Library Management System - Full Stack Project Exercise

🎯 Project Overview

You will build a comprehensive Library Management System for **"City Public Library"** - a community library that needs to digitize their book lending process, manage member registrations, and track book inventory. This system will handle book cataloging, member management, borrowing/returning books, and basic reporting.

📋 Business Requirements

Library Background

City Public Library serves a community of 5,000+ residents with:

- **Librarians** (3 staff): Manage books, assist members, handle day-to-day operations
- **Library Manager** (1 person): Oversee operations, view reports, manage staff
- **Members** (500+ registered): Browse books, borrow/return books, view history
- **Admin** (1 person): System configuration, user management, data maintenance

Core Business Problems to Solve

1. Manual book checkout process using paper cards
2. No easy way to search available books
3. Difficulty tracking overdue books and fines
4. No system to manage member information
5. Manual inventory management
6. No way for members to see their borrowing history

🔧 Technical Requirements

User Roles & Permissions Matrix

Feature/Action	Admin	Manager	Librarian	Member
User Management	Full	View Only	None	Own Profile
Book Management	Full	Full	Full	View Only
Member Management	Full	Full	Full	Own Profile
Issue/Return Books	Yes	Yes	Yes	No
View All Transactions	Yes	Yes	Yes	Own Only
Generate Reports	Yes	Yes	Limited	No
Fine Management	Yes	Yes	Yes	View Own
System Settings	Full	Limited	None	None

Core Entities & Relationships

1. Users

- id, email, password, first_name, last_name, role, phone
- address, membership_date, is_active, created_at, updated_at
- Roles: Admin, Manager, Librarian, Member

2. Books

- id, title, author, isbn, category, publisher, publication_year
- total_copies, available_copies, shelf_location, description
- added_date, updated_at, is_active

3. Categories

- id, name, description, created_at, updated_at
- Examples: Fiction, Non-Fiction, Science, History, Children, Technology

4. Transactions

- id, book_id, member_id, librarian_id, issue_date, due_date
- return_date, status, fine_amount, notes
- Status: Issued, Returned, Overdue

5. Fines

- id, member_id, transaction_id, amount, reason, status
- created_date, paid_date, waived_by
- Status: Pending, Paid, Waived

6. Reservations

- id, book_id, member_id, reservation_date, status, expiry_date
- Status: Active, Fulfilled, Expired, Cancelled

🎯 Functional Requirements

Phase 1: Authentication & Basic Setup

- Create user registration and login system
- Implement role-based access control (4 roles)
- Build user profile management
- Create admin panel for user management
- Set up basic navigation based on user roles
- Implement logout functionality

Phase 2: Book Management System

- Create book addition form with validation
- Build book catalog with search and filter functionality
- Implement category management
- Create book detail pages
- Add book editing and deletion (soft delete)
- Build inventory tracking (total vs available copies)
- Implement book image upload (optional)

Phase 3: Member Management

- Create member registration form
- Build member profile pages
- Implement member search functionality
- Create member status management (active/inactive)
- Add member borrowing history view
- Build member fine tracking

Phase 4: Book Borrowing System

- Create book issue/checkout functionality
- Implement book return process
- Build due date calculation (14-day loan period)
- Create overdue book tracking
- Implement fine calculation ($1/day overdue)
- Add book reservation system
- Build "My Books" page for members

Phase 5: Reports & Dashboard

- Create dashboard for different user roles
- Build popular books report
- Implement overdue books report
- Create member activity reports
- Add fine collection reports
- Build book inventory status
- Create simple analytics charts

🛠 Technical Implementation Guide

Technology Stack Options
Backend: Node.js/Express
Database: PostgreSQL or MySQL or MongoDB
Frontend: HTML/CSS/JavaScript or React
Authentication: Session-based (simpler than JWT)

Database Design Requirements
1. Create simple ERD with 6 main tables
2. Use foreign keys to maintain relationships
3. Include basic indexes on frequently searched fields
4. Create sample data for testing (20 books, 10 members)
5. Implement basic data validation

API Requirements (Simplified)
- Basic CRUD operations for each entity
- Simple input validation
- Basic error handling with user-friendly messages
- Clear API endpoints (RESTful preferred but not mandatory)

Frontend Requirements
- Clean, simple interface (Bootstrap recommended)
- Basic responsive design
- Form validation with clear error messages
- Simple navigation menu
- Basic search functionality

📊 Sample Data Scenarios

Test Data to Include

1. **Books**: 20 diverse books across different categories
    - Fiction: "The Great Gatsby", "To Kill a Mockingbird"
    - Science: "A Brief History of Time", "The Selfish Gene"
    - Children: "Harry Potter Series", "The Cat in the Hat"
    - Technology: "Clean Code", "The Pragmatic Programmer"

2. **Members**: 10 test members with different statuses
    - 3 active members with current borrowings
    - 2 members with overdue books
    - 2 members with fines
    - 3 new members with no borrowing history

3. **Transactions**: Various borrowing scenarios
    - Recently issued books
    - Books due soon
    - Overdue books with fines
    - Returned books

Business Rules to Implement

- Maximum 3 books per member at a time
- 14-day borrowing period
- $1 per day fine for overdue books
- Cannot borrow new books if fines exceed $10
- Books can be renewed once if no reservations exist

🔍 Testing Scenarios

Manual Testing Checklist

- Register new member and login
- Add new book to catalog
- Search for books by title/author/category
- Issue book to member
- Return book on time (no fine)
- Return book late (calculate fine)
- Try to borrow more than 3 books
- View member borrowing history
- Generate overdue books report
- Test different user role permissions

Simple Edge Cases

- Trying to issue unavailable books
- Returning already returned books
- Searching with empty or invalid inputs
- Accessing pages without proper permissions
- Handling duplicate ISBN entries

🚀 Deployment Requirements

Basic Deployment Checklist

- Environment configuration (database connection)
- Sample data seeding
- Basic security measures (password hashing)
- Simple error logging
- Basic performance optimization