

Distributed Operating Systems

Lab 2

Turning the Bazar into an Amazon: Replication, Caching and Consistency

Walaa Dweikat

Aisha Marmash

As we submitted in Lab 1, the program is built using flask with sqlite database, it is a web REST API which contains three web micro-serves. Every server resides on different machine and they communicate with each other using the network by sending http requests from a server to another.

But now we add three new books to the Bazar.com, and because of the popularity of the book store we need to rearchitecting it to handle the higher in the workload.

We add replication, caching, load balance and consistency to improve request processing latency.

Replication:

We implemented the replication by copying the catalog server to two new servers and also copying the order server to two new servers, so we now have 3 catalogs and 3 orders, but the front end server is not replicated, each one runs in a different port.

Consistency:

We implemented consistency by using http rest calls from the server which have a change in its database to the other two replicas.

Cache consistency needs to be addressed whenever a database entry is updated by buy requests or arrival of new stock of books, so we send invalidate request to the in-memory cache, which will cause the data for that item to be removed from the cache.

Load balance:

We implement load balance using round robin algorithm by send each request from the front end server to one of the replicas by using 2 counters from 1 to 3 one for catalog server and the other one for order server. According to the counter value 1, 2 or 3 we send the request to a specific server. If catalog counter equal 1 we send the request to the first replica, but if it is 2 we send the request to the second replica and so on.

Run Bazar.com:

To run our program, we run all the servers each one on its machine on a different port of the other replica port and use the postman program as a client send request to the front end server, which will resend it to the needed server.

How it is works

At first we ran 3 catalog servers on catalog server machine, each one on different port

3 replicas if catalog server

Catalog_1 on port 2000

```
* Serving Flask app "catalog_1/catalogServer.py" (lazy loading)
* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:2000/ (Press CTRL+L+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/__init__.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/__init__.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
█
```

Catalog_2 on port 3000

```
* Serving Flask app "catalog_2/catalogServer.py" (lazy loading)
* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:3000/ (Press CTRL+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/__init__.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/__init__.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
█
```

Catalog_3 on port 4000

```
* Serving Flask app "catalog_3/catalogServer.py" (lazy loading)
* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:4000/ (Press CTRL
+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages
/flask_marshmallow/_init_.py:26: UserWarning
: Flask-SQLAlchemy integration requires marshm
allow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages
/flask_marshmallow/_init_.py:26: UserWarning
: Flask-SQLAlchemy integration requires marshm
allow-sqlalchemy to be installed.
  warnings.warn(
█
```

Ran 3 replicas of order server on their machine each one on a different port

Order_1 on port 2000

```
C:\Users\dell\Desktop\Lab2>set FLASK_APP=order_1/order.py

C:\Users\dell\Desktop\Lab2>set FLASK_ENV=development

C:\Users\dell\Desktop\Lab2>set FLASK_RUN_PORT=2000

C:\Users\dell\Desktop\Lab2>flask run --host=0.0.0.0
* Serving Flask app 'order_1/order.py' (lazy loading)
* Environment: development
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 878-000-355
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a produ
ction deployment.
* Running on http://192.168.1.121:2000/ (Press CTRL+C to quit)
█
```

Order_2 on port 3000

```
C:\Users\dell\Desktop\Lab2>set FLASK_RUN_PORT=3000

C:\Users\dell\Desktop\Lab2>flask run --host=0.0.0.0
* Serving Flask app 'order_2/order.py' (lazy loading)
* Environment: development
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 878-000-355
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://192.168.1.121:3000/ (Press CTRL+C to quit)
```

Order_3 on port 6000

```
C:\Users\dell\Desktop\Lab2>set FLASK_RUN_PORT=6000

C:\Users\dell\Desktop\Lab2>flask run --host=0.0.0.0
* Serving Flask app 'order_3/order.py' (lazy loading)
* Environment: development
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 878-000-355
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://192.168.1.121:6000/ (Press CTRL+C to quit)
```

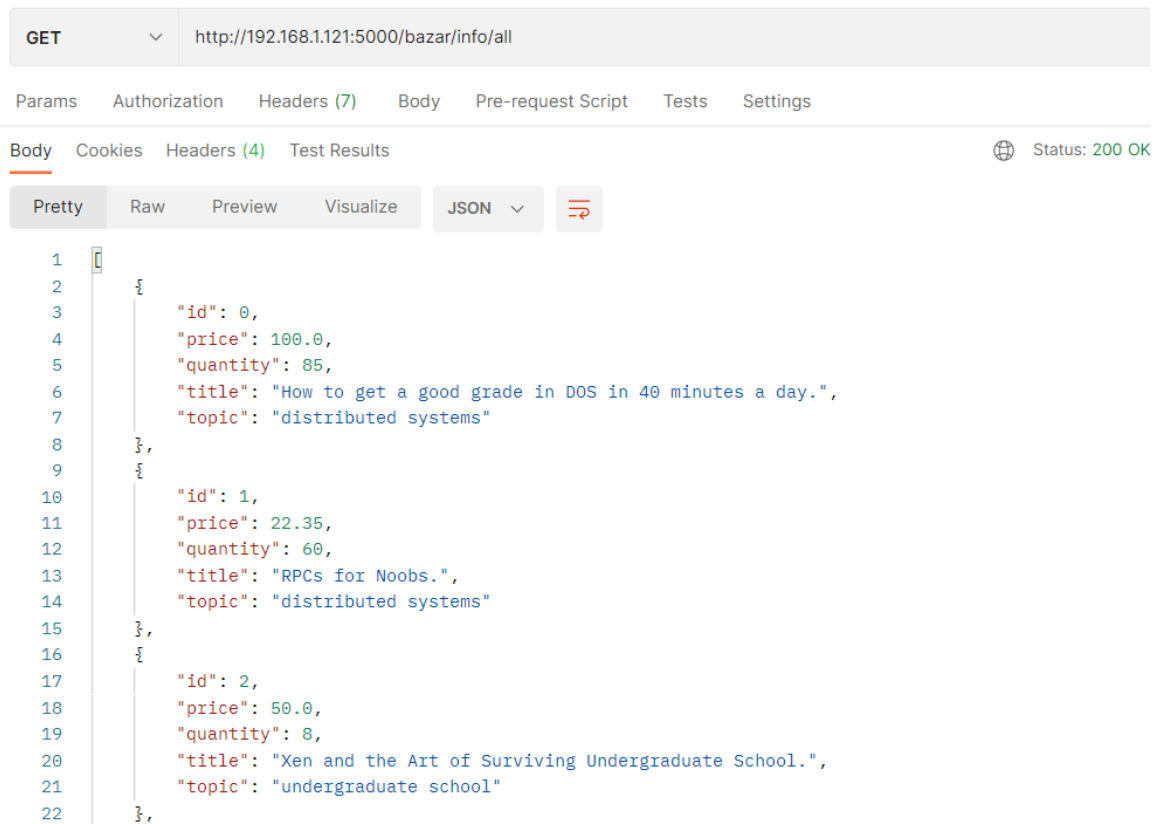
Ran one front end server on port 5000

```
C:\Users\dell\Desktop\Lab2>set FLASK_RUN_PORT=5000

C:\Users\dell\Desktop\Lab2>flask run --host=0.0.0.0
* Serving Flask app 'front-end/front-end server.py' (lazy loading)
* Environment: development
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 878-000-355
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://192.168.1.121:5000/ (Press CTRL+C to quit)
```


And we use postman as client.


- We add three new books to the store, so now we have 7 books



GET http://192.168.1.121:5000/bazar/info/all

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

Body Cookies Headers (4) Test Results  Status: 200 OK

Pretty Raw Preview Visualize JSON 

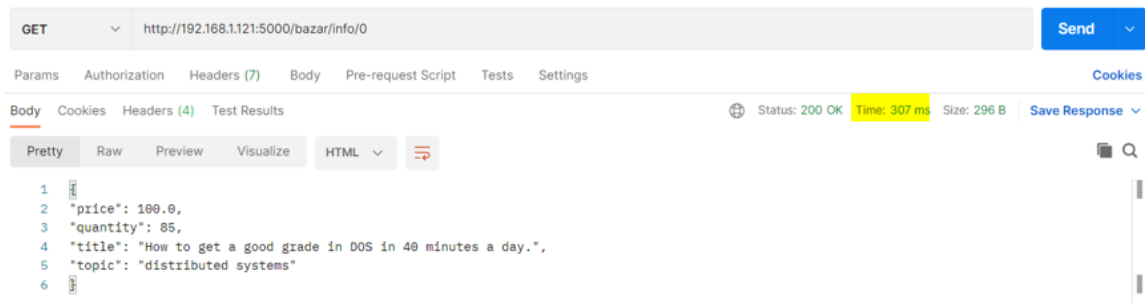
```

1  {
2    "id": 0,
3    "price": 100.0,
4    "quantity": 85,
5    "title": "How to get a good grade in DOS in 40 minutes a day.",
6    "topic": "distributed systems"
7  },
8  {
9    "id": 1,
10   "price": 22.35,
11   "quantity": 60,
12   "title": "RPCs for Noobs.",
13   "topic": "distributed systems"
14 },
15 {
16   "id": 2,
17   "price": 50.0,
18   "quantity": 8,
19   "title": "Xen and the Art of Surviving Undergraduate School.",
20   "topic": "undergraduate school"
21 },
22 }
```

```
23  ✓ {
24      "id": 3,
25      "price": 89.33,
26      "quantity": 65,
27      "title": "Cooking for the Impatient Undergrad.",
28      "topic": "undergraduate school"
29  },
30  ✓ {
31      "id": 4,
32      "price": 70.0,
33      "quantity": 40,
34      "title": "How to finish Project 3 on time ",
35      "topic": "new"
36  },
37  ✓ {
38      "id": 5,
39      "price": 100.0,
40      "quantity": 80,
41      "title": "Why theory classes are so hard ",
42      "topic": "new"
43  },
44  ✓ {
45      "id": 6,
46      "price": 700.0,
47      "quantity": 90,
48      "title": "Spring in the Pioneer Valley ",
49      "topic": "new"
50  }
51  ]
```

Cache and load balance:

Trial 1



At first, we started with empty cache, so when we sent request for “info/0” the front end server resent the request to the one of the catalog replicas because the cache is empty and didn’t have this book information.

```
* Serving Flask app 'front-end/front-end server.py' (lazy loading)
* Environment: development
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 878-000-355
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://192.168.1.121:5000/ (Press CTRL+C to quit)
{'price': 100.0, 'quantity': 85, 'title': 'How to get a good grade in DOS in 40 minutes a day.', 'topic': 'distributed systems'}
192.168.1.121 - - [14/Aug/2021 16:46:27] "GET /bazar/info/0 HTTP/1.1" 200 -
```

front-end server

the request sent to the catalog_1 to read the information of the book with id = 0

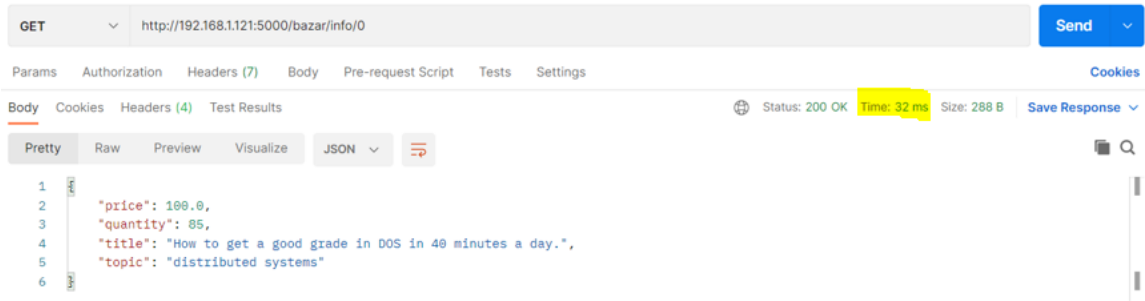
```
* Running on http://0.0.0.0:2000/ (Press CTRL+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
192.168.1.121 - - [14/Aug/2021 16:37:58] "GET /bazar/info/all HTTP/1.1" 200 -
192.168.1.121 - - [14/Aug/2021 16:46:27] "GET /bazar/info/0 HTTP/1.1" 200 -

* Serving Flask app "catalog_2/catalogServer.py" (lazy loading)
* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:3000/ (Press CTRL+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(

* Serving Flask app "catalog_3/catalogServer.py" (lazy loading)
* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:4000/ (Press CTRL+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
```

The response returned from the catalog_1 to the front end server then to the client and stored a copy of the information on the cache,so when the client sent this request for the first time it needed 307 ms but when he sent it again it spend 32 ms, because the needed information stored in the front end cash, and the front end server returnd it without sending request to the catalog server.

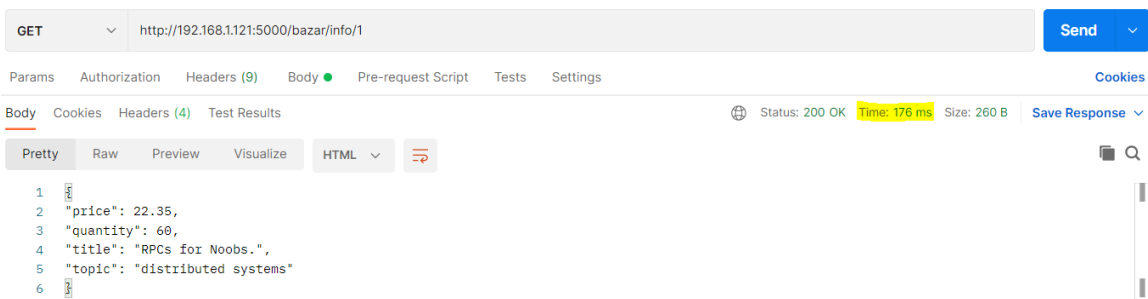
After caching



Trial 2

As previous trial when we asked for information not found in the cache, front end server brought it from the catalog server, but here we noticed load balance, where the front end server sent the request to the second replica of the catalog server, stored its response in the cache for the next requests.

Without cache



Front end server sent request to catalog_2

```
* Debugger is active!
* Debugger PIN: 878-000-355
* Running on all addresses.
  WARNING: This is a development server. Do not
  use it in a production deployment.
* Running on http://192.168.1.121:5000/ (Press C
TRL+C to quit)
{'price': 100.0, 'quantity': 85, 'title': 'How to
get a good grade in DOS in 40 minutes a day.', '
topic': 'distributed systems'}
192.168.1.121 - - [14/Aug/2021 16:46:27] "GET /ba
zar/info/0 HTTP/1.1" 200 -
192.168.1.121 - - [14/Aug/2021 16:47:55] "GET /ba
zar/info/0 HTTP/1.1" 200 -
{'price': 22.35, 'quantity': 60, 'title': 'RPCs f
or Noobs.', 'topic': 'distributed systems'}
192.168.1.121 - - [14/Aug/2021 16:49:07] "GET /ba
zar/info/1 HTTP/1.1" 200 -
```

The response brought from catalog_2 (load balance) and stored in the cache memory

```
* Running on http://0.0.0.0:2000/ (Press CTRL+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
192.168.1.121 - - [14/Aug/2021 16:37:58] "GET /bazar/info/all HTTP/1.1" 200 -
192.168.1.121 - - [14/Aug/2021 16:46:27] "GET /bazar/info/0 HTTP/1.1" 200 -

* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:3000/ (Press CTRL+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
192.168.1.121 - - [14/Aug/2021 16:49:07] "GET /bazar/info/1 HTTP/1.1" 200 -

* Serving Flask app "catalog_3/catalogServer.py" (lazy loading)
* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:4000/ (Press CTRL+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages/flask_marshmallow/_init_.py:26: UserWarning: Flask-SQLAlchemy integration requires marshmallow-sqlalchemy to be installed.
  warnings.warn(
```

With cache

```
GET http://192.168.1.121:5000/bazar/info/1
Status: 200 OK Time: 20 ms Size: 252 B
{"price": 22.35, "quantity": 60, "title": "RPCs for Noobs.", "topic": "distributed systems"}
```

Trial 3

Here the request sent from client to the front end server, then to catalog_3(load balance). Response returned from the catalog to the front end server, stored in the cache and returned the response to the client, and the time we needed was 294 ms.

```
GET http://192.168.1.121:5000/bazar/info/2
Status: 200 OK Time: 294 ms Size: 294 B
{"price": 50.0, "quantity": 8, "title": "Xen and the Art of Surviving Undergraduate School.", "topic": "undergraduate school"}
```

```

* Running on http://192.168.1.121:5000/ (Press C
TRL+C to quit)
{'price': 100.0, 'quantity': 85, 'title': 'How to
get a good grade in DOS in 40 minutes a day.', '
topic': 'distributed systems'}
192.168.1.121 - - [14/Aug/2021 16:46:27] "GET /ba
zar/info/0 HTTP/1.1" 200 -
192.168.1.121 - - [14/Aug/2021 16:47:55] "GET /ba
zar/info/0 HTTP/1.1" 200 -
{'price': 22.35, 'quantity': 60, 'title': 'RPCs f
or Noobs.', 'topic': 'distributed systems'}
192.168.1.121 - - [14/Aug/2021 16:49:07] "GET /ba
zar/info/1 HTTP/1.1" 200 -
{'price': 50.0, 'quantity': 8, 'title': 'Xen and
the Art of Surviving Undergraduate School.', 'top
ic': 'undergraduate school'}
192.168.1.121 - - [14/Aug/2021 16:51:25] "GET /ba
zar/info/2 HTTP/1.1" 200 -

```

Front end server

Sent to catalog_3

```

* Running on http://0.0.0.0:2000/ (Press CTR
L+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-package
s/flask_marshmallow/_init_.py:26: UserWarni
ng: Flask-SQLAlchemy integration requires mar
shmallow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-package
s/flask_marshmallow/_init_.py:26: UserWarni
ng: Flask-SQLAlchemy integration requires mar
shmallow-sqlalchemy to be installed.
  warnings.warn(
192.168.1.121 - - [14/Aug/2021 16:37:58] "GET
/bazar/info/all HTTP/1.1" 200 -
192.168.1.121 - - [14/Aug/2021 16:46:27] "GET
/bazar/info/0 HTTP/1.1" 200 -

* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:3000/ (Press CTRL+
C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages
/flask_marshmallow/_init_.py:26: UserWarning:
Flask-SQLAlchemy integration requires marshmall
ow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages
/flask_marshmallow/_init_.py:26: UserWarning:
Flask-SQLAlchemy integration requires marshmall
ow-sqlalchemy to be installed.
  warnings.warn(
192.168.1.121 - - [14/Aug/2021 16:49:07] "GET /
bazar/info/1 HTTP/1.1" 200 -

* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:4000/ (Press CTRL
+C to quit)
* Restarting with stat
/home/aisha/.local/lib/python3.8/site-packages
/flask_marshmallow/_init_.py:26: UserWarning:
Flask-SQLAlchemy integration requires marshm
allow-sqlalchemy to be installed.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 349-933-185
/home/aisha/.local/lib/python3.8/site-packages
/flask_marshmallow/_init_.py:26: UserWarning:
Flask-SQLAlchemy integration requires marshm
allow-sqlalchemy to be installed.
  warnings.warn(
192.168.1.121 - - [14/Aug/2021 16:51:25] "GET
/bazar/info/2 HTTP/1.1" 200 -

```

Second request got the response from the cache with 21 ms.

GET http://192.168.1.121:5000/bazar/info/2

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

Body Cookies Headers (4) Test Results Status: 200 OK Time: 21 ms Size: 286 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "price": 50.0,
3   "quantity": 8,
4   "title": "Xen and the Art of Surviving Undergraduate School.",
5   "topic": "undergraduate school"
6 }

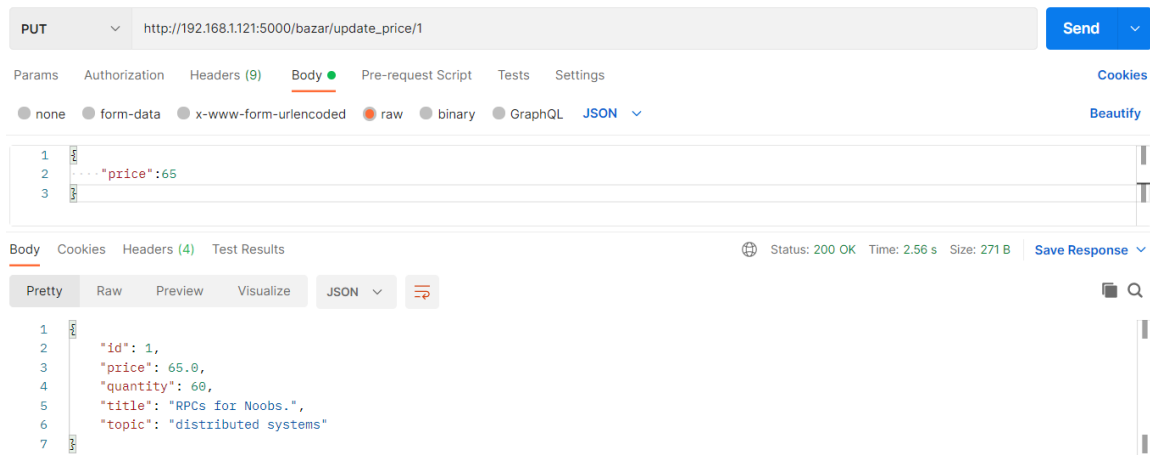
```

The cache has limited size, less than the catalog database size. So when the cache is full and we need to store new entry inside it we use LRU algorithm to replace it with one of the entries inside the cache.

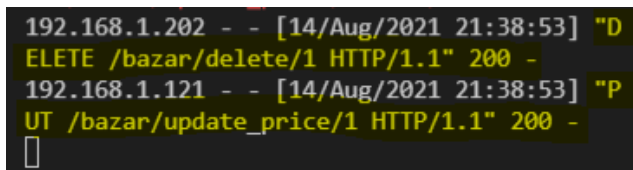
Consistency

When we update information in the catalog database with “PUT” or “POST” url calls, the same update sent to the other replicas by using http rest calls, and we send invalidate request to the cache memory, which will cause to delete the information about the changed book.

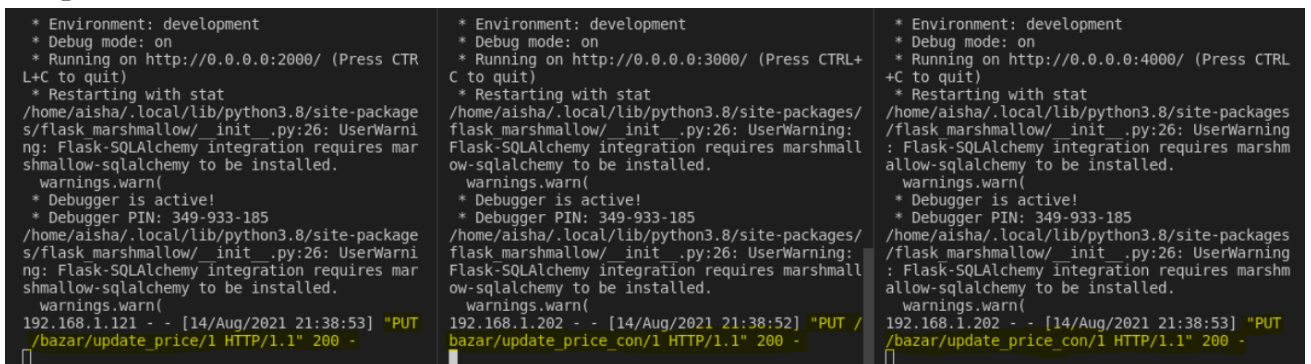
Here we sent request to update price of book1 (write operation)



So we delete the entry of this book from the cache by sending invalidate request from the catalog to the cache



The request went to catalog_1, but then requests sent from it to the other replicas to update the same information on their databases.

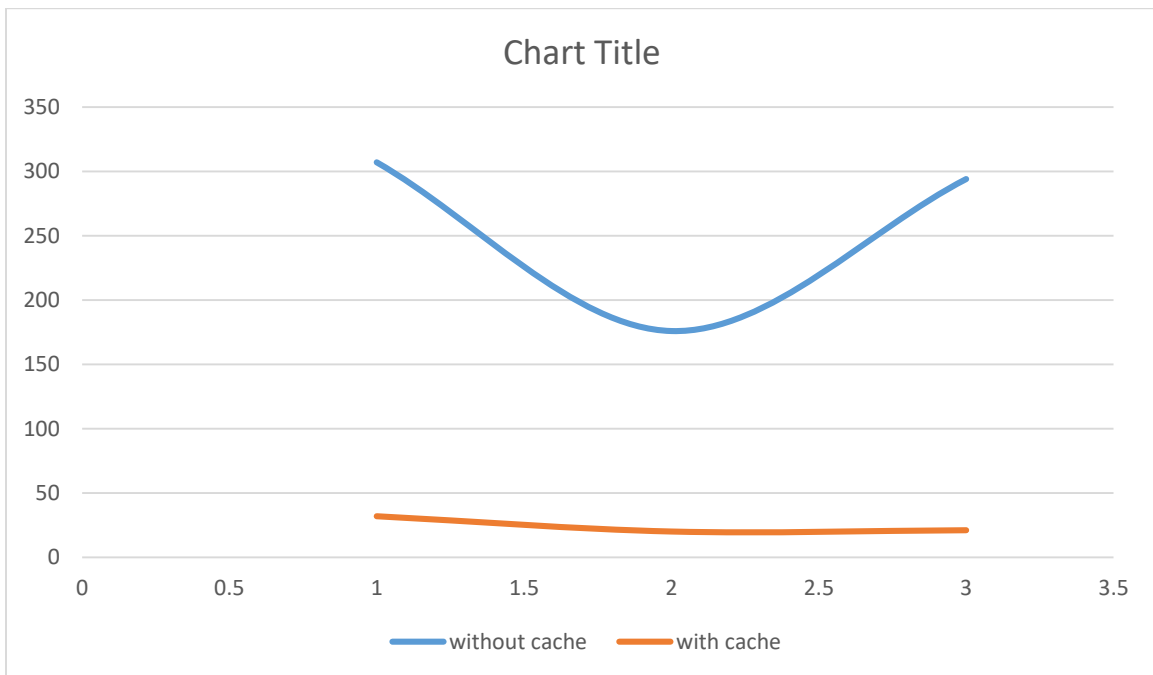


Here if we got the book 1 information from any catalog server, it will be the same.

And the same if we did buy operation on order server, it will store the new order in the all replicas databases, and update book quantity on all the catalog replicas.

- Compute the average response time (query/buy) of your new systems. What is the response time with and without caching? How much does caching help?

	Time without cache	Time with cache
Trial 1	307ms	32ms
Trial 2	176ms	20ms
Trial 3	294ms	21ms



average response time without cache = $(307+176+294)/3$

$$= 777/3$$

$$= 259\text{ms}$$

average response time with cache = $(32+20+21)/3$

$$= 73/3$$

$$= 24\text{ms}$$

Performance : $259/24.33 = 10.6$ so the new performance 10.6* old performance

Cache increase decrease the latency time so we got the data faster.

Search operation

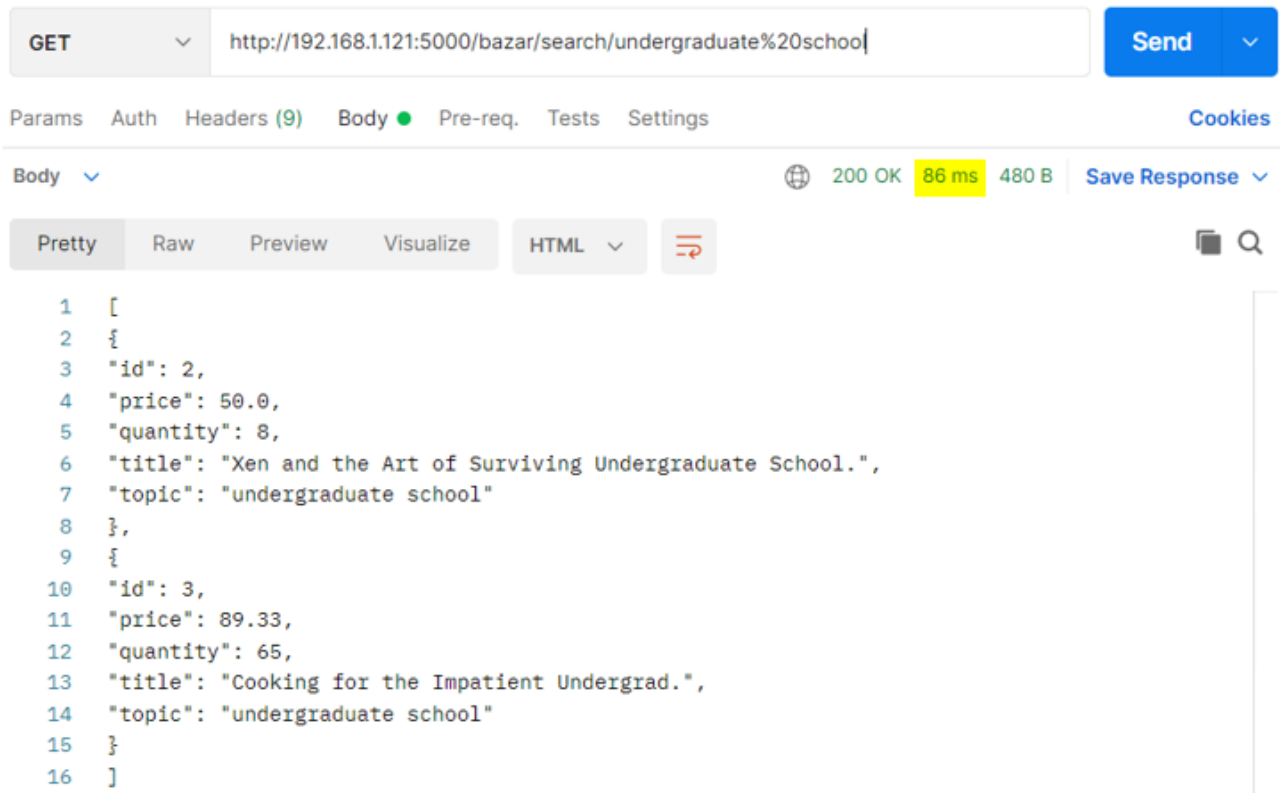
Also we use cache for search operation (read operation). For example, when I searched on a topic as a client and I got a response, returned the books from the catalog, and the books stored on the cache, so when I need to get information about one of them I got it from the cache memory with low latency. When the cache is full we will replace the new books with the least ordered book in the cache.

Starting with empty cache, we searched on “distributed systems” books, we got it from the catalog database and stored it in the cache database, it spent 127ms.

The screenshot shows a web browser's developer tools interface. At the top, a GET request is shown to the URL `http://192.168.1.121:5000/bazar/search/distributed%20systems`. The response status is 200 OK, with a response time of 127 ms and a body size of 460 B. The response body is displayed in JSON format, showing two book entries:

```
1 {
2   "id": 0,
3   "price": 100.0,
4   "quantity": 85,
5   "title": "How to get a good grade in DOS in 40 minutes a day.",
6   "topic": "distributed systems"
7 },
8 {
9   "id": 1,
10  "price": 22.35,
11  "quantity": 60,
12  "title": "RPCs for Noobs.",
13  "topic": "distributed systems"
14 }
15 }
```

And we searched on “undergraduate school”



GET Send

Params Auth Headers (9) Body ● Pre-req. Tests Settings Cookies

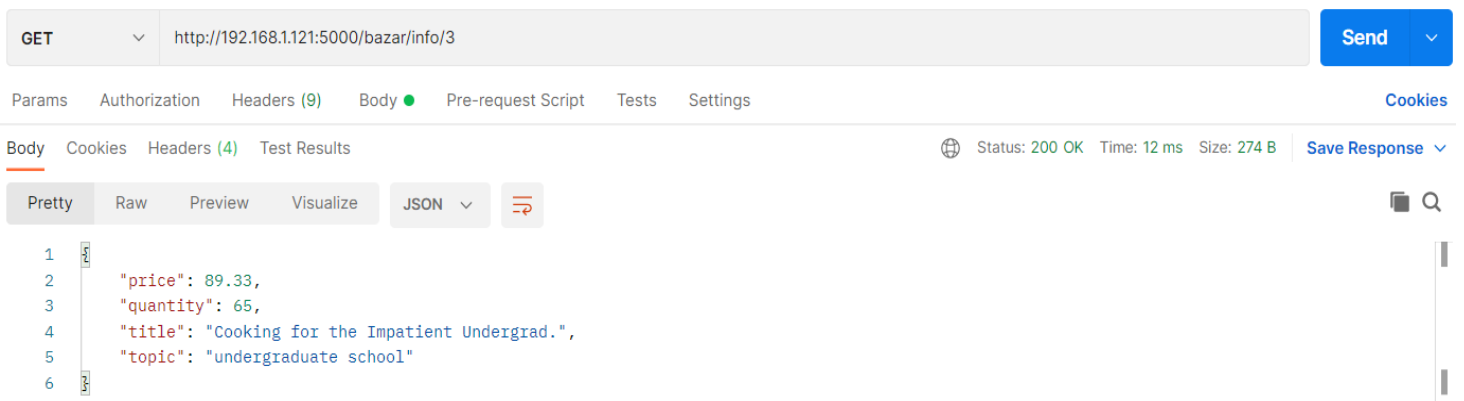
Body Save Response

Pretty Raw Preview Visualize HTML

```
1 [
2   {
3     "id": 2,
4     "price": 50.0,
5     "quantity": 8,
6     "title": "Xen and the Art of Surviving Undergraduate School.",
7     "topic": "undergraduate school"
8   },
9   {
10    "id": 3,
11    "price": 89.33,
12    "quantity": 65,
13    "title": "Cooking for the Impatient Undergrad.",
14    "topic": "undergraduate school"
15  }
16 ]
```

But the cache size is 5, and when we sent search requests of 2 different topics, cache now has 4 entries so there is a space for 1 book now and if I want to search of topic 3 “new” there are 3 new books come to the cache, so we replace the three new books with the least ordered ones.

So if I do requests for books 3 and 4 again their will ordered more than others on the cache.



GET Send

Params Authorization Headers (9) Body ● Pre-request Script Tests Settings Cookies

Body Cookies Headers (4) Test Results Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "price": 89.33,
3   "quantity": 65,
4   "title": "Cooking for the Impatient Undergrad.",
5   "topic": "undergraduate school"
6 }
```

GET ⌵ http://192.168.1.121:5000/bazar/info/4 Send ⌵

Params Authorization Headers (9) Body ● Pre-request Script Tests Settings Cookies

Body Cookies Headers (4) Test Results 🌐 Status: 200 OK Time: 18 ms Size: 252 B Save Response ⌵

Pretty Raw Preview Visualize JSON ⌵ ≡

```
1 [
2   "price": 70.0,
3   "quantity": 40,
4   "title": "How to finish Project 3 on time ",
5   "topic": "new"
6 ]
```

Notice the low latency, because we got this information from the cache.

Books 0, 1 and 2 are least ordered so when I search of “new” topic book it will replace them with 0, 1 and 2 books on the cache.

GET ⌵ http://192.168.1.121:5000/bazar/search/new Send ⌵

Params Authorization Headers (9) Body ● Pre-request Script Tests Settings Cookies

Body Cookies Headers (4) Test Results 🌐 Status: 200 OK Time: 215 ms Size: 556 B Save Response ⌵

Pretty Raw Preview Visualize JSON ⌵ ≡

```
1 [
2   {
3     "id": 4,
4     "price": 70.0,
5     "quantity": 40,
6     "title": "How to finish Project 3 on time ",
7     "topic": "new"
8   },
9   {
10    "id": 5,
11    "price": 100.0,
12    "quantity": 80,
13    "title": "Why theory classes are so hard ",
14    "topic": "new"
15  },
16  {
17    "id": 6,
18    "price": 700.0,
19    "quantity": 90,
20    "title": "Spring in the Pioneer Valley ",
21    "topic": "new"
22  }
23 ]
```

So now if we need book 0, 1 and 2 information the front end server will bring it for the client from the catalog server with low latency not from the cache.

- Construct a simple experiment that issues orders or catalog updates (i.e., database writes) to invalidate the cache and maintain cache consistency.

In the previous example of consistency, the time we needed was 2.56s to finish the update price operation, because we need to send delete request to the cache and update requests to the other catalog replicas, so it will spend a lot of time.

- What are the overhead of cache consistency operations?

Buy or update operation took long time, because it needs cache consistency.

- What is the latency of a subsequent request that sees a cache miss?

The latency increased because of cache miss, for example when we got the data from the cache it spent 20ms, but when there is a cache miss it spent 176ms to finish the operation.