Bazar.com Report

Nael Saleh

May 2020

1 Overview

I use lumen framework on Ubuntu OS using three virtual machines, all communication between servers are based on REST:

- Front-End Server.
- Order Server.
- Catalogue Server.

After editing the system we have 5 servers on three virtual machines so to use my system you have to install **lumen** micro service which use PHP scripting language, and then download the code, there is 5 folders one for each server as following:

- Catalogue and Rcatalogue for Catalogue server and it's replica, they should be in same machine.
- Order and Rorder for Order server and it's replica , the also should be in same machine.
- Frontend for Front-End server.

NOTE: I mean by (in same machine) to make my system work as I structured it, but in real life application the should be in different machines. Also, My laptop cant handle this number if virtual machines.
After that you can use the browser to make the requests you want and it's very interesting. I know my system is not complete as you expect but I do what I can do.

2 Cache

I add the cache functionality to the system in front-end server using lumen built-in functions, so lets see its effect on each operation:

Operation	Without cache	With cache
Successful Search	0.2698s	0.0016s
Unsuccessful Search	0.0277s	0.0010s
Successful Lookup	0.0083s	0.0010s
Unsuccessful Lookup	0.0996s	0.0008s
Successful Buy	1.1937s	0.0450s
Unsuccessful Buy	1.2035s	0.0008s

In read operation (Search , Lookup) the expired time for cache is 5 minutes and it is sufficient to handle the problem. However, in buy operation it just 1 minute for cache to be expired and if it is done successfully last time then the system send buy request to order server otherwise just return the reason of failure from the cache.

3 Replication

I add 2 new replication servers to the system, one for catalogue server and the other for order server. So now we want to handle the system with this new features and its effect in the performance of each operation.
For reading operations (search and lookup), when the query go to front-end server, first it check if the result exist in the cache. If it is there the server will response the content in the cache directly. Else, the front end server will choose random catalogue server to get the result from it and then put it in the cache and respond to the client. After try it many times the requests distributed equally to the two servers which balance the load when there is many connections of read.
If we see the next table we will conclude that the system performance is not change significantly and this is not the point because here we just prevent single point of failure and bottleneck from happening to our BAZAR.

Operation	Without cache	With cache
Successful Search	0.0350	0.0012s
Unsuccessful Search	0.0424	0.0012s
Successful Lookup	0.1094	0.0015s
Unsuccessful Lookup	0.0570s	0.0011s

For writing operation (Buy), when request send to front end server, first it check if the result in the cache. If it is there and negative (zero copies , wrong ID , No books in the store) then it will send the same result in cache. But if it (Buy Done) the server will remove it from cache, and choose random order server and get result from it. the same if request not in cache. Second, when

request reach order server , the server will send check request to catalogue if the response is negative it will send buy request directly which indicate what is wrong. However, when the response is positive, it will send another request to the other catalogue server to update the number of copies and then send buy request. After that it will respond (Buy Done).
The same here if we see the next table we will conclude that the system performance is not change significantly and this is not the point because here we just prevent single point of failure and bottleneck from happening to our BAZAR.

Operation	Without cache	With cache
Successful Buy	0.3182s	0.2450s
Unsuccessful Buy	0.0098s	0.0010s

4 Improvements

- We can use database instead of file system like (MySQL, Monogo).
- We can add good GUI instead of just show the result.
- We don't have enough time to make the system more powerful.