Portfolio Project: Reading List API Spec

Rachel Pratt

Data Model

Books

Property	Type	Required	Notes
id	Integer		The id of the book. Generated automatically upon creation.
title	String	Yes	Title of the book. Valid values are strings between 1-50 characters in length.
author	String	Yes	Author of the book. Valid values are strings between 1-50 characters in length.
genre	String	Yes	Genre of the book. Valid values are strings between 1-50 characters in length.
self	String		Self-link for the book. Not stored but generated dynamically and returned in GET requests.

Reading Lists

Property	Type	Required	Notes
id	Integer		The id of the reading list. Generated automatically upon creation.
name	String	Yes	Volume of the reading list. Valid values are strings between 1-50 characters in length.
description	String	Yes	Description of reading list. Valid values are strings between 1-100 characters in length.
books	Array		Array of book IDs contained in the reading list. Generated as an empty list upon creation.
user	String		The id of the user who created the reading list. Obtained from "sub" of JWT payload upon creation.
self	String		Self-link for the reading list. Not stored but generated dynamically and returned in GET requests.

Users

Property	Type	Notes
id	Integer	The id of the user. Obtained from "sub" of JWT payload.

Relationship between Books and Reading Lists:
A reading list contains an array of books represented by the book ID. Users can add books to their reading list, remove books from their reading list, and view a list of all of the books contained in any of their reading lists.

How the User Entity is modeled:
When a user registers or logs in for the first time using Auth0, a user object is created and stored in the Users collection. The session JWT's authenticity is verified and the value of the "sub" property in the JWT's payload becomes the user ID. Reading lists are linked to

users. When a reading list is created, its "user" property is set to the user's ID, which is the value of "sub" obtained from the JWT's payload. Because reading lists are protected, operations involving them require user authentication and authorization. This is accomplished by first verifying the JWT, and then comparing the "user" property of the reading list with the "sub" property of the JWT. There is no relationship between books and users, and operations involving books are unprotected.

Create a Book

Allows you to create a new book.

POST /books	**Unprotected**

Request

Path Parameters:
None

Request Body:
Required

Request Body Format:
JSON

Accepted MIME Types:
application/json

Request JSON Attributes

Name	**Description**	**Required?**
title	The title of the book.	Yes
author	The author of the book.	Yes
genre	The genre of the book.	Yes

Request Body Example

```
{
   "title": "Pride and Prejudice",
   "author": "Jane Austen",
   "genre": "romance"
}
```

Response

Response Body Format:
JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	201 Created	
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the book is not created, and 400 status code is returned.

Failure	406 Not Acceptable	If the request Accept header is set to a non-JSON MIME type, the book is not created, and 406 status code is returned.
Failure	415 Unsupported Media Type	If a non-JSON MIME type is sent to the endpoint, the book is not created, and 415 status code is returned.

Response Examples

Success

```
Status: 201 Created
{
   "id": 1234,
   "title": "Pride and Prejudice",
   "author": "Jane Austen",
   "genre": "romance",
   "self": "https://appspot.com/books/1234"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "Missing one or more attributes"
}
```

Failure

```
Status: 406 Not Acceptable
{
   "Error": "The server only sends JSON"
}
```

Failure

```
Status: 415 Unsupported Media Type
{
   "Error": "The server only accepts JSON"
}
```

View all Books

Allows you to view all existing books. Returns 5 books per page and includes a next link to get the next page of results unless there are no more pages of result. Also returns a count of the total number of existing books.

GET /books	**Unprotected**

Request

Path Parameters:
None

Request Body:
None

Response

Response Body Format:
JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	

Response Examples

Success

```
Status: 200 OK
{
   "books": [
      {
         "id": 1234,
         "title": "Pride and Prejudice",
         "author": "Jane Austen",
         "genre": "romance",
         "self": "https://appspot.com/books/1234"
      },
      {
         "id": 2239,
         "title": "The Great Gatsby",
         "author": "F. Scott Fitzgerald",
         "genre": "fiction",
         "self": "https://appspot.com/books/2239"
      },
    {
      "id": 1194,
      "title": "Great Expectations",
      "author": "Charles Dickens",
      "genre": "fiction",
      "self": "https://appspot.com/books/1194"
    },
    {
      "id": 1583,
      "title": "Dracula",
      "author": "Bram Stoker",
      "genre": "horror",
      "self": "https://appspot.com/books/1583"
    },
    {
      "id": 1388,
      "title": "Gone Girl",
      "author": "Gillian Flynn",
      "genre": "thriller",
      "self": "https://appspot.com/books/1388"
    }
  ],
  "count": 6,
  "next": "https://appspot.com/books?limit=5&offset=5"
}
```

View a Book

Allows you to view an existing book.

GET /books/:book_id	**Unprotected**

Request

Path Parameters:

Name	**Description**
book_id	ID of the book

Request Body:
None

Response

Response Body Format:
JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	404 Not Found	No book with this book_id exists

Response Examples

Success

```
Status: 200 OK
{
   "id": 1234,
   "title": "Pride and Prejudice",
   "author": "Jane Austen",
   "genre": "romance",
   "self": "https://appspot.com/books/1234"
}
```

Failure

```
Status: 404 Not Found
{
   "Error": "No book with this book_id exists"
}
```

Delete a Book

Allows you to delete an existing book. If a reading list’s "books" property contains the book, it will be removed.

DELETE /books/:book_id	**Unprotected**

Request

Path Parameters:

Name	Description
book_id	ID of the book

Request Body:
None

Response

No body

Response Body Format:
Success: No body
Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	404 Not Found	No book with this book_id exists

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found
{
  "Error": "No book with this book_id exists"
}
```

Edit a Book with PATCH

Allows you to modify one or more attributes of a book.

PATCH /books/:book_id	**Unprotected**

Request

Path Parameters:

Name	Description
book_id	ID of the book

Request Body:
Required

Request Body Format:
JSON

Accepted MIME types:
application/json

Request JSON Attributes

Name	Description	Required?
title	The name of the book.	No
author	The type of the book.	No
genre	The genre of the book.	No

Request Body Example

```
{
   "title": "Sense and Sensibility"
}
```

Response

Response Body Format:
JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No book with this book_id exists
Failure	400 Bad Request	If no attributes are provided or more than 3 attributes are provided, the book is not updated, and 400 status code is returned.

Failure	406 Not Acceptable	If the request Accept header is set to a non-JSON MIME type, the book is not updated, and 406 status code is returned.
Failure	415 Unsupported Media Type	If a non-JSON MIME type is sent to the endpoint, the book is not updated, and 415 status code is returned.

Response Examples

Success

```
Status: 200 OK
{
   "id": 1234,
   "title": "Sense and Sensibility",
   "author": "Jane Austen",
   "genre": "romance",
   "self": "https://appspot.com/books/1234"
}
```

Failure

```
Status: 404 Not Found
{
   "Error": "No book with this book_id exists"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "Too many attributes provided, must not exceed 3"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "No attributes provided "
}
```

Failure

```
Status: 406 Not Acceptable
{
   "Error": "The server only sends JSON"
}
```

Failure

```
Status: 415 Unsupported Media Type
{
   "Error": "The server only accepts JSON"
}
```

Edit a Book with PUT

Allows you to update a book with all new attributes.

PUT /books/:book_id	**Unprotected**

Request

Path Parameters:

Name	Description
book_id	ID of the book

Request Body:
Required

Request Body Format:
JSON

Accepted MIME types:
application/json

Request JSON Attributes

Name	Description	Required?
title	The title of the book.	Yes
author	The author of the book.	Yes
genre	The genre of the book.	Yes

Request Body Example

```
{
  "title": "David Copperfield",
  "author": "Charles Dickens",
  "genre": "fiction"
}
```

Response

Response Body Format:
JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No book with this book_id exists

Failure	400 Bad Request	If the request is missing any of the 3 required attributes, more than 3 attributes are provided, or no attributes are provided, the book is not updated, and 400 status code is returned.
Failure	406 Not Acceptable	If the request Accept header is set to a non-JSON MIME type, the book is not updated, and 406 status code is returned.
Failure	415 Unsupported Media Type	If a non-JSON MIME type is sent to the endpoint, the book is not updated, and 415 status code is returned.

Response Examples

Success

```
Status: 200 OK
{
   "id": 1234,
   "title": "David Copperfield",
   "author": "Charles Dickens",
   "genre": "fiction"
   "self": "https://appspot.com/books/1234"
}
```

Failure

```
Status: 404 Not Found
{
   "Error": "No book with this book_id exists"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "Missing one or more attributes"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "Too many attributes provided, must not exceed 3"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "No attributes provided"
}
```

Failure

```
Status: 406 Not Acceptable
{
   "Error": "The server only sends JSON"
}
```

Failure

```
Status: 415 Unsupported Media Type
{
   "Error": "The server only accepts JSON"
}
```

Create a Reading List

Allows you to create a new reading list.

POST /reading_lists	**Protected**

Request

Path Parameters:
None

Request Body:
Required

Request Body Format:
JSON

Accepted MIME Types:
application/json

Request JSON Attributes

Name	**Description**	**Required?**
name	The name of the reading list.	Yes
description	The description of the reading list.	Yes

Request Body Example

```
{
   "name": "My Summer Reading List",
   "description": "Books to read this Summer!"
}
```

Response

Response Body Format:
JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	201 Created	
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the reading list is not created, and 400 status code is returned.

Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the reading list is not created, and 401 status code is returned.
Failure	406 Not Acceptable	If the request Accept header is set to a non-JSON MIME type, the reading list is not created, and 406 status code is returned.
Failure	415 Unsupported Media Type	If a non-JSON MIME type is sent to the endpoint, the reading list is not created, and 415 status code is returned.

Response Examples

Success

```
Status: 201 Created
{
   "id": 2345,
   "name": "My Summer Reading List",
   "description": "Books to read this Summer!",
   "books": [],
   "user": "auth0|65736a4718710d662aeb6c73",
   "self": "https://appspot.com/reading_lists/2345"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "Missing one or more attributes"
}
```

Failure

```
Status: 401 Unauthorized
{
   "Error": "Unauthorized"
}
```

Failure

```
Status: 406 Not Acceptable
{
   "Error": "The server only sends JSON"
}
```

Failure

```
Status: 415 Unsupported Media Type
{
   "Error": "The server only accepts JSON"
}
```

View all Reading Lists

Allows you to view all existing reading lists for the user based on the session JWT. Returns 5 reading lists per page and includes a next link to get the next page of results unless there are no more pages of result. Also returns a count of the total number of existing reading lists for the user based on the session JWT.

GET /reading_lists	**Protected**

Request

Path Parameters:
None

Request Body:
None

Response

Response Body Format:
JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the reading lists are not returned, and 401 status code is returned.

Response Examples

Success

```
Status: 200 OK
{
   "reading_lists": [
      {
         "id": 2345,
         "name": "My Summer Reading List",
         "description": "Books to read this Summer!",
         "books": [1234, 1194, 1583],
         "user": "auth0|65736a4718710d662aeb6c73",
         "self": "https://appspot.com/reading_lists/2345"
      },
      {
         "id": 8475,
        "name": "Books Mom Recommended",
        "description": "Books mom told me I should read"
        "books": [1234, 4034, 8239, 1284],
        "user": "auth0|65736a4718710d662aeb6c73",
        "self": "https://appspot.com/reading_lists/8475"
      },
      {
        "id": 3745,
        "name": "English Literature 101 Reading List",
        "description": "Books to read for my English Lit class",
        "books": [8823, 9033, 7465, 9445, 1432],
        "user": "auth0|65736a4718710d662aeb6c73",
        "self": "https://appspot.com/reading_lists/3745"
      },
      {
        "id": 4822,
        "name": "Rainy Day Reading List",
        "description": "Books for a rainy day",
        "books": [7376, 9309, 1249],
        "user": "auth0|65736a4718710d662aeb6c73",
        "self": "https://appspot.com/reading_lists/4822"
      },
      {
        "id": 3210,
        "name": "My Favorite Romance Novels",
        "description": "Best love stories of all time!",
        "books": [1234, 4778, 2240],
        "user": "auth0|65736a4718710d662aeb6c73",
        "self": "https://appspot.com/reading_lists/3210"
      }
  ],
  "count": 7,
  "next": "https://appspot.com/reading_lists?limit=5&offset=5"
}
```

Failure

```
Status: 401 Unauthorized
{
  "Error": "Unauthorized"
}
```

View a Reading List

Allows you to view an existing reading list.

GET /reading_lists/:reading_list_id	**Protected**

Request

Path Parameters:

Name	**Description**
reading_list_id	ID of the reading_list

Request Body:
None

Response

Response Body Format:
JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	404 Not Found	No reading_list with this reading_list_id exists
Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the reading list is not returned, and 401 status code is returned.
Failure	403 Forbidden	If "user" property of the specified reading list does not match session JWT's "sub" property, the reading list is not returned, and 403 status code is returned.

Response Examples

Success

```
Status: 200 OK
{
   "id": 2345,
   "name": "My Summer Reading List",
   "description": "Books to read this Summer!",
   "books": [1234, 1194, 1583],
   "user": "auth0|65736a4718710d662aeb6c73",
   "self": "https://appspot.com/reading_lists/2345"
}
```

Failure

```
Status: 404 Not Found
{
   "Error": "No reading_list with this reading_list_id exists"
}
```

Failure

```
Status: 401 Unauthorized
{
   "Error": "Unauthorized"
}
```

Failure

```
Status: 403 Forbidden
{
   "Error": "Forbidden - Not owner of reading list"
}
```

Delete a Reading List

Allows you to delete an existing reading_list.

DELETE /reading_lists/:reading_list_id	**Protected**

Request

Path Parameters:

Name	Description
reading_list_id	ID of the reading_list

Request Body:
None

Response

No body

Response Body Format:
Success: No body
Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	404 Not Found	No reading_list with this reading_list_id exists
Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the reading list is not returned, and 401 status code is returned.
Failure	403 Forbidden	If "user" property of the specified reading list does not match session JWT's "sub" property, the reading list is not returned, and 403 status code is returned.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found
{
   "Error": "No reading_list with this reading_list_id exists"
}
```

Failure

```
Status: 401 Unauthorized
{
   "Error": "Unauthorized"
}
```

Failure

```
Status: 403 Forbidden
{
   "Error": "Forbidden - Not owner of reading list"
}
```

Edit a Reading List with PATCH

Allows you to modify one or more attributes of a book.

PATCH /reading_lists/:reading_list_id	**Protected**

Request

Path Parameters:

Name	Description
reading_list_id	ID of the reading_list

Request Body:
Required

Request Body Format:
JSON

Accepted MIME types:
application/json

Request JSON Attributes

Name	Description	Required?
name	The name of the reading list.	No
description	The description of the reading list.	No

Request Body Example

```
{
   "description": "Books to read Summer 2023"
}
```

Response

Response Body Format:
JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No reading_list with this reading_list_id exists
Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the reading list is not updated, and 401 status code is returned.

Failure	403 Forbidden	If "user" property of the specified reading list does not match session JWT's "sub" property, the reading list is not updated, and 403 status code is returned.
Failure	400 Bad Request	If no attributes are provided or more than 2 attributes are provided, the reading list is not updated, and 400 status code is returned.
Failure	406 Not Acceptable	If the request Accept header is set to a non-JSON MIME type, the reading list is not updated, and 406 status code is returned.
Failure	415 Unsupported Media Type	If a non-JSON MIME type is sent to the endpoint, the reading list is not updated, and 415 status code is returned.

Response Examples

Success

```
Status: 200 OK
{
   "id": 2345,
   "name": "My Summer Reading List",
   "description": "Books to read Summer 2023",
   "books": [],
   "user": "auth0|65736a4718710d662aeb6c73",
   "self": "https://appspot.com/reading_lists/2345"
}
```

Failure

```
Status: 404 Not Found
{
   "Error": "No reading_list with this reading_list_id exists"
}
```

Failure

```
Status: 401 Unauthorized
{
   "Error": "Unauthorized"
}
```

Failure

```
Status: 403 Forbidden
{
   "Error": "Forbidden - Not owner of reading list"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "Too many attributes provided, must not exceed 2"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "No attributes provided "
}
```

Failure

```
Status: 406 Not Acceptable
{
   "Error": "The server only sends JSON"
}
```

Failure

```
Status: 415 Unsupported Media Type
{
   "Error": "The server only accepts JSON"
}
```

Edit a Reading List with PUT

Allows you to update a reading list with all new attributes.

PUT /reading_lists/:reading_list_id	**Protected**

Request

Path Parameters:

Name	Description
reading_list_id	ID of the reading_list

Request Body:
Required

Request Body Format:
JSON

Accepted MIME types:
application/json

Request JSON Attributes

Name	Description	Required?
name	The name of the reading list.	Yes
description	The description of the reading list.	Yes

Request Body Example

```
{
   "name": "Winter 2023 Reading List",
   "description": "Books to read this Winter"
}
```

Response

Response Body Format:
JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No reading_list with this reading_list_id exists
Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the reading list is not updated, and 401 status code is returned.

Failure	403 Forbidden	If "user" property of the specified reading list does not match session JWT's "sub" property, the reading list is not updated, and 403 status code is returned.
Failure	400 Bad Request	If the request is missing any of the 2 required attributes, more than 2 attributes are provided, or no attributes are provided, the reading list is not updated, and 400 status code is returned.
Failure	406 Not Acceptable	If the request Accept header is set to a non-JSON MIME type, the reading list is not updated, and 406 status code is returned.
Failure	415 Unsupported Media Type	If a non-JSON MIME type is sent to the endpoint, the reading list is not updated, and 415 status code is returned.

Response Examples

Success

```
Status: 200 OK
{
   "id": 2345,
   "name": "Winter 2023 Reading List",
   "description": "Books to read this Winter",
   "books": [],
   "user": "auth0|65736a4718710d662aeb6c73",
   "self": "https://appspot.com/reading_lists/2345"
}
```

Failure

```
Status: 404 Not Found
{
   "Error": "No reading_list with this reading_list_id exists"
}
```

Failure

```
Status: 401 Unauthorized
{
   "Error": "Unauthorized"
}
```

Failure

```
Status: 403 Forbidden
{
   "Error": "Forbidden - Not owner of reading list"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "Missing one or more attributes"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "Too many attributes provided, must not exceed 2"
}
```

Failure

```
Status: 400 Bad Request
{
   "Error": "No attributes provided"
}
```

Failure

```
Status: 406 Not Acceptable
{
   "Error": "The server only sends JSON"
}
```

Failure

```
Status: 415 Unsupported Media Type
{
   "Error": "The server only accepts JSON"
}
```

Add a Book to a Reading List

Allows you to add a book to a reading list’s "books" property array.

PUT /reading_lists/:reading_list_id/books/:book_id	**Protected**

Request

Path Parameters:

Name	Description
reading_list_id	ID of the reading_list
book_id	ID of the book

Request Body:
None

Response

No body

Response Body Format:
Success: No body
Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	404 Not Found	The specified book and/or reading_list does not exist
Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the book is not added to the reading list, and 401 status code is returned.
Failure	403 Forbidden	If the book is already in the specified reading list or the "user" property of the specified reading list does not match session JWT’s "sub" property, the book is not added to the reading list, and 403 status code is returned.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found
{
   "Error": "The specified book and/or reading_list does not exist"
}
```

Failure

```
Status: 401 Unauthorized
{
   "Error": "Unauthorized"
}
```

Failure

```
Status: 403 Forbidden
{
   "Error": "Forbidden - Not owner of reading list"
}
```

Failure

```
Status: 403 Forbidden
{
   "Error": "Book already in reading list"
}
```

Remove a Book from a Reading List

Allows you to remove a book from a reading list’s "books" property array. Removing a book does not delete it.

DELETE /reading_lists/:reading_list_id/books/:book_id	**Protected**

Request

Path Parameters:

Name	**Description**
reading_list_id	ID of the reading_list
book_id	ID of the book

Request Body:
None

Response

No body

Response Body Format:
Success: No body
Failure: JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	204 No Content	
Failure	404 Not Found	The specified book and/or reading_list does not exist, or the specified book is not in the reading list.
Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the book is not removed from the reading list, and 401 status code is returned.
Failure	403 Forbidden	If the "user" property of the specified reading list does not match session JWT’s "sub" property, the book is not removed from the reading list, and 403 status code is returned.

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found
{
   "Error": "The specified book and/or reading_list does not exist"
}
```

Failure

```
Status: 404 Not Found
{
   "Error": "The specified book is not in the reading list"
}
```

Failure

```
Status: 401 Unauthorized
{
   "Error": "Unauthorized"
}
```

Failure

```
Status: 403 Forbidden
{
   "Error": "Forbidden - Not owner of reading list"
}
```

View all Books in a Reading List

Allows you to view all books in a reading list’s "books" property array.

GET /reading_lists/:reading_list_id/books/	**Protected**

Request

Path Parameters:

Name	**Description**
reading_list_id	ID of the reading_list

Request Body:
None

Response

Response Body Format:
JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	
Failure	404 Not Found	No reading_list with this reading_list_id exists
Failure	401 Unauthorized	If the session JWT is unable to be verified for any reason, the books are not returned, and 401 status code is returned.
Failure	403 Forbidden	If the "user" property of the specified reading list does not match session JWT’s "sub" property, the books are not returned, and 403 status code is returned.

Response Examples

Success

```
Status: 200 OK
[
   {
      "id": 1234,
      "title": "Pride and Prejudice",
      "author": "Jane Austen",
      "genre": "romance",
      "self": "https://appspot.com/books/1234"
   },
   {
        "id": 2239,
        "title": "The Great Gatsby",
        "author": "F. Scott Fitzgerald",
        "genre": "fiction",
        "self": "https://appspot.com/books/2239"
    }
]
```

Failure

```
Status: 404 Not Found
{
    "Error": "No reading_list with this reading_list_id exists"
}
```

Failure

```
Status: 401 Unauthorized
{
    "Error": "Unauthorized"
}
```

Failure

```
Status: 403 Forbidden
{
    "Error": "Forbidden - Not owner of reading list"
}
```

View all Users

Allows you to view all existing users.

GET /users	**Unprotected**

Request

Path Parameters:
None

Request Body:
None

Response

Response Body Format:
JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	200 OK	

Response Examples

Success

```
Status: 200 OK
[
   {
      "id": "auth0|65736a4718710d662aeb6c73"
   },
   {
      "id": "auth0|19336a457671ksjdflkasdf837"
   }
]
```