**BDD Scenarios:**

Scenario Outline: Get all books

Given the '/books?title={<StringParam>}' resource

When 'GET' request is executed

Then response status code is '200'

And response contains correct books data with '<StringParam>' parameter

| *Title* | *Description* | *Author* |

| TestTitle | TestDescription | TestAuthor |

Examples:

| *StringParam* |

| Test |

| |

Scenario: Get a book

Given the '/books/1' resource

When 'GET' request is executed

Then response status code is '200'

And response body contains the following book

| *Id* | *Title* | *Description* | *Author* |

| 1 | TestTitle | TestDescription | TestAuthor |

Scenario: Get a book with no existing id

Given the '/books/0' resource

When 'GET' request is executed

Then response status code is '404'

And response body contains the following message 'Book with id 0 not found!'

Scenario: Create a book

Given the '/books/' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthor | TestTitle | TestDescription |

When 'POST' request is executed

Then response status code is '200'

And response body contains the following book

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthor | TestTitle | TestDescription |

Scenario: Try to create a book with author and title exceeding allowed characters

Given the '/books/' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthorTestAuthorTestAuthor1 | TestTitle | TestDescription |

When 'POST' request is executed

Then response status code is '400'

And response body contains the following message 'Book.Author should not exceed 30 characters!\r\nParameter name: Book.Author'

Given the '/books/' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthor | TestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTe | TestDescription |

When 'POST' request is executed

Then response status code is '400'

And response body contains the following message 'Book.Title should not exceed 100 characters!\r\nParameter name: Book.Title'

Scenario: Update a book

Given the '/api/books/1' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthor | TestTitle | TestDescription |

When 'PUT' request is executed

Then response status code is '200'

And response body contains the following book

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthor | TestTitle | TestDescription |

Scenario: Try to update a book with author and title exceeding allowed characters

Given the '/books/' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthorTestAuthorTestAuthor1 | TestTitle | TestDescription |

When 'PUT' request is executed

Then response status code is '400'

And response body contains the following message 'Book.Author should not exceed 30 characters!\r\nParameter name: Book.Author'

Given the '/books/' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthor | TestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTestTitleTe | TestDescription |

When 'PUT' request is executed

Then response status code is '400'

And response body contains the following message 'Book.Title should not exceed 100 characters!\r\nParameter name: Book.Title'

Scenario: Update a book no existing id

Given the '/api/books/1' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 101 | TestAuthor | TestTitle | TestDescription |

When 'PUT' request is executed

Then response status code is '404'

And response body has error message 'Book with id 444 not found!'

Scenario: Delete a book

Given the '/api/books/1' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 1 | TestAuthor | TestTitle | TestDescription |

When 'DELETE' request is executed

Then response status code is '204'

And response has no body

Scenario: Delete a book no existing id

Given the '/api/books/1' resource

And request has the following body

| *Id* | *Author* | *Title* | *Description* |

| 100 | TestAuthor | TestTitle | TestDescription |

When 'DELETE' request is executed

Then response status code is '404'

And response body has error message 'Book with id 444 not found!'

**Bugs**:

* POST request executes with null value for Author property.

Description: Author property is posted with null value when POST request is executed

Steps to reproduce:

1. Make a POST request http://localhost:8080/api/books/ with following body as an example:

{

"Id": 1,

"Title": "TestTitle1",

"Description": "TestDescription1",

"Author": "TestAuthor1"

}

Actual result: The response body is returned with Author property value null

Expected result: The response body contains all values from the POST request

* Get request for a specific book doesn’t return properties in the correct order in the response body

Pre-conditions: Add one book using POST request, using following body in the request:

POST /api/books

{

"Id": 1,

"Author": "TestAuthor",

"Title": "TestTitle",

"Description": "TestDescription"

}

Steps to reproduce:

1. Make a GET request http://localhost:8080/api/books/1

Actual result: The response body is returned with following order where Author is last property, and it should be after Id:

{

    "Id": 1,

    "Title": " TestTitle",

    "Description": "TestDescription",

    "Author": " TestAuthor"

}

Expected result: The response body is returned with following order:

{

    "Id": 1,

    "Author": "TestAuthor",

    "Title": " TestTitle",

    "Description": "TestDescription"

}

* POST and PUT request create and update resources with incorrect order in the response body

Steps to reproduce:

1. Make a POST/PUT request http://localhost:8080/api/books/1

Actual result: The response body is returned with following order where Author is last property, and it should be after Id:

{

    "Id": 1,

    "Title": " TestTitle",

    "Description": "TestDescription",

    "Author": " TestAuthor"

}

Expected result: The response body is returned with following order:

{

    "Id": 1,

    "Author": "TestAuthor",

    "Title": " TestTitle",

    "Description": "TestDescription"

}