**Library Management App**

Library Management API helps you automate library functions in a secure way. This API provides a good exposure to its functionalities Even for a layman and it makes it very handy to deal with Users ,Librarians and Books' data in back-end. 🚀

**Library Management API**

Operations that can be performed:

* **Create users**
* **Create Librarians**
* **Get Book details from DB**
* **Inserting New Books in DB**
* **Update users' credentials( Refreshing of Tokens )**
* **Delete users**
* **Issuing book**

### **POST**

**Post request for signing up of new users**

URL: <http://127.0.0.1:8000/users>

this request is used for creating a new user in database

This endpoint lets you create a new user in database. As you provide Name , Password and Role fields as BODY() param. This POST request will insert a new user row in database schema.

**Parameters:**

**Request Body ( Required ): Example**

{

"Name": "Joelay field",

"Password": "any",

"Role": "user or Librarian"

}

Response:

Code: Description

200 Successful Response

"string"

422 Validation Error

{

"detail": [

{

"loc": [

"string"

],

"msg": "string",

"type": "string"

}

]

}

### **POST**

**Post request for user login**

URL: <http://127.0.0.1:8000/user/login>

This endpoint is to get refreshed tokens for registered users.

This endpoint will let you generate new refreshed jwt tokens for already registered users. incase if your token gets expired just use this request and get a new one.

Example Value:

{

"Name": "sitaram ",

"Password": "any"

}

Response:

Code: Description

200 Successful Response

"string"

422 Validation Error

{

"detail": [

{

"loc": [

"string"

],

"msg": "string",

"type": "string"

}

]

}

**DELETE:**

**DELETE request for user deletion**

URL: <http://127.0.0.1:8000/user/{user_id}/del>

**You can perform delete action for users.**

{Path Param} user\_id : String( ID of user)

Response:

Code: Description

200 Successful Response

"string"

422 Validation Error

{

"detail": [

{

"loc": [

"string"

],

"msg": "string",

"type": "string"

}

]

}

**GET**

**GET request To see all books from database**

URL: <http://127.0.0.1:8000/book/all>

This endpoint lets you see all the books that are available in database. It exposes unique ID of books that can be used while issuing any book.

Response:

Code: Description

200 Successful Response

[

{

"Publication": "Gita Press",

"id": "fcc2aa6b-24a6-4324-a7b9-eaf2191eac2e",

"No\_of\_copies": 511,

"Subject": "Religious",

"Title": "sitaram",

"Author": "any"

}

|  |  |
| --- | --- |
| 404 | Resources Not available |
|  | **POST**  **POST request To add up book in database**  URL: <http://127.0.0.1:8000/book/add>  This endpoint lets you create a new book in database. As you provide Title , Author , Publication,'Subject' and No\_of\_copies fields as BODY() param. This POST request will insert a new user row in databse schema. Note that only Librarians are authorized and entitled to perform this task.  **Parameters:**  **Request Body ( Required ): Example**  {  "Title": "sitaram",  "Author": "any",  "Publication": "Gita Press",  "Subject": "Religious",  "No\_of\_copies": "2"  }  Response:  Code: Description  403 Error: Forbidden  {  "detail": "Not authenticated"  }  200 Successful Response  "string"  422 Validation Error  {  "detail": [  {  "loc": [  "string"  ],  "msg": "string",  "type": "string"  }  ]  } |
|  | **POST**  **POST request To get a book issued for user**  URL: <http://127.0.0.1:8000/book/issue>  To get any book issued from library , use this API end point. Any user , be it a librarian or student, can issue book from database.  **Parameters:**  {Query Param} user\_id : UUID( ID of user)  {Query Param} book\_id : UUID( ID of book)  **Request Body ( Required ): Example**  {  "Title": "sitaram",  "Author": "any",  "Publication": "Gita Press",  "Subject": "Religious",  "No\_of\_copies": "1"  }  Response:  Code: Description  200 Successful Response  "string"  422 Validation Error  {  "detail": [  {  "loc": [  "string"  ],  "msg": "string",  "type": "string"  }  ]  } |