Requirements Specification

Table of Contents

[1. System Overview 2](#_r7jy1ljk8hs4)

[2. Functional Requirements 2](#_msbnsxd002fs)

[User Management 2](#_vemd3gm96bxg)

[Book Management 2](#_kqic56ulombp)

[Borrowing and Returning: 2](#_kv7ccgrjzn74)

[Reporting and Analytics 2](#_87tjqd9iw11)

[3. Non-Functional Requirements 3](#_kz3jaecnxcmp)

[Performance 3](#_qh46lsg8oer2)

[Security 3](#_7i2duehzxts2)

[Usability 3](#_n3g1mvv5fwhl)

[4. Interfaces 3](#_qxhcrtiflkl1)

[Backend Interfaces 3](#_3y2wessxw2c9)

[Frontend Interfaces 3](#_7vmt84wz5ecp)

[5. Data Structures 3](#_jp24llx8npym)

[6. Design Constraints 4](#_5xpw8mntutvf)

# 

# **Library Management System**

The System Requirements Specification (SRS) outlines the detailed technical specifications for the Library Management System (LMS). This system aims to efficiently manage library resources, encompassing book cataloging, user management, and borrowing activities.

### **1. System Overview**

The LMS utilizes Spring Boot for the backend, ReactJS for the frontend, and MySQL as the database. It facilitates the comprehensive management of books, users, and borrowing processes.

### **2. Functional Requirements**

#### User Management

* Create, Edit, and Delete User Accounts
* Manage User Roles and Permissions
* Record and Track User Borrowing History

#### Book Management

* Add, Edit, and Delete Book Records (Librarians)
* Categorize and Assign Books Using Genres and Subjects
* Maintain and Track Book Availability Status

#### Borrowing and Returning:

* Users can search for books based on availability and criteria
* Allow users to request available books for borrowing
* Set borrowing durations and handle due dates
* Process book returns and update availability status accordingly

#### Reporting and Analytics

* Generate Reports on Book Usage and Borrowing Trends
* Identify trends in user borrowing behavior
* Alert users (Notifications) or librarians about overdue book returns.

### **3. Non-Functional Requirements**

#### Performance

* Handle Concurrent Users: Support high concurrent user loads without performance degradation.
* Efficient Book Search and Filtering: Ensure swift and responsive book search operations. (Names and Genre search)

#### Security

* Secure User Authentication and Authorization: Implement robust authentication mechanisms.
* Protect Sensitive Data: Encrypt and safeguard user information and borrowing records.
* Prevent Unauthorized Access: Ensure access control to library resources.

#### Usability

* User-Friendly Interface: Create an intuitive interface for easy navigation.
* Clear Instructions and Guidance: Provide comprehensive guidance for users.
* Consistent Design Patterns: Ensure coherence between frontend and backend design patterns.

### **4. Interfaces**

#### Backend Interfaces

RESTful APIs for user management, book operations, and borrowing processes.

#### Frontend Interfaces

User interface components for user interactions, book browsing, and borrowing.

### **5. Data Structures**

* Tables: Users, Books, BorrowingHistory, Genres, Subjects.
* Relationships and constraints between tables.

### **6. Design Constraints**

Spring Boot and ReactJS frameworks are used due to their robustness, ease of development, and popularity in the industry. MySQL was chosen due to compatibility, scalability, and familiarity. Best coding practices, documentation and design principles were utilized to facilitate future modifications.