Assignment: Library Management System

Objective

Develop the backend for a library management system using your chosen backend technology (e.g. Django, FastAPI, Flask). The system will include models representing books and library users, demonstrating your ability to implement 1-1, 1-M, and M-M relationships. Also, create APIs to interact with these models.

Task Breakdown

Task 1: Project Setup

- Choose your backend framework and set up the development environment.
- Initialise a new project in a git repo and configure the database (e.g., MySQL, PostgreSQL).

Task 2: Database Schema Design

- Design and implement the database schema for the library system.
- Create the following models with specified attributes and relationships:
    1. User Model
        - Attributes: UserID, Name, Email, MembershipDate
        - Relationships: 1-M with BorrowedBooks (A user can borrow multiple books)
    2. Book Model
        - Attributes: BookID, Title, ISBN, PublishedDate, Genre
        - Relationships: 1-1 with BookDetails(Each book has one set of details)
    3. BookDetails Model (for 1-1 relationship)
        - Attributes: DetailsID, BookID (FK), NumberOfPages, Publisher, Language
        - Relationships: 1-1 with Book(Each set of book details is linked to exactly one book)
    4. BorrowedBooks Model (to demonstrate 1-N relationship)
        - Attributes: UserID (FK), BookID (FK), BorrowDate, ReturnDate
        - Relationships: 1-M with User(A user can borrow multiple books)

Task 3: API Development

- Develop the following APIs for each model:
  1. User APIs
     - Create a New User: Endpoint to add a new user to the system with details like name, email, and membership date.
     - List All Users: Endpoint to retrieve a list of all users in the system.
     - Get User by ID: Endpoint to fetch a user's details using their UserID.
  2. Book APIs
     - Add a New Book: Endpoint to add a new book record, including title, ISBN, published date, and genre.
     - List All Books: Endpoint to retrieve a list of all books in the library.
     - Get Book by ID: Endpoint to fetch details of a specific book using its BookID.
     - Assign/Update Book Details: Endpoint to assign details to a book or update existing book details, like number of pages, publisher, language.
  3. BorrowedBooks APIs
     - Borrow a Book: Endpoint to record the borrowing of a book by linking a user with a book.
     - Return a Book: Endpoint to update the system when a book is returned.
     - List All Borrowed Books: Endpoint to list all books currently borrowed from the library.

Task 4: Testing and Validation

- Test each API for functionality and reliability.
- Ensure all CRUD operations work as intended for each model.

Task 5: Code Quality and Error Handling

- Ensure your code is clean, readable, and well-documented.
- Implement robust error handling to cover edge cases and potential errors.

Task 6: Authentication Implementation (Bonus)

- As a bonus challenge, implement user authentication.
- Secure your APIs with appropriate authentication mechanisms.

Evaluation Criteria

1. Model Design: Correct implementation of 1-1 and 1-M relationships.
2. API Functionality: All APIs should perform the intended operations.
3. Code Quality: Clean, readable, and well-documented code.

4. Error Handling: Proper handling of edge cases and errors.
5. Bonus: Implement authentication for users and secure the APIs.

Submission Instructions

- Provide the source code via a **Git repository** through mail.
- Include a README with **setup instructions** and **API documentation**.
- Clearly state any additional notes or assumptions made during development.

This assignment provides an opportunity to showcase your skills in backend development, database design, and API creation, with a focus on implementing complex relationships and ensuring robust functionality.