Library Management System Using Python

SOFTWARE REQUIREMENT SPECIFICATIONS (SRS)

Team Members:
Khandakar Nafees Hossain (12023052004014)
Dhritiman Bera (12023052004006)
Parthib Mahapatra (12023052004014)

1. Introduction

1.1 Purpose

This Library Management System (LMS) aims to transform traditional library operations into a digital format streamlining the management of library resources using a user-friendly, web-based application developed in Python and Streamlit. The system will support role-based functionalities for **Librarians**, **Students**, and **Other Users**, including user authentication, book management, tracking of issued/returned books, and personalized book recommendations.

1.2 Scope

The LMS provides an intuitive interface for users to interact with digital library resources. Key features include user registration and authentication, role-based access to functions (adding/deleting books), viewing book catalogue with cover images and descriptions, book issuing/returning with due-date tracking, personal book list management, and AI-based book recommendations. The system uses **JSON** for book storage and is deployable on **GitHub** via **Streamlit**. To provide web deployment for real-time access.

1.3 Definitions, Acronyms, and Abbreviations

- **LMS** – Library Management System
- **JSON** – JavaScript Object Notation
- **UI** – User Interface
- **AI** – Artificial Intelligence
- **OTP** – One-Time Password
- **DB** – Database

2. Overall Description

2.1 Product Perspective

This LMS is a standalone web application with a JSON-based backend and a Streamlit-powered frontend. It provides a modern interface and customizable workflows tailored to librarians and students.

2.2 Product Functions

- User sign-up and login with role selection (Student/Librarian/Others)
- View all books with cover images, index, description, and availability
- Librarians can add or delete books
- Users can add books to their personal "To Read" list
- Books can be issued and tracked with due dates
- Books can be returned and marked available again
- Book recommendation based on past user interactions
- Overview of issued books and due date alerts

2.3 User Characteristics

- **Librarians:** Can manage book inventory
- **Students:** Can view, issue, return, and maintain a personal reading list
- **Others:** Similar to students but may have restricted access
- All users need basic digital literacy

2.4 Operating Environment

- Web browser (Chrome, Firefox, Edge)
- Streamlit front-end hosted on GitHub or Streamlit Community Cloud
- Python environment for backend logic
- JSON for book data
- Local or cloud storage for user sessions and logs

2.5 Design and Implementation Constraints

- Frontend: Streamlit (Python-based)
- Storage: JSON file for books, optional local storage or Firebase for users
- Backend: Python libraries (Pandas, datetime)
- Hosting: GitHub Pages, Streamlit Cloud
- Secure handling of user data (email, passwords)

2.6 User Documentation

- **User Manual:** Instructions on registration, viewing/issuing/returning books
- **Admin Guide:** How to add/remove books, monitor issued books
- All documentation will be made available via PDF and embedded help sections

2.7 Assumptions and Dependencies

- Users have access to a stable internet connection
- The Streamlit service remains publicly accessible
- User credentials are securely managed
- Admin users are pre-assigned or approved by a supervisor

3. Specific Requirements

3.1 Functional Requirements

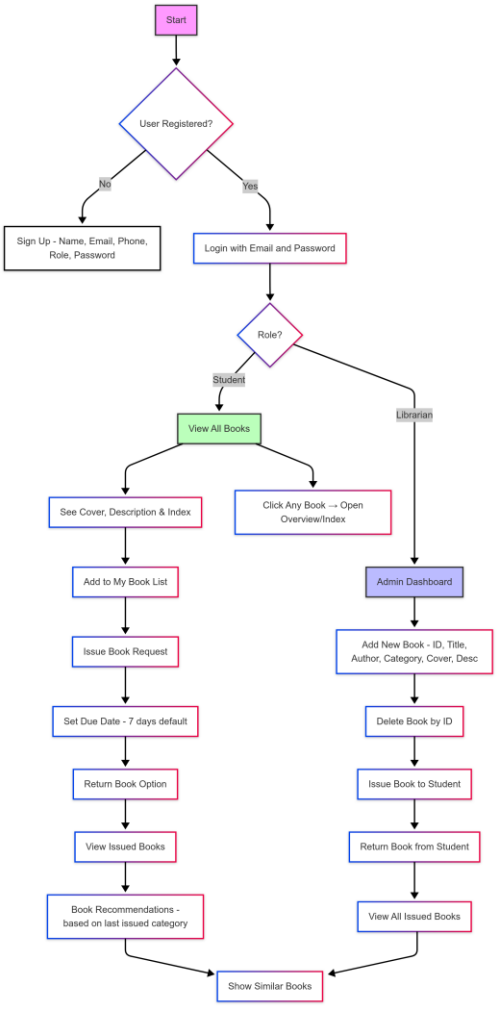
ID	Functional Requirement
FR1	Users can register using name, mobile number, email, and password and select a role
FR2	Users can log in with correct credentials
FR3	View all books with details like ID, title, author, availability, cover page, and description
FR4	Only librarians can add new books
FR5	Students and others can add books to a personal book list
FR6	Issue book with due date tracking (default: 14 days)
FR7	Return issued books; update availability status
FR8	View all issued books and their return deadlines
FR9	Only librarians can delete books
FR10	View index page and overview for each book by clicking on it
FR11	Recommend books based on user history or last borrowed book category

ID	Functional Requirement
FR12	Book data is stored and retrieved using a JSON file format

3.2 Non-Functional Requirements

ID	Non-Functional Requirement
NFR1	The system should support 100+ concurrent users
NFR2	All operations should complete in under 3 seconds
NFR3	Responsive UI on desktop and mobile devices
NFR4	System availability of 99.5%
NFR5	User data should be encrypted where applicable

4. Tech Stack



Component	Technology
Frontend	Streamlit (Python)
Backend	Python (Pandas, datetime, JSON)
Database	JSON files
Authentication	Custom password-based login (OTP optional)
Deployment	GitHub + Streamlit Community Cloud

5. Risks and Mitigations

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

5. 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)

6. Sustainability and SDGs

This project aligns with the following United Nations Sustainable Development Goals:

Goal	Contribution
SDG 4: Quality Education	Enhances access to learning materials and promotes digital learning.
SDG 9: Industry, Innovation & Infrastructure	Encourages innovation through digital infrastructure in education.
SDG 12: Responsible Consumption & Production	Promotes digital reading and reduces paper-based resource dependency.

8. Team Members Roles

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

7. Sign-off and Approval

Name	Signature	Date
Mr. Subhabrata Sengupta		
Dr. Rupayan Das		
Khandakar Nafees Hossain		
Dhritiman Bera		
Parthib Mahapatra		