Library Management System

Design Document

Name: Vaibhav Tyagi (VXT200018)

Course: CS 6360 Database Design

Introduction:

To build library management system consisting of 1000s of book records and borrower information, we will need a GUI (frontend) and a backend to store and a middleware which would help the user in interacting with the database. We can use a MVC architecture for this, as it will provide clear separation between frontend, middleware, and backend. This architecture will also prevent users from directly accessing the database and keep the database from any unwanted and erroneously made changes.

Design Decisions and Assumptions:

We will discuss all the design decisions and assumptions that were made at each level of the system.

To begin with, I have chosen the Django Framework to build this project. As it is a MVC architecture and easy to use. The applications built on Django are easy to maintain, modify and faster to develop. Further it is much easier to organize and divide the project into dedicated modules. Therefore, to find a specific part of code and debug is much quicker. Further, an authentication module is also added to the project. The user will be asked to Login/Signup before using the system. Django provides inbuilt modules to implement user authentication.

1. **Database (Backend):**

I am using MySQL database for the storing the information. MySQL is an open source, easy to install and connect with Python. The database can be connected to the Django application with the help of PyMySQL and mysqlclient modules. Further, the following tables are created in the library management database – book, authors, book_authors, book_loans, borrower, and fines. The schema of the tables is same as given in the requirements document. An additional field ‘available’ is added in the book table to know whether a book is in stock or not. Also, the field ISBN13 is not used in the schema as it was not mentioned in the requirement specifications.

2. **Middleware (Django framework - Python):**

As mentioned earlier, I have used Django framework to build the system. The middleware (business logic) is written in Python. I have used pandas to initially create the schema of the database and initialize the tables with the given data. Built on MVC architecture, it handles the user request for the data with the help of written business logic and protect the data from any unauthorized access and manipulation by providing the separation between view and database. Further, the application has multiple features like check-in, check-out, fines, adding borrower. All these features can be segregated to single module (a.k.a. application in Django) which makes it much easier to maintain the code and locate the errors.

3. **Frontend (GUI):**

Frontend of the system is built using HTML, CSS and Bootstrap. The GUI is built using HTML pages and styling is done with CSS. Bootstrap is also used to create and style some of the elements of the HTML pages. Bootstrap is very convenient to use as it has simple ways to style the elements and validate the form fields.

Programming Languages and Libraries used:

This is the list of programming languages, libraries, web framework and database used.

Web Framework: Django
Languages: Python, SQL, HTML, CSS, Bootstrap
Libraries: Pandas, PyMySQL, mysqlclient
Database: MySQL

Architecture:

Below is the architecture diagram of the system:

At the backend, we have MySQL database to store all the data related to the books and borrowers. This data can be only accessed through middleware and not directly by the user.

Then we have a middleware where all the business logic is written. It consists of multiple apps (modules) for each required functionality. As you can see, we have, Login app to manage the authentication of the user. Then we have Search Book app which can be used to search a book and then check it out. We also have a separate app to check in the book. There is also a Pay fine app which is used to refresh the fines, pay the fines and list the users with unpaid and paid fines. Lastly, we got borrower management app which is used to add the new borrower information.

All the above functionalities are accessed with the help of a GUI (frontend), through which the user interacts and gives their input as per requirement to carry out some function.

Further, the interaction between all the layers is bi-directional. The flow of data happens both ways between given two layers.