Library Management System — Project Documentation

1. Project Title

Library Management System
Saylani Assignment — Python Project

---

2. Project Overview

The Library Management System is a **Python-based application** designed to simulate the core functionality of a library:
✓ Manage books (add, remove, view)
✓ Manage users (admin, user roles)
✓ Issue and return books
✓ Store persistent data in JSON files
✓ Simple menu or interface for interaction

This project uses Python's standard libraries and **flat-file storage (JSON)** instead of a database, making it simple yet effective.

---

3. Purpose

The goal of this project is to create a digital system to:

- Replace manual logbooks used in libraries
- Track book inventory (stock, available copies)
- Track user accounts
- Allow issuing and returning of books
- Enforce admin and user roles

---

4. Technologies Used

Technology	Purpose
Python	Main programming language
JSON	Data storage for books and user accounts
Standard Library (`json`)	JSON file handling

---

5. File Structure

```
Library-Management-System--Saylani-Assignment/
```

```
├── admin.py        # Admin logic
├── user.py         # User logic
├── main.py         # Main application entry point
├── data.json       # Books data
├── user.json       # User accounts
```

---

6. Functional Modules

6.1 Admin Module (`admin.py`)

Admin features typically include:

- Login authentication for admin
- Add new books
- Edit or delete book data
- View all books
- Manage users

(Actual methods depend on code logic in `admin.py`)

6.2 User Module (`user.py`)

User functions typically include:

- User login
- View available books
- Issue a book
- Return a book
- Search books

6.3 Main Application (`main.py`)

- Starts the program
- Shows menu UI (possibly via CLI or Streamlit)
- Handles user input for different actions
- Calls admin or user methods based on role

---

7. How the System Works Internally

Login Logic

- Prompt for username/password
- Check against stored credentials
- If valid → set session state
- If invalid → error message

JSON Handling

Python `json.load()` and `json.dump()` are used to:

- Read data from JSON
- Update the list (add / edit / remove)
- Save back with indentation (`indent=4` for readability)

---

8. Limitations & Future Enhancements

Current Limitations

- Flat-file storage → not scalable
- No search filtering (partial matches)
- No UI (pure text/Streamlit buttons)
- No role permissions beyond admin/user

Future Possibilities

✓ Switch to database (SQLite/MySQL)
✓ Add search filters (author, title)
✓ Add due dates and fines
✓ Add GUI web interface (Django/Flask)
✓ Add reports and charts

---

9. Conclusion

This Library Management System is a practical Python project demonstrating:
✓ File handling with JSON
✓ User and admin workflows
✓ Data creation, reading, updating, and deletion

It is suitable as an assignment or a learning project for basic full-stack application design.