



COLLEGE CODE: 9222

COLLEGE NAME: THENI KAMMAVAR SANGAM

COLLEGEOFTECHNOLOGY

DEPARTMENT: B.tech(IT)

STUDENT NM-ID: aut922223it015

ROLL NO : 23it015

DATE : 19-09-2025

Completed the project named as

Phase__ TECHNOLOGY PROJECT

NAME: library book management

SUBMITTED BY,

NAME: JEYASURYA K

MOBILE NO: 6374954151

Problem statement & Requirement

Problem Statement

Libraries need an efficient and organized system to manage book inventory, member registration, issue/return processes, and fines. Traditional manual methods are time-consuming, error-prone, and lack real-time tracking.

A Library Book Management System will:

- Digitize and streamline book cataloging
- Track borrowing and returns
- Manage users (members, librarians)
- Enforce due dates and late fees

Users & Stakeholders

Role Description

Manages the system: adds books, handles check-in/check-out, registers

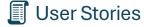
users

Member/User Can search books, borrow/return books, view history

Admin (Optional) Manages librarians, analytics, system settings

IT/Admin

Maintains the infrastructure and backend Staff



As a librarian, I want to:

- Add, edit, and remove books from the catalog
- Register and manage members
- Check in and check out books
- View reports on overdue books and fines

🙎 As a member, I want to:

- Search for available books
- View book details and availability
- Borrow and return books
- View my borrowing history and due dates

% As an admin, I want to:

- Monitor usage statistics
- Manage user roles and permissions
- View system logs

MVP Features (Minimum Viable Product)

| • | | |
|---|--|--|
| Module | Features | |
| User Management | Register/LoginRole-based access (Member/Librarian) | |
| Book Management | - Add/Edit/Delete books- View book list- Search & filter | |
| Borrow/Return | Issue book to userReturn bookDue date tracking | |
| Overdue/Fines | - Auto-calculate overdue days- Fine computation | |
| Reports (Basic) | - List of borrowed books- Overdue reports per user | |



You can build this as a web app or REST API. Here's a sample of what both might look like:

A. Wireframes (Text Description)

- Login Page (Email, Password)
- Dashboard
 - o Tabs: Books, Members, Borrowed, Reports
- Books Page
 - List with filters/search
 - o Add/Edit/Delete buttons
- Borrow/Return Page
 - Select user + book
 - o Issue with due date
 - Return book
- Member Profile
 - Borrowing history
 - Current borrowed books
 - o Fine info

(Let me know if you'd like image wireframes!)

B. API Endpoints (REST)

| Method | Endpoint | Description |
|--------|-------------------------|---------------------------|
| POST | /api/login | User login |
| GET | /api/books | Get all books |
| POST | /api/books | Add a book |
| PUT | /api/books/:id | Edit book |
| DELETE | /api/books/:id | Delete book |
| POST | /api/users | Register user |
| GET | /api/users/:id/borrowed | Get user's borrowed books |
| POST | /api/borrow | Issue a book |
| POST | /api/return | Return a book |
| GET | /api/reports/overdue | Get overdue report |
| | | |

✓ Acceptance Criteria

Book Management

- Librarian can add/edit/delete books
- Books have title, author, ISBN, status (available/borrowed)

✓ User Management

- Members can register/login
- Librarians can view member borrowing history

✓ Borrow/Return

- A user can borrow only if the book is available
- A due date is set at borrowing time (e.g., 14 days)
- Returned books are marked available

✓ Fines

- Fines are calculated as: # of days late × daily fine rate
- Users can view outstanding fines

Security & Access

- Only librarians can modify books or users
- Members can only borrow/return and view their own records