Software Requirements Specification (SRS)

1. Introduction

1.1 Project Name:

Library Management System (LMS)

1.2 Purpose:

The LMS is a full-stack web-based system that automates book management, borrowing, and user operations. It ensures efficient tracking of library inventory and streamlines user experience through advanced filtering and borrowing history tracking.

1.3 Technologies Used:

- **Backend:** Java Spring Boot (RESTful API)
- **Frontend:** React.js (Component-based UI)
- **Database:** MySQL (Relational Data Management)
- **Authentication:** JWT (JSON Web Token) with Spring Security
- **UI Framework:** Bootstrap/Material-UI

2. System Features

Feature	Description
User Management	Users can register, login, reset passwords, and manage profiles.
Book Management	Admins can add, edit, delete books, and manage book availability.
Borrowing System	Users can borrow and return books, and track due dates.
Search Feature	Advanced search with filters (title, author, category, publication year).
Overdue Notifications	Scheduled tasks notify users of overdue books and calculate fines.
Role-Based Access	Admins manage the system; members borrow and search books.

3. System Architecture

The system follows the **MVC (Model-View-Controller)** design pattern:

- **Model** → Database (MySQL) and ORM using JPA/Hibernate.
- **View** → React.js frontend for user interaction.
- **Controller** → Spring Boot RESTful API for business logic.

4. Database Design

Table Name	Description
users	Stores user details (name, email, password, role, status).
books	Stores book details (title, author, category, publication year, availability).
borrowed_books	Tracks borrowed books, return dates, and overdue fines.

5. Security Requirements

- All passwords must be encrypted using **BCrypt hashing**.
- JWT authentication will be used for user sessions.
- **Role-based access control** will restrict system functions (Librarian vs. Member).
- Input validation and SQL injection prevention mechanisms will be implemented.

6. Performance Requirements

- The system should handle **up to 5,000 users** and **100,000 books** efficiently.
- The response time for book searches should be **<500ms**.
- APIs should handle at least **100 concurrent requests** without failure.

7. Out of Scope

The following features are beyond the scope of this project:

- Mobile application development (iOS/Android).
- Integration with external APIs such as Google Books.
- Online payment processing for fines.
- Multi-language support.
- AI-driven book recommendations.
- Advanced data analytics and trend reporting.

8. Testing & Validation

Testing Type	**Description**
Unit Testing	Individual components such as authentication, book management, and search functionalities will be tested separately.

Integration Testing	Ensuring the backend and frontend communicate effectively through API calls.
User Acceptance Testing (UAT)	End-users will test the system to ensure it meets functional requirements.
Performance Testing	Load testing to ensure response times meet the specified performance criteria.

9. Deployment Strategy

- The system will be **deployed on a cloud server** with database hosting.
- A **CI/CD pipeline** will be implemented for automated deployment.
- Versioning control will be managed via **GitHub**.

10. Maintenance & Future Enhancements

- Regular security patches and system updates.
- Enhancements such as **mobile app integration, AI-based recommendations, and real-time reporting** can be considered in future versions.