Business Requirements Document: Digital Library Management System

1. Project Overview

Project Name: LibraryHub - Digital Library Management Platform

Objective: Create a modern digital library system for book lending and management

Business Context: Modernize traditional library operations with digital tools and online services

2. Stakeholders

- Library Staff
- Library Members
- System Administrators
- IT Support Team

3. Functional Requirements

3.1 Book Management

- Add, update, and remove books from catalog
- Book attributes: ISBN, title, author, genre, publication year, copies available
- Book search and filtering capabilities
- Digital book cover image upload

3.2 Member Management

- Member registration and profile management
- Member attributes: ID, name, email, phone, address, membership type
- Membership renewal and status tracking
- Member activity history

3.3 Lending Operations

- Book checkout and return processing
- Due date calculation and tracking
- Fine calculation for overdue books
- Reservation system for unavailable books

- Lending history and statistics

3.4 Reporting and Analytics

- Popular books report
- Member activity reports
- Overdue books tracking
- Financial reports for fines and fees

4. Technical Requirements

4.1 Backend (Python/FastAPI)

- RESTful API architecture
- SQLite/PostgreSQL database
- Pydantic data validation
- JWT authentication

4.2 Frontend (React)

- Single Page Application (SPA)
- Responsive design
- Search and filter components
- Data tables for book and member listing

5. Data Models

Books:

- id: integer (primary key)
- isbn: string (unique)
- title: string
- author: string
- genre: string
- publication_year: integer
- total_copies: integer
- available_copies: integer
- created_at: datetime

Members:

- id: integer (primary key)
- member_id: string (unique)
- first_name: string
- last_name: string
- email: string (unique)
- phone: string
- address: text
- membership_type: enum(regular, premium, student)
- join_date: date
- status: enum(active, inactive, suspended)

Loans:

- id: integer (primary key)
- book_id: integer (foreign key)
- member_id: integer (foreign key)
- checkout_date: date
- due_date: date
- return_date: date (nullable)
- fine_amount: decimal
- status: enum(active, returned, overdue)

6. API Endpoints

Books:

- GET /api/books - List all books
- POST /api/books - Add new book
- GET /api/books/{id} - Get book details
- PUT /api/books/{id} - Update book
- DELETE /api/books/{id} - Remove book

Members:

- GET /api/members - List all members
- POST /api/members - Register new member

- GET /api/members/{id} - Get member details
- PUT /api/members/{id} - Update member
- DELETE /api/members/{id} - Remove member

Loans:

- POST /api/loans/checkout - Checkout book
- PUT /api/loans/{id}/return - Return book
- GET /api/loans/overdue - List overdue loans
- GET /api/members/{id}/loans - Member loan history

7. Non-Functional Requirements

- Support up to 10,000 books and 5,000 members
- Response time under 1 second
- Daily automated backups
- Role-based access control
- Audit trail for all operations

8. Success Criteria

- Reduce book checkout time by 70%
- Eliminate manual record keeping
- Generate automated reports
- Improve member experience with online services