Library Management System - Project Design Document

Project Goal

Create a robust, easy-to-use Library Management System (LMS) that enables admins to manage books, members, and transactions efficiently, while providing reports, fine management, and optional advanced features like reservations and dashboards.

System Overview

The LMS is a CRUD-based web application with authentication for administrators, allowing them to manage books, members, issue/return transactions, track fines, generate reports, and provide search/filtering capabilities.

Core Modules

A. Books Management:
- Add, update, delete books.
- Search/filter by title, author, category, availability.
- Auto-update availability on issue/return.

B. Member Management:
- Add, update, delete members.
- Assign membership types.
- Search by name or ID.

C. Issue & Return:
- Record issues with member & book IDs.
- Prevent duplicate issues.
- Auto-update availability on return.

D. Fine Management:
- Calculate late fees.
- Record payments.
- View fine history per member.

E. Reports:
- Issued books, member history, popular books, availability summary.

F. Admin Controls:
- Admin login/logout.
- Manage all records.
- Optional DB export.

Database Design (Entities)

Library Management System - Project Design Document

1. Books: book_id, title, author, genre, isbn, price, availability, edition, created_at
2. Members: member_id, name, email, phone, address, membership_type, active_status, created_at
3. IssuedBooks: issue_id, book_id, member_id, issue_date, due_date, return_date, fine_amount, fine_paid
4. Fines (optional): fine_id, member_id, issue_id, amount, paid_status, created_at
5. Reservations (optional): reservation_id, member_id, book_id, reservation_date, status

Tech Stack

Frontend: React.js / Angular / Vue.js
Styling: Tailwind CSS / Bootstrap
Backend: Node.js + Express.js / Django / Spring Boot
Database: MySQL / PostgreSQL / MongoDB
Auth: JWT / Session-based
Hosting: Vercel / Render / AWS / Azure
Reports: Chart.js / Recharts

API Endpoints (Example)

Books:
POST /books, GET /books, GET /books/:id, PUT /books/:id, DELETE /books/:id

Members:
POST /members, GET /members, GET /members/:id, PUT /members/:id, DELETE /members/:id

Issued Books:
POST /issue, PUT /return/:issue_id, GET /issued

Fines:
GET /fines, PUT /fines/:id/pay

User Flow

1. Login
2. Dashboard
3. Books Management
4. Members Management
5. Issue/Return
6. Fines
7. Reports
8. Logout

Development Plan (Agile)

Sprint 1: Project setup, Admin login, Books CRUD

Library Management System - Project Design Document

Sprint 2: Members CRUD, Issue/Return module
Sprint 3: Fine calculation, Reports module
Sprint 4: Search/filters, Dashboard, Final testing & deployment

Security Considerations

Validate all inputs, implement authentication, optional role-based access, and secure password storage (bcrypt).