**Software Requirements Specification (SRS)**

1. **Introduction**

This document outlines the functional and non-functional requirements for the Library Management System. The system will allow users to search for books, borrow and return them, and receive overdue notifications. Administrators will be able to manage the book inventory. The system will be developed using React for the frontend, Spring Boot for the backend, and MySQL for data storage.

1. **Functional Requirements**
   1. **User Registration and Authentication**
2. Users can register with a username, email, and password.
3. Registered users can log in to access the system.
4. Users can reset their password via UI (**intermediate**).
5. The system provides role-based access (Librarian vs. Member) (**intermediate**)
   1. **Book Search**
6. Users can search for books by title, author, or category.
7. Advanced filters (e.g., category, publication year) are available. (**intermediate**)
8. The system displays a list of matching books.
   1. **Borrowing / Returning Books**
9. Users can borrow available books.
10. Users can return borrowed books.
11. The system updates the book’s status accordingly (available, borrowed, reserved).
12. Users can view their borrowing history. (**intermediate**)
    1. **Admin Functions**
13. Administrators can add new books.
14. Administrators can update or delete existing books. (**intermediate**)
15. The system tracks book status (available, borrowed, reserved).
16. Administrators can search and filter books. (**intermediate**)
    1. **Overdue Notifications and Fine Calculation**
17. The system sends overdue notifications for borrowed books.
18. Scheduled tasks check for overdue books (**Intermediate**).
19. The system calculates fines based on overdue duration (**Intermediate**)
    * $0.50 per day for overdue books
    * Fine starts accumulating from day after due date
    * Maximum fine per book: $20
    1. **Testing and Documentation**
20. The system undergoes unit and integration testing.
21. Write comprehensive documentation including BRD, SRS, API documentation, and a README guide.
22. **Non-Functional Requirements**
    1. **Performance Requirements**
23. The system should handle up to 100 simultaneous users without performance degradation.
24. Search queries should return results within 3 seconds.
    1. **Security Requirements**
25. User passwords must be encrypted using a secure hashing algorithm.
26. Role-based access control (Librarian vs. Member) should restrict functionalities based on user roles (**intermediate**).
27. Input validation and sanitization should be implemented to prevent security vulnerabilities.
    1. **Usability Requirements**
28. The system should have a user-friendly interface for both members and administrators.
29. Navigation should be intuitive and responsive.