LIBRARY MANAGEMENT SYSTEM DOCUMENTATION

1. Problem Statement

Managing a library involves keeping track of books, their availability, borrowers, and the overall status of the library's inventory. The problem arises from the need to manage books efficiently, allow users to borrow and return books, and search for books based on specific criteria. Existing systems may lack flexibility and ease of use, which hinders the user experience.

2. Requirements

- **A system with python installed**
- **Visual Studio Code**
- **Python Modules: os, json**

3. Proposed Solution

The solution is a Python-based Library Management System that fulfills the requirements outlined above. This solution uses JSON files to persist data, and the operations include adding books, viewing available books, borrowing and returning books, and searching for books.

4. Features

5. **Add Book**:

   Add new books with a title, author, and quantity. If the book already exists, the quantity is updated.

6. **View Books**:

   View a list of all books with their availability status (available or borrowed).

7. **Borrow Book**:

   Users can borrow books if available, and the system keeps track of who borrowed the book.

8. **Return Book**:

Users can return borrowed books, and the system updates the quantity and status accordingly.

9. **Search Book**:

   Search for books by title or part of the title. The search is case-insensitive.

5. Future Scope

1. **Enhanced Search**

   The search functionality can be expanded to support partial matches for author names and more advanced filtering based on availability.

2. **User Management**

   Adding user authentication for tracking borrowed books per user and implementing user roles

3. **Reservation System**

   Implementing a feature where users can reserve books that are currently borrowed and be notified when they become available.

4. **Overdue Book Tracking**

   Adding due dates and overdue tracking to manage returned books and fines.

6. Conclusion

The Library Management System successfully meets the requirements of a simple book management application. By using a text-based interface and storing data in a persistent JSON file, the system ensures that users can interact with the library effectively. The system is expandable, allowing for future features and improvements as needed.

This Library Management System implementation covers basic library management features such as book addition, borrowing, returning, and searching, and can serve as a foundation for more advanced systems.

7. Code

```
import json
import os

Library_File = 'library_data.json'

def load_library():
    if not os.path.exists(Library_File):
        return {}
    try:
        with open(Library_File, 'r') as file:
            return json.load(file)
    except Exception as e:
        print(f"Error loading library data: {e}")
        return {}

def save_library(library):
    try:
        with open(Library_File, 'w') as file:
            json.dump(library, file)
    except Exception as e:
        print(f"Error saving library data: {e}")

def add_book(library, title, author, quantity):
    while True:
        if title in library:
            print("Book already exists. Updating the quantity.")
            library[title]['quantity'] += quantity
        else:
            library[title] = {'author': author, 'quantity': quantity, 'borrowed_by': None}
        save_library(library)
        print(f"Book '{title}' by {author} added successfully!")

        add_more = input("Do you want to add more books? (Y/N): ").upper()
        if add_more == 'Y':
            title = input("Enter book title: ").title()  # Use title() for consistency
            author = input("Enter author name: ").title()  # Use title() for consistency
            try:
                quantity = int(input("Enter the quantity of books: "))
            except ValueError:
                print("Invalid input for quantity. Please enter an integer.")
                break
        elif add_more == 'N':
            break
        else:
            print("Invalid choice! Exiting the loop.")
            break

def view_books(library):
    if not library:
        print("The library is empty.")
        return
    for title, details in library.items():
        status = f"Available ({details['quantity']})" if not details['borrowed_by'] else f"Borrowed by {details['borrowed_by']}"
        print(f"Title: {title}, Author: {details['author']}, Status: {status}")

def borrow_book(library, title, borrower_name):
    if title not in library:
        print("Book not found in the library.")
```

```
            return
        if library[title]['quantity'] == 0:
            print(f"All copies of {title} are currently borrowed.")
            return
        if library[title]['borrowed_by']:
            print(f"The book '{title}' is currently borrowed by {library[title]['borrowed_by']}.")
            return
        library[title]['quantity'] -= 1
        library[title]['borrowed_by'] = borrower_name
        save_library(library)
        print(f"'{title}' has been borrowed by {borrower_name}.")

    def return_book(library, title):
        if title not in library:
            print("Book not found in the library.")
            return
        if not library[title]['borrowed_by']:
            print(f"The book '{title}' was not borrowed.")
            return
        borrower_name = library[title]['borrowed_by']
        library[title]['quantity'] += 1
        library[title]['borrowed_by'] = None
        save_library(library)
        print(f"'{title}' has been returned by {borrower_name}.")

    def search_book(library, search):
        search = search.title()  # Standardize search term
        found = {title: details for title, details in library.items() if search in title.title()}  # Use title() for matching

        if not found:
            print(f"No books found with the title containing '{search}'.")
        else:
            for title, details in found.items():
                status = f"Available ({details['quantity']})" if not details['borrowed_by'] else f"Borrowed by {details['borrowed_by']}"
                print(f"Title: {title}, Author: {details['author']}, Status: {status}")

    def main():
        library = load_library()
        while True:
            print("\nLibrary Management System")
            print("1. Add Book")
            print("2. View Books")
            print("3. Borrow Book")
            print("4. Return Book")
            print("5. Search Book")
            print("6. Exit")

            choice = input("Enter your choice: ")

            if choice == "1":
                title = input("Enter book title: ").title()
                author = input("Enter author name: ").title()
                try:
                    quantity = int(input("Enter the quantity of books: "))
                    add_book(library, title, author, quantity)
                except ValueError:
                    print("Invalid input for quantity. Please enter an integer.")

            elif choice == "2":
                view_books(library)
```

```
        elif choice == "3":
            title = input("Enter the book title: ").title()
            borrower_name = input("Enter your name: ").title()
            borrow_book(library, title, borrower_name)

        elif choice == "4":
            title = input("Enter the book title: ").title()
            return_book(library, title)

        elif choice == "5":
            search = input("Enter the title or part of the title to search: ").title()
            search_book(library, search)

        elif choice == "6":
            print("Exiting the system.")
            break

        else:
            print("Invalid choice!")

if __name__ == "__main__":
    main()
```

Output

```
Library Management System
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Search Book
6. Exit
Enter your choice: 2
The library is empty.

Library Management System
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Search Book
6. Exit
Enter your choice: 1
Enter book title: Harry Potter
Enter author name: James
Enter the quantity of books: 5
Book 'Harry Potter' by James  added successfully!
Do you want to add more books? (Y/N): y
Enter book title: Atomic Habits
Enter author name: 5
Enter the quantity of books: 2
Book 'Atomic Habits' by 5 added successfully!
Do you want to add more books? (Y/N): n
```

```
Library Management System
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Search Book
6. Exit
Enter your choice: 2
Title: Harry Potter, Author: James , Status: Available (5)
Title: Atomic Habits, Author: 5, Status: Available (2)

Library Management System
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Search Book
6. Exit
Enter your choice: 3
Enter the book title: Harry Potter
Enter your name: Rineesh
'Harry Potter' has been borrowed by Rineesh.
```

```
Library Management System
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Search Book
6. Exit
Enter your choice: 4
Enter the book title: Harry Potter
'Harry Potter' has been returned by Rineesh.

Library Management System
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Search Book
6. Exit
Enter your choice:
```