



## **DOS OUTPUT HW2**

**By:**

**Hala Barqawi (11821121) ,**

**Asala Tibi (11819226).**

**GET:** request the information of book with id =1

The screenshot shows a web browser's developer tools interface. The top bar indicates the URL is `http://localhost:3001/info/1`. The request method is `GET`. The response status is `200 OK` with a time of `589 ms` and a size of `358 B`. The response body is displayed in JSON format:

```
{
  "id": 1,
  "title": "How to get a good grade in DOS in 40 minutes a day",
  "topic": "distributed systems",
  "stock": 3,
  "price": "20"
}
```

Frontend Server running on port 3001 Console for the first time we requested information of the above book.

```
C:\Windows\System32\cmd.exe - node app.js
Microsoft Windows [Version 10.0.19043.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Users\barqa\OneDrive\Desktop\DOS-Project>node app.js
Frontend Server is Running!
the required information is : not in cache
Catalog server replica to get the information is = 1
*****
```

2nd time we requested the same book , it was found in cache.

```
C:\Windows\System32\cmd.exe - node app.js
Microsoft Windows [Version 10.0.19043.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Users\barqa\OneDrive\Desktop\DOS-Project>node app.js
Frontend Server is Running!
the required information is : not in cache
Catalog server replica to get the information is = 1
*****
The required information is : in cache
*****
```

**GET:** request the information of book with id=2

http://localhost:3001/info/2

GET http://localhost:3001/info/2

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results

Status: 200 OK Time: 192 ms Size: 322 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 2,
3   "title": "RPCs for Noobs",
4   "topic": "distributed systems",
5   "stock": 15,
6   "price": "30"
7 }
```

```
The required information is : in cache
*****
the required information is : not in cache
Catalog server replica to get the information is = 0
*****
```

**GET:** The first search about book with distributed systems topic

http://localhost:3001/search/distributed systems

GET http://localhost:3001/search/distributed systems

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results

Status: 200 OK Time: 35 ms Size: 448 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "title": "How to get a good grade in DOS in 40 minutes a day",
5     "topic": "distributed systems",
6     "stock": 3,
7     "price": "20"
8   },
9   {
10    "id": 2,
11    "title": "RPCs for Noobs",
12    "topic": "distributed systems",
13    "stock": 15,
14    "price": "30"
15  }
16 ]
```

Fronted server Console after the above search

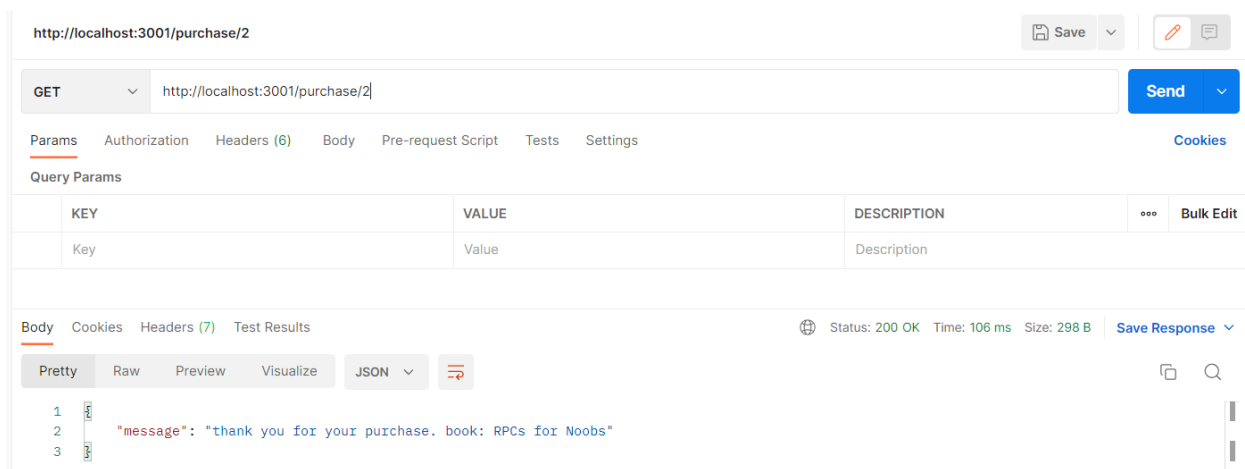
```
Catalog server replica to get the information is = 0
*****
not in cache
Return the book with topic distributed systems and indexCatalog= 1
*****
```

Search again about the above topic (send Request again) :

```
not in cache
Return the book with topic distributed systems and indexCatalog= 1
*****
in cache
*****
```

## PURCHASE :

**GET:** send a purchase request on book with id =2.



The screenshot shows a REST client interface with the following details:

- URL: `http://localhost:3001/purchase/2`
- Method: `GET`
- Send button: `Send`
- Query Params table:

KEY	VALUE	DESCRIPTION
Key	Value	Description
- Body tab selected, showing JSON response:

```
{  "message": "thank you for your purchase. book: RPCs for Noobs"}
```
- Status: `200 OK`, Time: `106 ms`, Size: `298 B`

The Stock in the database of both replicas has been updated. Also when requesting the information of that book .

Shown below:

GET http://localhost:3001/info/2 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 12 ms Size: 322 B Save Response

Pretty Raw Preview Visualize JSON

```

1  {
2    "id": 2,
3    "title": "RPCs for Noobs",
4    "topic": "distributed systems",
5    "stock": 14,
6    "price": "30"
7  }

```

Console shows that the updating operation of the stock value was done correctly and the cache invalidated so the book with id=2 is not in the cache anymore.

```

order server number for purchase is=1
*****
the required information is : not in cache
Catalog server replica to get the information is = 0
*****

```

## UPDATING:

**PUT** : request to update the price value of the book with id=2.

The price was =20

Then it should =40

The request was done successfully.

http://localhost:3001/book/1?stock=0&price=40 Save

PUT http://localhost:3001/book/1?stock=0&price=40 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> stock	0			
<input checked="" type="checkbox"/> price	40			
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 73 ms Size: 269 B Save Response

Pretty Raw Preview Visualize JSON

```

1  {
2    "message": "Updated successfully"
3  }

```

Requesting the information of the book with id=1 to ensure that price was updated , the updating done on both replicas.

GET http://localhost:3001/info/1

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 13 ms Size: 358 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "title": "How to get a good grade in DOS in 40 minutes a day",
4   "topic": "distributed systems",
5   "stock": 3,
6   "price": "40"
7 }
```

Console shows that after updating and requesting the information it's not in the cache any more because cache received invalidating which means that the object inside it is not correct or exists now.

```
*****
catalog server number for updating is = 1
*****
the required information is : not in cache
Catalog server replica to get the information is = 0
*****
```

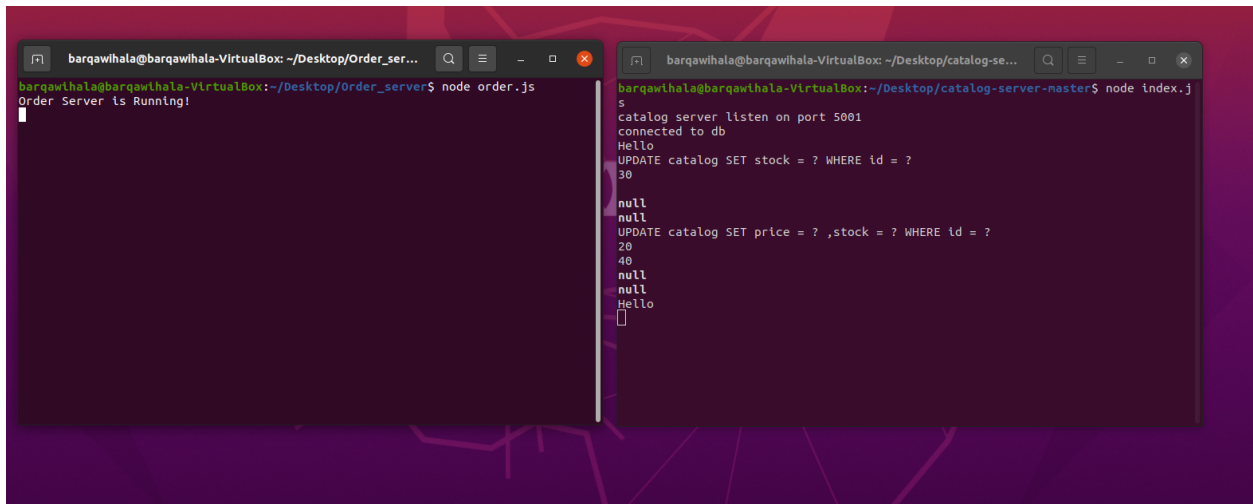
```
hala@Barqawi: ~/Desktop/DOS-order-main
hala@Barqawi:~/Desktop/DOS-order-main$ node order.js
Order Server is Running!

hala@Barqawi:~/Desktop/catalog-server-master$ node index.js
catalog server listen on port 5000
connected to db
HIIIIII
HIIIIII
UPDATE catalog SET stock = ? WHERE id = ?
null
HIIIIII
UPDATE catalog SET price = ? ,stock = ? WHERE id = ?
null
HIIIIII
```

Replica 1 (192.168.1.167) containing a copy of both order and catalog server.

Order server running on port 3004

Catalog server running on port 5000



The image shows two terminal windows side-by-side. The left window is titled 'barqawihala@barqawihala-VirtualBox: ~/Desktop/Order\_ser...' and shows the command 'node order.js' being executed, with the output 'Order Server is Running!'. The right window is titled 'barqawihala@barqawihala-VirtualBox: ~/Desktop/catalog-se...' and shows the command 'node index.js' being executed. The output of the right window includes: 'catalog server listen on port 5001', 'connected to db', 'Hello', 'UPDATE catalog SET stock = ? WHERE id = ?', '30', 'null', 'null', 'UPDATE catalog SET price = ? ,stock = ? WHERE id = ?', '20', '40', 'null', 'null', 'Hello', and a cursor at the end of the line.

```
barqawihala@barqawihala-VirtualBox: ~/Desktop/Order_ser...
barqawihala@barqawihala-VirtualBox:~/Desktop/Order_server$ node order.js
Order Server is Running!

barqawihala@barqawihala-VirtualBox:~/Desktop/catalog-se...
barqawihala@barqawihala-VirtualBox:~/Desktop/catalog-server-master$ node index.js
catalog server listen on port 5001
connected to db
Hello
UPDATE catalog SET stock = ? WHERE id = ?
30
null
null
UPDATE catalog SET price = ? ,stock = ? WHERE id = ?
20
40
null
null
Hello
█
```

Replica 2(192.168.1.136) containing a copy of both order and catalog server.

Order server running on port 3003

Catalog server running on port 5001