📚 Library Management System (LMS)

Phase 1 – Backend Engineering Final Project

🕒 Duration: **1 Month**

🎯 Goal: **Apply ALL Phase-1 backend skills in one real system**

🌍 Project Overview (Big Picture)

A **Library Management System** is a backend system that helps a library:

- Track **books**
- Manage **users (members & librarians)**
- Control **borrowing and returning**
- Enforce **rules** (limits, fines, availability)
- Keep **data consistent, secure, and reliable**

This system is **API-based** (no frontend focus).

Think of it as the **brain of the library**.

🧠 Storytelling: How the Client Uses This System

📖 The Client

A **small university library** with:

- 1 head librarian
- 3 assistant librarians
- 500+ students
- Thousands of books

They want to **stop using paper and Excel**.

👤 Characters in the Story

- **Admin Librarian**

  Manages system, books, users, rules

- **Student (Library Member)**

  Borrows and returns books

- **System (Your Backend)**

  Enforces rules and stores truth

🧩 A Day in the Library (Story Flow)

1. A student comes to the library
2. Librarian searches for a book
3. System checks availability
4. Student borrows the book
5. System updates stock
6. Deadline is calculated
7. Student returns book (late or on time)
8. System updates status and fines

👉 **All decisions happen in your backend**

🏗️ System Architecture (What You're Building)

🔧 Type of System

- RESTful Backend API
- Stateless
- JSON-based communication

📦 Core Components

- Server (request handling)
- Business logic (rules)
- Database (persistent storage)
- Authentication & authorization
- Error handling & validation

🗃️ Core Modules (VERY IMPORTANT)

You must clearly separate responsibilities.

1️⃣ User Management Module 👤

Purpose

Control **who can access the system** and **what they can do**.

User Types

- **Admin**
- **Librarian**
- **Member (Student)**

User Properties

Each user must have:

- Unique identity
- Role
- Account status (active / suspended)
- Secure credentials

Rules

- Only **Admins** can create librarians
- Librarians can register members
- Members **cannot access admin actions**
- Suspended users **cannot borrow books**

Skills Used Here

- Authentication concepts
- Authorization (role-based access)
- Secure password handling
- Environment variables
- Data validation

2️⃣ Book Management Module 📚

Purpose

Store and manage all book data.

Book Information

Each book must include:

- Title
- Author
- Category
- ISBN (unique)
- Total copies
- Available copies
- Status (available / unavailable)

Rules

- ISBN must be unique

- Available copies ≤ total copies
- Book cannot be deleted if currently borrowed
- Only librarians/admins can add or update books

Real-Life Meaning

This replaces:

- Physical book cards
- Excel spreadsheets
- Manual counting

3️⃣ Borrowing System 🔄

Purpose

Control book lending logic.

Borrowing Process

1. Member requests a book
2. System checks:
    - User status
    - Borrow limit
    - Book availability
3. If valid:
    - Borrow record is created
    - Available copies decrease
    - Due date is calculated

Borrowing Rules

- Max books per user (e.g. 3)
- Fixed borrowing period (e.g. 14 days)
- Cannot borrow same book twice

- Suspended users cannot borrow

Important Concept

👉 **Borrowing is NOT just a table**

It is a **business process**

4️⃣ Return & Fine System 💰

Purpose

Handle returns and penalties.

Return Process

1. Book is returned
2. System checks due date
3. If late:
    - Fine is calculated
4. Book status updates
5. Borrow record closes

Fine Rules

- Fixed fine per late day
- Fine accumulates
- Member cannot borrow if fine exceeds limit

Why This Matters

You are modeling **real-world consequences** using backend logic.

5️⃣ Database Design 🗄️

Entities You Must Have

- Users
- Roles

- Books
- Categories
- Borrow records
- Fine records (or derived logic)

Relationships

- One user → many borrow records
- One book → many borrow records
- One borrow record → one user + one book

Concepts Applied

- Primary keys
- Foreign keys
- Constraints
- Normalization
- Referential integrity

6️⃣ API Design 🌐

What Your API Must Support

- User registration & login
- Book CRUD operations
- Borrow book
- Return book
- View borrow history
- View user status

Principles

- Clear endpoints
- Meaningful HTTP status codes

- Consistent responses
- No business logic in routes

7️⃣ Validation & Error Handling ⚠️

You Must Handle

- Invalid input
- Missing fields
- Unauthorized access
- Forbidden actions
- Resource not found
- Logical errors (e.g. borrow unavailable book)

Goal

Your API should **never crash**

It should **always respond meaningfully**

8️⃣ Security Basics 🔐

Required Concepts

- Password hashing
- Token-based authentication
- Protected routes
- Environment variables
- No sensitive data in responses

This is **real backend security**, not theory.

🧪 Testing Mindset (Even Without Tests)

You should think:

- What if user sends bad data?

- What if same request is sent twice?
- What if database fails?
- What if user role is wrong?

This builds **engineering thinking**.

🧠 What You Will Achieve After Finishing This Project

🎓 Technical Skills

- Build a full backend system from scratch
- Design relational databases
- Apply authentication & authorization
- Implement real business logic
- Handle real-world edge cases
- Structure scalable backend projects

👨‍💻 Engineering Skills

- Think like a backend engineer
- Translate requirements → system design
- Separate logic cleanly
- Write maintainable backend architecture
- Understand **why**, not just **how**

🏁 Phase 1 Final Outcome 🚀

After this project, you are:

✅ **Junior Backend Engineer – Ready**

✅ Comfortable with real CRUD systems

✅ Ready for **Phase 2**

✅ Able to explain your system to clients or interviewers

🧩 Final Summary

- This project is **not about code**
- It is about **thinking like an engineer**
- If you can build this correctly:
  - You understand backend fundamentals
  - You are no longer a beginner

📌 **This Library Management System is your Phase-1 graduation project.**