

# Library Management System Using Python

SOFTWARE ENGINEERING LAB (PCCCS594)

Project Supervision	SL No.	Name	Enrollment No.
Prof. Subhabhrata Sengupta	1.	Khandakar Nafees Hossain	12023052004014
Prof. Dr. Rupayan Das	2.	Dhritiman Bera	12023052004006
	3.	Parthib Mahapatra	12023052004019



## Project Summary

The Library Management System is a Python- and Streamlit-based web application designed to automate the management of books, users, and book transactions in an educational institution or public library. It offers role-based authentication, allowing librarians to add, delete, issue, and return books, while students can view, request, and manage issued books. The system includes AI-powered book recommendations, digital bookshelves, due date tracking, fine alerts, and a clean interactive interface.










## Objective

- To build a feature-rich, user-friendly library system that manages users and books digitally.
- To support librarians in managing inventory, issuing books, and tracking return dates.
- To offer students a personalized interface with recommendations and self-service options.
- To digitize the library ecosystem using open-source, lightweight, and easily deployable technology.

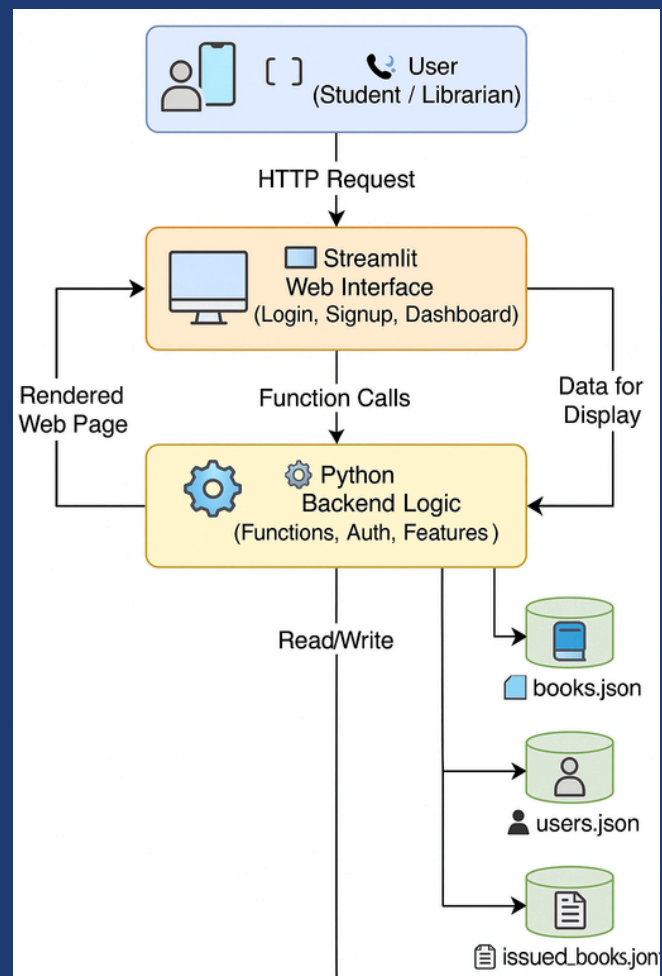
## Scope of the Project

- Role-based access: Librarian and Student.
- Sign-up/login system with secure authentication.
- View, add, delete, issue, and return books with status tracking.
- Digital bookshelf with cover images and descriptions.
- Due date tracking with fine calculation.
- AI-based book recommendation based on category.
- Data storage using JSON (no external DB required).
- Designed for deployment on platforms like Streamlit and GitHub.

## Problem Statement

-  Manual library systems are time-consuming and prone to human error.
-  Lack of secure access for different user roles (e.g., Students vs Librarians).
-  No structured way to track issued or returned books with due dates.
-  Students cannot view detailed book descriptions or preview pages before borrowing.
-  No automated reminders or countdowns for book return deadlines.
-  Lack of book recommendation based on past borrowings reduces user engagement.
-  Book data and user records are often not digitized, making it hard to analyze usage trends.

## System Architecture:



## Project Timeline

Phase	Start Date	End Date
Planning	18/07/2025	24/07/2025
Model Development	25/07/2025	19/08/2025
Backend	08/08/2025	26/08/2025
Frontend	25/07/2025	07/08/2025
Testing	29/08/2025	10/09/2025

# Technology Stack



# Deliverables

- Complete Python code with Streamlit UI
- JSON book database with cover image URLs and descriptions
- User data storage (with login credentials and personal lists)
- Book issuing, return, and tracking system
- AI-based book recommendation module
- PDF user manual and admin guide
- Live hosted application (Streamlit cloud)

# Risk & Mitigations

Risk	Mitigation Strategy
<b>Unauthorized access to admin (librarian) features</b>	Implement strict role-based access control and verify roles during login
<b>Loss or corruption of book data (JSON file)</b>	Maintain regular JSON backups and implement data validation before read/write operations
<b>Data privacy and user credential leaks</b>	Use encrypted password storage (e.g., hashing with salt) and secure login validation
<b>System crashes or unavailability during peak usage</b>	Host on scalable platforms like Streamlit Cloud or Render with proper performance
<b>Users forget to return books or miss due dates</b>	Implement a live countdown system with return deadlines and optional email alerts
<b>Incorrect or inconsistent book information</b>	Restrict book addition/editing to librarian role; use form validation
<b>Recommendation system gives irrelevant suggestions</b>	Start with rule-based logic; improve over time using user interaction analytics
<b>Incompatibility on mobile devices</b>	Use responsive Streamlit layout and test across devices

# Technical Feasibility

Tools Used: Python, Streamlit, Pandas, JSON, GitHub.

Platform Compatibility: Works on any device with a web browser.

Libraries: Streamlit for UI, Pandas for data handling.

Conclusion: Highly feasible due to minimal dependencies, ease of development, and strong open-source support.

# Economic Feasibility

Development Cost: No licensing cost — Python, Streamlit, and other tools are free.

Maintenance Cost: Low; only requires occasional data updates or feature enhancements.

Hosting: Can be deployed on free platforms (Streamlit Cloud, GitHub Pages).

Conclusion: Economically feasible with near-zero operational costs.

# Operational Feasibility

User Friendly: Intuitive UI with tabs, forms, and interactive elements.

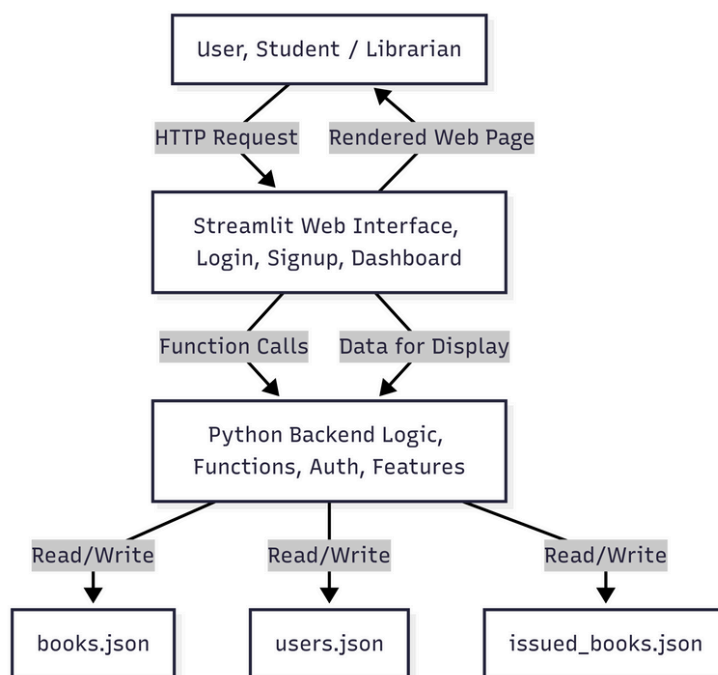
Role-based Access: Ensures streamlined workflows for students and librarians.

Automation: Reduces manual tracking, paperwork, and errors.

Training: Minimal training required due to simple and guided UI.

Conclusion: Highly operable in real-world environments like schools and colleges.

# System Architecture



# Team Members Role

Name	Role
Khandakar Nafees Hossain	Project Lead & Developer
Dhritiman Bera	Developer & Designer
Parthib Mahapatra	Tester & Deployment



## Sign of Supervisor

Prof. Subhabrata Sengupta

Prof. Dr. Rupyan Das

## Team Members

Khandakar Nafees Hossain

Dhritiman Bera

Parthib Mahapatra

