library api

User

POST userLogin

http://localhost:3000/api/users/login

API POST /api/users/login

This endpoint is used to log in a user.

Request Body

- `email` (text, required): The email of the user.
- `password` (text, required): The password of the user.

Response

- Status: 200
- Content-Type: application/json

Success Response

json

```
{
    "success": true,
    "message": "",
    "token": ""
}
```

Body raw (json)

json

```
{
    "email": "navabdarveshali@gmail.com",
    "password": "#@ap22R1948"
}
```

POST userRegister

http://localhost:3000/api/users/register

The `POST /api/users/register` endpoint is used to register a new user.

Request Body

- `name` (text, required): The name of the user.
- `email` (text, required): The email address of the user.
- `password` (text, required): The password for the user's account.
- `contact` (text, required): The contact information of the user.

Response (application/json)

The response schema for the successful registration is as follows:

json

```
{
  "type": "object",
```

```
  "properties": {
    "success": {
      "type": "boolean"
    },
    "message": {
      "type": "string"
    },
    "userId": {
```


Body raw (json)


```
json

{
    "name": "navab darveshali",
    "email": "navabdarveshali@gmail.com",
    "password": "#@ap22R1948",
    "contact": "9014008288"
}
```


PUT updateUser

http://localhost:3000/api/users/update

Update User Information

This endpoint allows the client to update user information.

Request

The request should be sent as an HTTP PUT to the following URL:

`http://localhost:3000/api/users/update`

Request Body

The request body should be in JSON format and include the following parameters:

- `name` (string, optional): The updated name of the user.
- `email` (string, optional): The updated email address of the user.
- `contact` (string, optional): The updated contact information of the user.

Example:


```
json

{
    "name": "Updated Name",
    "email": "updated@example.com",
    "contact": "1234567890"
}
## Response
The response to this request will be a JSON object conforming to the follo
```json
{
    "type": "object",
    "properties": {
```


The response will contain a `status` key with a string value indicating the status of the update operation, and a `message` key with a string value providing additional information about the update.

HEADERS

Authorization	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwiZW1haWwiOiJuYXZhYmRhcnZlc2hhbGlAZ21haWwuY29tIiwiaWF0IjoxNzI3ODQ4MjcwLCJleHAiOjE3Mjc4NTE4NzB9.6VgTSr5pkXh-RzzQOh3oD_eIU8KRZN4FMZ-uhoBVRK8

Body raw (json)
```

```
json
```

```
{
  "name": "New Name",
  "email": "newemail@example.com",
  "contact": "1234567890"
}
```

books

GET getAllBooks

```
http://localhost:3000/api/books/genre/bumchik
```

This endpoint retrieves a list of books belonging to a specific genre.

The response of this request can be documented as a JSON schema as follows:

```
json
```

```
{
    "type": "object",
    "properties": {
        "success": {
            "type": "boolean"
        },
        "books": {
            "type": "array",
            "items": {
                "type": "object",
                "properties": {
```

POST addBook

```
http://localhost:3000/api/books/add
```

Add Book

This endpoint allows the addition of a new book to the database.

Request

- Method: POST
- URL: `http://localhost:3000/api/books/add`
- Body:
  - `title` (text, required): The title of the book.
  - `author` (text, required): The author of the book.
  - `genre` (text, required): The genre of the book.
  - `available` (text, required): Indicates the availability of the book.

Response

The response is in JSON format and has the following schema:

```
json
```

```
{
    "type": "object",
    "properties": {
        "success": {
            "type": "boolean"
        },
        "message": {
            "type": "string"
        },
        "bookId": {
            "type": "integer"
```

- Status: 201 Created
- Content-Type: application/json
- `success` (boolean): Indicates if the request was successful.

- `message` (string): Additional information or error message.
- `bookId` (integer): The unique identifier of the newly added book.

HEADERS

Authorization	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwiZW1haWwiOiJuYXZhYmRhcnZlc2hhbGlAZ21haWwuY29tIiwiaWF0IjoxNzI3ODQ4MjcwLCJleHAiOjE3Mjc4NTE4NzB9.6VgTSr5pkXh-RzzQOh3oD_eIU8KRZN4FMZ-uhoBVRK8

Body raw (json)

```json
{
    "title":"50 shades of gray",
    "author":"navab",
    "genre":"bumchik",
    "available":"available"
}
```

PUT updateBook

http://localhost:3000/api/books/update/1

Update Book Details

This endpoint is used to update the details of a specific book.

Request Body

- `title` (string, required): The updated title of the book.
- `author` (string, required): The updated author of the book.
- `genre` (string, required): The updated genre of the book.
- `available` (boolean, required): Indicates the availability of the book.

Response

The response is in JSON format and follows the schema below:

```json
{
    "type": "object",
    "properties": {
        "success": {
            "type": "boolean"
        },
        "message": {
            "type": "string"
        },
        "updatedBook": {
            "type": "integer"
```

- `success` (boolean): Indicates if the update operation was successful.
- `message` (string): A message related to the update operation.
- `updatedBook` (integer): The ID of the updated book.

HEADERS

Authorization	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwiZW1haWwiOiJuYXZhYmRhcnZlc2hhbGlAZ21haWwuY29tIiwiaWF0IjoxNzI3ODQ4MjcwLCJleHAiOjE3Mjc4NTE4NzB9.6VgTSr5pkXh-RzzQOh3oD_eIU8KRZN4FMZ-uhoBVRK8

Body raw (json)

```
json
```

```
{
    "title":"50 shades of gray",
    "author":"uday",
    "genre":"sexeducation",
    "available":"available"
}
```

DELETE deleteBook

```
http://localhost:3000/api/books/delete/2
```

This endpoint sends an HTTP DELETE request to delete a specific book with the ID 3.

Request Body

The request does not require a request body.

Response

- Status: 200
- Content-Type: application/json

The response contains a JSON object with the following fields:

- `success` (boolean): Indicates whether the deletion was successful.
- `message` (string): Provides a message related to the deletion process.

To save the `success` field from the response to an environment or global variable, you can use Postman's test scripts with the following code:

```
bash
```

```
pm.environment.set("successVariable", pm.response.json().success);
```

HEADERS

Authorization	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwiZW1haWwiOiJuYXZhYmRhcnZlc2hhbGlAZ21haWwuY29tIiwiaWF0IjoxNzI3ODQ4MjcwLCJleHAiOjE3Mjc4NTE4NzB9.6VgTSr5pkXh-RzzQOh3oD_eIU8KRZN4FMZ-uhoBVRK8

Body raw (json)

```
json
```

```
{
    "title":"50 shades of gray",
    "author":"uday",
    "genre":"sexeducation",
    "available":"available"
}
```

---

borrows

POST borrow

```
http://localhost:3000/api/borrows/borrow
```

Borrow Book

This endpoint allows the user to borrow a book by providing the user ID, book ID, and email.

Request Body

- `user_id` (string): The ID of the user borrowing the book.
- `book_id` (string): The ID of the book being borrowed.
- `email` (string): The email of the user borrowing the book.

Response

The response is in JSON format and follows the schema below:

```json
{
    "type": "object",
    "properties": {
        "success": {
            "type": "boolean"
        },
        "message": {
            "type": "string"
        },
        "data": {
            "type": "number"
```

HEADERS

Authorization	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MywiZW1haWwiOiJuYXZhYmRhcnZlc2hhbGlAZ21haWwuY29tIiwiaWF0IjoxNzI3ODYzNDUyLCJleHAiOjE3Mjc4ODE0NTJ9.Vv42xAGeq_1p7eMG-Y6o20BMowlJUldMePze9t3lD6M

Body raw (json)

```json
{
    "user_id":"1",
    "book_id":"3",
    "email":"navabdarveshali33@gmail.com"

}
```

POST returnBook

http://localhost:3000/api/borrows/return

The endpoint `/api/borrows/return` is a POST request used to return a borrowed item. The request should include a payload with the `id` of the borrowed item.

Request Body

- `id` (string, required): The ID of the borrowed item to be returned.

Response

The response is in JSON format and has the following schema:

```json
{
  "type": "object",
  "properties": {
    "success": {
      "type": "boolean"
    },
    "message": {
      "type": "string"
    },
    "data": {
      "type": "number"
```

The response includes:

- `success` (boolean): Indicates if the return operation was successful.
- `message` (string): A message related to the return operation.

- `data` (number): Additional data related to the return operation.

HEADERS

Authorization	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MywiZW1haWwiOiJuYXZhYmRhcnZlc2hhbGlAZ21haWwuY29tIiwiaWF0IjoxNzI3ODUxOTYzLCJleHAiOjE3Mjc4Njk5NjN9.XyeAKwPbnH3zW1ZWqygiz6b223xzWJnSDLtNJDKkJVQ

PARAMS

Authorization

Body raw (json)

```
json
```

```
{
    "id":"12"
}
```

GET UserBorrowedBooks

http://localhost:3000/api/borrows/user/1

This endpoint retrieves a list of borrowed books for a specific user.

Request

- Method: GET
- Endpoint: `http://localhost:3000/api/borrows/user/1`

Response

The response is a JSON object with the following schema:

```
json
```

```
{
  "type": "object",
  "properties": {
    "success": {
      "type": "boolean"
    },
    "message": {
      "type": "string"
    },
    "data": {
      "type": "array",
```

HEADERS

Authorization	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MywiZW1haWwiOiJuYXZhYmRhcnZlc2hhbGlAZ21haWwuY29tIiwiaWF0IjoxNzI3ODUxOTYzLCJleHAiOjE3Mjc4Njk5NjN9.XyeAKwPbnH3zW1ZWqygiz6b223xzWJnSDLtNJDKkJVQ

GET getOverdueBooks

http://localhost:3000/api/borrows/getdues