PERFORMANCE METRICS DOCUMENT

COMPSCI-677

LAB 3 - PYGMY.COM

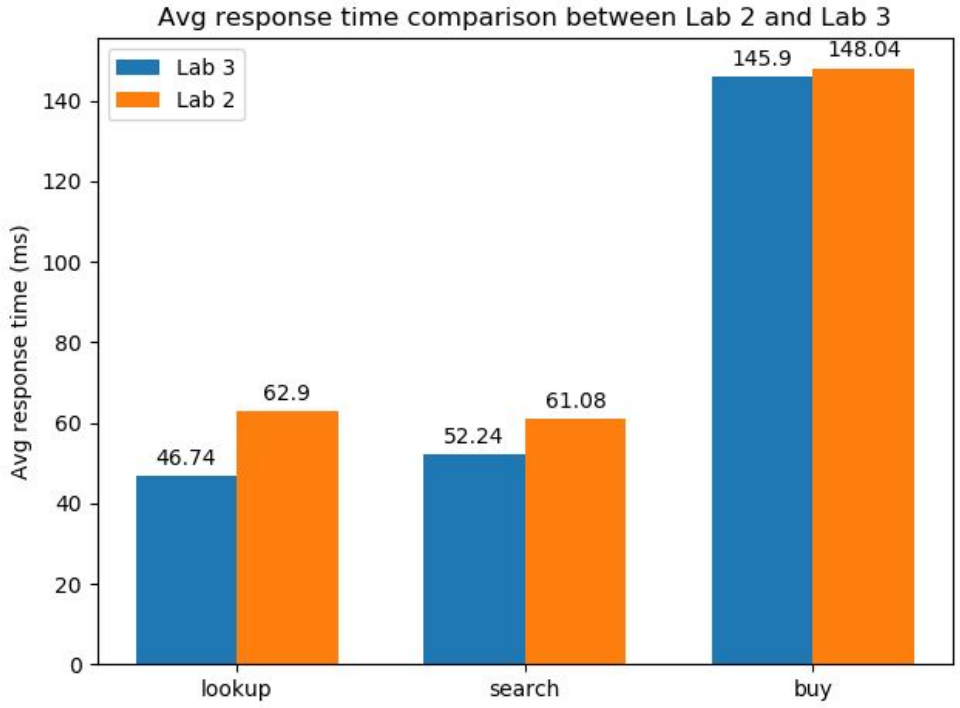
In order to conduct performance metrics evaluation of our system, we implemented the following steps:

- Record the starting time stamp for each new request that is generated by any client
- Generate a unique ID for that request, which is forwarded along with the request to each server that is involved with completing that request
- Record the time stamp when a response is received and calculate the difference in milliseconds
- Record the time taken for each request to get completed along with its unique request ID
- Calculate total time taken for all requests to be completed for a particular client as “end to end time of completion”
- Track number of requests generated in order to calculate average response time for each client across all its requests
- Save the list of request IDs, response times, avg response times and end to end completion times to file titled ‘client_<client number>_metrics.txt’ in the appropriate tests subdirectory
- In addition to this, each of the servers (front-end, catalog, order) also logs the request ID and time taken to service that request for each request they receive at each API endpoint - Thus, the path of each request can be traced across the client and all the servers by finding the appropriate request ID within the log files of each of the servers and seeing how much time was taken to process that specific request at that specific server.

Below, we can see the recorded metric values from one execution cycle of tests on EDLAB machines, along with comparison from lab 2:

Request	**Number of requests**	**Avg. response time (Lab 2)(ms)**	**Avg. response time (Lab 3)(ms)**
lookup	1800	62.90	46.74
search	1800	61.08	52.24
buy	1800	148.04	145.90

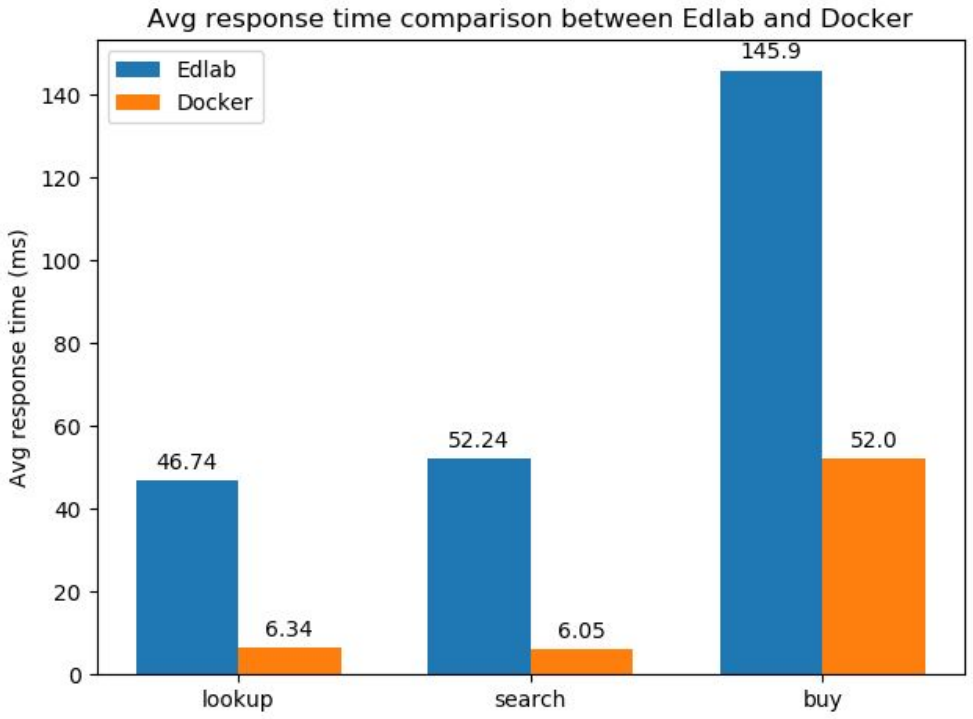
As we can see, oor system in Lab 3 performs better in all of the requests. This is primarily due to caching for read requests, and load balancing for write requests. The following plot shows this comparison:



The following table shows a comparison between the system running on EdLab and Docker:

Request	**Number of requests**	**Avg. response time (Docker)(ms)**	**Avg. response time (EdLab)(ms)**
lookup	1800	6.34	46.74
search	1800	6.05	52.24
buy	1800	52.0	145.90

As we can see from the above table, the system running locally on Docker is much faster than the system running on Edlab. The following plot shows this:



Cache Consistency:

We are using flask-caching to maintain the cache. To maintain cache consistency, we are invalidating the cache for a particular data item, every time a buy request is received. As far the overhead of cache invalidation is concerned, we have observed latency of an average of **10.21ms** for every request subsequent to a cache miss.

The following shows a sample of the log file generated with the response times:

Request ID	Request completion time (milliseconds)
c17d9498-782f-11ea-9714-7304c17ec9ca	226.526
c1a93cd8-782f-11ea-9714-7304c17ec9ca	183.067
c20d6294-782f-11ea-8781-28f10e08bdf0	250.796
c21572f4-782f-11ea-a5ba-cbfbcd6061b9	450.153
c217453e-782f-11ea-a5ba-cbfbcd6061b9	571.889
c299be42-782f-11ea-b838-f7046ed58e93	197.609
c2bf8a82-782f-11ea-b838-f7046ed58e93	149.331
c2de5408-782f-11ea-b838-f7046ed58e93	181.373
c30222f2-782f-11ea-b838-f7046ed58e93	177.507
c323a710-782f-11ea-b838-f7046ed58e93	171.033
c3514148-782f-11ea-b838-f7046ed58e93	183.396