Bookshelf REST API Test Suite Created by Priti Gaddam

Test case Id: 0001

Test Description: Add a book object to HashMap.

Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.

Test steps:

1. Create new -> HTTP request in Postman tool
2. Select POST as HTTP method to create book data in Postman
3. Enter URI as http://localhost:8080/books in Postman
4. Select message body to be sent as request in Postman
5. Select ‘raw’ and ‘JSON’ in request message body pane in Postman
6. Write book data in the message body in JSON Format. The input JSON data will look as below:

```
{
    "id": 2,
    "name": "Hansen OM Book",
    "author": "abc"
}
```

7. Click on ‘Send’ button
8. View the response data in response pane of Postman tool.

Expected Results:

1. HTTP Response Code should be 201 Created.
2. true

.

Actual Results:

1. 201 Created.
2. true

Test case Id: 0002

Test Description: Add a book object to hashmap which is already present in Hashmap.

Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.

Test steps:

9. Create new -> HTTP request in Postman tool
10. Select POST as HTTP method to create book data in Postman
11. Enter URI as http://localhost:8080/books in Postman

12. Select message body to be sent as request in Postman
13. Select ‘raw’ and ‘JSON’ in request message body pane in Postman
14. Write book data in the message body in JSON Format. The input JSON data will look as below:

```
{
  "id": 2,

   "name": "Hansen OM1 Book",

   "author": "abc1"
}
```

15. Click on ‘Send’ button
16. View the response data in response pane of Postman tool.

Expected Results:

1. HTTP Response Code should be 406 Not acceptable.
2. false

Actual Result :

1. Status : 406 Not Acceptable.
2. false

Test case Id :0003

Test Description: Find all book data in booklist

Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.

Test steps:

1) Create new -> HTTP request in Postman tool
2) Select get GET as HTTP method to see all book data in Postman
3) Enter URI as http://localhost:8080/books in Postman

4) Click on ‘Send’ button
5) View the response data in response pane of Postman tool

Expected Results:

1. HTTP Response Code should be 200 OK.
2. The response message body should show below JSON (same as the request):

```
[
    {
        "id": 1,
        "name": "Hansen CSD Book",
        "author": "Satyen Pandhare"
    },
    {
        "id": 2,
        "name": "Hansen OM Book"
        "author": "abc"
    },
    {
        "id": 3,
        "name": "Hansen CPQ Book",
        "author": "pqr"
    },
    {
        "id": 4,
        "name": "Hansen IF Book",
        "author": "xyz"
    }
]
```

Actual Result :

1. Status : 200 Ok
2. Result :

```
[
  {
      "id": 1,
      "name": "Hansen CSD Book",
      "author": "Satyen Pandhare"
  },
  {
      "id": 2,
      "name": "Hansen OM Book"
      "author": "abc"
  },
  {
      "id": 3,
      "name": "Hansen CPQ Book",
      "author": "pqr"
  },
  {
      "id": 4,
      "name": "Hansen IF Book",
      "author": "xyz"
  }
]
```

Test case Id :0004

Test Description: Find specific book data from HashMap for a given id/key

Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.

Test steps:
1) Create new -> HTTP request in postman
2) Select GET as HTTP method to find specific book data in Postman
3) Enter URI as http://localhost:8080/books/2 in Postman
4) Click on 'Send' button
5) View the response data in response pane of Postman tool.

Expected Results:

1. HTTP Response Code should be 200 OK.
2. The response message body should show below JSON (same as the request):

```
{
    "id": 2,
    "name": "Hansen OM Book",
    "author": "abc"
 }
```

Actual Result :

1. Status : 200 Ok
2. Result :

```
{
    "id": 2,
    "name": "Hansen OM Book",
    "author": "abc"
 }
```

Test case Id :0005

Test Description: Find specific book data from HashMap for a given id/key which is not present in Hashmap.

Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.

Test steps:
1) Create new -> HTTP request in postman
2) Select GET as HTTP method to find specific book data in Postman
3) Enter URI as http://localhost:8080/books/2 in Postman
4) Click on ‘Send’ button
5) View the response data in response pane of Postman tool

Expected Results:

1. HTTP Response Code should be 404 NOT FOUND.
2. The response message body should show below:

(If given book id is not present in book list then): Given book id is not present in booklist.
OR false

Actual Result :

1. Status : 200 Ok
2. The body of result is blank.

Test case Id :0006
Test Description: Update book data in HashMap
Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.
Test steps:
1) Create new -> HTTP request in postman
2) Select PATCH as HTTP method to update book data in Postman
3) Enter URI as http://localhost:8080/books in Postman
4) Select message body to be sent as request in Postman
5) Select ‘raw’ and ‘JSON’ in request message body pane in Postman
6) Write book data in body in JSON Format. The input JSON data will look as below:

```
{
    "id": 4,
    "name": "Hansen PF1 Book",
    "author": "wersion"
}
```

7) Click on ‘Send’ button
8) View the response data in response pane of Postman tool

Expected Results:

1. HTTP Response Code should be 201 Created.
2. The response message body should show
true

Actual Result :

1. Status : 201 Created
2. true

Test case Id :0007

Test Description: Update book data which is not present in Hashmap.

Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.

Test steps:
1) Create new -> HTTP request in postman
2) Select PATCH as HTTP method to update book data in Postman
3) Enter URI as http://localhost:8080/books in Postman
4) Select message body to be sent as request in Postman
5) Select 'raw' and 'JSON' in request message body pane in Postman
6) Write book data in body in JSON Format. The input JSON data will look as below:

```
{
    "id": 10,

    "name": "Hansen OPU Book",
    "author": "lkgy"
}
```

7) Click on 'Send' button
8) View the response data in response pane of Postman tool

Expected Results:

1. HTTP Response Code should be 404 Not found.
2. The response message body should show
   false

Actual Result :

3. Status : 404 Not found
4. false

Test case Id :0008

Test Description: Delete book data in HashMap with the specified key/id.

Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.

Test steps:
1) Create new -> HTTP request in postman
2) Select DELETE as HTTP method to delete specific book data in Postman
3) Enter URI as http://localhost:8080/books/2 in Postman
4) Click on ‘Send’ button
5) View the response data in response pane of Postman tool

Expected Results:

1. HTTP Response Code should be 200 OK.
2. The response message body should show below:

true (if delete data)

Actual Result :

1. Status : 200 Ok
2. true

Test case Id :0009

Test Description: Delete book data   which is not present in hashmap.

Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.

Test steps:
1) Create new -> HTTP request in postman
2) Select DELETE as HTTP method to delete specific book data in Postman
3) Enter URI as http://localhost:8080/books/10 in Postman
4) Click on ‘Send’ button
5) View the response data in response pane of Postman tool

Expected Results:
1. HTTP Response Code should be 404 NOT FOUND.
2. The response message body should show below:
   False(if data not present)

Actual Result :

1. Status : 404 Not Found
2. false

Test case Id :0010
Test Description: Add a book object to HashMap
Pre-condition: The bookshelf RESTApi should be running in STS on port 8080 and Postman should be in working condition.
Test steps:
1) Create new -> HTTP request in postman
2) Select POST as HTTP method to create book data in Postman
3) Enter URI as http://localhost:8080/books in Postman
4) Select message body to be sent as request in Postman
5) Select 'raw' and 'JSON' in request message body pane in Postman
6) Write book data in body in JSON Format. The input JSON data will look as below:

```
{
   "id": 1,
   "name1": "Book 1",
   "author1": "Author 1"
}
```

7) Click on 'Send' button
8) View the response data in response pane of Postman tool

Expected Results:

1. HTTP Response Code should be 201 CREATED.
2. The response message body should show below JSON (same as the request):

```
{
   "id": 1,
   "name": "NULL",
   "author": "NULL"
}
```