**SRS DOCUMENT**

1. **Project Abstract:**

The **Library Management System (LMS)** is a web-based platform designed to modernize and streamline library operations, enhancing accessibility and efficiency for students, faculty, and administrators. The system offers seamless book search functionality, enabling users to find books either through course-based recommendations ranked by professor reviews or via direct searches with AI-powered suggestions for related books. Users can track borrowed books, receive automated due-date reminders, pay overdue fines online, and provide feedback on returned books. Additionally, the system facilitates book reservations, ensuring structured access through a queue-based allocation with email notifications.

Faculty members benefit from additional privileges, including the ability to add books and research papers, label books, and provide expert reviews. Meanwhile, administrators gain complete control over database management, book approvals, user request handling, and automated report generation for efficient decision-making.

The LMS is developed using **React, Bootstrap, Node, and MongoDb**, ensuring a scalable, user-friendly, and cost-effective solution tailored for educational institutions and research libraries. With role-based authentication and structured access control, the system enhances security while minimizing manual workload. By integrating AI-driven recommendations, automated notifications, and digital transactions, the LMS significantly optimizes resource management, fostering a technology-driven and efficient library ecosystem.

1. **Document Revision History:**

|  |  |  |
| --- | --- | --- |
| Version No. | Date | Key Features |
| 1.0 | 16-03-2025 | * User features limited. * Possible Actions by students – search (direct matching), reserve, renew, paying fine(offline), notifications when book becomes available. * Possible actions by faculty – similar to students + add a book, providing reviews. * Admin actions- as in current version. |
| 2.0 | 24-03-2025 | * User features extended – paying fine online, search strategy improved, AI based recommendations included. |
| 2.1 | 25-03-2025 | * Feature to add a research paper by faculty or admin and student able to download it and transfer to their system via email. |

1. **Customer:**

The **Library Management System (LMS)** is designed to cater to three primary customer groups: students (general users), faculty members, and administrators. Each group has unique requirements that shape the system’s functionality.

**Customer A: Student (General User Perspective)**

Students require an intuitive and feature-rich portal to efficiently manage their library interactions. Key expectations include:

* **Secure authentication** with a personalized user portal.
* **Borrowed book overview** displaying due dates, with near-due books listed at the top.
* **Online fine payment system** for overdue book returns.
* **Smart search functionality** that not only retrieves the searched book but also suggests related books using AI.
* **Course-based book recommendations**, ranked by professor ratings and reviews.
* **Automated availability notifications** when a reserved book becomes available.
* **Digital research paper access**, including an option to download and email them to personal devices.
* **Book location assistance**, helping users find books within the library through an online portal.

**Customer B: Faculty (Professor Perspective)**

Professors play a crucial role in curating reading materials for students and contributing academic resources. Their key requirements include:

* **A dynamic course book list**, allowing them to add, modify, and prioritize recommended books throughout the semester.
* **Tagging functionality** to classify books based on subject relevance and priority.
* **A research paper management system**, enabling them to upload, categorize, and organize research papers for students.
* **Book review and feedback submission**, allowing them to share insights on the quality and relevance of books and research papers.

**Customer C: Administrator (Library Management Perspective)**

Administrators oversee the overall operation of the library and require robust management tools. Their key needs include:

* **Automated report generation**, providing insights into library usage, book circulation, and overdue statistics.
* **A structured request approval system**, prioritizing faculty requests over student requests for book additions.
* **Data visualization tools**, such as graphs and charts, to analyse library trends and optimize resource allocation.

1. **Competitive Landscape in Library Management Systems (LMS)**

Library Management Systems (LMS) have evolved to improve accessibility and efficiency in academic institutions. Traditional LMS solutions, such as Koha, Evergreen, and Ex Libris Alma, focus on core functionalities like book cataloging, borrowing, and inventory management. More advanced platforms now integrate AI-driven recommendations, cloud-based access, and mobile compatibility.

While these established LMS platforms offer extensive features, they often lack customization for specific institutional needs. A key differentiator of our system is the **search by course** feature, allowing students to quickly find academic resources relevant to their coursework, reducing the time spent browsing large catalogs.

Additionally, our LMS enhances automation by incorporating real-time book availability updates, automated overdue notifications, and personalized recommendations. Unlike many existing systems that require manual updates, our solution streamlines operations, reducing administrative workload and improving user experience.

**Competitive Differentiators and Barriers**

Our LMS stands out due to its high level of adaptability and automation. Features like **customized search filters, real-time tracking, and smart resource recommendations** create a competitive advantage. Moreover, the integration of **cloud-based accessibility and AI-powered analytics** helps optimize library resource utilization.

Competitive barriers include the **adoption challenge** faced by institutions with legacy systems and the need for initial training. However, our user-friendly interface and comprehensive documentation mitigate these concerns, ensuring a smoother transition.

**Patent Considerations**

While general LMS functionalities are widely available, unique features such as our **search by course system, AI-driven recommendations, and automated tracking mechanisms** may have potential for patent protection. A patent landscape analysis would be necessary to determine existing intellectual property rights in this area and assess the feasibility of obtaining a unique patent position.

By addressing gaps in customization, automation, and competitive barriers, our LMS provides a more user-centric and efficient solution compared to traditional platforms. These improvements make our system highly competitive while setting a new standard for adaptability in academic libraries.

1. **System Requirements:**

**5.1 Functional Requirements**

The Library Management System (LMS) must support the following functional requirements:

**User Management**

* The system must store information about users (students, faculty, librarians, and administrators), including login credentials, personal details, and book transaction history.
* The system must allow the admin to add, remove, and manage users.
* Role-based authentication and authorization must be enforced to ensure proper access control.
* Users must be able to update personal details but cannot modify system data.

**Book Management**

* The system must store details of books, including title, author, publisher, edition, ISBN, availability status, and location.
* Books should be categorized by subjects, genres, and courses recommended by faculty.
* Admins must be able to add, remove, and update book records.
* A real-time inventory of book availability must be maintained.

**Book Issuing, Returning, and Reservations**

* Users can borrow books, subject to borrowing limits (e.g., students: max 4 books, faculty: max 6 books).
* The system must track issue dates, due dates, and overdue fines.
* Late returns must be subject to fines, payable online.
* Users can reserve unavailable books and receive notifications when they become available.
* Reserved books must be collected within a set timeframe, failing which the reservation is cancelled.

**Book Search and Recommendations**

* Users must be able to search for books using title, author, course, or keywords.
* If searching by course, books recommended by faculty should be displayed first, sorted by professor ratings.
* If searching by book name, related books should be suggested using an AI-based recommendation system.

**Feedback and Reviews**

* Faculty members can provide feedback on books, which will be displayed to students.
* Users can rate and review books after returning them.

**Research Paper Repository**

* Faculty and students can upload research papers to a dedicated repository.
* Research papers can be searched, downloaded, and transferred via email.

**Admin Functions**

* Admins can approve books added by faculty, manage user requests, and generate system reports.
* Reports can include overdue books, fines collected, most borrowed books, etc.
* The admin can update rules regarding borrowing limits, fines, and reservation policies.

**Notifications and Alerts**

* Users should receive email notifications for due dates, fines, book availability, and important announcements.
* Users in the reservation queue should be notified when a book becomes available.

**5.2 Use Case Specification**

Each major feature of the LMS is represented as a use case.

**Use Case 1: User Login**

**Actors:** Student, Faculty, Librarian, Admin  
**Preconditions:** User must be registered  
**Description:** User enters credentials; system verifies them and grants access based on the user’s role.  
**Acceptance Criteria:**

* Successful login with valid credentials.
* Error message for invalid credentials.

**Use Case 2: Book Search**

**Actors:** Student, Faculty, Librarian  
**Preconditions:** Books must be recorded in the system  
**Description:** Users search for books using different filters (title, author, course, etc.).  
**Acceptance Criteria:**

* Matching results displayed in relevance order.
* If no books match, a message is displayed.

**Use Case 3: Book Issuing**

**Actors:** Librarian  
**Preconditions:** Book must be available  
**Description:** Librarian assigns books to users, updating availability status.  
**Acceptance Criteria:**

* Issued books are assigned to the user.
* System prevents issuing if a book is unavailable.

**Use Case 4: Book Return**

**Actors:** Student, Faculty, Librarian  
**Preconditions:** Book must be issued to the user  
**Description:** User returns the book; system updates status and calculates fines if overdue.  
**Acceptance Criteria:**

* Books are marked as returned.
* Fines are recorded for overdue returns.

**Use Case 5: Book Reservation**

**Actors:** Student, Faculty  
**Preconditions:** Book must be unavailable  
**Description:** User reserves the book; system queues them and notifies when the book is available.  
**Acceptance Criteria:**

* Reservation is successfully recorded.
* Users are notified when the book is ready for pickup.

**5.3 Non-Functional Requirements**

**Performance and Speed**

* System must support at least 100 concurrent users without performance degradation.
* Login response time should not exceed 2 seconds.
* Book search results should load in under 5 seconds.

**Security and Privacy**

* Role-based access control must be implemented.
* User passwords must be securely encrypted.
* Only authorized users can modify critical system data.
* Payment transactions for fines should be securely processed.

**Memory and Storage Requirements**

* System must handle at least 1000 book records.
* User transaction logs should be retained for at least 5 years.
* Regular backups should be implemented to prevent data loss.

**Availability and Reliability**

* LMS must be available 24/7 with minimal downtime.
* Automatic backups should be scheduled to avoid data loss.
* A failover mechanism should be implemented for system recovery.

**Scalability**

* System should support increasing user traffic and book records.
* Architecture should allow future enhancements like AI-based book recommendations.

**Error Handling**

* System must handle expected and unexpected errors gracefully.
* Error logs should be maintained for debugging purposes.

**5.4 Software and Hardware Requirements**

**Software Requirements:**

* **Frontend:** React.js, Bootstrap, HTML, CSS, JavaScript
* **Backend:** Node.js, Express.js
* **Database:** MongoDB
* **Authentication:** OAuth authentication
* **Payment Gateway:** Stripe (for fine payments)
* **Operating System:** Windows 10, Linux, or macOS
* **Cloud Storage:** AWS S3 for research papers

**Hardware Requirements:**

* **Processor:** Intel Core i5 or higher
* **RAM:** Minimum 8GB recommended for smooth operation
* **Hard Disk:** Minimum 100GB storage space
* **Server Requirements:** Cloud-based deployment on AWS/Azure with auto-scaling support