**Setup and Installation**

1. Prerequisites

Before you start the installation, ensure that your system meets the following requirements:

- \*\*PHP Version\*\*: 7.4 + versions

- \*\*Laravel Version\*\*: 8

- \*\*Database\*\*: MySQL

- \*\*Composer\*\*: 2.1.8

2. Installation and Setup

To install and set up the project, follow these steps:

1. Clone the project from the Git repository:

**git clone <repository\_url>**

2. Update Composer to install the project's dependencies:

**composer update**

3. Copy the content of the **.env.example** file and create a new .env file. Then, paste the copied content into the .env file and set up the database name as specified in the .env file

4. Migrate the database to create the required tables:

**php artisan migrate**

5. Run the necessary seeders:

- To seed permissions:

php artisan db:seed --class=PermissionTableSeeder

- To create an admin user:

php artisan db:seed --class=CreateAdminUserSeeder

6. Start the application:

php artisan serve

By following these steps, you'll ensure that the project is correctly set up and ready for use. The seeders will populate the database with initial data, including permissions and an admin user.

**Following modules in this project**

Roles and Permission (Spatie):

Roles are used to group users and grant them specific permissions. To define a role, use the Role model provided by Spatie

This project currently have an admin role and user role, admin can perform all the action like

**Create Book,**

**Update Book,**

**Delete Book ,**

**Borrow Book,**

**Return Book,**

**View User List ,**

**View Book list.**

User role can perform all the action like.

**Borrow Book,**

**Return Book,**

Authentication:  
  
JWT Authentication is used for authentication and authorization in the registration and login process

Api List:  
  
**1.Register**

**Endpoint:** POST /register (<http://127.0.0.1:8000/api/register>)

**Description:** Registers a new user.

**Request Example:**

{

"name": "John Doe",

"email": "john@example.com",

"password": "password123"  
}  
  
  
**2.Login**

**Endpoint:** POST /login (http://127.0.0.1:8000/api/login)

**Description:** Login user.

**Request Example:**

{

"email": "john@example.com",

"password": "password123"  
}  
  
  
**3.UserList**

**Endpoint:** GET / get\_users (http://127.0.0.1:8000/api/get\_users)

**Description:** get list of users.  
**Head:**  
{

"Authorization": "Bearer YOUR\_JWT\_TOKEN",

"Content-Type": "application/json"

}  
  
  
**4.Create Book**

**Endpoint:** POST / book-store (http://127.0.0.1:8000/api/ book-store)

**Description:** create a new book.  
**Head:**  
{

"Authorization": "Bearer YOUR\_JWT\_TOKEN",

"Content-Type": "application/json"

}  
  
**Request Example:**

{

"name":"nehru",

"category":"Story",

"author": "nehru",

"publisher": "nehru1",

"isbn": "test123455"

}  
  
**5.update Book**

**Endpoint:** PUT/ book-update/{id}(http://127.0.0.1:8000/api/ book-update/1)

**Description:** update existing book.  
**Head:**  
{

"Authorization": "Bearer YOUR\_JWT\_TOKEN",

"Content-Type": "application/json"  
}  
  
**Request Example:**

{

"name":"nehru1",

"category":"Story1"  
}  
  
**6.Delete Book**

**Endpoint:** DELETE/ book-delete/{id}(http://127.0.0.1:8000/api/ book-delete/1)

**Description:** delete existing book.  
**Head:**  
{

"Authorization": "Bearer YOUR\_JWT\_TOKEN",

"Content-Type": "application/json"  
}

**7.Get Book**

**Endpoint:** GET/ get-books (http://127.0.0.1:8000/api/ get-books)

**Description:** Get book with status(available/borrowed)and who borrow the book.  
**Head:**  
{

"Authorization": "Bearer YOUR\_JWT\_TOKEN",

"Content-Type": "application/json"  
}  
  
  
**8.Borrow Book**

**Endpoint:** POST/ borrowed-book/{id}(http://127.0.0.1:8000/api/ borrowed-book /1)

**Description:** Borrow a book.  
**Head:**  
{

"Authorization": "Bearer YOUR\_JWT\_TOKEN",

"Content-Type": "application/json"  
}  
  
  
  
  
  
  
  
  
**8.Return Book**

**Endpoint:** POST/ return-book/{id}(http://127.0.0.1:8000/api/ return-book /1)

**Description:** Return a borrowed book.  
**Head:**  
{

"Authorization": "Bearer YOUR\_JWT\_TOKEN",

"Content-Type": "application/json"  
}

Repository Link : <https://github.com/nehruajith/library-management.git>