A Multi-tier Online Book Store

Malak Bawwab

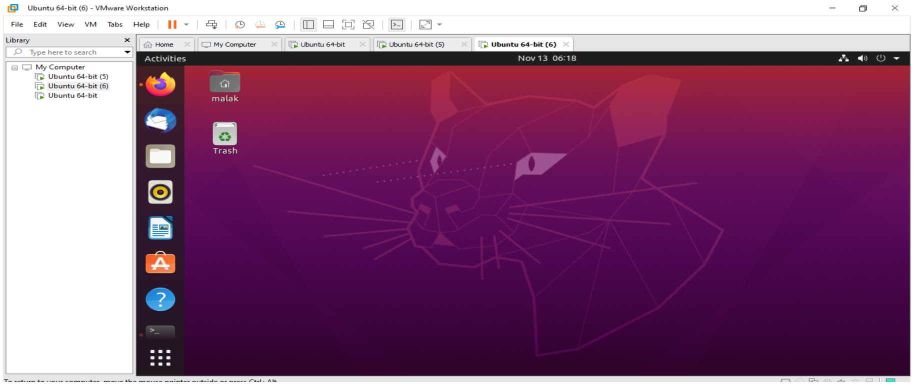
September 2020

Contents

1 Introduction

The store will employ a two-tier web design - a front-end and a backend - and use microservices at each tier. The front-end tier will accept user requests and perform initial processing. The backend consists of two components: a catalog server and an order server.

- There are 3 services:catalog service,order service and front service .
- Each service runs on it's virtual machine,so I have 3 instances of VMs running with ubuntu.
- The communication between services is done through Rest APIs and Guzzle http client.



2 Framework

I used PHP Lumen to implement the 3 services.Lumen is a micro-framework built on top of Laravel's core components,it utilizes a lot of familiar concepts from Laravel such as Eloquent, caching, routing, middleware and it's service container. One major advantage of Lumen is it's speed, as a result the framework is designed for building fast micro-services or APIs.

3 Project Structure

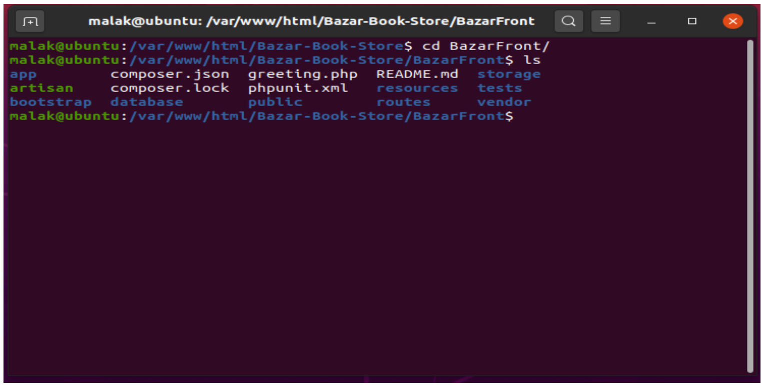
Each service (lumen project) has the following structure.The most important folders are:-

- In routes folder-web.php you can define your own routes/requests.
- In app - Http- Controllers :here you can define your own controllers that have methods to handle your routes /requests.I added BooksController for each service.

- In .env file you can configure the database you want to use and in database folder you can define your sqlite DB(database.sqlite).I used sqlite only in the catalog and order services.

- In resources folder-views you can define your GUI,I added only greeting.php in front service.

- In bootstrap folder-app.php you can uncomment some lines to be able to use Laravel API Facades DB or Eloquent.



malak@ubuntu: /var/www/html/Bazar-Book-Store/BazarFront

```
malak@ubuntu:/var/www/html/Bazar-Book-Store$ cd BazarFront/
malak@ubuntu:/var/www/html/Bazar-Book-Store/BazarFront$ ls
app        composer.json  greeting.php  README.md  storage
artisan    composer.lock  phpunit.xml   resources  tests
bootstrap  database       public        routes     vendor
malak@ubuntu:/var/www/html/Bazar-Book-Store/BazarFront$
```

4 Project Design

- Front service:search,buy,lookup,commands.

- Catalog service:search,lookup,update cost,increase quantity,decrease quantity.

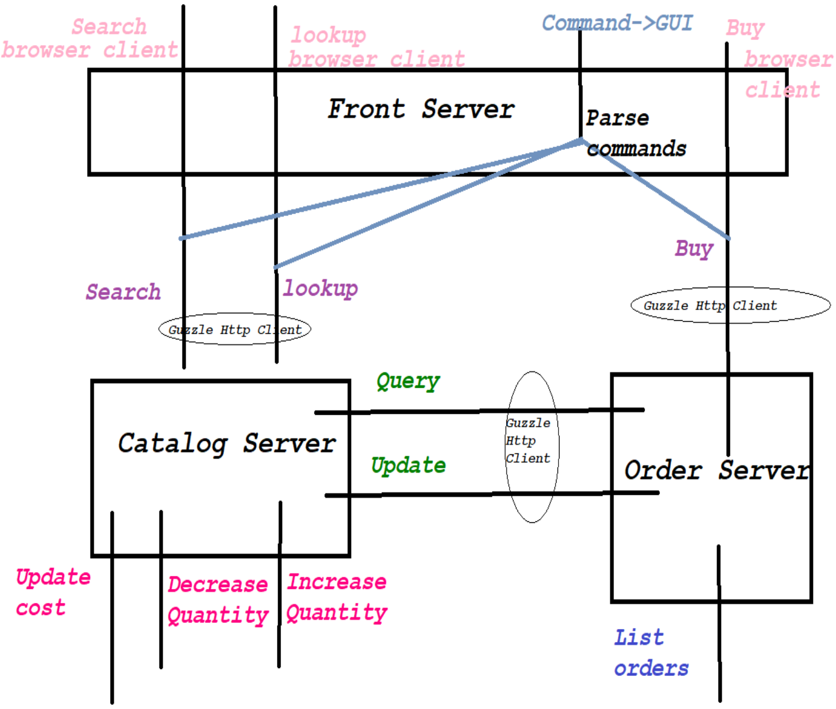
- Order service:buy,listOrders.

- GUI greeting.php that resides in the front tier send command request to front service.

- Front service talks to catalog service in case of search,lookup through Guzzle http client.

- Front service talks to order service in case of buy through Guzzle http client.

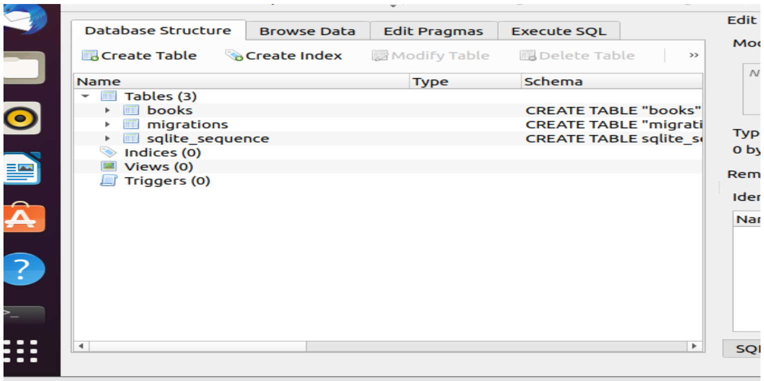
- Order service talks to catalog service in case of buy through Guzzle http



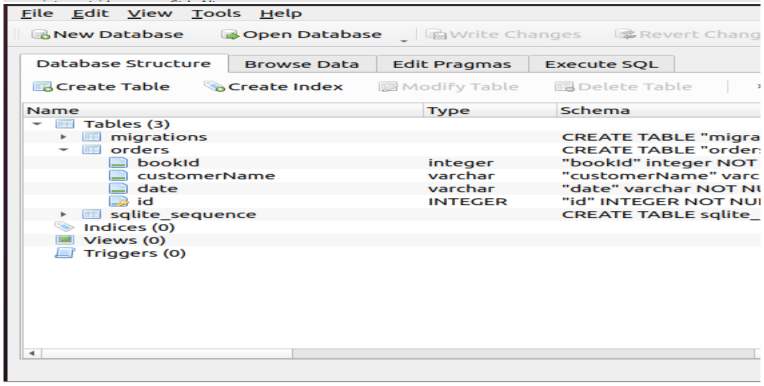
client.

5 Database

- I used sqlite database since it is a lightweight database and it is easy to use.
- Front service doesn't use sqlite ,it just does an initial processing and forward requests to catalog and order services.
- Catalog service uses sqlite with a table named books,books table contains books information(itemNumber,price,quantity,topic,title).
- Order service uses sqlite with a table named orders, orders table contains a list of all orders received for the books(customerName,itemNumber,date)



```
Name                               Type        Schema
▾ Tables (3)
    ▸ books                                    CREATE TABLE "books"
    ▸ migrations                               CREATE TABLE "migrati
    ▸ sqlite_sequence                          CREATE TABLE sqlite_s
  Indices (0)
  Views (0)
  Triggers (0)
```



```
Name                               Type        Schema
▾ Tables (3)
    ▸ migrations                               CREATE TABLE "migra
    ▾ orders                                   CREATE TABLE "order
          bookId                   integer     "bookId" integer NOT
          customerName             varchar     "customerName" varc
          date                     varchar     "date" varchar NOT NU
          id                       INTEGER     "id" INTEGER NOT NUI
    ▸ sqlite_sequence                          CREATE TABLE sqlite_
  Indices (0)
  Views (0)
  Triggers (0)
```

6 GitHub

Bazar-Book-Store repository contains 4 branches:

- Main(contains this document and images for the output).
- Catalog-Service(contains lumen catalog service project).
- Order-Service(contains lumen order service project).
- Front-Server(contains lumen front service project).

7 Catalog Service

Server ip for the catalog service is 192.168.209.134.

- **Catalog service** does query operations (by itemNumber ,by topic)and update operations(cost,quantity decrease or increase) and also does some operations that are required to complete buy process which is defined in the order service.

- **GET http://192.168.209.134/search/{topic}** : this request search for a book in books table based on the topic.

- **GET http://192.168.209.134/lookup/{itemNumber}** :this request search for a book in books table based on the itemNumber.

- **GET http://192.168.209.134/query/{itemNumber}**:this request check if the book exists or not,out of stock or not,it is called by order service through Guzzle http client in buy operation.

- **PUT http://192.168.209.134/update/{itemNumber}**:this request decreases the quantity by one(buy operation is successful).It is called in buy operation(order service) through Guzzle http client after making sure that the book exists and not out of stock due to query request above.

- **PUT http://192.168.209.134/update/book/{itemNumber}/cost/{newCost}**:this request update the price of the book defined by itemNumber.

- **PUT http://192.168.209.134/increase/book/{itemNumber}/quantity/{numberOfItems}**: this request increase the quantity of the book ,so if the original quantity is 45 and numberOfItems pathVariable is 3,the quantity will be 47(there is 47 items avaliable of this book with this itemNumber).

- **PUT http://192.168.209.134/decrease/book/{itemNumber}/quantity/{numberOfItems}**: this request decrease the quantity of the book ,so if the original quantity is 45 and numberOfItems pathVariable is 3,the quantity will be 42(there is 42 items avaliable of this book with this itemNumber).

8 Order Service

Server ip for the catalog service is 192.168.209.131.

- Order service does buy operations and getListOfOrders operation.Through buy operation ,it sends external requests to catalog service through Guzzle http client to complete buy process.

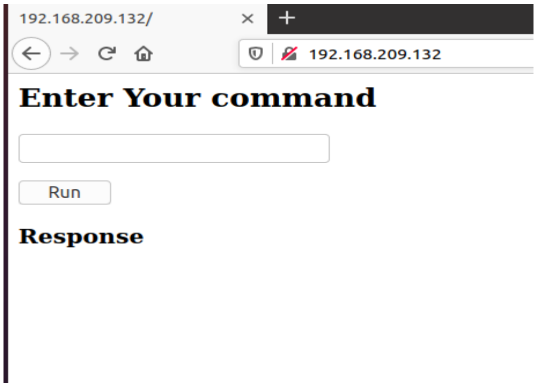
- **GET http://192.168.209.131/list/orders/{itemNumber}** : this request return a list for all the received orders of the book with this itemNumber.

- **POST http://192.168.209.131/buy/{itemNumber}** :this request send external requests to catalog service through Guzzle http client ,the first request is http://192.168.209.134/query/{itemNumber} to check if the book exists on the store and is not out of sock(there are items available to buy).if yes,it sends a second request PUT http://192.168.209.134/update/{itemNumber} also through Guzzle http client to decrease the quantity of the book by 1(buy operation is successful).

9 Front Service

Server ip for the front service is 192.168.209.132. Front services does search,lookup,buy operations by forwarding requests to catalog service(search,lookup) and order service(buy) through Guzzle http client.I implemented two ways that the client can do:

- **First:**The client can enter these first 2 urls directly in the browser and the POST request using postman:
  - **GET http://192.168.209.132/search/topic/{topic}** : this request send external request GET http://192.168.209.134/search/{topic}to catalog service through Guzzle http client.
  - **GET http://192.168.209.132/lookup/number/{itemNumber}** : this request send external request GET http://192.168.209.134/lookup/{itemNumber}to catalog service through Guzzle http client.
  - **POST http://192.168.209.132/buy/number/{itemNumber}**: this request send external request POST http://192.168.209.131/buy/{itemNumber}to order service through Guzzle http client.
- **Second:**the user can enter http://192.168.209.132 in the browser,once he entered a GUI(greeting.php page)will appear.



Greeting page contains a textInput where the user can enter some commands and contains a response section that is resulted from running the entered command.
If the user doesn't enter any of the 3 commands(search ¡topic¿,lookup ¡itemNumber¿,buy ¡itemNumber¿,the response will be command not found ,else a POST request with a body(containing the full command) will be sent to front server :

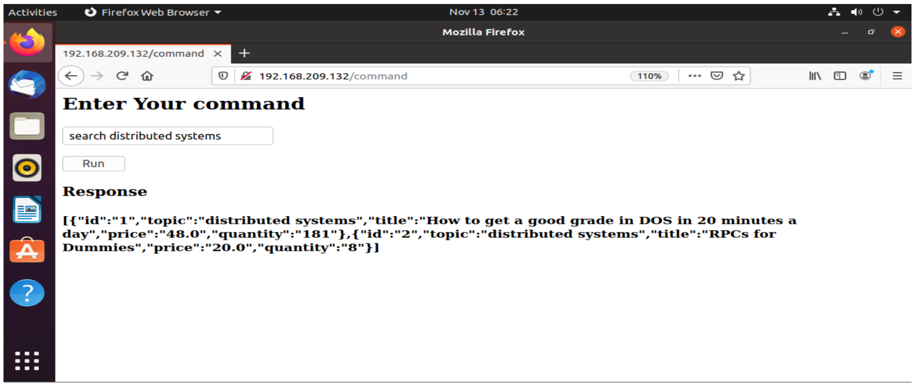
POST http://192.168.209.132 /{command}: this request will parse the entered command ,and based on the command type(search,lookup,buy)it will send external requests to catalog(search,lookup) and order (buy)services through Guzzle http clients.

10 How to run

- **Run the system as whole:**

  1. Power on the 3 VMs..
  2. After setting Up lumen project and moving it inside apache2 server, start apache2 server.

     First way:-
     *Open the Front service VM,open the browser and enter http://192.168.209.132
     *Enter the command in the textInput and press run.
     *Results:**

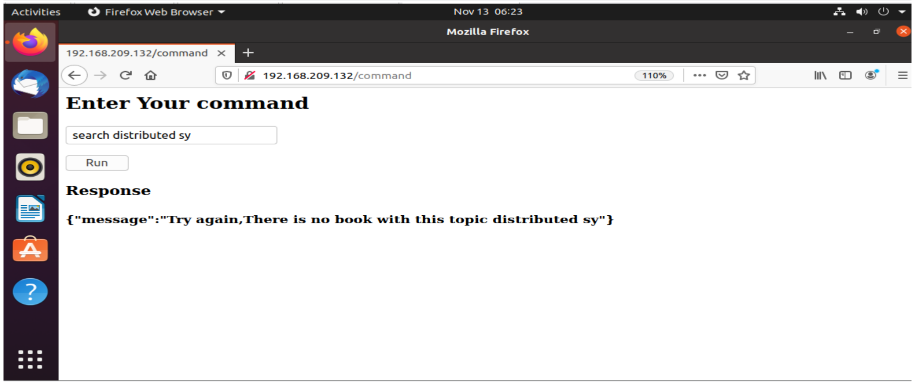


Enter Your command

search distributed systems

Response

```
[{"id":"1","topic":"distributed systems","title":"How to get a good grade in DOS in 20 minutes a day","price":"48.0","quantity":"181"},{"id":"2","topic":"distributed systems","title":"RPCs for Dummies","price":"20.0","quantity":"8"}]
```

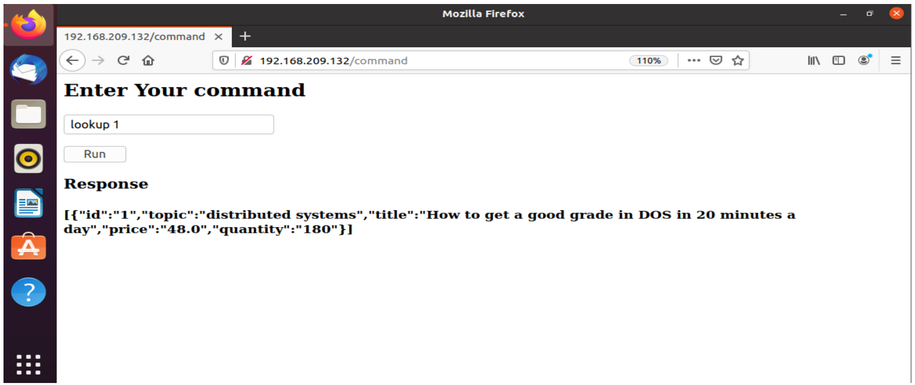


Enter Your command

search distributed sy

Response

```
{"message":"Try again,There is no book with this topic distributed sy"}
```



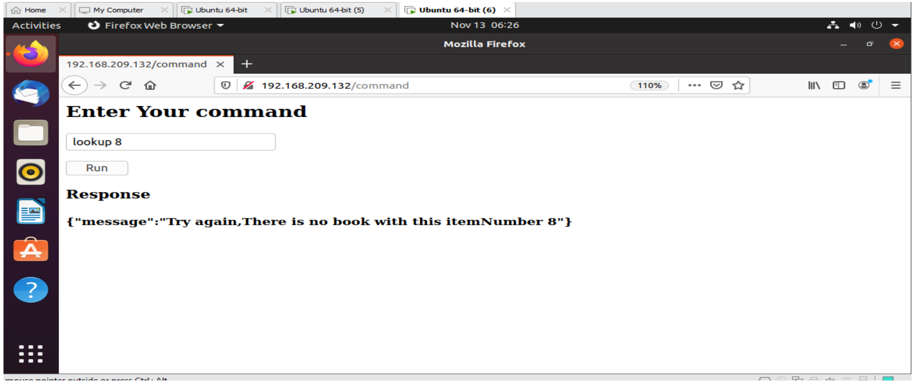
Enter Your command

lookup 1

Run

Response

```
[{"id":"1","topic":"distributed systems","title":"How to get a good grade in DOS in 20 minutes a
day","price":"48.0","quantity":"180"}]
```



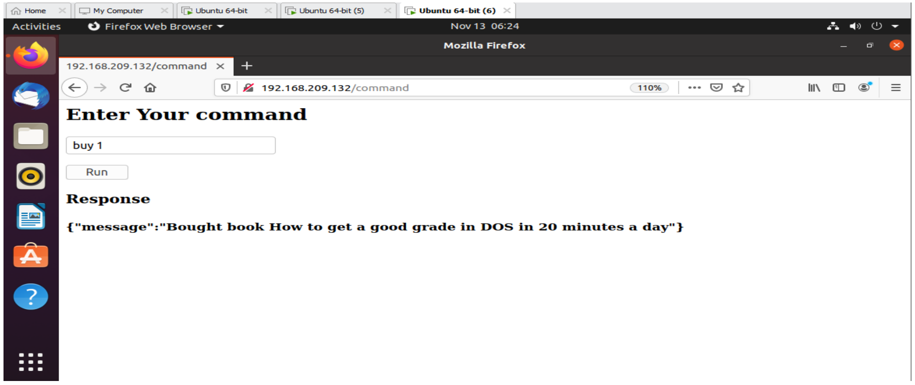
Enter Your command

lookup 8

Run

Response

```
{"message":"Try again,There is no book with this itemNumber 8"}
```



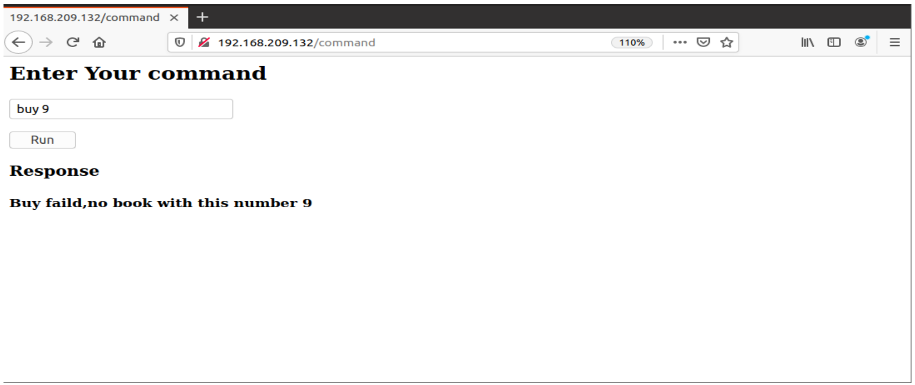
Enter Your command

buy 1

Run

Response

```
{"message":"Bought book How to get a good grade in DOS in 20 minutes a day"}
```



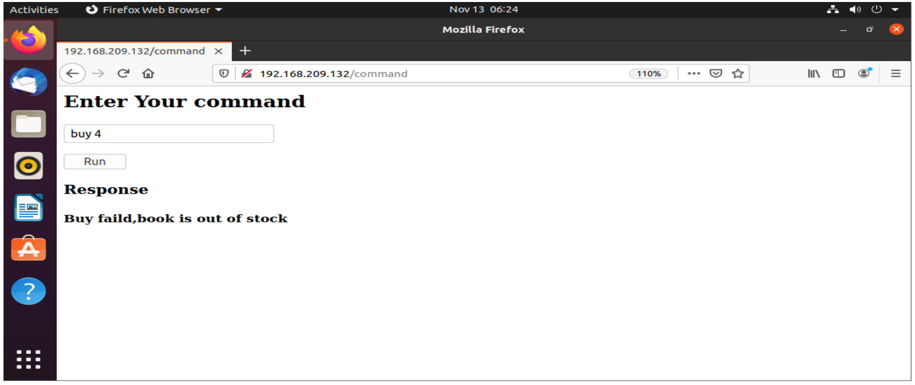
Enter Your command

buy 9

Run

Response

Buy faild,no book with this number 9



Enter Your command

buy 4

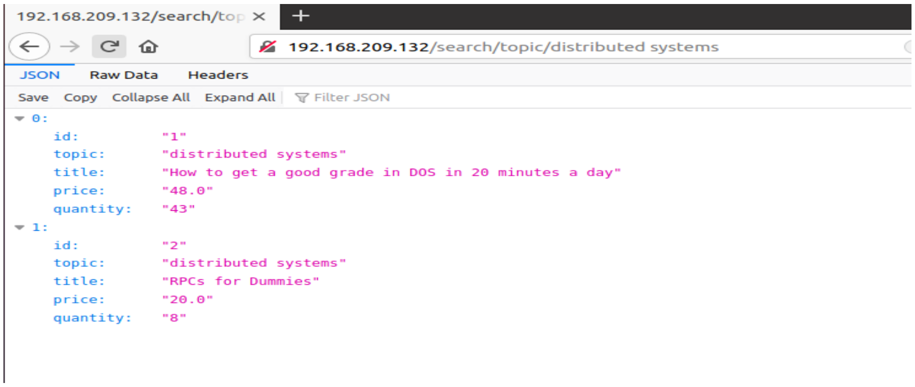
Run

Response

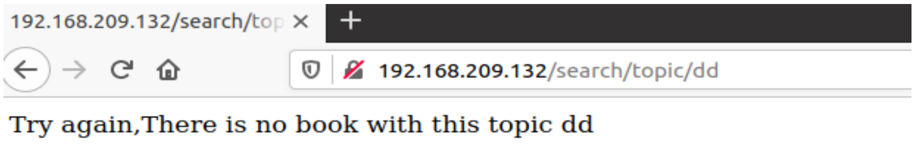
Buy faild,book is out of stock

Second way:-

*open the browser and enter http://192.168.209.132/search/topic/{topic}:

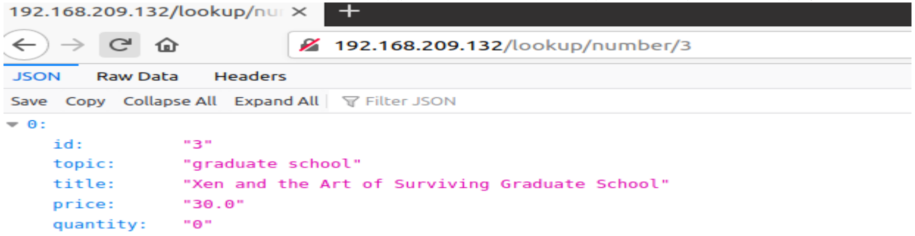

192.168.209.132/search/top
192.168.209.132/search/topic/distributed systems
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON

```
▼ 0:
    id:         "1"
    topic:      "distributed systems"
    title:      "How to get a good grade in DOS in 20 minutes a day"
    price:      "48.0"
    quantity:   "43"
▼ 1:
    id:         "2"
    topic:      "distributed systems"
    title:      "RPCs for Dummies"
    price:      "20.0"
    quantity:   "8"
```

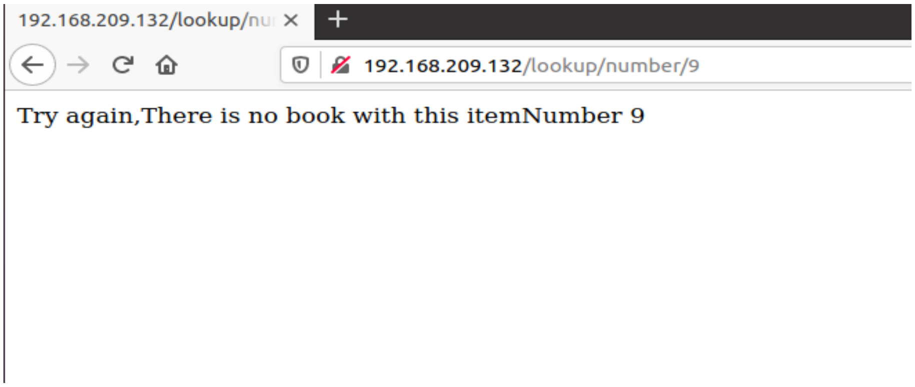


Try again,There is no book with this topic dd

*open the browser and enter http://192.168.209.132/lookup/number/{itemNumber}

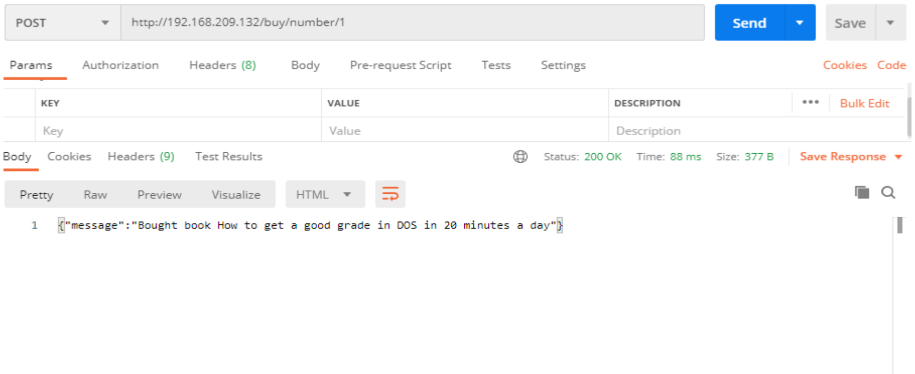


```
▾ 0:
    id:          "3"
    topic:       "graduate school"
    title:       "Xen and the Art of Surviving Graduate School"
    price:       "30.0"
    quantity:    "0"
```



Try again,There is no book with this itemNumber 9

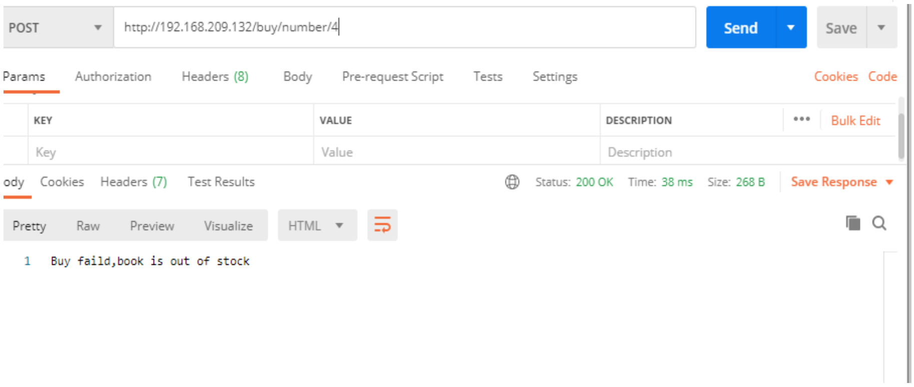
*Open the postmant to execute post request: http://192.168.209.132/buy/number/{itemNumber}



http://192.168.209.132/buy/number/1

Status: 200 OK Time: 88 ms Size: 377 B

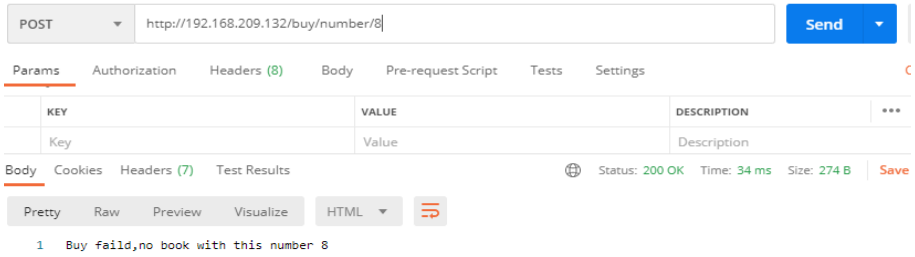
```
{"message":"Bought book How to get a good grade in DOS in 20 minutes a day"}
```



http://192.168.209.132/buy/number/4

Status: 200 OK Time: 38 ms Size: 268 B

```
Buy faild,book is out of stock
```



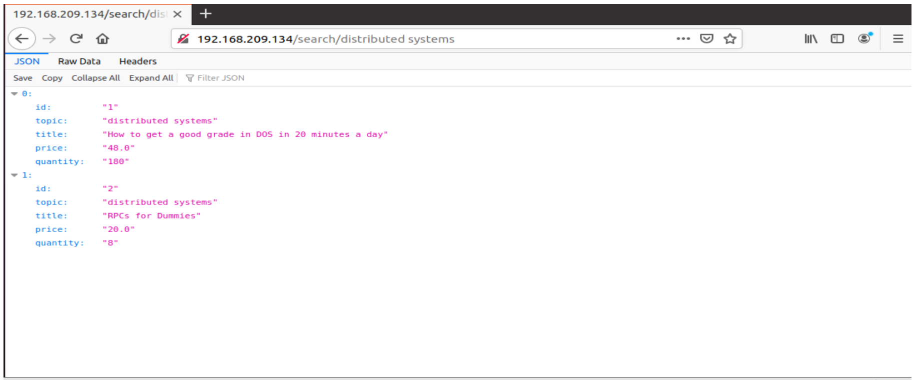
http://192.168.209.132/buy/number/8

Status: 200 OK Time: 34 ms Size: 274 B

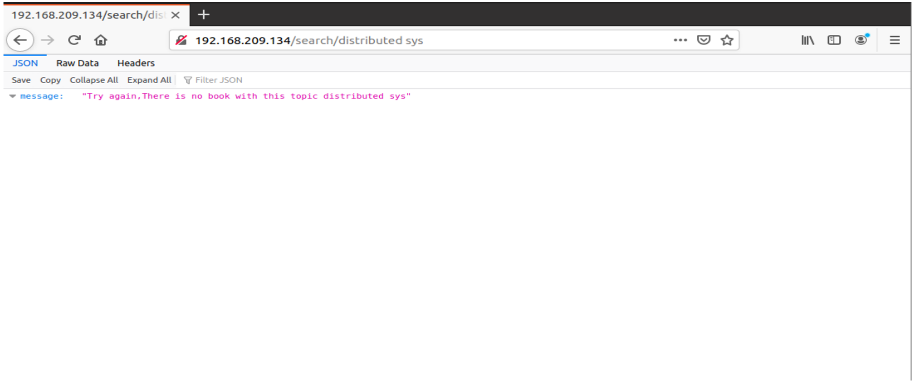
```
Buy faild,no book with this number 8
```

- **Run the catalog service alone:**

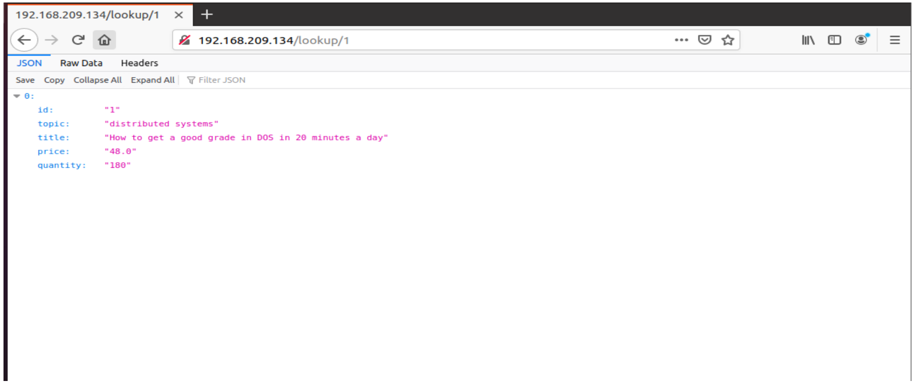
  1. Power on the catalog service VM..
  2. After setting Up lumen project and moving it inside apache2 server, start apache2 server.


192.168.209.134/search/dist
192.168.209.134/search/distributed systems
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON

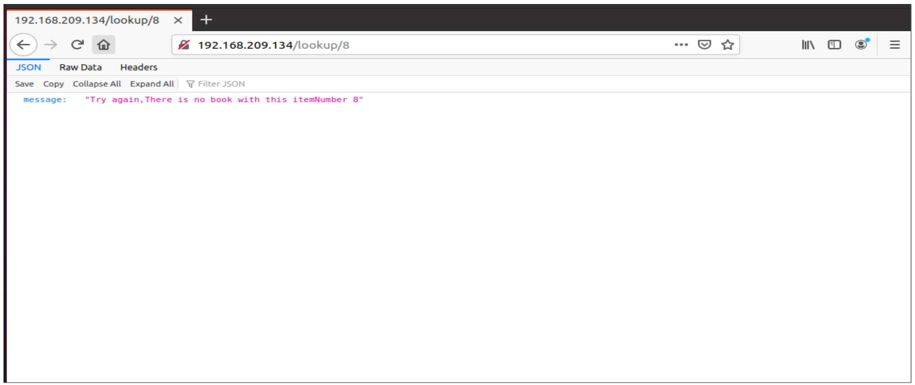
```
▼ 0:
    id:         "1"
    topic:      "distributed systems"
    title:      "How to get a good grade in DOS in 20 minutes a day"
    price:      "48.0"
    quantity:   "180"
▼ 1:
    id:         "2"
    topic:      "distributed systems"
    title:      "RPCs for Dummies"
    price:      "20.0"
    quantity:   "8"
```


192.168.209.134/search/dist
192.168.209.134/search/distributed sys
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON

```
▼ message:   "Try again,There is no book with this topic distributed sys"
```

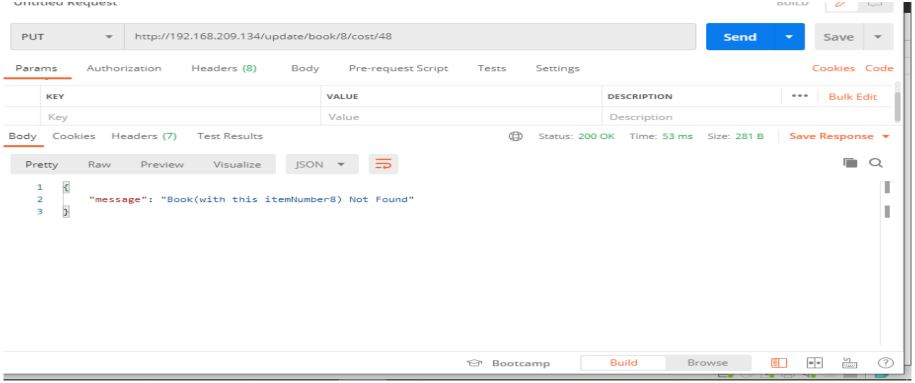


```
▼ 0:
    id:         "1"
    topic:      "distributed systems"
    title:      "How to get a good grade in DOS in 20 minutes a day"
    price:      "48.0"
    quantity:   "180"
```



192.168.209.134/lookup/8

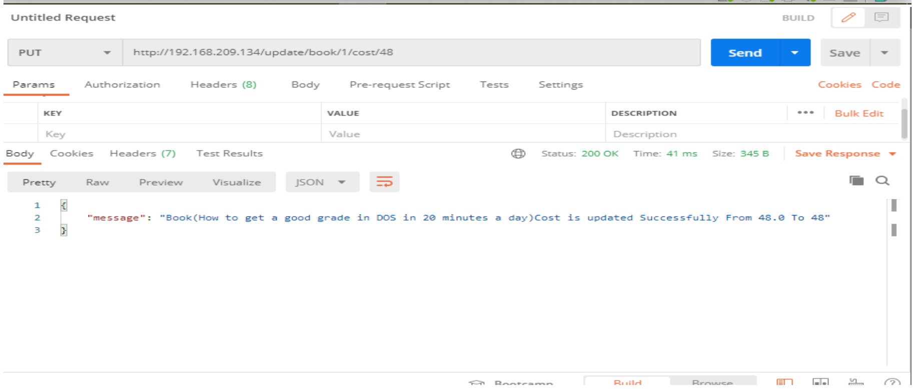
```
message:   "Try again.There is no book with this itemNumber 8"
```



PUT http://192.168.209.134/update/book/8/cost/48

Status: 200 OK   Time: 53 ms   Size: 281 B

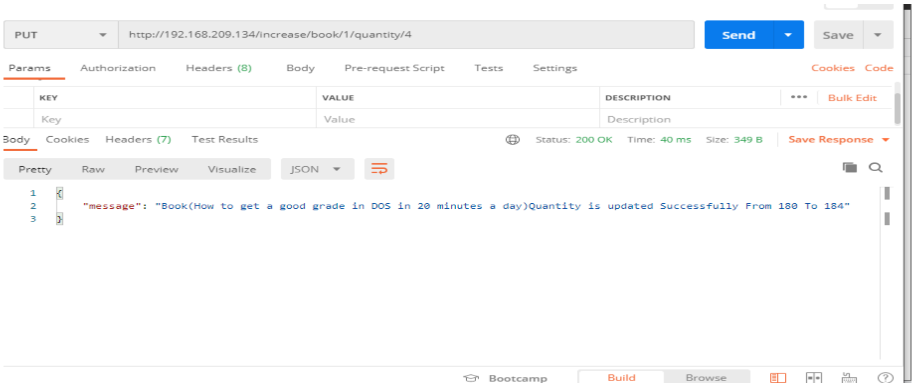
```
{
    "message": "Book(with this itemNumber8) Not Found"
}
```



PUT http://192.168.209.134/update/book/1/cost/48

Status: 200 OK   Time: 41 ms   Size: 345 B

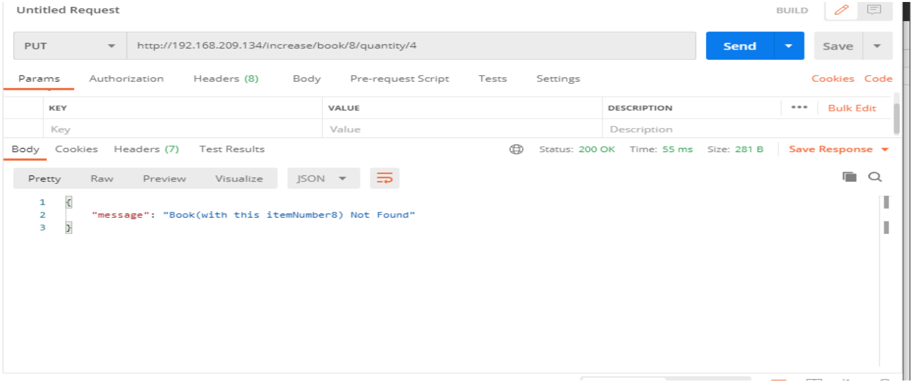
```
{
    "message": "Book(How to get a good grade in DOS in 20 minutes a day)Cost is updated Successfully From 48.0 To 48"
}
```



PUT http://192.168.209.134/increase/book/1/quantity/4

Status: 200 OK Time: 40 ms Size: 349 B

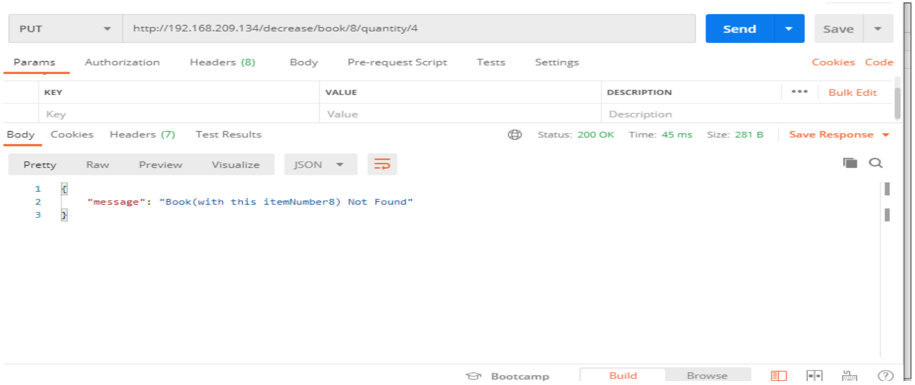
```
{
    "message": "Book(How to get a good grade in DOS in 20 minutes a day)Quantity is updated Successfully From 180 To 184"
}
```



PUT http://192.168.209.134/increase/book/8/quantity/4

Status: 200 OK Time: 55 ms Size: 281 B

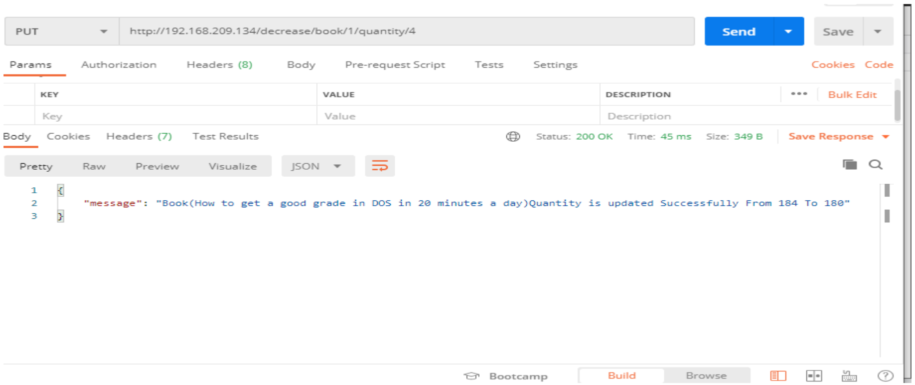
```
{
    "message": "Book(with this itemNumber8) Not Found"
}
```



PUT http://192.168.209.134/decrease/book/8/quantity/4

Status: 200 OK Time: 45 ms Size: 281 B

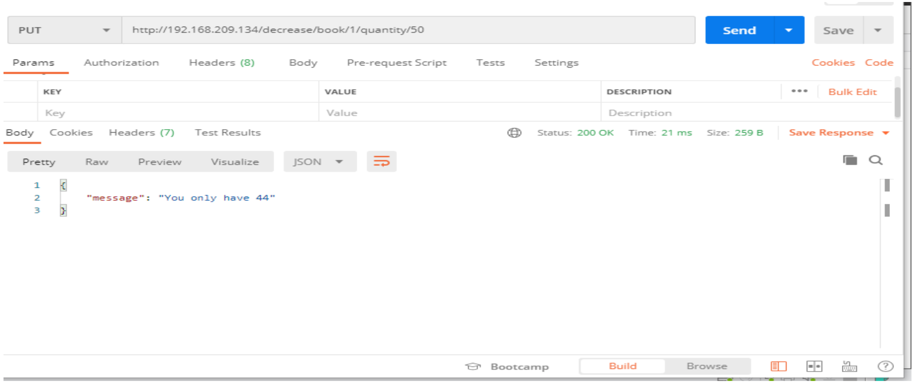
```
{
    "message": "Book(with this itemNumber8) Not Found"
}
```



PUT http://192.168.209.134/decrease/book/1/quantity/4

Status: 200 OK Time: 45 ms Size: 349 B

```
{
    "message": "Book(How to get a good grade in DOS in 20 minutes a day)Quantity is updated Successfully From 184 To 180"
}
```



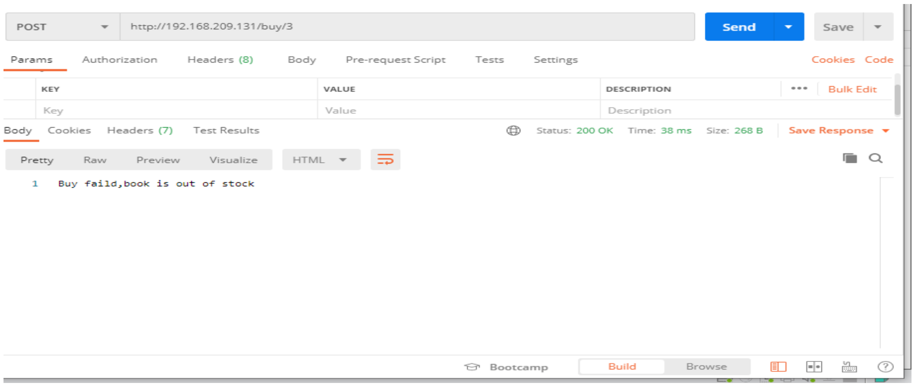
PUT http://192.168.209.134/decrease/book/1/quantity/50

Status: 200 OK Time: 21 ms Size: 259 B

```
{
    "message": "You only have 44"
}
```

- **Run the orders service alone:**

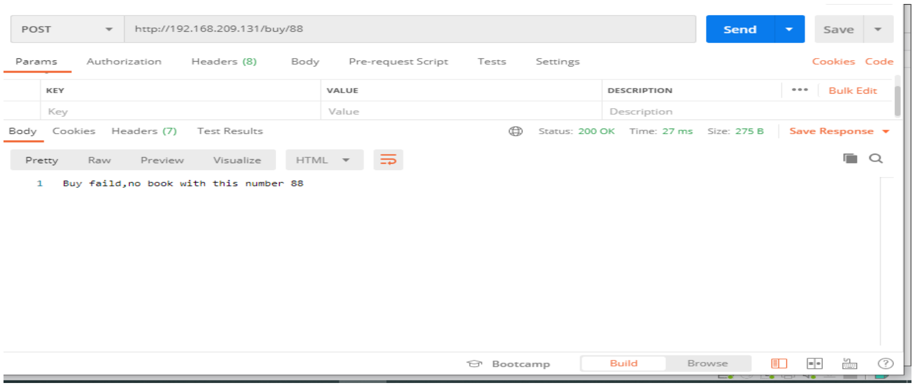
1. Power on the catalog service VM and order Service VM..
2. After setting Up lumen project and moving it inside apache2 server, start apache2 server.



POST http://192.168.209.131/buy/3

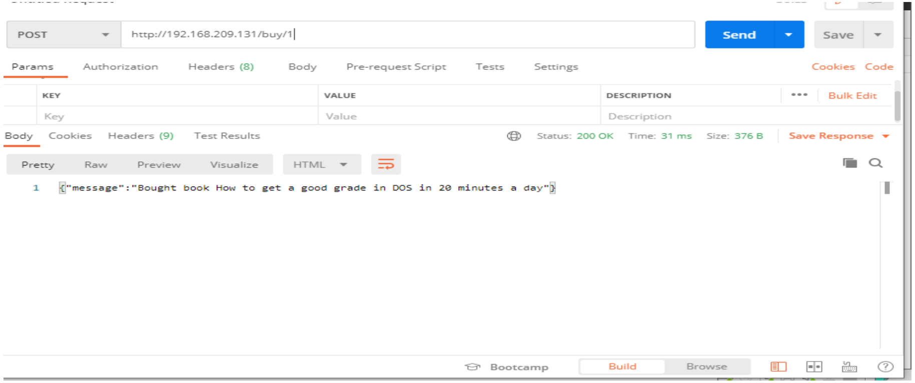
Status: 200 OK Time: 38 ms Size: 268 B

```
Buy faild,book is out of stock
```


POST http://192.168.209.131/buy/88

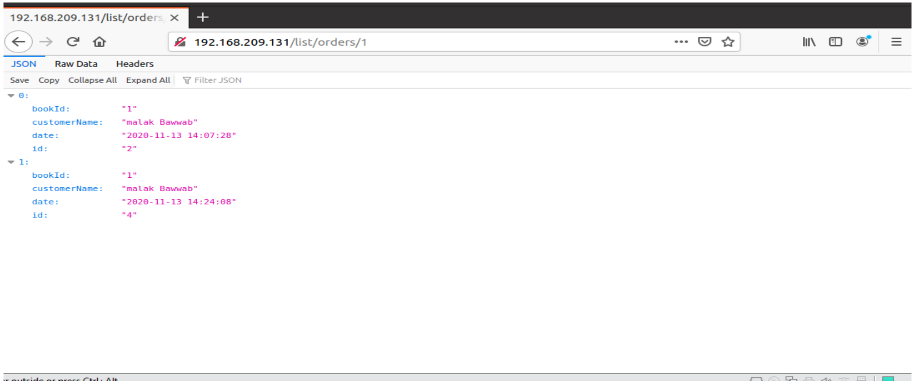
Status: 200 OK Time: 27 ms Size: 275 B

```
Buy faild,no book with this number 88
```

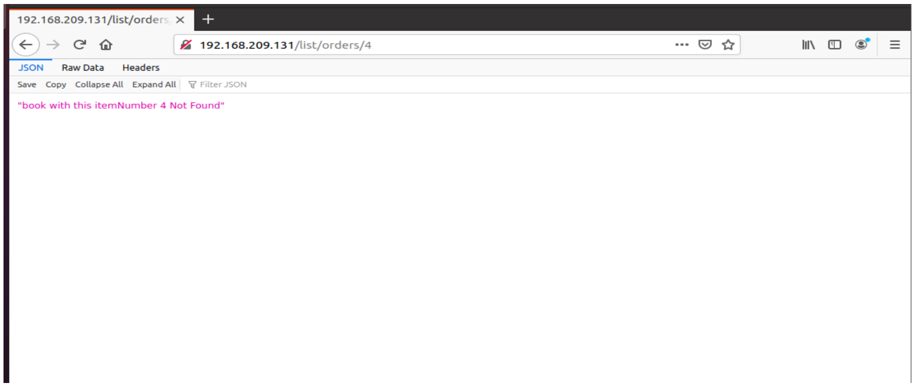

POST http://192.168.209.131/buy/1

Status: 200 OK Time: 31 ms Size: 376 B

```
{"message":"Bought book How to get a good grade in DOS in 20 minutes a day"}
```


192.168.209.131/list/orders/1

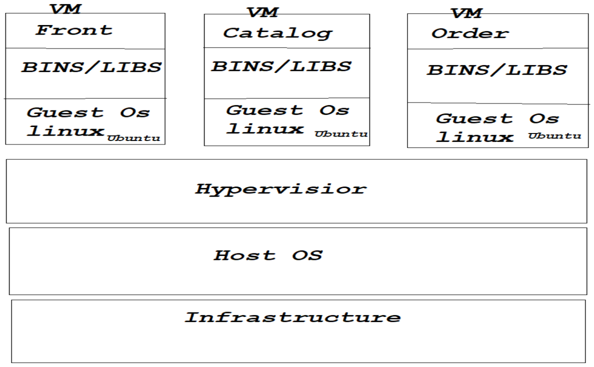
```
0:
    bookId:          "1"
    customerName:    "malak Bawwab"
    date:            "2020-11-13 14:07:28"
    id:              "2"
1:
    bookId:          "1"
    customerName:    "malak Bawwab"
    date:            "2020-11-13 14:24:08"
    id:              "4"
```

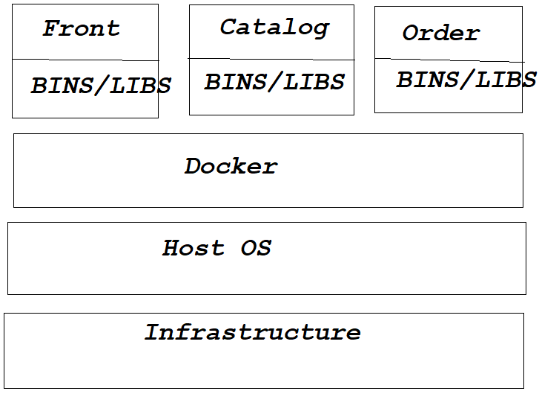


```
"book with this itemNumber 4 Not Found"
```

11 Improvements and design tradoffs

- Use Docker instead of 3 instances of VMs with ubuntu because:
  - Docker containers are lightweight bit VMs are heavyweight.
  - Virtual machines have host OS and the guest OS inside each VM. In contrast, Docker containers host on a single physical server with a host OS, which shares among them. Sharing the host OS between containers makes them light .
  - VMs are slow, each VM includes a full copy of an operating system, the application, and necessary binaries and libraries — taking up tens of GBs.
  - Containers take up less space than VMs .
  - Docker eliminate the need for a guest OS and we save the amount of memory that was occupied by it.




Front
BINS/LIBS
Catalog
BINS/LIBS
Order
BINS/LIBS
Docker
Host OS
Infrastructure