Distributed Operating Systems

Final project - Part 2

Project Report

Student Name: Duaa Braik

Student ID: 11820272

Submitted to: Dr. Samer Arandi

Introduction:

In this part of the project, we want to add replication and caching to the bookstore system in order to improve the performance and lower the overhead of requests between client and servers.

We will implement this by:

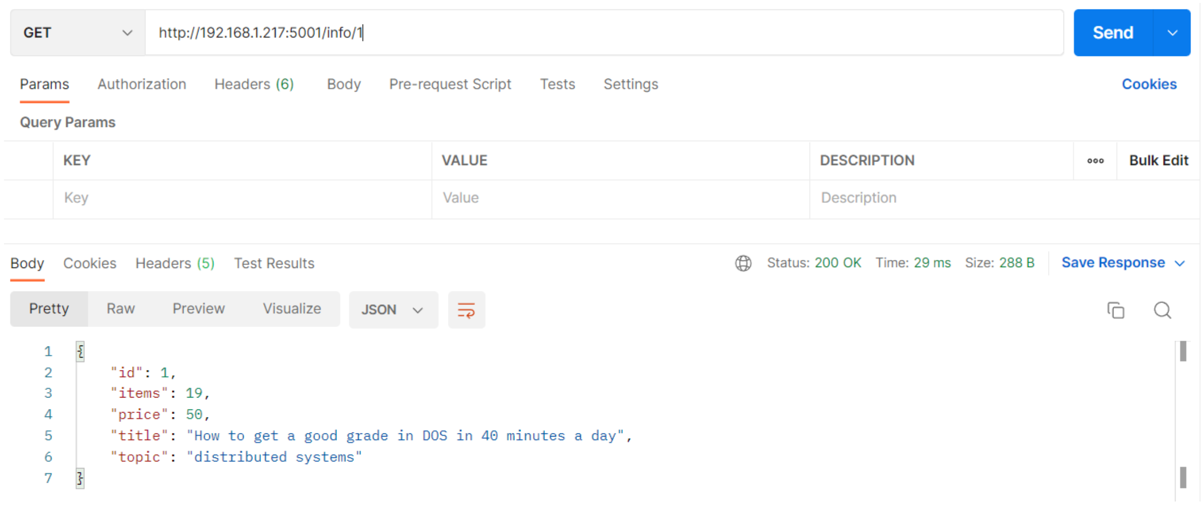
- **Adding a cache server**, using Redis which is a technology used to implement a cache server, between the frontend server and backend servers that caches the responses coming from catalog server in order to reduce the number of requests going to servers and decrease response time.
- **Replication:** each of the catalog server and order servers are replicated (each has 2 copies), and each of the catalog servers has its own database.

- **Load balancing:** is implemented using round-robin algorithm.

- **Consistency:** replication requires consistency; each of the databases should be updated on any update request, and cache could be updated on invalidated, but in this part I updated the cache directly.

1) Caching:

Using Redis, I implemented a cache server that saves the requests results coming from the servers, and this technique reduced the response time as shown in figure 1.a & figure 1.b:

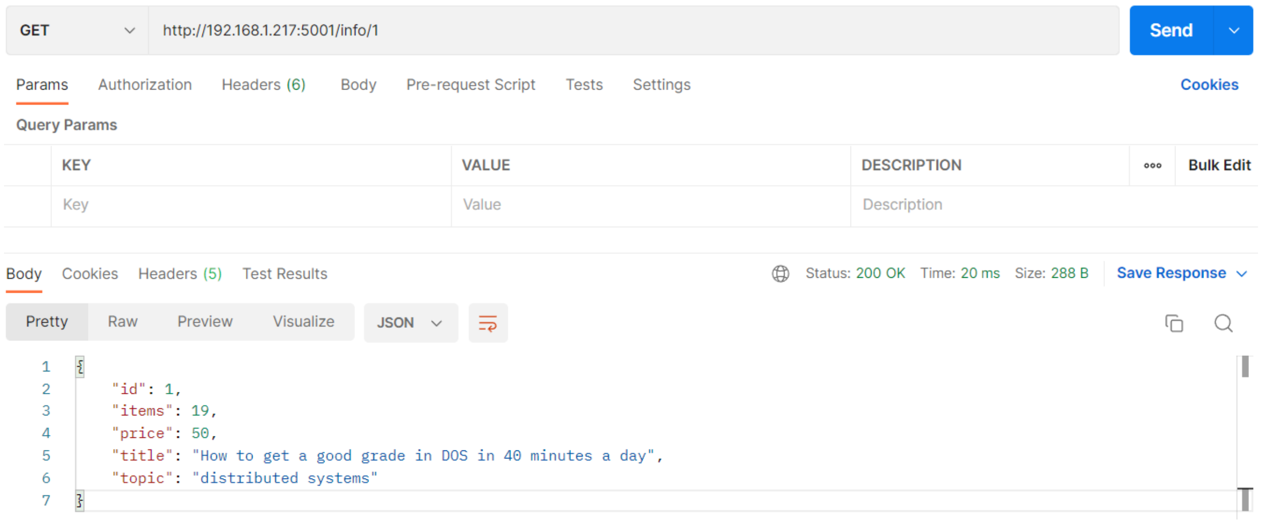


http://192.168.1.217:5001/info/1

Status: 200 OK Time: 29 ms Size: 288 B

```
{
    "id": 1,
    "items": 19,
    "price": 50,
    "title": "How to get a good grade in DOS in 40 minutes a day",
    "topic": "distributed systems"
}
```

figure 1.a

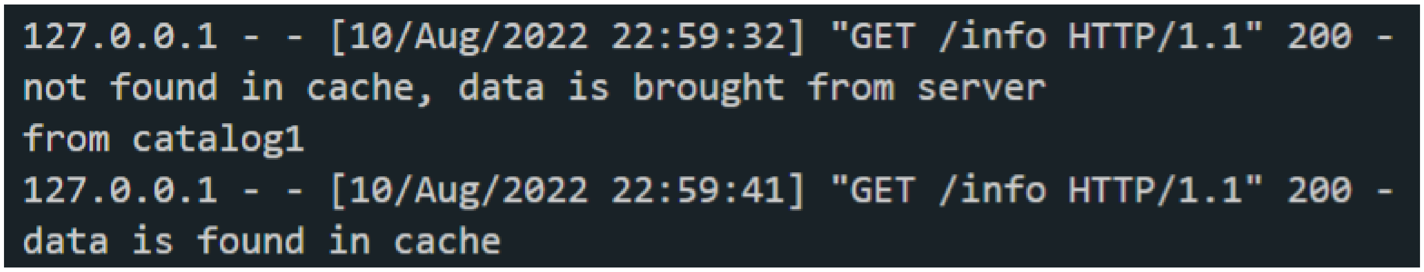


```
{
    "id": 1,
    "items": 19,
    "price": 50,
    "title": "How to get a good grade in DOS in 40 minutes a day",
    "topic": "distributed systems"
}
```

figure 1.b

Response time is reduced from 29ms to 20ms

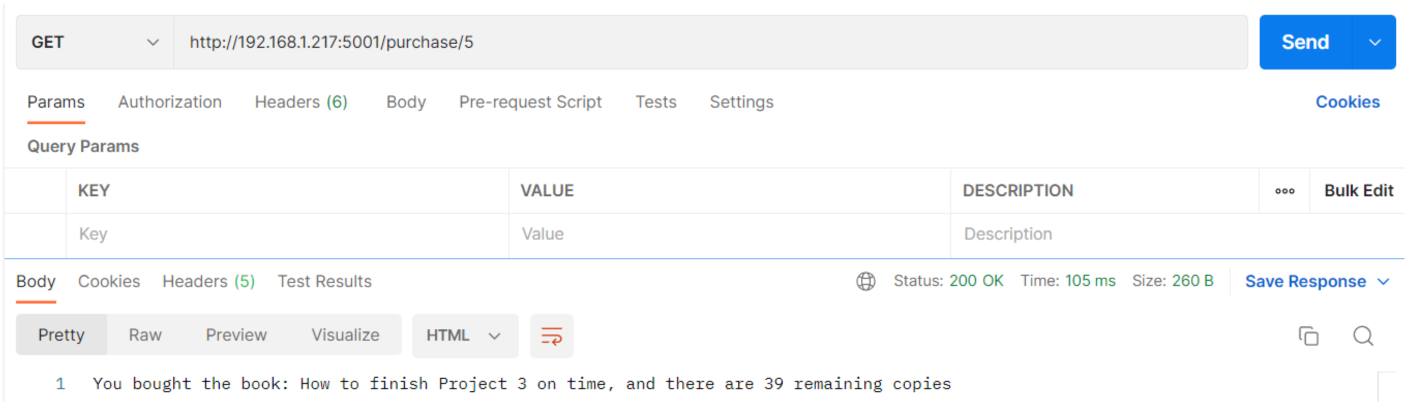
Output in console:



```
127.0.0.1 - - [10/Aug/2022 22:59:32] "GET /info HTTP/1.1" 200 -
not found in cache, data is brought from server
from catalog1
127.0.0.1 - - [10/Aug/2022 22:59:41] "GET /info HTTP/1.1" 200 -
data is found in cache
```

2) Load balancing:

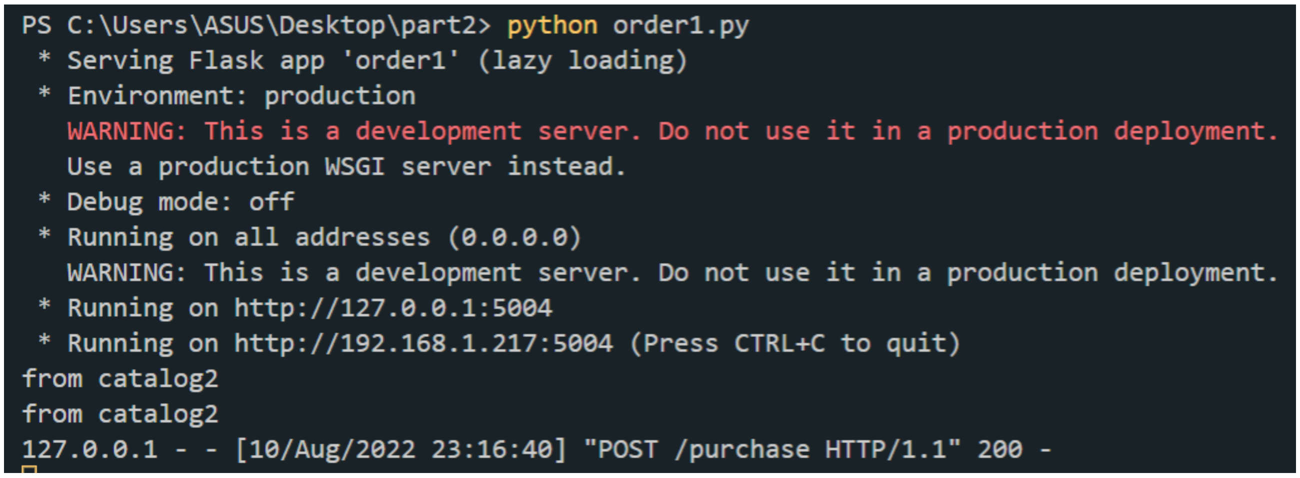
Using round-robin algorithm, requests are sent to different servers.



```
http://192.168.1.217:5001/purchase/5
Status: 200 OK   Time: 105 ms   Size: 260 B
You bought the book: How to finish Project 3 on time, and there are 39 remaining copies
```

figure2

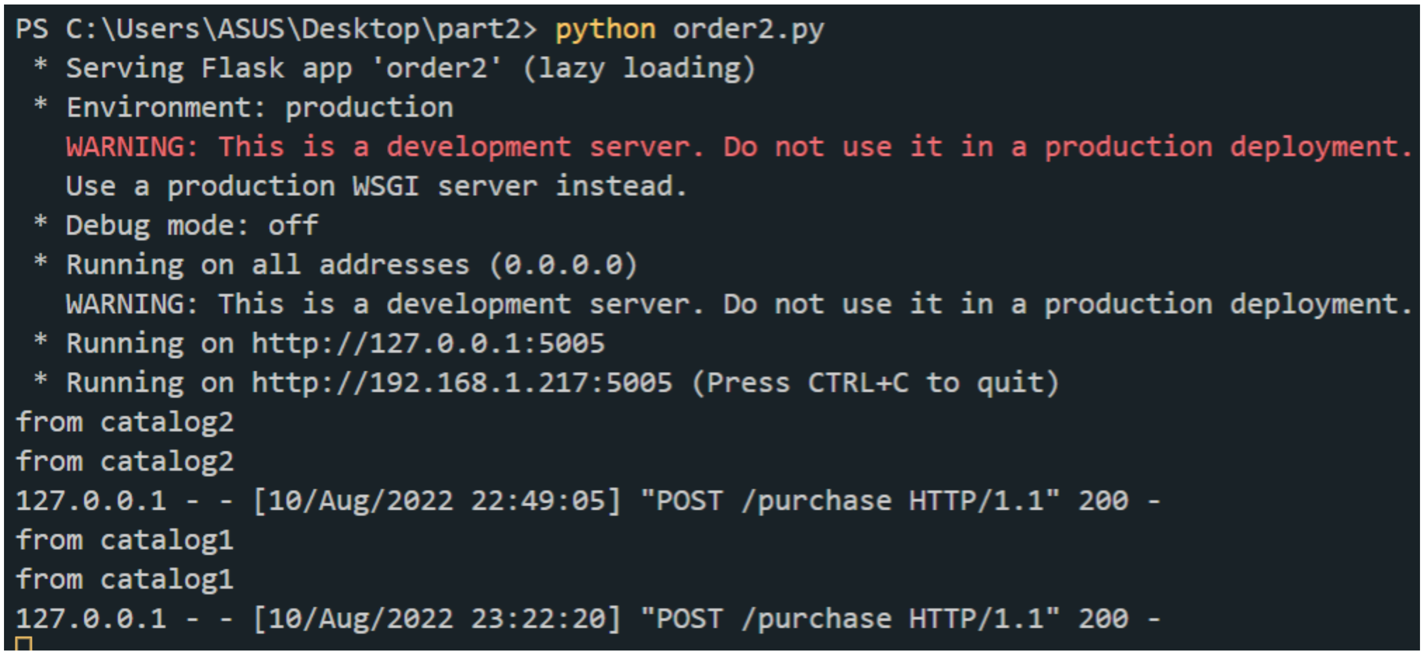
The request in figure2 is sent to order1 and it sends info and update requests to catalog2, as shown in figure 3.



```
PS C:\Users\ASUS\Desktop\part2> python order1.py
 * Serving Flask app 'order1' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses (0.0.0.0)
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://127.0.0.1:5004
 * Running on http://192.168.1.217:5004 (Press CTRL+C to quit)
from catalog2
from catalog2
127.0.0.1 - - [10/Aug/2022 23:16:40] "POST /purchase HTTP/1.1" 200 -
```

figure 3

Then I sent the same purchase request one more time, and it was sent to order2 and it sent info and update requests to catalog1, as shown in figure 4.

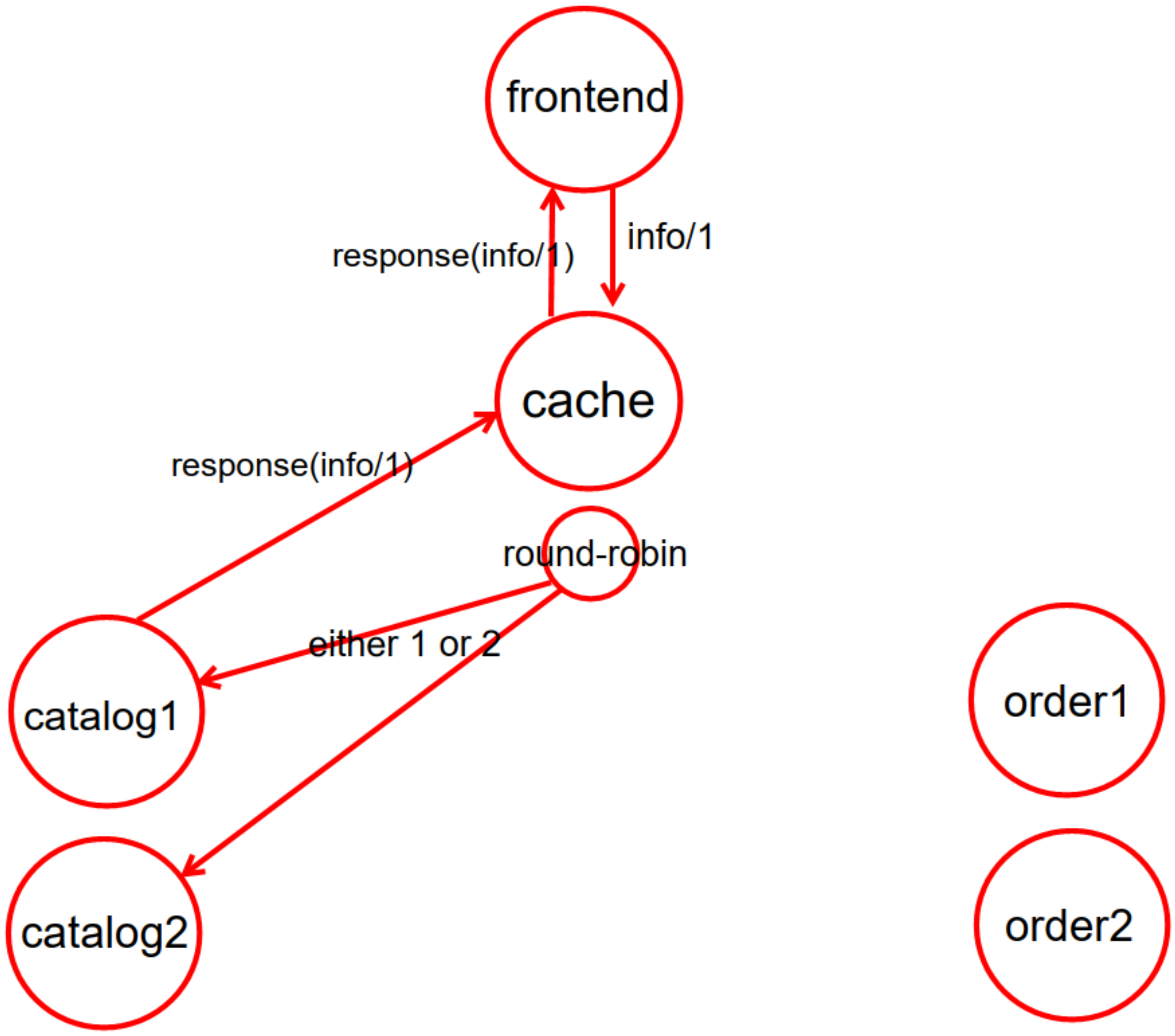


```
PS C:\Users\ASUS\Desktop\part2> python order2.py
 * Serving Flask app 'order2' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses (0.0.0.0)
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://127.0.0.1:5005
 * Running on http://192.168.1.217:5005 (Press CTRL+C to quit)
from catalog2
from catalog2
127.0.0.1 - - [10/Aug/2022 22:49:05] "POST /purchase HTTP/1.1" 200 -
from catalog1
from catalog1
127.0.0.1 - - [10/Aug/2022 23:22:20] "POST /purchase HTTP/1.1" 200 -
```

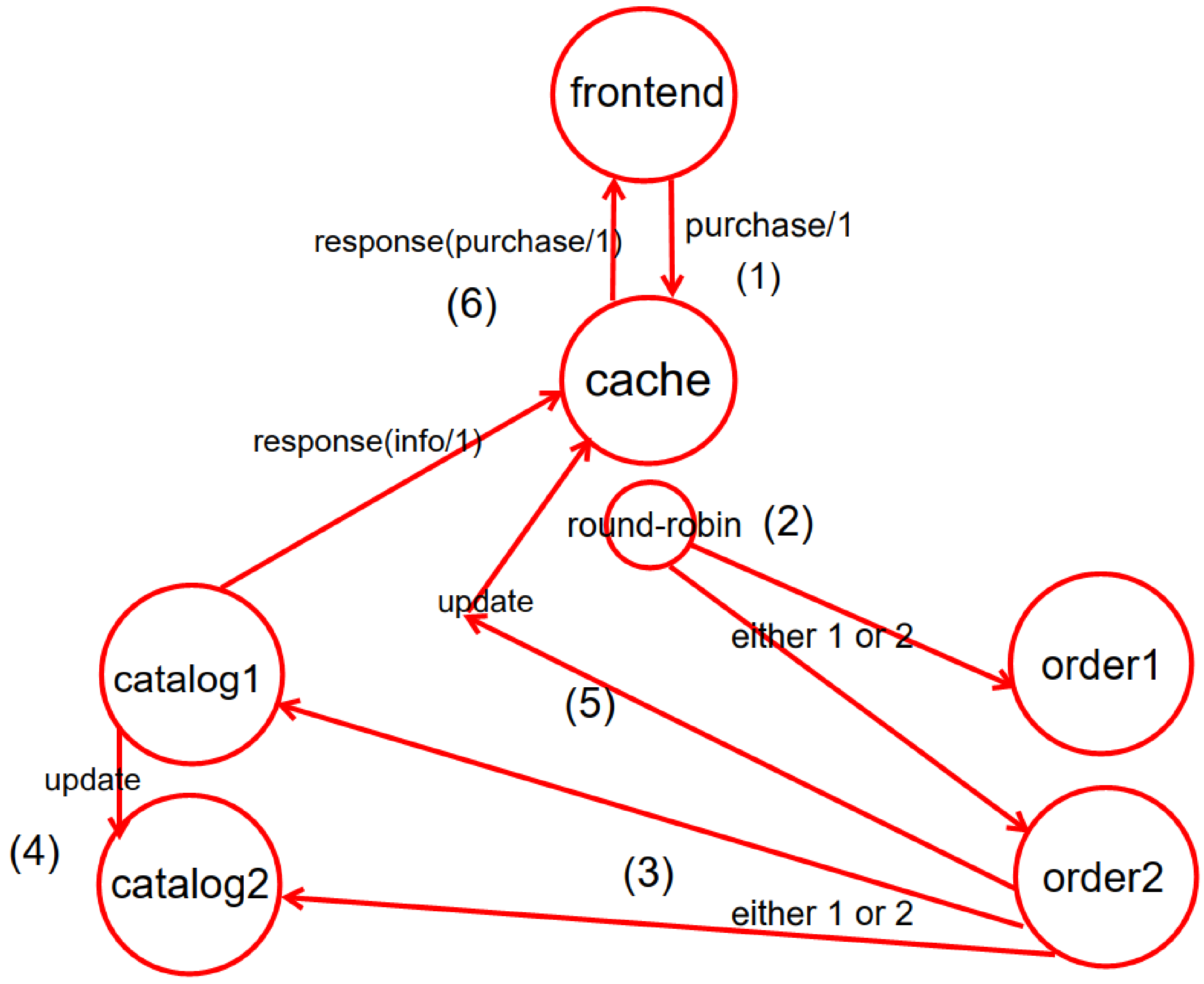
Figure 4

3) Consistency:

Because we have replicated data over the 2 catalog servers, we have to maintain some kind of consistency; each purchase request is sent to any of the catalog servers should then send update request to the other server to update its database.



Info request



Purchase request