Libro.WebApi

Overview

An ASP.NET Core Web API for Library System

Terms Of Service

Example License

Security

Type	Name	Scopes
apiKey	Bearer	

Paths

POST `/Authentication/Rgister` Rgister New User

Sample request:

```
 POST /Authentication/Rgister
 {
      userName: userNameTest,
      email: userEmail@gmail.com,
      phoneNumber: 0593497415,
      password: testPassword
 }
Sample request2:
```

```
POST /Rgister
{
     userName: userNameTest,
     email: userEmail@gmail.com,
     password: testPassword
}
```

Responses

Code	Description	Links
200	Return the User’s info when Rgister sucessfully *Content* **text/plain** **application/json** **text/json**	No Links

<table>
<tr><th>Code</th><th>Description</th><th>Links</th></tr>
<tr><td>400</td><td>

When user info (Email,UserName,Phone) Already Exists

Content

text/plain

Example

```
{"status":400,"errors":[{"fieldName":"PhoneNumber","mes
sage":"the error message - example -The PhoneNumber
field is not a valid phone
number."},{"fieldName":"Email","message":" the error
message - example - The Email is used or
required"},{"fieldName":"UserName","message":" the
error message - example - The UserName is used or
required"},{"fieldName":"Password","message":" the
error message - example - The Password field is
required"}]}
```

application/json

Example

```
{"status":400,"errors":[{"fieldName":"PhoneNumber","mes
sage":"the error message - example -The PhoneNumber
field is not a valid phone
number."},{"fieldName":"Email","message":" the error
message - example - The Email is used or
required"},{"fieldName":"UserName","message":" the
error message - example - The UserName is used or
required"},{"fieldName":"Password","message":" the
error message - example - The Password field is
required"}]}
```

text/json

Example

```
{"status":400,"errors":[{"fieldName":"PhoneNumber","mes
sage":"the error message - example -The PhoneNumber
field is not a valid phone
number."},{"fieldName":"Email","message":" the error
message - example - The Email is used or
required"},{"fieldName":"UserName","message":" the
error message - example - The UserName is used or
required"},{"fieldName":"Password","message":" the
error message - example - The Password field is
required"}]}
```

</td><td>No Links</td></tr>
<tr><td>404</td><td>When Role patron is not Found</td><td>No Links</td></tr>
</table>

Code	Description	Links
500	Internal Server Error	No Links

POST `/Authentication/Login` Login to account

Sample request:

```
POST /Authentication/Login
{
     email: user@example.com,
     password: string
}
```

Responses

Code	Description	Links
200	Return the User’s Token when login sucessfully *Content* **text/plain** **application/json** **text/json**	No Links
401	When user’s credentials are Invalid	No Links
500	Internal Server Error	No Links

GET `/Author/{AuthorId}` Get Author’s Details by Id

Sample request:

```
GET /Author/7A0A07DF-C8AE-4407-BD73-9E737D9FBF15
```

Parameters

Type	Name	Description	Schema
path	**AuthorId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Author’s Detailts *Content* **text/plain** **application/json** **text/json**	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin,Librarian or Patron	No Links
404	When Author Is Not Found	No Links
500	Internal Server Error	No Links

PUT `/Author/{AuthorId}` Update author by Id

Sample request:

```
PUT /Author/7A0A07DF-C8AE-4407-BD73-9E737D9FBF15
{
name: authorName,
dateOfBirth: 2023-06-16T16:21:38.124Z
}
```

Parameters

Type	Name	Description	Schema
path	**AuthorId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when Author is updated	No Links

Code	Description	Links
400	When AuthorId in the route does not match the one in the body or DateOfBirth is in the future *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"DateOfBirth","message":"Date Of Birth must be less than now"},{"fieldName":"Id","message":"Bad Id"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"DateOfBirth","message":"Date Of Birth must be less than now"},{"fieldName":"Id","message":"Bad Id"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"DateOfBirth","message":"Date Of Birth must be less than now"},{"fieldName":"Id","message":"Bad Id"}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Librarian	No Links
404	When Author is Not Found	No Links
500	Internal Server Error	No Links

DELETE `/Author/{AuthorId}` Remove Author By Id

Sample request:

```
DELETE /Author/3D9402DC-A780-431D-884D-4EE7DDF73FEC
```

Parameters

Type	Name	Description	Schema
path	**AuthorId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Success when Author is deleted	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Librarian	No Links
404	When Author is Not Found	No Links
500	Internal Server Error	No Links

POST `/Author` Create new author

Sample request:

```
POST /Author
{
name: authorName,
dateOfBirth: 2023-06-16T16:21:38.124Z
}
```

Responses

Code	Description	Links
201	Returns the newly created Author and it’s route in the header:location *Content* **text/plain** **application/json** **text/json**	No Links

Code	Description	Links
400	when DateOfBirth is in the future *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"DateOfBirth","message":"'DateOfBirth' must be less than now"},{"fieldName":"Name","message":"The Name field is required."}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"DateOfBirth","message":"'DateOfBirth' must be less than now"},{"fieldName":"Name","message":"The Name field is required."}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"DateOfBirth","message":"'DateOfBirth' must be less than now"},{"fieldName":"Name","message":"The Name field is required."}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Librarian	No Links
500	Internal Server Error	No Links

GET /Book Search Books with optional Params To get the paginated Books

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET /Book?PageNumber=0&Count=5
{
```

```
    }
sample request2:
```

```
GET /Book?PageNumber=1&Count=7
{
   Title : "test"
}
```

Parameters

Type	Name	Description	Schema
query	**AuthorName** *optional*		string
query	**PageNumber** *optional*		integer (int32)
query	**Title** *optional*		string
query	**Genre** *optional*		string
query	**Count** *optional*		integer (int32)

Responses

Code	Description	Links
200	List of Books with pagination *Content* **text/plain** *Example* `{"books":[{"id":"dcd8b4a7-0ddb-4318-a25b-d8b4b009df43","title":"titleTest","genre":"genre","publishedDate":"2023-02-24T08:26:49.7660902+00:00"},{"id":"2d8b64e1-8164-49af-a9a5-a82e7e406052","title":"titleTest1","genre":"genre1","publishedDate":"2023-02-14T08:26:49.7660974+00:00"}],"pages":2}` **application/json** *Example* `{"books":[{"id":"dcd8b4a7-0ddb-4318-a25b-d8b4b009df43","title":"titleTest","genre":"genre","publishedDate":"2023-02-24T08:26:49.7660902+00:00"},{"id":"2d8b64e1-8164-49af-a9a5-a82e7e406052","title":"titleTest1","genre":"genre1","publishedDate":"2023-02-14T08:26:49.7660974+00:00"}],"pages":2}` **text/json** *Example* `{"books":[{"id":"dcd8b4a7-0ddb-4318-a25b-d8b4b009df43","title":"titleTest","genre":"genre","publishedDate":"2023-02-24T08:26:49.7660902+00:00"},{"id":"2d8b64e1-8164-49af-a9a5-a82e7e406052","title":"titleTest1","genre":"genre1","publishedDate":"2023-02-14T08:26:49.7660974+00:00"}],"pages":2}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin,Librarian or Patron	No Links
500	Internal Server Error	No Links

POST /Book Create New Book

Sample request:

```
POST /Book
{
    title: string,
    genre: string,
    publishedDate: "2023-06-16T09:09:56.356Z",
    isAvailable: true
}
```

Responses

Code	Description	Links
201	Returns the newly created Book and it’s route in the header:location *Content* **text/plain** **application/json** **text/json**	No Links
400	when PublishedDate is in the future *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"PublishedDate","message":"Published Date' must be less than now"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"PublishedDate","message":"Published Date' must be less than now"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"PublishedDate","message":"Published Date' must be less than now"}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Librarian	No Links
500	Internal Server Error	No Links

GET `/Book/{BookId}` Get a specific book Details With the Authors by BookId

Sample request:

```
GET /Book/02AA22F4-66E0-45A2-9735-0DB690147662
```

Parameters

Type	Name	Description	Schema
path	**BookId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Book Detailts With Authors *Content* **text/plain** **application/json** **text/json**	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin,Librarian or Patron	No Links
404	When Book Is Not Found	No Links
500	Internal Server Error	No Links

PUT `/Book/{BookId}` Update Book by Id

Sample request:

```
PUT /Book/02AA22F4-66E0-45A2-9735-0DB690147662
{
    id: 02AA22F4-66E0-45A2-9735-0DB690147662,
    title: string,
    genre: string,
    publishedDate: 2023-06-16T15:06:59.064Z,
    isAvailable: true
}
```

Parameters

Type	Name	Description	Schema
path	**BookId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when book is updated *Content* **text/plain** **application/json** **text/json**	No Links
400	When BookId in the route does not match the one in the body or PublishedDate is in the future *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"PublishedDate","message":"Published Date' must be less than now"},{"fieldName":"Id","message":"Bad Id"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"PublishedDate","message":"Published Date' must be less than now"},{"fieldName":"Id","message":"Bad Id"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"PublishedDate","message":"Published Date' must be less than now"},{"fieldName":"Id","message":"Bad Id"}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Librarian	No Links
404	When Book is Not Found	No Links
500	Internal Server Error	No Links

DELETE `/Book/{BookId}` Remove Book By Id

Sample request:

```
DELETE /Book/02AA22F4-66E0-45A2-9735-0DB690147662
```

Parameters

Type	Name	Description	Schema
path	**BookId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Success when book is deleted *Content* **text/plain** **application/json** **text/json**	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Librarian	No Links
404	When Book is Not Found	No Links
500	Internal Server Error	No Links

GET `/Book/{BookId}/Reviews` Get Book's reviews by BookId

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET Book/02AA22F4-66E0-45A2-9735-0DB690147662/Reviews?PageNumber=0&Count=5
```

Parameters

Type	Name	Description	Schema
path	**BookId** *required*		string (uuid)
query	**PageNumber** *optional*		integer (int32)
query	**Count** *optional*		integer (int32)

Responses

Code	Description	Links

200	List of BookReviews with pagination
	Content
	text/plain
	Example
	`{"reviews":[{"user":{"id":"bb5cc5ac-8953-444c-8d5d-46f54ff47e1c","userName":"userName"},"bookId":"e704b2b9-bd7e-4ba5-8df6-6e456769cbfc","userId":"bb5cc5ac-8953-444c-8d5d-46f54ff47e1c","rate":4,"review":"stringTest"},{"user":{"id":"0d633364-7c26-4e61-9fee-a50386af61f4","userName":"userName"},"bookId":"f7211769-bbef-4a1c-a98a-d31cfd80c5dc","userId":"0d633364-7c26-4e61-9fee-a50386af61f4","rate":4,"review":"stringTest2"}],"pages":2}`
	application/json
	Example
	`{"reviews":[{"user":{"id":"bb5cc5ac-8953-444c-8d5d-46f54ff47e1c","userName":"userName"},"bookId":"e704b2b9-bd7e-4ba5-8df6-6e456769cbfc","userId":"bb5cc5ac-8953-444c-8d5d-46f54ff47e1c","rate":4,"review":"stringTest"},{"user":{"id":"0d633364-7c26-4e61-9fee-a50386af61f4","userName":"userName"},"bookId":"f7211769-bbef-4a1c-a98a-d31cfd80c5dc","userId":"0d633364-7c26-4e61-9fee-a50386af61f4","rate":4,"review":"stringTest2"}],"pages":2}`
	text/json
	Example
	`{"reviews":[{"user":{"id":"bb5cc5ac-8953-444c-8d5d-46f54ff47e1c","userName":"userName"},"bookId":"e704b2b9-bd7e-4ba5-8df6-6e456769cbfc","userId":"bb5cc5ac-8953-444c-8d5d-46f54ff47e1c","rate":4,"review":"stringTest"},{"user":{"id":"0d633364-7c26-4e61-9fee-a50386af61f4","userName":"userName"},"bookId":"f7211769-bbef-4a1c-a98a-d31cfd80c5dc","userId":"0d633364-7c26-4e61-9fee-a50386af61f4","rate":4,"review":"stringTest2"}],"pages`
		No Links

Code	Description	Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
404	When Book is Not Found	No Links
500	Internal Server Error	No Links

POST `/Book/{BookId}/Review` add a review on a book by BookId

Sample request:

```
POST Book/02AA22F4-66E0-45A2-9735-0DB690147662/Review
{
  rate: 5,
  review: feedback, good
}
```

Parameters

Type	Name	Description	Schema
path	**BookId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns the newly created Review *Content* **text/plain** **application/json** **text/json**	No Links

Code	Description	Links
400	When Book is Not Returned or already reviewd or bad rate value *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"BookReviewNotAllowed","message":"Not allowed to review book, you need to borrow and return the book"},{"fieldName":"BookIsReviewed","message":"you Already have Reviewed this book"},{"fieldName":"Rate","message":"The field Rate must be between 1 and 5."}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"BookReviewNotAllowed","message":"Not allowed to review book, you need to borrow and return the book"},{"fieldName":"BookIsReviewed","message":"you Already have Reviewed this book"},{"fieldName":"Rate","message":"The field Rate must be between 1 and 5."}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"BookReviewNotAllowed","message":"Not allowed to review book, you need to borrow and return the book"},{"fieldName":"BookIsReviewed","message":"you Already have Reviewed this book"},{"fieldName":"Rate","message":"The field Rate must be between 1 and 5."}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
404	When Book is Not Found	No Links
500	Internal Server Error	No Links

GET `/Book/Transactions/{TransactionId}` Get Book Transaction By Transaction Id

Sample request:

```
GET /Book/Transactions/455AE5E6-D447-4A66-AB59-5E48B3E69E03
```

Parameters

Type	Name	Description	Schema
path	**TransactionId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns BookTransaction Details *Content* **text/plain** **application/json** **text/json**	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin,Librarian or Patron	No Links
404	When BookTransaction Is Not Found	No Links
500	Internal Server Error	No Links

POST `/Book/{BookId}/Reserve` Reserve book by BookId

Sample request:

```
POST Book/02AA22F4-66E0-45A2-9735-0DB690147662/Reserve
```

Parameters

Type	Name	Description	Schema
path	**BookId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when book is Reserved	No Links
400	When Book is Not not available *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"the Book test is not avaialble at the moment"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"the Book test is not avaialble at the moment"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"the Book test is not avaialble at the moment"}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not patron	No Links
404	When Book is Not Found	No Links
500	Internal Server Error	No Links

PUT `/Book/Transactions/{TransactionId}/Borrow` Checkout/Borrow a reserved book for patron transactionId

Sample request:

```
PUT Book/Transactions/BBAB7E04-92FD-42D0-BB67-050379BE5585/Borrow
```

Parameters

Type	Name	Description	Schema
path	**TransactionId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when book is Borrowed	No Links
400	When Book is Already Borrowed *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"book is already Borrowed"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"book is already Borrowed"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"book is already Borrowed"}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not librarian	No Links
404	When Transaction is Not Found	No Links
500	Internal Server Error	No Links

PUT `/Book/Transactions/{TransactionId}/Return` Return book by transactionId

Sample request:

```
PUT Book/Transactions/BBAB7E04-92FD-42D0-BB67-050379BE5585/Return
```

Parameters

Type	Name	Description	Schema
path	**TransactionId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when book is Returned	No Links
401	Authorization has been denied for this request	No Links
403	When user is not librarian	No Links
404	When Transaction is Not Found	No Links
500	Internal Server Error	No Links

GET `/Book/Due-Date-Transactions` get all transactions for the borrowed books with due dates with pagination

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET /Book/Due-Date-Transactions?PageNumber=0&Count=5
```

Parameters

Type	Name	Description	Schema
query	**PageNumber** *optional*		integer (int32)
query	**Count** *optional*		integer (int32)

Responses

Code	Description	Links
200	List of Transactions with pagination *Content* **text/plain** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"title","genre":"genre","publishedDate":"2023-01-25T08:26:49.7877627Z"},"dueDate":"2023-06-19T08:26:49.7877602Z","status":1},{"book":{"isAvailable":false,"title":"title1","genre":"genre1","publishedDate":"2022-08-28T08:26:49.78781Z"},"dueDate":"2023-06-14T08:26:49.7878082Z","status":1}],"pages":2}` **application/json** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"title","genre":"genre","publishedDate":"2023-01-25T08:26:49.7877627Z"},"dueDate":"2023-06-19T08:26:49.7877602Z","status":1},{"book":{"isAvailable":false,"title":"title1","genre":"genre1","publishedDate":"2022-08-28T08:26:49.78781Z"},"dueDate":"2023-06-14T08:26:49.7878082Z","status":1}],"pages":2}` **text/json** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"title","genre":"genre","publishedDate":"2023-01-25T08:26:49.7877627Z"},"dueDate":"2023-06-19T08:26:49.7877602Z","status":1},{"book":{"isAvailable":false,"title":"title1","genre":"genre1","publishedDate":"2022-08-28T08:26:49.78781Z"},"dueDate":"2023-06-14T08:26:49.7878082Z","status":1}],"pages":2}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not librarian	No Links
500	Internal Server Error	No Links

GET `/Librarian/{UserId}` Get Librarian Profile Details by Id

Sample request:

```
GET /librarian/A63B4C43-AA43-4E90-A97E-FB8DABC3162D
```

Parameters

Type	Name	Description	Schema
path	**UserId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Librarian’s Profile *Content* **text/plain** **application/json** **text/json**	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin, or the managed user is not librarian	No Links
404	When Librarian is Not Found	No Links
500	Internal Server Error	No Links

PUT `/Librarian/{UserId}` update Librarian Profile

Sample request:

```
PUT /librarian/A63B4C43-AA43-4E90-A97E-FB8DABC3162D
{
    id: A63B4C43-AA43-4E90-A97E-FB8DABC3162D,
    userName: librarianUserName,
    email: test@test.com,
    phoneNumber: 059248215
}
```

Parameters

Type	Name	Description	Schema
path	**UserId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when Librarian is updated	No Links

Code	Description	Links
400	When the new Librarian info already exists or the LibrarianId in the route does not match the one in the body *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}`	No Links
401	Authorization has been denied for this request	No Links

Code	Description	Links
403	When user is not Admin, or the managed user is not librarian	No Links
404	When Librarian is Not Found	No Links
500	Internal Server Error	No Links

DELETE `/Librarian/{UserId}` Delete Librarian account

Sample request:

```
DELETE /librarian/A63B4C43-AA43-4E90-A97E-FB8DABC3162D
```

Parameters

Type	Name	Description	Schema
path	**UserId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when Librarian is Deleted	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin, or the managed user is not librarian	No Links
404	When Librarian is Not Found	No Links
500	Internal Server Error	No Links

POST `/Librarian` Create New Librarian account

Sample request:

```
 POST /librarian
 {
     "userName": "string",
     "email": "user@example.com",
     "phoneNumber": "string",
     "password": "string"
  }
Sample request2:
```

```
POST /librarian
{
     userName: LibrarianNameTest,
     email: LibrarianEmail@gmail.com,
      password: testPassword
  }
```

Responses

Code	Description	Links
201	Returns the newly created Librarian and it's route in the header:location *Content* **text/plain** **application/json** **text/json**	No Links

Code	Description	Links
400	When user info (Email,UserName,Phone) Already Exists *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"}]}`	No Links
401	Authorization has been denied for this request	No Links

Code	Description	Links
403	When user is not Admin	No Links
404	When Role librarian is not Found	No Links
500	Internal Server Error	No Links

GET `/Notification` Returns user’s notification with pagination

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET /Notification?PageNumber=0&Count=5
```

Parameters

Type	Name	Description	Schema
query	**PageNumber** *optional*		integer (int32)
query	**Count** *optional*		integer (int32)

Responses

Code	Description	Links
200	List of Notifications with pagination *Content* **text/plain** *Example* `{"notifications":[{"id":"c84998bd-d0ba-4e04-a24a-62904ef6745d","message":"Notification Message","isRead":false},{"id":"acd7e32c-96ed-44a5-8c85-b2fcfead9ca6","message":"Notification Message2","isRead":true}],"pages":2}` **application/json** *Example* `{"notifications":[{"id":"c84998bd-d0ba-4e04-a24a-62904ef6745d","message":"Notification Message","isRead":false},{"id":"acd7e32c-96ed-44a5-8c85-b2fcfead9ca6","message":"Notification Message2","isRead":true}],"pages":2}` **text/json** *Example* `{"notifications":[{"id":"c84998bd-d0ba-4e04-a24a-62904ef6745d","message":"Notification Message","isRead":false},{"id":"acd7e32c-96ed-44a5-8c85-b2fcfead9ca6","message":"Notification Message2","isRead":true}],"pages":2}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
500	Internal Server Error	No Links

POST `/Notification/ReservedBooks` Notify Patrons About their reserved books(Push - database - Email notification)

Sample request:

```
POST /Notification/ReservedBooks
```

Responses

Code	Description	Links
200	Succes when Patrons are Notified	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Librarian	No Links
500	Internal Server Error	No Links

POST `/Notification/DueDates` Notify Patrons About their DueDate books(Push - database - Email notification)

Sample request:

```
POST /Notification/DueDates
```

Responses

Code	Description	Links
200	Succes when Patrons are Notified	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Librarian	No Links
500	Internal Server Error	No Links

GET `/Patron/{UserId}` Get patron user by admin or librarian by userId

Sample request:

```
GET Patron/CC0D402A-D4AA-40CE-B675-B4F67EC4B650
```

Parameters

Type	Name	Description	Schema
path	**UserId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Patron’s Profile *Content* **text/plain** **application/json** **text/json**	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin or Librarian, or the managed user is not Patron	No Links
500	Internal Server Error	No Links

PUT `/Patron/{UserId}` Updates Patron’s Profile

Sample request:

```
PUT /Patron/A63B4C43-AA43-4E90-A97E-FB8DABC3162D
{
    id: A63B4C43-AA43-4E90-A97E-FB8DABC3162D,
    userName: PatronUserName,
    email: test@test.com,
    phoneNumber: 0592482135
}
```

Parameters

Type	Name	Description	Schema
path	**UserId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when Patron is updated	No Links

Code	Description	Links
400	When the new Patron info already exists or the PatronId in the route does not match the one in the body *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}`	No Links
401	Authorization has been denied for this request	No Links

Code	Description	Links
403	When user is not Admin or Librarian, or the managed user is not Patron	No Links
500	Internal Server Error	No Links

GET `/Patron/{UserId}/Borrowing-History` Get patron’s Borrowing History

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET /Patron/A63B4C43-AA43-4E90-A97E-FB8DABC3162D/Borrowing-
History?PageNumber=0&Count=5
```

Parameters

Type	Name	Description	Schema
query	**PageNumber** *optional*		integer (int32)
path	**UserId** *required*		string (uuid)
query	**Count** *optional*		integer (int32)

Responses

Code	Description	Links
200	List of Transactions with pagination *Content* **text/plain** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"test","genre":"genre","publishedDate":"2023-02-24T08:26:49.7976999+00:00"},"dueDate":"2023-06-24T08:26:49.7976957+00:00","status":1},{"book":{"isAvailable":false,"title":"test1","genre":"genre1","publishedDate":"2023-02-14T08:26:49.7977084+00:00"},"dueDate":"2023-06-24T08:26:49.7977016+00:00","status":0}],"pages":2}` **application/json** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"test","genre":"genre","publishedDate":"2023-02-24T08:26:49.7976999+00:00"},"dueDate":"2023-06-24T08:26:49.7976957+00:00","status":1},{"book":{"isAvailable":false,"title":"test1","genre":"genre1","publishedDate":"2023-02-14T08:26:49.7977084+00:00"},"dueDate":"2023-06-24T08:26:49.7977016+00:00","status":0}],"pages":2}` **text/json** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"test","genre":"genre","publishedDate":"2023-02-24T08:26:49.7976999+00:00"},"dueDate":"2023-06-24T08:26:49.7976957+00:00","status":1},{"book":{"isAvailable":false,"title":"test1","genre":"genre1","publishedDate":"2023-02-14T08:26:49.7977084+00:00"},"dueDate":"2023-06-24T08:26:49.7977016+00:00","status":0}],"pages":2}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin or Librarian, or the managed user is not Patron	No Links
500	Internal Server Error	No Links

GET /Patron/RecommendedBooks Get Recommended Books for a patron based in borrowing History

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET /Patron/RecommendedBooks?PageNumber=0&Count=5
```

Parameters

Type	Name	Description	Schema
query	**PageNumber** *optional*		integer (int32)
query	**Count** *optional*		integer (int32)

Responses

Code	Description	Links
200	List of Recommended Books with pagination *Content* **text/plain** *Example* `{"recommendedBooks":[{"id":"1c6f0e13-7cc6-4de9-8e4d-82f06f885107","authors":[{"id":"22ae2b2a-580c-416a-a98b-ef0a000935e8","name":"name","dateOfBirth":"2003-06-24T08:26:49.7993715Z"},{"id":"fdc82c24-a6c2-4c68-a625-31331ee7c20e","name":"name1","dateOfBirth":"1998-06-24T08:26:49.7993748Z"}],"title":"title","genre":"genre","publishedDate":"2022-08-08T08:26:49.7993517Z"},{"id":"724e3278-5c70-4516-a138-390425d849ec","authors":[{"id":"44d42c11-4f9e-4dff-868e-c5b8ffb2a0fd","name":"name2","dateOfBirth":"1996-06-24T08:26:49.7993785Z"},{"id":"1c6d52dc-f514-44af-a104-b5e7de8dd1c7","name":"name3","dateOfBirth":"1993-06-24T08:26:49.7993799Z"}],"title":"title2","genre":"genre2","publishedDate":"2023-01-25T08:26:49.7993772Z"}],"pages":2}` **application/json** *Example* `{"recommendedBooks":[{"id":"1c6f0e13-7cc6-4de9-8e4d-82f06f885107","authors":[{"id":"22ae2b2a-580c-416a-a98b-ef0a000935e8","name":"name","dateOfBirth":"2003-06-24T08:26:49.7993715Z"},{"id":"fdc82c24-a6c2-4c68-a625-31331ee7c20e","name":"name1","dateOfBirth":"1998-06-24T08:26:49.7993748Z"}],"title":"title","genre":"genre","publishedDate":"2022-08-08T08:26:49.7993517Z"},{"id":"724e3278-5c70-4516-a138-390425d849ec","authors":[{"id":"44d42c11-4f9e-4dff-868e-c5b8ffb2a0fd","name":"name2","dateOfBirth":"1996-06-24T08:26:49.7993785Z"},{"id":"1c6d52dc-f514-44af-a104-b5e7de8dd1c7","name":"name3","dateOfBirth":"1993-06-24T08:26:49.7993799Z"}],"title":"title2","genre":"genre2","publishedDate":"2023-01-25T08:26:49.7993772Z"}],"pages":2}` **text/json**	No Links

Code	Description	Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
500	Internal Server Error	No Links

GET `/ReadingList` Get ReadingLists

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET /ReadingList?PageNumber=0&Count=5
```

Parameters

Type	Name	Description	Schema
query	**PageNumber** *optional*		integer (int32)
query	**Count** *optional*		integer (int32)

Responses

Code	Description	Links
200	List of ReadingLists with pagination *Content* **text/plain** *Example* `{"readingLists":[{"id":"06d47ee0-67c6-4261-b1b1-c47112f03910","name":"readingListName1","private":false},{"id":"852448ca-465f-4193-a61c-e21c93420a2c","name":"readingListName2","private":true}],"pages":2}` **application/json** *Example* `{"readingLists":[{"id":"06d47ee0-67c6-4261-b1b1-c47112f03910","name":"readingListName1","private":false},{"id":"852448ca-465f-4193-a61c-e21c93420a2c","name":"readingListName2","private":true}],"pages":2}` **text/json** *Example* `{"readingLists":[{"id":"06d47ee0-67c6-4261-b1b1-c47112f03910","name":"readingListName1","private":false},{"id":"852448ca-465f-4193-a61c-e21c93420a2c","name":"readingListName2","private":true}],"pages":2}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
500	Internal Server Error	No Links

POST `/ReadingList` Create new ReadingList

Sample request:

```
POST /ReadingList
 {
    name: string,
    private: true
 }
```

Responses

Code	**Description**	**Links**
201	Returns the newly created ReadingList and it’s route in the header:location *Content* **text/plain** **application/json** **text/json**	No Links
400	when Creating new ReadingList without private field *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":"The Name field is required."},{"fieldName":"Private","message":"The Private field is required."}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":"The Name field is required."},{"fieldName":"Private","message":"The Private field is required."}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":"The Name field is required."},{"fieldName":"Private","message":"The Private field is required."}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
500	Internal Server Error	No Links

GET `/ReadingList/{ReadingListId}/Books` Get ReadingList with it’s books with pagination

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET /ReadingList/D9BBC76C-AFBB-4AB0-BE01-ADCCD95C38D8/Books?PageNumber=0&Count=5
```

Parameters

Type	Name	Description	Schema
query	**PageNumber** *optional*		integer (int32)
query	**Count** *optional*		integer (int32)
path	**ReadingListId** *required*		string (uuid)

Responses

Code	Description	Links
200	List of readlingList with it’s Books - pagination *Content* **text/plain** *Example* `{"readingList":{"id":"26374608-df16-4e68-9f65-4ae842ff18e7","books":[{"title":"title","genre":"genre","publishedDate":"2002-06-24T08:26:49.8054719Z"},{"title":"title1","genre":"genre1","publishedDate":"2013-06-24T08:26:49.8054757Z"},{"title":"title2","genre":"genre2","publishedDate":"2022-06-24T08:26:49.8054759Z"}],"name":"readingListName","private":false},"pages":2}` **application/json** *Example* `{"readingList":{"id":"26374608-df16-4e68-9f65-4ae842ff18e7","books":[{"title":"title","genre":"genre","publishedDate":"2002-06-24T08:26:49.8054719Z"},{"title":"title1","genre":"genre1","publishedDate":"2013-06-24T08:26:49.8054757Z"},{"title":"title2","genre":"genre2","publishedDate":"2022-06-24T08:26:49.8054759Z"}],"name":"readingListName","private":false},"pages":2}` **text/json** *Example* `{"readingList":{"id":"26374608-df16-4e68-9f65-4ae842ff18e7","books":[{"title":"title","genre":"genre","publishedDate":"2002-06-24T08:26:49.8054719Z"},{"title":"title1","genre":"genre1","publishedDate":"2013-06-24T08:26:49.8054757Z"},{"title":"title2","genre":"genre2","publishedDate":"2022-06-24T08:26:49.8054759Z"}],"name":"readingListName","private":false},"pages":2}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
500	Internal Server Error	No Links

PUT `/ReadingList/{ReadingListId}` Update ReadingList by Id

Sample request:

```
PUT /ReadingList/D9BBC76C-AFBB-4AB0-BE01-ADCCD95C38D8
{
    id: D9BBC76C-AFBB-4AB0-BE01-ADCCD95C38D8,
    name: string,
    private: true
}
```

Parameters

Type	Name	Description	Schema
path	**ReadingListId** *required*		string (uuid)

Responses

Code	Description	Links
200	Success when ReadingList is updated	No Links

Code	Description	Links
400	When ReadingList Id in the route does not match the one in the body or request without name *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":"The Name field is required."},{"fieldName":"Id","message":"bad Id"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":"The Name field is required."},{"fieldName":"Id","message":"bad Id"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":"The Name field is required."},{"fieldName":"Id","message":"bad Id"}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
404	When ReadingList is Not Found	No Links
500	Internal Server Error	No Links

DELETE `/ReadingList/{ReadingListId}` Delete readingList by Id

Sample request:

```
DELETE /ReadingList/D9BBC76C-AFBB-4AB0-BE01-ADCCD95C38D8
```

Parameters

Type	Name	Description	Schema
path	**ReadingListId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Success when ReadingList is deleted	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links
404	When ReadingList is Not Found	No Links
500	Internal Server Error	No Links

POST `/ReadingList/{ReadingListId}/Books/{BookId}` Add book to readingList

Sample request:

```
POST /ReadingList/D9BBC76C-AFBB-4AB0-BE01-ADCCD95C38D8/Books/BBAB7E04-92FD-42D0-BB67-
050379BE5585
```

Parameters

Type	Name	Description	Schema
path	**BookId** *required*		string (uuid)
path	**ReadingListId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Success when Book is added to the reading List	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links

Code	Description	Links
404	When ReadingList or Book Not Found *Content* **text/plain** *Example* `{"status":404,"errors":[{"fieldName":"Book","message":"404 Book NotFound"},{"fieldName":"ReadingList","message":"404 ReadingList NotFound"}]}` **application/json** *Example* `{"status":404,"errors":[{"fieldName":"Book","message":"404 Book NotFound"},{"fieldName":"ReadingList","message":"404 ReadingList NotFound"}]}` **text/json** *Example* `{"status":404,"errors":[{"fieldName":"Book","message":"404 Book NotFound"},{"fieldName":"ReadingList","message":"404 ReadingList NotFound"}]}`	No Links
409	when ReadingList already has The added Book	No Links
500	Internal Server Error	No Links

DELETE `/ReadingList/{ReadingListId}/Books/{BookId}` Remove book from readingList

Sample request:

```
DELETE /ReadingList/D9BBC76C-AFBB-4AB0-BE01-ADCCD95C38D8/Books/BBAB7E04-92FD-42D0-
BB67-050379BE5585
```

Parameters

Type	Name	Description	Schema
path	**BookId** *required*		string (uuid)

Type	Name	Description	Schema
path	**ReadingListId** *required*		string (uuid)

Responses

Code	Description	Links
200	Returns Success when Book is removed from the reading List	No Links
400	When ReadingList Does Not Contain the Book *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"ReadingList Does Not Contain The Book"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"ReadingList Does Not Contain The Book"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"Book","message":"ReadingList Does Not Contain The Book"}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Patron	No Links

Code	Description	Links

404	When ReadingList or Book Not Found
	Content
	text/plain
	Example
	`{"status":404,"errors":[{"fieldName":"Book","message":"`
	`404 Book`
	`NotFound"},{"fieldName":"ReadingList","message":"404`
	`ReadingList NotFound"}]}`
	application/json
	Example
	`{"status":404,"errors":[{"fieldName":"Book","message":"`
	`404 Book`
	`NotFound"},{"fieldName":"ReadingList","message":"404`
	`ReadingList NotFound"}]}`
	text/json
	Example
	`{"status":404,"errors":[{"fieldName":"Book","message":"`
	`404 Book`
	`NotFound"},{"fieldName":"ReadingList","message":"404`
	`ReadingList NotFound"}]}`
		No Links

500	Internal Server Error	No Links

POST `/Role` Create new role

Sample request:

```
POST /Role
{
    name = roleName
}
```

Responses

Code	Description	Links
200	Success and returns the newly created role *Content* **text/plain** **application/json** **text/json**	No Links
400	when Role Name Already Exists *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":" The Role With Name patron Already Exists"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":" The Role With Name patron Already Exists"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"Name","message":" The Role With Name patron Already Exists"}]}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin	No Links
500	Internal Server Error	No Links

GET `/User` Rerurns profile

Sample request:

```
GET /User
```

Responses

Code	Description	Links
200	Returns User’s Profile *Content* **text/plain** **application/json** **text/json**	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin, Librarian or Patron	No Links
404	When User is Not Found	No Links
500	Internal Server Error	No Links

PUT `/User` Update profile

Sample request:

```
PUT /User
{
  id: 3fa85f64-5717-4562-b3fc-2c963f66afa6,
  userName: string,
  email: user@example.com,
  phoneNumber: string
}
```

Responses

Code	Description	Links
400	When the new User info already exists or the User Id in the route does not match the one in the body *Content* **text/plain** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}` **application/json** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}` **text/json** *Example* `{"status":400,"errors":[{"fieldName":"PhoneNumber","message":"the error message - example -The PhoneNumber field is not a valid phone number."},{"fieldName":"Email","message":" the error message - example - The Email is used or required"},{"fieldName":"UserName","message":" the error message - example - The UserName is used or required"},{"fieldName":"Password","message":" the error message - example - The Password field is required"},{"fieldName":"Id","message":"Bad Id"}]}`	No Links
401	Authorization has been denied for this request	No Links

Code	Description	Links
403	When user is not Admin, Librarian or Patron	No Links
500	Internal Server Error	No Links

GET `/User/Borrowing-History` Get Borrowing History

Route Defualts:

```
{
Defualt:
    PageNumber=0,
    Count=5
```

```
    Max:
        Count=10
    }
Sample request:
```

```
GET /User/Borrowing-History?PageNumber=0&Count=5
```

Parameters

Type	Name	Description	Schema
query	**PageNumber** *optional*		integer (int32)
query	**Count** *optional*		integer (int32)

Responses

Code	Description	Links
200	List of Borrowing History (Transactions) with pagination *Content* **text/plain** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"title","genre":"genre","publishedDate":"2023-01-25T08:26:49.8167957Z"},"dueDate":"2023-06-19T08:26:49.8167922Z","status":1},{"book":{"isAvailable":false,"title":"title1","genre":"genre1","publishedDate":"2022-08-28T08:26:49.8167963Z"},"dueDate":"2023-06-14T08:26:49.816796Z","status":1}],"pages":2}` **application/json** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"title","genre":"genre","publishedDate":"2023-01-25T08:26:49.8167957Z"},"dueDate":"2023-06-19T08:26:49.8167922Z","status":1},{"book":{"isAvailable":false,"title":"title1","genre":"genre1","publishedDate":"2022-08-28T08:26:49.8167963Z"},"dueDate":"2023-06-14T08:26:49.816796Z","status":1}],"pages":2}` **text/json** *Example* `{"transactions":[{"book":{"isAvailable":false,"title":"title","genre":"genre","publishedDate":"2023-01-25T08:26:49.8167957Z"},"dueDate":"2023-06-19T08:26:49.8167922Z","status":1},{"book":{"isAvailable":false,"title":"title1","genre":"genre1","publishedDate":"2022-08-28T08:26:49.8167963Z"},"dueDate":"2023-06-14T08:26:49.816796Z","status":1}],"pages":2}`	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin, Librarian or Patron	No Links
500	Internal Server Error	No Links

POST `/User/{UserId}/Role/{RoleId}` Assign Role To User by admin

Sample request:

```
POST /User/7841A4A4-4E6A-4D5A-AADD-AA162EA1CABF/Role/BBAB7E04-92FD-42D0-BB67-
050379BE5784
```

Parameters

Type	Name	Description	Schema
path	**UserId** *required*		string (uuid)
path	**RoleId** *required*		string (uuid)

Responses

Code	Description	Links
200	when add role to user successfully	No Links
401	Authorization has been denied for this request	No Links
403	When user is not Admin	No Links
404	When User Or Role Not Found	No Links
409	when User already has The Assigned Role	No Links
500	Internal Server Error	No Links

GET `/WeatherForecast`

Responses

Code	Description	Links
200	Success *Content* **text/plain** **application/json** **text/json**	No Links

GET `/WeatherForecast/aa`

Responses

Code	Description	Links
200	Success	No Links

Components

Schemas

AddBookToReadingListErrorResponseExample

AuthorDto

Properties

Name	Description	Schema
name *required*	**Maximum Length**: 256	string
dateOfBirth *optional*	*nullable*	string (date-time)
id *optional*		string (uuid)

BookStatus

BookTransactionWithStatusAndIdDto

Properties

Name	Description	Schema
id *optional*		string (uuid)
book *optional*		CreateBookDto
dueDate *optional*	*nullable*	string (date-time)
status *optional*		BookStatus

BookWithAuthorsDto

Properties

Name	Description	Schema
title *required*	**Maximum Length**: 256	string
genre *optional*	*nullable*	string
publishedDate *required*		string (date-time)
id *optional*		string (uuid)

Name	Description	Schema
authors *optional*	*nullable*	< AuthorDto > array

CheckOutBookErrorResponseExample

CreateAuthorDto

Properties

Name	Description	Schema
name *required*	**Maximum Length**: 256	string
dateOfBirth *optional*	*nullable*	string (date-time)

CreateBookDto

Properties

Name	Description	Schema
title *required*	**Maximum Length**: 256	string
genre *optional*	*nullable*	string
publishedDate *required*		string (date-time)
isAvailable *required*		boolean

CreateBookErrorResponseExample

CreateBookReviewDto

Properties

Name	Description	Schema
rate *required*	**Maximum**: 5 **Minimum**: 1	integer (int32)
review *optional*	*nullable*	string

CreateBookReviewErrorResponseExample

CreateReadingListDto

Properties

Name	Description	Schema
name *required*	**Minimum Length**: 1	string
private *optional*		boolean

CreateReadingListErrorResponseExample

CreateRoleDto

Properties

Name	Description	Schema
name *required*	**Maximum Length**: 256	string

CreateRoleErrorResponseExample

CreateUserDto

Properties

Name	Description	Schema
userName *required*	**Maximum Length**: 256	string
email *required*	**Maximum Length**: 256	string (email)
phoneNumber *optional*	*nullable*	string (tel)
password *required*		string

DueDateDto

Properties

Name	Description	Schema
dueDate *required*	**Minimum Length**: 1	string (date-time)

ErrorModel

Properties

Name	Description	Schema
fieldName *optional*	*nullable*	string

Name	Description	Schema
message *optional*	*nullable*	string

ErrorResponse

Properties

Name	Description	Schema
status *optional*		HttpStatusCode
errors *optional*	*nullable*	< ErrorModel > array

GetBookPaginationOkResultExample

GetBookReviewsPaginationOkResultExample

GetNotificationsPaginationOkResultExample

GetPatronBorrowingHistoryOkResponseExample

GetPatronRecommendedBooksPaginationOkResultExample

GetReadingListDto

Properties

Name	Description	Schema
name *required*		string
private *optional*		boolean
id *optional*		string (uuid)

GetReadingListPaginationOkResultExample

GetReadingListWithBooksPagonationOkResultExample

GetTransactionsPaginationOkResultExample

HttpStatusCode

LoginUserDto

Properties

Name	Description	Schema
email *required*	**Maximum Length**: 256	string (email)
password *required*		string

RegisterErrorResponseExample

RegisterLibraraianErrorResponseExample

RemoveBookFromReadingListBadRequestErrorResponseExample

RemoveBookFromReadingListNotFoundErrorResponseExample

ReserveBookErrorResponseExample

RoleDto

Properties

Name	Description	Schema
name *required*	**Maximum Length**: 256	string
id *optional*		string (uuid)

UpdateAuthorErrorResponseExample

UpdateBookDto

Properties

Name	Description	Schema
title *required*	**Maximum Length**: 256	string
genre *optional*	*nullable*	string
publishedDate *required*		string (date-time)
id *optional*		string (uuid)
isAvailable *optional*	*nullable*	boolean

UpdateBookErrorResponseExample

UpdateLibraraianErrorResponseExample

UpdatePatronErrorResponseExample

UpdateReadingListDto

Properties

Name	Description	Schema
id *optional*		string (uuid)
name *required*	**Maximum Length**: 250	string
private *optional*	*nullable*	boolean

UpdateReadingListErrorResponseExample

UpdateUserDto

Properties

Name	Description	Schema
id *optional*		string (uuid)
userName *optional*	*nullable* **Maximum Length**: 256	string
email *optional*	*nullable* **Maximum Length**: 256	string (email)
phoneNumber *optional*	*nullable*	string (tel)

UpdateUserErrorResponseExample

UserDto

Properties

Name	Description	Schema
userName *required*	**Maximum Length**: 256	string
email *required*	**Maximum Length**: 256	string (email)
phoneNumber *optional*	*nullable*	string (tel)

UserDtoWithId

Properties

Name	Description	Schema
userName *required*	**Maximum Length**: 256	string

Name	Description	Schema
email *required*	**Maximum Length**: 256	string (email)
phoneNumber *optional*	*nullable*	string (tel)
id *optional*		string (uuid)

WeatherForecast

Properties

Name	Description	Schema
id *optional*		string (uuid)
date *optional*		string (date-time)
temperatureC *optional*		integer (int32)
temperatureF *optional*	*read only*	integer (int32)
summary *optional*	*nullable*	string