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 library operations including book management, borrowing, returning and reservation. It improves efficiency for librarians and offers a user-friendly interface for members to manage their interactions with the library.

1.3 Technologies Used:
- Backend: Java Spring Boot (RESTful API)
- Frontend: React.js
- Database: MySQL
- Authentication: JWT with Spring Security
- UI Framework: Bootstrap or Material-UI

2. System Features

- User Management: Users can register, login and reset passwords.
- Role-Based Access: Admins manage the system; members can borrow and search books.
- Book Management: Admins can add, edit, delete, and view books, and manage availability.
- Borrowing System: Users can borrow books.
- Reservation System: Users can reserve books and cancel reservations.
- Fine Enforcement: Users cannot borrow books if he has reached the limit of 3 borrowed books.
- Search Feature: Basic search by title, author, ISBN, category.
- Borrowing History: Admins can view history of borrowed and returned books.

3. System Architecture

The system follows the MVC (Model-View-Controller) design pattern:
- Model: Database and ORM layer using JPA/Hibernate
- View: React.js for user interface
- Controller: RESTful APIs developed with Spring Boot

4. Database Design

- users: Stores user details (id, name, email, password, role, status)
- books: Stores book details (id, title, author, category, year, status)
- borrow_records: Tracks borrow/return dates
- reservations: Tracks book reservations and cancellation status

5. Security Requirements

- All passwords are encrypted using BCrypt.
- JWT is used for session authentication.
- Role-based access control enforced through Spring Security.
- Input validation and SQL injection prevention implemented.

6. Performance Requirements

- System supports up to 5,000 users and 100,000 books.
- Book search response time < 500ms.
- API can handle at least 100 concurrent requests.

7. Out of Scope

- Mobile application development.
- External API integration (e.g., Google Books).
- Online fine payment system.
- Multi-language support.
- AI-based recommendation features.

8. Testing & Validation

- Unit Testing: Covers authentication, borrowing, and book logic.
- Integration Testing: Verifies API interaction with frontend.
- User Acceptance Testing: Validates full feature flow.
- Performance Testing: Simulates load and measures response.

9. Deployment Strategy

Application is deployed to a cloud server with MySQL hosting.
CI/CD pipeline handles automated build and deployment.
GitHub is used for version control.

10. Maintenance & Future Enhancements

- Regular security patches and updates.

- Potential mobile app version.
- Addition of analytics dashboard for librarians.
- Integration with notification systems (email/SMS).