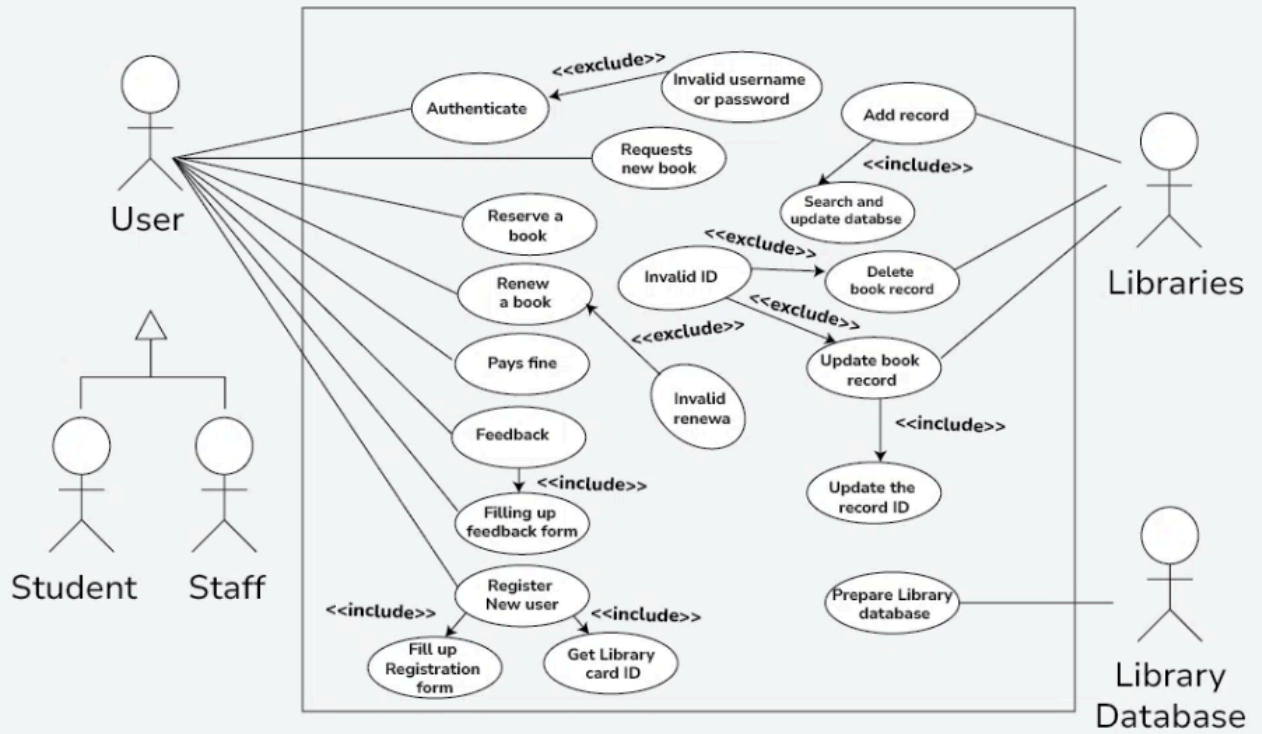
Takes title and author as input → appends to library list.

```
view_books()
```

Displays all books with serial numbers.

### ***5. Design diagram :***

## Use Case Diagram of Library Management System



### 6. Output screenshots:

```
1. Add Book
2. View Books
3. Search Book
4. Issue Book
5. Return Book
6. Exit
Choose an option: 1
Enter book title: physics
Enter author name: hc verma
Book added successfully!
```

```
1. Add Book
2. View Books
3. Search Book
4. Issue Book
5. Return Book
6. Exit
Choose an option: 2
```

```
--- Library Books ---
1. physics by hc verma - Available
```

```
1. Add Book
2. View Books
3. Search Book
4. Issue Book
5. Return Book
6. Exit
Choose an option: 3
Enter book title to search: physics
Book Found: physics by hc verma - Available
```

1. Add Book
2. View Books
3. Search Book
4. Issue Book
5. Return Book
6. Exit

Choose an option: 4

Enter book title to issue: physics

Book issued successfully!

1. Add Book
2. View Books
3. Search Book
4. Issue Book
5. Return Book
6. Exit

Choose an option: 5

Enter book title to return: physics

Book returned successfully!

1. Add Book
2. View Books
3. Search Book
4. Issue Book
5. Return Book
6. Exit

Choose an option: 6

Exiting program...