DOS project part2

By:

Asala Tibi (11819226)

Hala Barqawi (11821121)

Overall program :

In this part we modify part1 to add replication and caching to improve request processing latency .

In frontend server we add in-memory cache that caches the results of requests to the order and catalog server . It use load balancing algorithm (round robin) to distribute the requests to one of the replicas.

The catalog and order server are replicated and run on virtual machines . when update one of replica , another replica will update so I need to consistency , I used push technique where backend servers send invalidate to cache when update happens ,so the information in cache will delete .

How it works:

- When query request sends to frontend server ,The frontend first check the cache if it has the required information in cache it will return it and doesn't need to contact with backend servers. If not ,it will forward the request to one replica of catalog server using robin-round algorithm .

- When buy request sends to frontend server. It will forward to one of replica of order server . the update occurs on database .so another replica will updates and the server push invalidate to cache to delete the old information.

Trade-offs:

The frontend server may become a bottleneck if there are overhead requests .

Round-robin algorithm simple but there is a lot of algorithm get a better results .

Possible improvement :

Use better consistency , may update the value in cache instead of delete it .

How run the program:

- on my machine i run the frontend server on port 3001 , so clients can send requests on it by using browser or postman

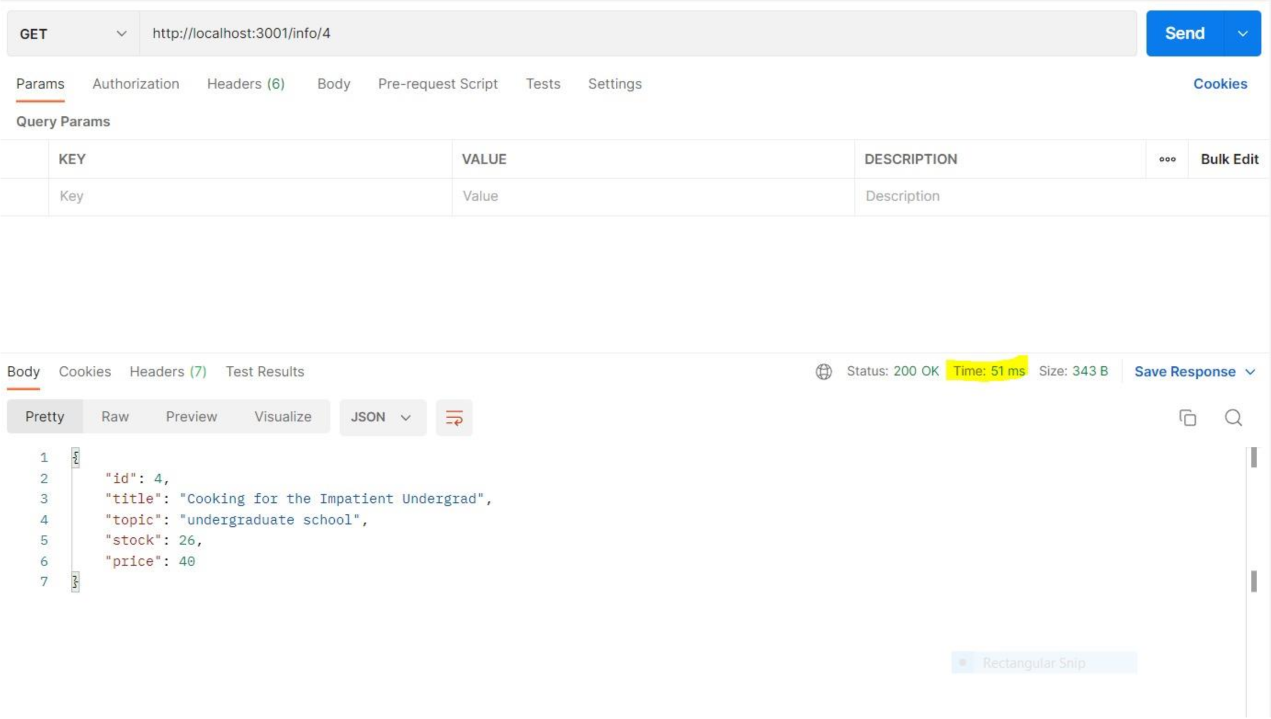
- on the first virtual machine i run one replica of catalog server on port 5000 and order server on port 3004 .

- on the second virtual machine i run another replica of catalog server on port 5001 and order server on port 3003 .

Experimental Evaluation and Measurements

	With cache	Without cache
average response time	8 ms	51 ms

Without cache

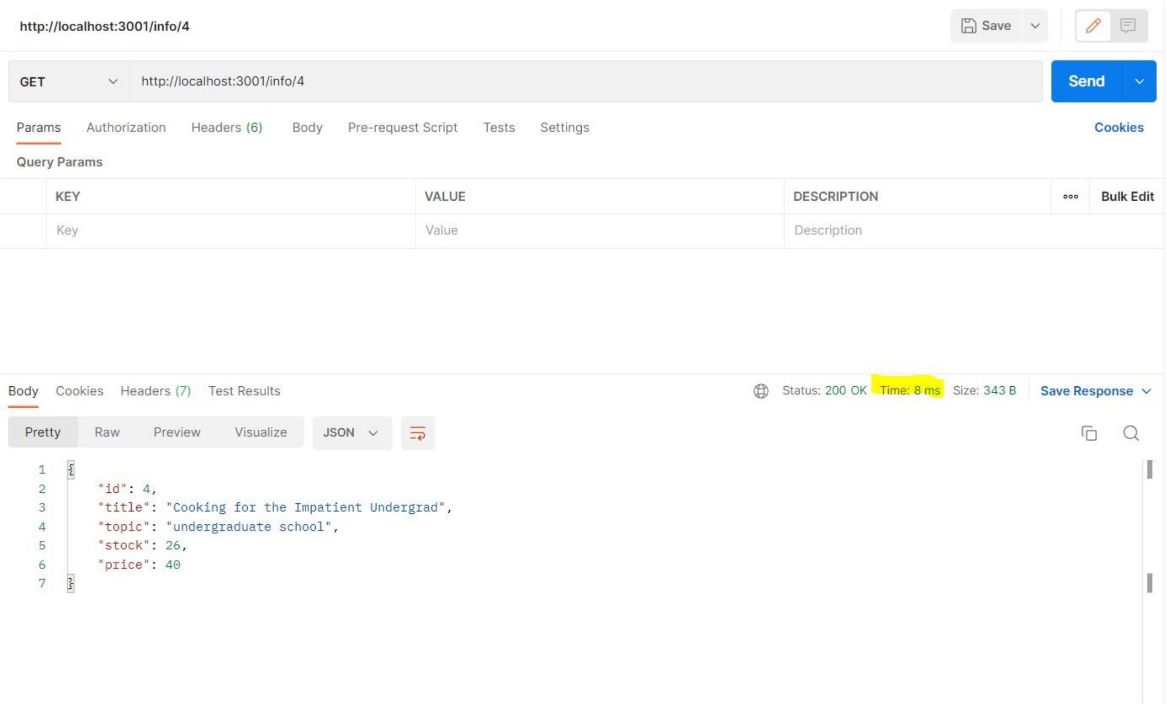


GET http://localhost:3001/info/4

Status: 200 OK Time: 51 ms Size: 343 B

```
{
    "id": 4,
    "title": "Cooking for the Impatient Undergrad",
    "topic": "undergraduate school",
    "stock": 26,
    "price": 40
}
```

With cache



http://localhost:3001/info/4

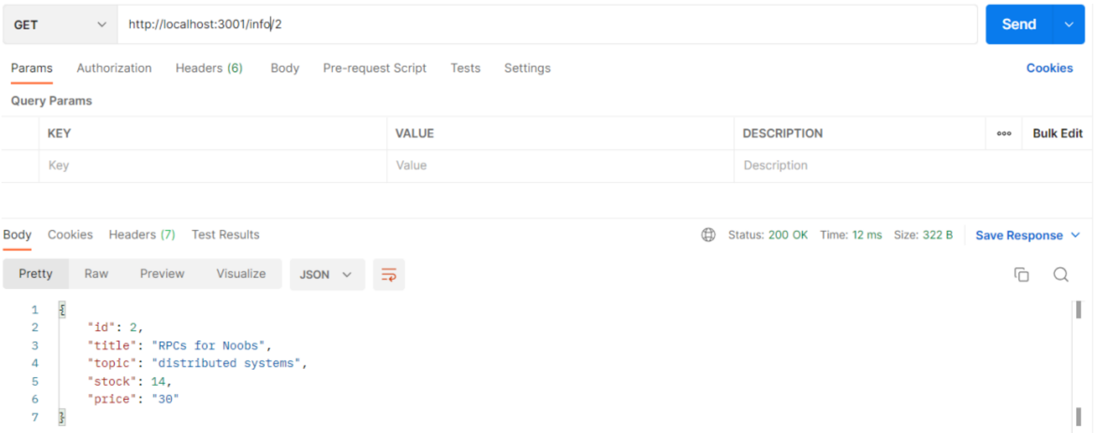
GET http://localhost:3001/info/4 Send

Status: 200 OK Time: 8 ms Size: 343 B

```
{
    "id": 4,
    "title": "Cooking for the Impatient Undergrad",
    "topic": "undergraduate school",
    "stock": 26,
    "price": 40
}
```

The cache will decrease the response time

- Construct a simple experiment that issues orders or catalog updates (i.e., database writes) to invalidate the cache and maintain cache consistency. What are the overhead of cache consistency operations? What is the latency of a subsequent request that sees a cache miss?



GET http://localhost:3001/info/2 Send

Status: 200 OK Time: 12 ms Size: 322 B

```
{
    "id": 2,
    "title": "RPCs for Noobs",
    "topic": "distributed systems",
    "stock": 14,
    "price": "30"
}
```

Cache miss = 12ms