Application Overview

The library management application is designed to allow users to perform various operations related to managing books, members, loans, reservations, and genres. Here is an overview of the technical choices and architecture adopted:

Technical Choices

Programming Language and Framework

PHP and Laravel: PHP is used as the primary development language with Laravel as the backend framework. Laravel provides a robust MVC structure, advanced security features, and seamless integration with third-party services through its extensive package ecosystem.

Database

MySQL: MySQL is chosen as the relational database management system. It is well-supported by Laravel, offering high performance, reliability, and ease of data management.

Security

Authentication and Authorization: Laravel Sanctum is used for API token-based authentication. This ensures secure management of user sessions and access to sensitive resources through authentication tokens.

RESTful API

RESTful API: A RESTful API is designed to facilitate easy integration with other systems and applications. Endpoints are designed following REST best practices, providing a consistent and predictable user experience.

Application Architecture

MVC Structure

Model-View-Controller (MVC): The application follows the MVC architecture to separate business logic (controllers), data management (models), and presentation (views or JSON API).

Core Modules

- **Book Management**: Modules for adding, updating, deleting, and viewing book details. Books are associated with genres, enabling efficient classification and search.
- **Member Management**: Features to manage library members, including adding, updating, deleting, and viewing member details.
- **Loans and Reservations Management**: Features for issuing and returning books, managing reservations, and sending notifications to concerned members.
- **Genre Management**: Ability to create, read, update, and delete genres associated with books, facilitating effective library organization.

Security and Session Management

Authentication and Tokens: Secure user authentication using JWT (JSON Web Tokens) generated by Laravel Sanctum. Tokens are used to control access to resources and protect sensitive API routes.

Conclusion

This technical documentation provides a clear overview of the technical choices and architecture of the library management application, highlighting decisions made to ensure optimal performance, enhanced security, and easy integration. It serves as a valuable reference for developers working on the project and anyone looking to understand the internal workings of the application.

testing Endpoints in Postman

1. Register a User

Request:

- **Method:** POST
- **URL:** http://your-api-url/register
- **Body:**
- {
-   "name": "John Doe",
-   "email": "john.doe@example.com",
-   "password": "password123",
-   "role": "user"
- }

Expected Response:

- Status: 200 OK
- Response Body:
- {
-   "message": "User registered successfully",

- "user": {
- "id": 1,
- "name": "John Doe",
- "email": "john.doe@example.com",
- "role": "user"
- }
- }

2. Login User

- **Request:**
- **Method:** POST
- **URL:** http://your-api-url/login
- **Body:**
- {
- "email": "john.doe@example.com",
- "password": "password123"
- }
- **Expected Response:**
- Status: 200 OK
- Response Body (contains the access_tokenfor subsequent requests):
- {
- "token_type": "Bearer",
- "expires_in": 3600,
- "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6Il9YQVRVZVJXdGxhSWw0em9UeWt3ZyJ9..."
- }

3. Add a Book

- **Request:**
- **Method:** POST
- **URL:** http://your-api-url/books
- **Headers:**
  - Authorization: Bearer <access_token> (from login response)
- **Body:**
- {
- "title": "Book Title",
- "author": "Book Author",
- "genre": "Fiction",
- "published_year": "2023",
- "ISBN": "978-3-16-148410-0",
- "copies_available": 10
- }
- **Expected Response:**
- Status: 201 Created

- Response Body:
- {
- "message": "Book added successfully",
- "data": {
- "id": 1,
- "title": "Book Title",
- "author": "Book Author",
- "genre": "Fiction",
- "published_year": "2023",
- "ISBN": "978-3-16-148410-0",
- "copies_available": 10,
- "created_at": "2024-07-10T12:00:00Z",
- "updated_at": "2024-07-10T12:00:00Z"
- }
- }

4. Get All Books

- **Request:**
- **Method:** GET
- **URL:** http://your-api-url/books
- **Headers:**
    - Authorization: Bearer <access_token>
- **Expected Response:**
- Status: 200 OK
- Response Body:
- json
- Copy code
- {
- "data": [
- {
- "id": 1,
- "title": "Book Title",
- "author": "Book Author",
- "genre": "Fiction",
- "published_year": "2023",
- "ISBN": "978-3-16-148410-0",
- "copies_available": 10,
- "created_at": "2024-07-10T12:00:00Z",
- "updated_at": "2024-07-10T12:00:00Z"
- }
-]
- }

5. Update a Book

- **Request:**
- **Method:** PUT
- **URL:** http://your-api-url/books/1 (assuming book ID is 1)
- **Headers:**

  - Authorization: Bearer <access_token>
- **Body:**
- {
-    "title": "Updated Book Title",
-    "author": "Updated Book Author",
-    "genre": "Science Fiction",
-    "published_year": "2024",
-    "ISBN": "978-3-16-148410-0",
-    "copies_available": 8
- }
- **Expected Response:**
- Status: 200 OK
- Response Body:
- {
-    "message": "Book updated successfully",
-    "data": {
-        "id": 1,
-        "title": "Updated Book Title",
-        "author": "Updated Book Author",
-        "genre": "Science Fiction",
-        "published_year": "2024",
-        "ISBN": "978-3-16-148410-0",
-        "copies_available": 8,
-        "created_at": "2024-07-10T12:00:00Z",
-        "updated_at": "2024-07-10T12:30:00Z"
-    }
- }

6. Delete a Book

- **Request:**
- **Method:** DELETE
- **URL:** http://your-api-url/books/1 (assuming book ID is 1)
- **Headers:**
  - Authorization: Bearer <access_token>
- **Expected Response:**
- Status: 200 OK
- Response Body:
-    "message": "Book deleted successfully"
- }
-