



An-Najah National University

Computer Engineering department

Distributed and Operating Systems Course

Micro Web Services HW#2 - Online BookStore

Moath AbuSada

14 Nov 2020

Task

Design Bazar.com - the World's smallest book store. Bazar.com carries only four books for sale:

1. How to get a good grade in DOS in 20 minutes a day.
2. RPCs for Dummies.
3. Xen and the Art of Surviving Graduate School.
4. Cooking for the Impatient Graduate Student

each book has a unique ID, title, topic group, quantity and price attributes.

The user can do these three operations:

1. Lookup for a book by book_id
2. Search for the books by topic name
3. Buy a book by book_id

System Design

The system was designed by three modules:

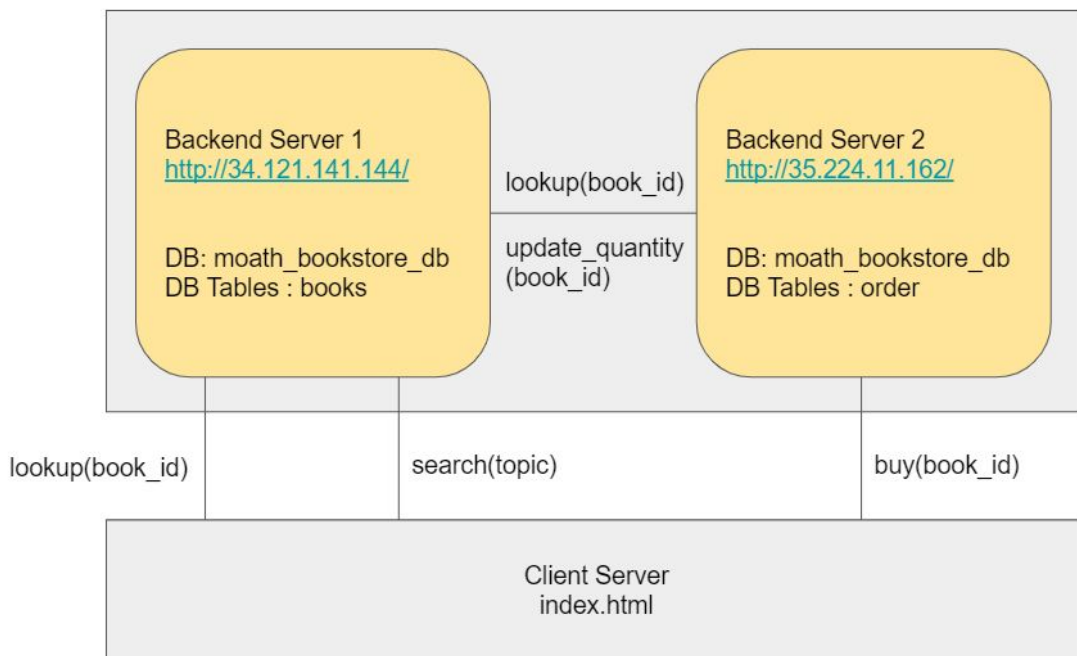
1. Backend server to support lookup and search interactions
2. Backend server to support buy interaction
3. Client server which can be live server or user PC

Each **backend server** is a live virtual machine run with ubuntu server OS with apache server installed on it alongside with sql server to support data storage. Each server has its own ip address.

And the **client server** is a basic html page that makes interaction with the backend servers by ajax and http apis. So the client page can be located on a live server or can be run on a local client pc.

Regarding the **Database design**, the database is distributed horizontally to the same database with the same user was distributed on the both Backend servers, but the backend server 1 contains the database with just books table, and the backend server 2 contains the database with just order table.

System model diagram



Database schemas

1. DB on Backend server 1

- Structure


#	Name	Type
1	book_id 	int(11)
2	topic	varchar(64)
3	title	varchar(255)
4	price	int(11)
5	quantity	int(11)

- Records

book_id	topic	title	price	quantity
1	distributed systems	How to get a good grade in DOS in 20 minutes a day	23	60
2	distributed systems	RPCs for Dummies	45	33
3	graduate school	Xen and the Art of Surviving Graduate School	25	0
4	graduate school	Cooking for the Impatient Graduate Student	90	23

2. DB on Backend server 2

- Structure

#	Name	Type
1	order_id 	int(11)
2	book_id	int(11)
3	order_date	datetime

- Records (subset)

order_id	book_id	order_date
10617	3	2020-11-13 23:57:19
10618	3	2020-11-13 23:58:04
10619	3	2020-11-14 00:05:50
10620	3	2020-11-14 00:08:37
10621	2	2020-11-14 01:57:24
10622	3	2020-11-14 01:59:29
10623	3	2020-11-14 01:59:35
10624	3	2020-11-14 01:59:35

Code implementation

1. Client server

Simple html page with ajax library included, through this page we interact with the different backend servers.

You find client server code on github repo -> frontend folder -> index.html

Moath Book Store

Search, Lookup and Buy your intersted books quickly.

<input type="text" value="distributed systems"/>	<input type="button" value="Search"/>
<input type="text" value="enter the book id you look for"/>	<input type="button" value="Lookup"/>
<input type="text" value="enter the book id you want to buy"/>	<input type="button" value="Buy"/>
<pre>[{"book_id": "1", "topic": "distributed systems", "title": "How to get a good grade in DOS in 20 minutes a day", "price": "23", "quantity": "60"}, {"book_id": "2", "topic": "distributed systems", "title": "RPCs for Dummies", "price": "45", "quantity": "33"}]</pre>	

2. Backend servers

On each backend server we have a gateway that controls all server operations, after the user passes the operation and targeted item on URI, api gateway manubulate the operation and requests the target operation code.

So for example if we want to **buy a book**, how will this operation be done ?

1. On client page we request buy api from Backend server 2 and pass the book_id parameter from text input

```
$("#button-buy").click(function () {  
    $.ajax({  
        url: "http://35.224.11.162/api.php/buy/" + $("#input-buy").val(),  
        dataType: 'json',  
        success: function (resp) {  
            $("#output-result").val(JSON.stringify(resp));  
        }  
    });  
});
```

URI : http://35.224.11.162/api.php/buy/4

2. On api gateway we will explode the operation and the value then request the local controller (on backend server 2) that responsible for buying operation then return the operation result as JSON

```
if ($uri[2] == 'buy') {  
  
    $url = "http://35.224.11.162/buy.php?book_id=" . $uri[3];  
    $curl = curl_init($url);  
    curl_setopt($curl, CURLOPT_URL, $url);  
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);  
    $resp = curl_exec($curl);  
    curl_close($curl);  
  
    echo $resp;  
}
```

3. On local buy controller, first we call the lookup api from backend server 1 to validate if book is exit also to validate if book not out of stock

```
$book_id = filter_input(INPUT_GET, 'book_id', FILTER_SANITIZE_STRING);  
  
$url = "http://34.121.141.144/api.php/lookup/" . $book_id;  
$curl = curl_init($url);  
curl_setopt($curl, CURLOPT_URL, $url);  
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);  
$resp = curl_exec($curl);  
curl_close($curl);  
  
$result = json_decode($resp, true);  
  
if ($result['status'] != "FALSE") {  
    if ($result['quantity'] < 1) {  
        $json['msg'] = "book out of stock";  
        $json['status'] = "FALSE";  
    } else {  
        $db = new DataBaseInstance();  
  
        $sql = "INSERT INTO orders SET book_id = '" . $book_id . "', order_date = NOW()";  
  
        $query = $db->query($sql);  
  
        $order_id = $db->getLastId();  
    }  
}
```

4. If order was added, we call another api from backend server 1 which responsible for updating the quantity by decreasing it by one after each order

```
// update quantity
$url = "http://34.121.141.144/api.php/update_quantity/" . $book_id;
$curl = curl_init($url);
curl_setopt($curl, CURLOPT_URL, $url);
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
$res = curl_exec($curl);
curl_close($curl);

$json['msg'] = "Order was added with order_id #" . $order_id;
$json['status'] = "TRUE";
```

5. Check printed response json on textarea at client page, also re-do lookup on the book_id to validate that quantity was decreased by one

4	Buy
<pre>{ "msg": "Order was added with order_id #10699", "status": "TRUE" }</pre>	

Github Repository

Link : <https://github.com/moathabusada/doshw1.git>

Content:

- Project PDF Report
- frontend folder contain client page
- backend_server_1 folder, contains files to put on the main apache server directory (e.g: www, public-html)
- backend_server_2 folder, contains files to put on the main apache server directory (e.g: www, public-html)
- db_schemas folder, contain the database schema for both backend servers

The End