AIM: Prepare broad SRS (software requirement software) for the selected project

SRS stands for **Software Requirements Specification**. It is a detailed document that describes the functional and non-functional requirements of a software system. The purpose of an SRS is to provide a clear and comprehensive description of the system to be developed, ensuring that both the development team and the client (or stakeholders) have a shared understanding of the system's goals and functionality.

Here are the key components typically included in an SRS document:

1. **Introduction**: This section provides an overview of the software, including its purpose, scope, and any definitions or acronyms that will be used in the document.
2. **System Overview**: A high-level description of the system, explaining what it will do without diving into technical details.
3. **Functional Requirements**: These are detailed descriptions of the functionalities the system must support. It includes user interactions, data processing, and specific tasks the software will perform.
4. **Non-Functional Requirements**: These specify the quality attributes the system must have, such as performance, security, usability, and reliability.
5. **System Architecture and Design**: This section outlines the architectural design of the system, including any components and technologies that will be used to implement the system.
6. **Appendix**: This section may include additional information like acronyms, glossary of terms, assumptions, and dependencies.

Software Requirements Specification (SRS) for Library Management System

1. Introduction

1.1 Purpose The purpose of this document is to define the requirements for the Library Management System (LMS). This system is intended to automate the functions of a library, including book management, member management, borrowing, returning, and searching.

1.2 Scope

The Library Management System will handle the following:

- Catalog management of books
- Member registration and management
- Book borrowing and returning process

- Search functionality for books
- Overdue notifications
- Reports generation

1.3 Definitions, Acronyms, and Abbreviations

- **LMS**: Library Management System
- **Admin**: Library administrator with full access to the system
- **Member**: Library user registered with the system
- **Book**: Physical or digital item in the library catalog

1.4 References

- ISO/IEC 9126 - Software Engineering – Product Quality
- IEEE 830 - IEEE Recommended Practice for Software Requirements Specifications

2. System Overview

The Library Management System will provide users with the ability to search for books, borrow them, and return them. Administrators will manage user accounts, monitor book availability, and oversee transactions. It will also include features to track overdue items and send notifications to users.

---

3. Functional Requirements

3.1 User Management

- **3.1.1 Registration**: A user should be able to register as a member in the system by providing necessary personal details (name, contact information, etc.).
- **3.1.2 Login/Logout**: Users (members and admins) must be able to log in and log out using credentials (username and password).

- **3.1.3 Profile Management**: Members should be able to view and update their personal details.

3.2 Book Management

- **3.2.1 Add Books**: Admin should be able to add new books to the library catalog, including title, author, ISBN, genre, and availability status.
- **3.2.2 Edit Book Information**: Admin should be able to modify the book details.
- **3.2.3 Remove Books**: Admin should be able to delete books from the catalog when needed.

3.3 Borrowing and Returning Books

- **3.3.1 Borrow Books**: Members should be able to borrow available books, with details recorded in the system (borrow date, return due date).
- **3.3.2 Return Books**: Members should be able to return books by marking them as returned in the system.
- • **3.3.3 Overdue Management**: System should track overdue books and generate overdue notifications to members.

3.4 Search Functionality

- **3.4.1 Search Books**: Members and admins should be able to search for books by title, author, genre, or ISBN.
- **3.4.2 Advanced Search**: Admins should be able to perform detailed searches with filters (e.g., availability, genre, publication date).

3.5 Notifications

- **3.5.1 Overdue Notifications**: The system should automatically notify members of overdue books via email or SMS.
- **3.5.2 Reservation Notifications**: If a borrowed book is returned, the system should notify members on the waiting list (if applicable).

3.6 Reporting

- **3.6.1 Book Availability Report**: Admin should be able to generate reports on the availability of books.

- **3.6.2 Member Activity Report**: Admin should be able to view reports on members' borrowing and returning activities.
- **3.6.3 Overdue Book Report**: Admin should be able to generate a report of overdue books and associated fines.

4. Non-Functional Requirements

4.1 Performance Requirements

- The system should handle up to 500 simultaneous users.
- The system response time for search functionality should be less than 2 seconds.

4.2 Security Requirements

- User passwords must be stored securely using encryption.
- The system should have role-based access control (admin, member).
- Sensitive member information should be encrypted in the database.

4.3 Availability

- The system should be available 24/7 with an uptime of 99.9%.

4.4 Backup

- The system must back up its data daily and allow restoration from the backup in case of a failure.

4.5 Usability

- The system interface should be intuitive and easy to navigate, with a user-friendly design.
- The system should be accessible on both desktop and mobile devices.

5. System Architecture and Design

5.1 Overview

The system will follow a client-server architecture. The client-side will be a web-based interface accessible via browsers, while the server-side will handle requests, process data, and interface with the database.

5.2 Components

- **Client**: Web-based user interface for interaction with the system.
- **Server**: Handles requests, processes data, and interacts with the database.
- **Database**: Stores user and book data, transaction logs, and other relevant information.

5.3 Technologies

- Frontend: HTML, CSS, JavaScript (React, Angular, etc.)
- Backend: Node.js, Python (Django/Flask), or Java (Spring Boot)
- Database: MySQL, PostgreSQL, or MongoDB

6. Appendix

6.1 Glossary

- **ISBN**: International Standard Book Number
- **CRUD**: Create, Read, Update, Delete (Basic operations for managing data)
- **UI/UX**: User Interface/User Experience

6.2 Assumptions and Dependencies

- Users must have internet access to interact with the system.
- System must operate on standard web browsers.