Design & Implementation of a Library Database System

Taylor M Smith

Principles of Database Systems

New York University

SPR25

Project Overview

- The Library Management System automates book cataloging, member records, loans, reservations, and fines. Built on a 3-tier architecture for maintainability and scalability.

Library Management System

System Architecture

Describes the 3-tier architecture:

- Presentation Layer - Flask Web App (HTML templates)
- Business Logic Layer - Python functions, routing, loan/fine logic
- Data Layer - SQLite database with schema, triggers, views

ER Diagram

- Show the ER diagram: includes >7 entities like Books, Members, Publishers, Loans, Reservations, Fines, Librarians. Highlight PKs, FKs, and normalized relationships.

Database Design (DDL)

Overview of CREATE TABLE statements

Includes constraints:

- PKs/FKs
- UNIQUE ISBN
- Views: AvailableBooks, BookLoansReport
- Triggers: Loan/Return inventory updates

Advanced SQL Features

Implemented SQL features:

- Views: logical abstraction and filtering
- Triggers: automate inventory updates
- Simulated Procedure: `loan_book()`
- Function: `calculate_fine()` to compute overdue fees

CRUD Functionality

Screenshots:

- Homepage (Select)
- Add Book Form (Insert)
- Edit Book (Update)
- Delete Book Action

Each operation tested and functional

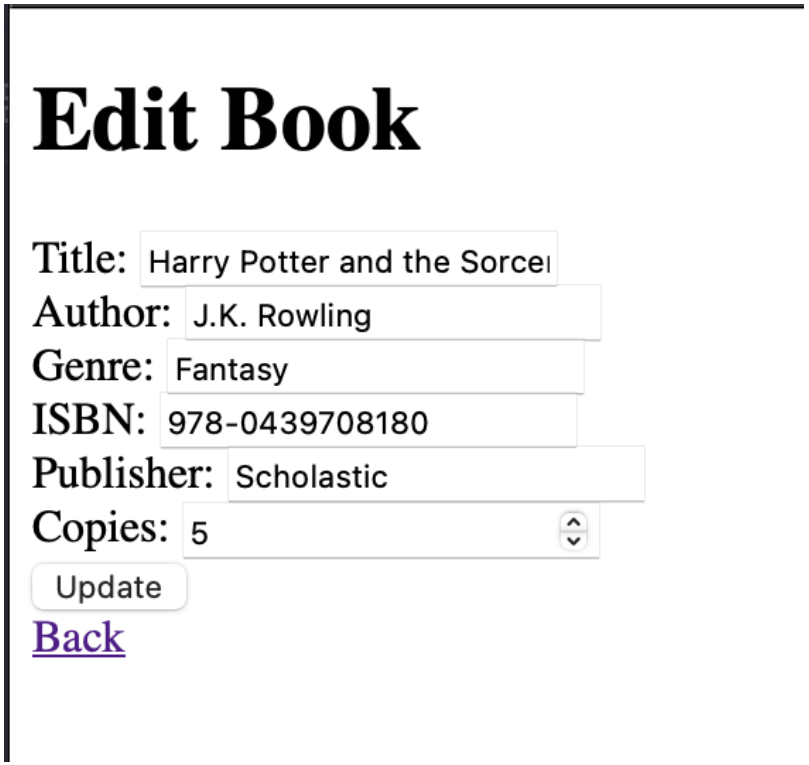
Homepage

Books | Members | Reservations | Fines

Library Books

Add New Book

Add & Edit Books



Add & Edit Members

Library Members

Add New Member

Back to Books

Add New Member

Name: Jane Doe

Email: 1234@nyu.edu

Add Member

Back to Members

Delete Book

Library Books

Add New Book

Report & Aggregation

- BookLoansReport view groups loans by member using `GROUP BY` and `COUNT(*)`. Demonstrates database reporting capabilities for analytics.

Normalization, Integrity, Isolation

- Normalization: separate entities for books, publishers, etc.
- Integrity: enforced through PK, FK, UNIQUE ISBN, triggers
- Isolation: SQLite's SERIALIZABLE isolation level (default); sufficient for single-user DB use case