

# Microservices Spring Boot - Final Project

## Overview:

The Book Store REST service using Mysql database show how to use the RESTful service to add and retrieve books from a book store.

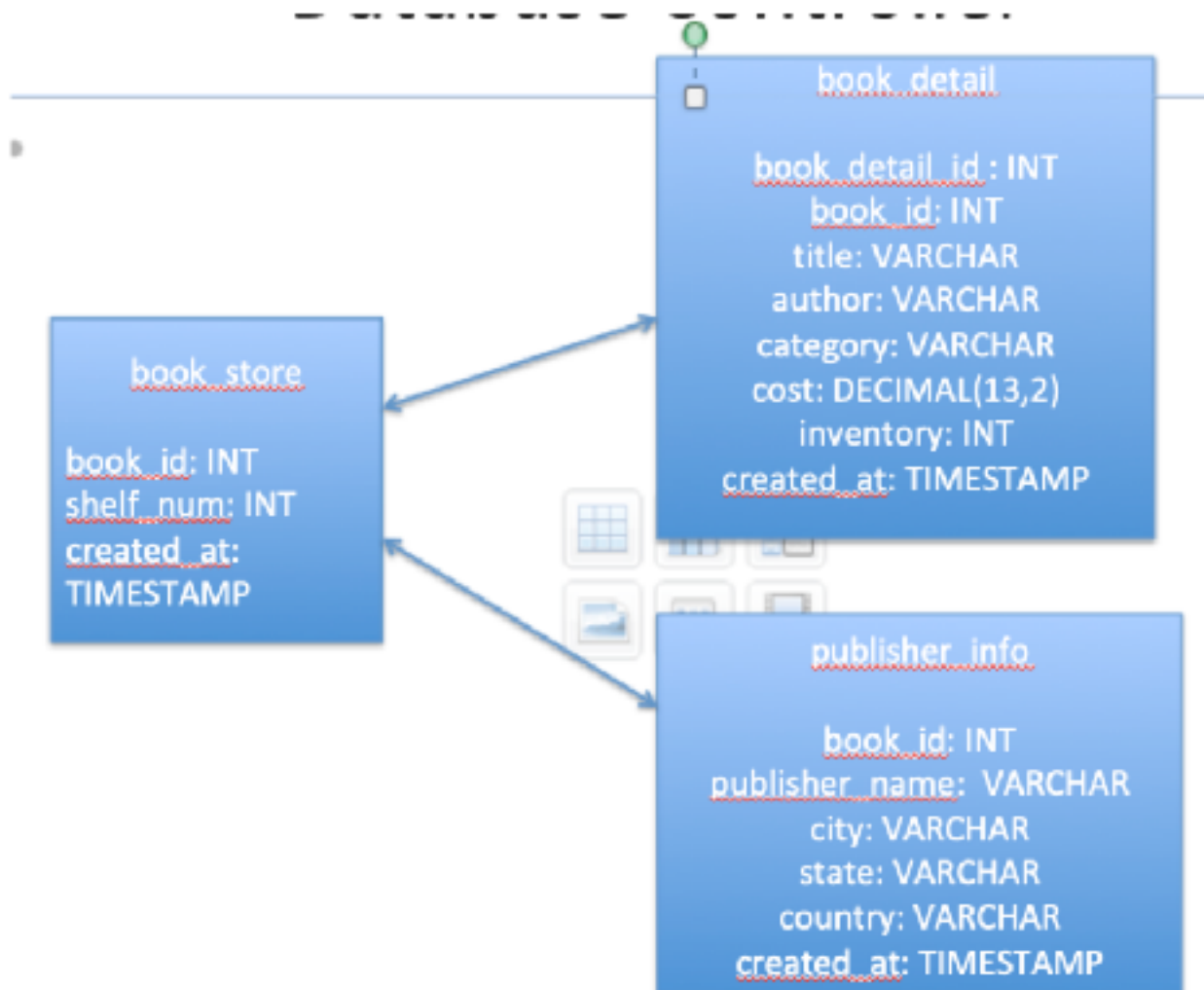
The Book Store REST service performs ten operations:

1. printing all books from bookstore,
2. printing book which has zero inventory
3. printing all books from all shelves order by shelf number
4. finding book using shelf number which is passed as a query parameter to the POST operation
5. finding book title using a search string which is passed as a query parameter to the POST operation
6. getting inventory using book id, which is passed as a query parameter to the POST operation
7. finding publisher name using a search string which is passed as a query parameter to the POST operation

8. adding a book using a java object class which is passed as datas to the POST operation
9. adding a publisher information using a java object class which is passed as datas to the POST operation
10. Finding book category using a search string which is passed as a query parameter to the POST operation

This is a REST Client process which uses Invoke REST API activity to retrieve data from the REST Server. You can POST and GET books by specifying the HTTP Client, HTTP Method and Request Type in the Invoke REST API activity.

## DataBase Architecture



## REST Services:

### GitHub:

<https://github.com/learn-OC/dev-ent-java-microserv-spring-final-proj-template>

### URL :

http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/ spring-proj-template/

### Methods:

---

1)/[printAllBooks](#): print all books from book detail

- Method: GET
- URL Param: None
- Success Response : 200
- Sample Curl call:

http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template// printAllBooks

- Expected Output:

```

Lirhe-MacBook-Pro-2:Downloads lintrun$ curl http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template//printAllBooks
SELECT * from book_detail;
book_id,title, author, category, cost, inventory
1. Absalom,Absalom, William Faulkner, action, 34.00, 0,
2. A time to kill, John Grisham, action, 22.50, 1,
3. East of Eden, John Steinbeck, romance, 22.50, 10,
4. Vile Bodies, Evelyn Waugh, romance, 32.50, 7,
5. Behold, here is poison, Georgette Heyer, romance, 12.50, 20,
6. Band of brothers, Stephen E. Ambrose, fiction, 12.50, 10,
7. Mortal Engines, Philip Reeve, history, 18.50, 0,
20. Mortal Engines, Philip Reeve, action, 21.90, 20,
21. Mortal Engines, Philip Reeve, action, 21.90, 20,
22. Mortal Engines, Philip Reeve, action, 21.90, 20,
24. Mortal Engines, Philip Reeve, action, 21.90, 20,
25. Mortal Engines, Philip Reeve, action, 21.90, 20,
26. Mortal Engines, Philip Reeve, action, 21.90, 20,
27. Mortal Engines, Philip Reeve, action, 21.90, 20.

```

2)/[printBookHasZeroInventory](#) : print all books which have zero inventory

- Method: GET
- URL Param: None
- Success Response : 200
- Sample Curl:
  - curl http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template// printBookHasZeroInventory
- Expected Output:

```

Lirhe-MacBook-Pro-2:Downloads lintrun$ curl http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template//printBookHasZeroInventory
SELECT * from book_detail where inventory = 0;
book_id,title, author, category, cost, inventory
1. Absalom,Absalom, William Faulkner, action, 34.00, 0,
7. Mortal Engines, Philip Reeve, history, 18.50, 0,

```

3)/[showBookOnAllShelves](#): list all books on shelves and order them by shelf number

- Method: GET
- URL Param: None
- Success Response : 200

- Sample Curl call:

- curl http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template// showBookOnAllShelves

- Expected Output:

```
aws history | grep browserless.iam.site
Linux-MacBook-Pro-2:~$ curl http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template/showBookOnAllShelves
select t1.shelf_number, t2.book_id, t2.title, t2.author, t2.category, t2.inventory, t2.cost from book_store t1 INNER JOIN book_detail t2 ON t1.book_id = t2.book_id order
by t1.shelf_number;

shelf_number, book_id, title, author, category, inventory, cost
111, 25, Mortal Engines, Philip Reeve, action, 21.90, 20,
111, 6, Band of Brothers, Stephen E. Ambrose, Fiction, 12.56, 16,
111, 29, Mortal Engines, Philip Reeve, action, 21.90, 28,
111, 1, Absalom, Absalom, William Faulkner, action, 34.99, 6,
111, 21, Mortal Engines, Philip Reeve, action, 21.90, 28,
111, 22, Mortal Engines, Philip Reeve, action, 21.90, 28,
111, 24, Mortal Engines, Philip Reeve, action, 21.90, 28,
112, 2, A Time to Kill, John Grisham, action, 22.50, 1,
113, 5, Denold, here is poison, Georgette Heyer, romance, 12.56, 16,
113, 7, Mortal Engines, Philip Reeve, history, 15.50, 6,
113, 3, East of Eden, John Steinbeck, romance, 22.50, 16,
114, 4, Wild Dillies, Evelyn Waugh, romance, 12.56, 7,
```

---

4)/[findBookOnShelf](#) : list all books on shelves and order them by shelf number

- Method: GET

- URL Param: shelfNum=[integer]

- Success Response : 200

- Sample Curl call:

- curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:

8080/spring-proj-template//  
findBookOnShelf?shelfNum=111”

- Expected Output:

```
193 history | grep findBookOnShelf
Linko-MacBook-Pro-2:Devsleace linko$ curl -s -H "Accept: application/json" -H "Content-Type:application/json" -X POST "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template//findBookOnShelf?shelfNum=111"
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 489
Server: Jetty(9.4.0.v20180311)

select t1.shelf_number, t2.book_id, t2.title, t2.author, t2.category, t2.inventory, t2.cost from book_shelfs t1 join book_detail t2 on t1.book_id = t2.book_id where
t1.shelf_number = 111;

shelf number, book id, title , author, category, inventory , cost
111, 1, Absalom,Absalom, William Faulkner, action, 34.88, 4,
111, 4, Band of brothers, Stephen E. Ambrose, fiction, 12.06, 26,
111, 20, Mortal Engines, Philip Reeve, action, 21.50, 30,
111, 21, Mortal Engines, Philip Reeve, action, 21.50, 30,
111, 22, Mortal Engines, Philip Reeve, action, 21.50, 30,
111, 24, Mortal Engines, Philip Reeve, action, 21.50, 30,
111, 25, Mortal Engines, Philip Reeve, action, 21.50, 30,
```

---

5)/findBookTitle : search all books which have a substring  
which provided from URL param

- Method: POST
- URL Param:
- title=[String] Required
- Success Response : 200
- Sample Curl call:
  - curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST --data "{\"title\":\"Ba\"}" "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template//findBookTitle”

## ● Expected Output:

```
light-MacBook-Pro-7:Downloads liantran$ curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST --data '{"title":"BookA"}' "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template/getBookTitle"
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 155
Server: Jetty/9.4.25.v20200117

select * from book_detail where title like '%BookA';
book_id, title, author, category, cost, inventory
1, Band of brothers, Stephen E. Ambrose, fiction, 32.56, 26,
```

6)/getInventoryForBookId : get an inventory for specific bookid

- Method: POST URL Param:
- bookid =[integer] Required
- Success Response : 200
- Sample Curl call:
  - curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template/getInventoryForBookId?bookid=1"
- Expected Output:

```
liantran$ curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template/getInventoryForBookId?bookid=1"
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 182
Server: Jetty/9.4.26.v20200117

select book_id, title, author, inventory, category from book_detail where book_id = 1;
book_id, title, author, inventory, category
1, Nooson,Aboson, William Kaulkner, 0, action,
```

7)/[findPublisherName](#): find publisher name based on user input

- Method: POST URL Param:
- name =[String] Required
- Success Response : 200
- Sample Curl call:
  - `curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST --data "{\"name\":\"Ha\"}" "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template// findPublisherName"`
  - Expected Output:

```
499 history | grep findPublisherName
linhs-MacBook-Pro-2:Downloads linhsraaf$ curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST --data '{"name":"Ha"}' "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template//findPublisherName"
HTTP/1.1 200 OK
Content-Type: application/json;charset=utf-8
Content-Length: 216
Server: Jetty(9.4.26.v20200117)

{"publisher_id": 2, "book_id": 2, "publisher_name": "Hachette Livre", "city": "Duportino", "state": "CA", "country": "USA"}, {"publisher_id": 3, "book_id": 3, "publisher_name": "HarperCollins", "city": "San Jose", "state": "TX", "country": "USA"}
```

8)/[addBookDetail](#): add a book into book\_detail database table

- Method: POST



- URL Param: None
- Data Param:
  - {"title": [String], "author" : [String], "category": [String], "cost": [Decimal], "inventory": [Int], "shelfNum": [Int]}
  - Required
- Success Response : 200
- Sample Curl call:
  - curl -i -H "Accept: application/json" -H "Content-Type: application/json" -X POST --data '{"title": "Mortal Enginges", "author": "Philip Reeve", "category": "action", "cost": 21.9, "inventory": 20, "shelfNum": 111}' "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template//addBookDetail"
  - Expected Output:

```

Lirhs-MacBook-Pro-2:Downloads lihtiree$ curl -i -H "Accept: application/json" -H "Content-Type: application/json" -X POST --data '{"title": "Mortal Enginges", "author": "Philip Reeve", "category": "action", "cost": 21.9, "inventory": 20, "shelfNum": 111}' "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template//addBookDetail"
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 81
Server: Jett/9.4.20.v2020117

{"id": 1, "title": "Mortal Enginges", "author": "Philip Reeve", "category": "action", "cost": 21.9, "inventory": 20, "shelfNum": 111}

Number of row added into 'book_detail' and 'book_store' = 1 with book_id = 17

```

9)/addpublisher : add publisher info into publisher\_info

- Method: POST
- URL Param: None
- Data Param:
  - {"bookid": [Int], "publisher\_name": [String], "city": [String], "state": [String], "country": [String]}
- Required: "bookid" must existing in book\_store
- Success Response : 200
- Sample Curl call:
  - curl -i -H "Accept: application/json" -H "Content-Type: application/json" -X POST --data '{"bookid": 1, "publisher\_name": "Stephen King", "city": "San Jose", "state": "CA", "country": "USA"}' "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template/addPublisher"
  - Expected Output:

```
Linux-MacBook-Pro-2:DevTools linhtszs3 curl -i -H "Accept: application/json" -H "Content-Type: application/json" -X POST --data '{"bookid": 1, "publisher_name": "Stephen King", "city": "San Jose", "state": "CA", "country": "USA"}' "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template/addPublisher"
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 33
Server: Jetty/9.4.24.v20200811

{"bookid": 1, "publisher_name": "Stephen King", "city": "San Jose", "state": "CA", "country": "USA"}
== Added 1 row in publisher_info linhtszs3 | grep findBookId
```

10)/findBookWithCategory : find all books has the search category

- Method: POST
- URL Param: None
- Data Param:
  - {"name":[String]} Required
- Success Response : 200
- Sample Curl call:
  - curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST --data '{"name":["action"]}' "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template/findBookByCategory"
  - Expected Output:

```
Linux-MacBook-Pro-3:Downloads liantress
Linux-MacBook-Pro-3:Downloads liantress$ curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST --data '{"name":["action"]}' "http://ec2-18-216-171-213.us-east-2.compute.amazonaws.com:8080/spring-proj-template/findBookByCategory"
HTTP/1.1 200 OK
Content-Type: application/json;charset=utf-8
Content-Length: 486
Server: Jetty(9.4.56.v20201117)

select * from book_detail where category like 'Action%';

book_id, title, author, category, cost, inventory
1, anshelm,anselm, william saulnier, action, 34.88, 8,
2, a time to kill, John Grisham, action, 22.98, 1,
20, Mortal Engines, Philip Reeve, action, 21.98, 20,
21, Mortal Engines, Philip Reeve, action, 21.98, 20,
22, Mortal Engines, Philip Reeve, action, 21.98, 20,
26, Mortal Engines, Philip Reeve, action, 21.98, 20,
25, Mortal Engines, Philip Reeve, action, 21.98, 20,
26, Mortal Engines, Philip Reeve, action, 21.98, 20,
27, Mortal Engines, Philip Reeve, action, 21.98, 20,
```

## How to run it locally:

Assume you have IntelliJ installed in your laptop and your AWS server is ready

- 1)Goto <https://github.com/learn-OC/dev-ent-java-microserv-spring-final-proj-template> and 'Fork'
- 2)Open IntelliJ
- 3)File->New->Project From Version Control
- 4)enter your github
- 5)right click on 'src/main/java/JDBCApplication.java and 'Run'
- 6)Goto the terminal and run Curl command like

- curl <http://localhost:8080/printAllBooks>
- curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST "http://localhost:8080/findBookOnShelf?shelfNum=111"

7)If you want to use Postman, use below steps:

- for 'GET' call, put <http://localhost:8080/printAllBooks> into url
- for 'POST' call, put <http://localhost:8080/findBookTitle> into url and put {"title": "Ba"} into 'Body' tab

## How to debug it locally:

Assume you have IntelliJ installed in your laptop and your AWS server is ready

- 1)Goto <https://github.com/learn-OC/dev-ent-java-microserv-spring-final-proj-template> and 'Fork'
- 2)Open IntelliJ
- 3)File->New->Project From Version Control
- 4)enter your github

5)right click on 'src/main/java/JDBCApplication.java and 'Debug JDBCApplication'

6)In Postman,

- for 'GET' call, put `http://localhost:8080/printAllBooks` into url

7)Set a breakpoint where you want to trace

8)From Postman, hit 'Send' and debug it from there

---

