

## *DOS course*

### **Multi-tier Online Book Store 2**

Name: Omnya Hmouda & Linda Abu Khalil


The concept:

Frontend server responsible for forwarding requests to the two servers, and implements load balancing (using round-robin algorithm). It also has an in memory cache to enhance the performance, for this we used a hash map for caching requests.

To implement replication we made two replicas of both Catalog server and order server with different ports

In this project we use Java/Spark and Netbeans IDE/Sqlite database.

Before caching:


HTTP New Collection / New Request 


GET ⌵ http://localhost:8082/CATALOG\_WEBSERVICE\_IP/info/2 ↗

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

Query Params

	Key	Value	Description
	Key	Value	Description

Body Cookies Headers (4) Test Results  200 OK 390 ms 235

Pretty Raw Preview Visualize JSON ⌵ 

```
1 {  
2   "quantity": 5,  
3   "price": 50,  
4   "title": "How to get a good grade in DOS in 40 minutes a day"  
5 }
```

---

```
not in cache  
retrieved from db  
2  
finding time = 12544200
```

After caching:

GET http://localhost:8082/CATALOG\_WEBSERVICE\_IP/info/2

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

Query Params

	Key	Value	Description
	Key	Value	Description

Body Cookies Headers (4) Test Results 200 OK 6 ms 242 B Save

Pretty Raw Preview Visualize HTML

```
1 {"quantity":5,"price":50,"title":"How to get a good grade in DOS in 40 minutes a day"}
```

found in cache  
finding time = 11500

Before cache:

GET http://localhost:8082/CATALOG\_WEBSERVICE\_IP/search/distributed systems

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

Query Params

	Key	Value	Description
	Key	Value	Description

Body Cookies Headers (4) Test Results 200 OK 23 ms 254

Pretty Raw Preview Visualize JSON

```
1 [  
2   {  
3     "id": 1,  
4     "title": "RPCs for Noobs"  
5   },  
6   {  
7     "id": 2,  
8     "title": "How to get a good grade in DOS in 40 minutes a day"  
9   }  
]
```

```
finding time = 427900
not found in cache
distributed+systems
retrieved from db
1
```

After cache:

GET

http://localhost:8082/CATALOG\_WEBSERVICE\_IP/search/distributed systems

Send

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettingsCookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

BodyCookiesHeaders (4)Test Results200 OK6 ms270 BSave as example

PrettyRawPreviewVisualizeHTML

```
1 {"data":[{"id":1,"title":"RPCs for Noobs"}, {"id":2,"title":"How to get a good grade in DOS in 40 minutes a day"}]}
```

```
finding time = 6200
- found in cache
```

Before cache:

The screenshot shows a REST client interface. At the top, a PUT request is configured to `http://localhost:8082/order_webservice_ip/purchase/2`. Below the URL bar, tabs for Params, Authorization, Headers (7), Body, Pre-request Script, Tests, and Settings are visible. The 'Query Params' section is expanded, showing a table with columns Key, Value, and Description. The 'Body' tab is selected, displaying the response in 'Pretty' JSON format. The response status is 200 OK with a response time of 474 ms. The response body contains a message and two variables: `finding time = 3479200` and `current cache size:0`.

PUT ▼ `http://localhost:8082/order_webservice_ip/purchase/2`

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

Query Params

	Key	Value	Description
	Key	Value	Description

Body Cookies Headers (4) Test Results 🌐 200 OK 474 ms

Pretty Raw Preview Visualize JSON ▼ 🔄

```
1 Update successful, and you purchased a book.
2
```

```
finding time = 3479200
current cache size:0
```

After cache:

```
finding time = 26400
```

Code: <https://github.com/Omnya-Hmoda/dos>