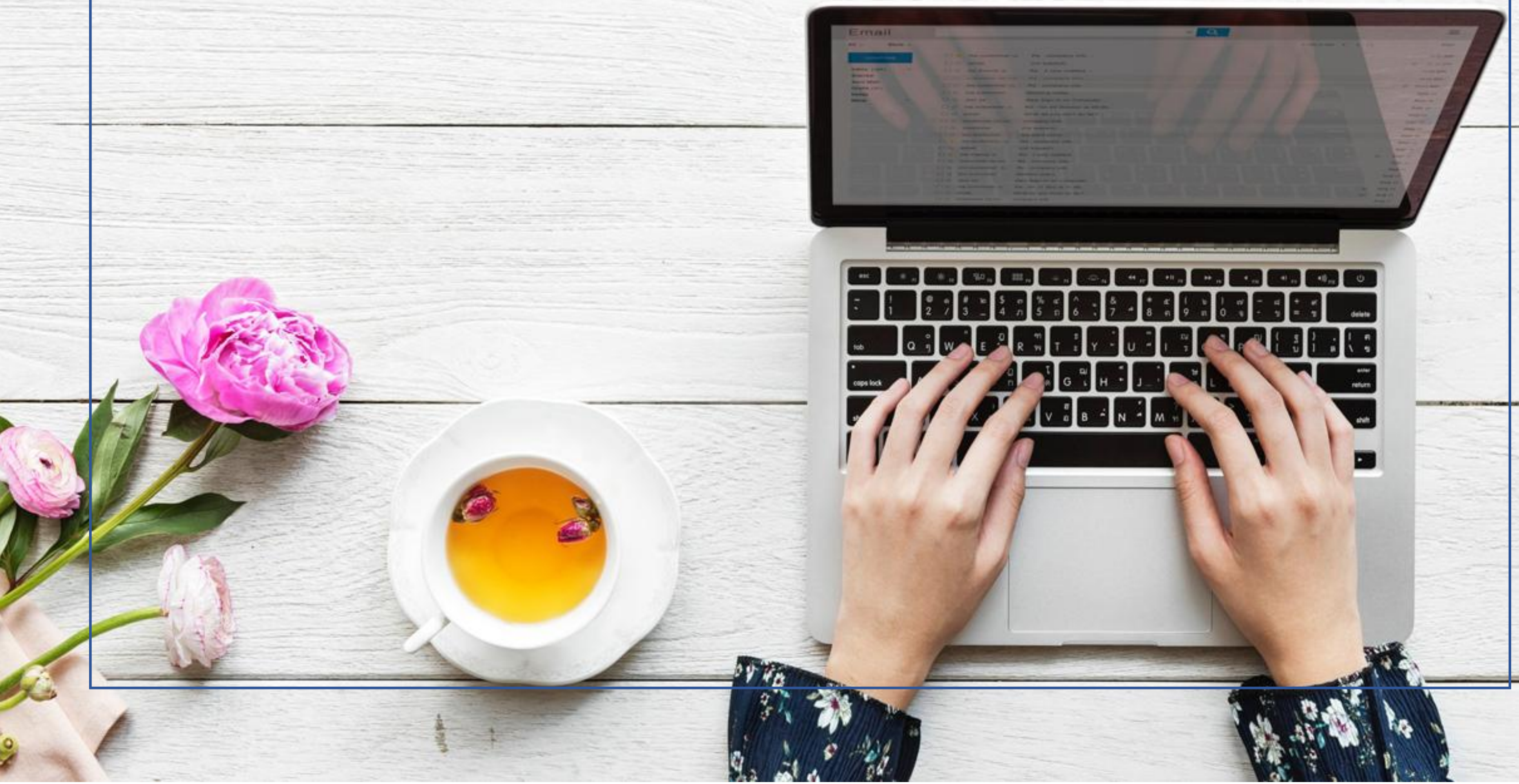
Experimental Evaluations and Measurements

Lina Qurom

11924435



Section 1: Average Response Time

Firstly, I carried out the experiment three times per request (info/search/purchase) for each of these cases:

Request	Without caching	With caching / Cache Miss	With caching / Cache Hit
info	2.05886	2.07598	0.00000
info	2.02253	2.06085	0.00000
info	2.05509	2.05519	0.00000
search	2.06290	2.05673	0.00000
search	2.05665	2.04173	0.00000
search	2.04607	2.05441	0.00000
purchase	20.57228	20.46887	20.45994
purchase	22.66776	22.54819	22.52714
purchase	20.58426	20.51303	20.51401

- Without Caching:
  Average response time (query: info/search) = 2.05018 seconds.
  Average response time (buy: purchase) = 21.27477 seconds.
  Average response time (**query/buy**) = **8.458489** seconds.

- With Caching (Cache Miss):
  Average response time (query: info/search): 2.0575 seconds.
  Average response time (buy: purchase) = 21.1767 seconds.
  Average response time (**query/buy**) = **8.458489** seconds.

- With Caching (Cache Hit):
  Average response time (query: info/search): 0.00000 seconds.
  Average response time (buy: purchase) = 21.1767 seconds.
  Average response time (**query/buy**) = **5.294174** seconds.

✓ Caching provides a significant performance improvement for query responses when there is a **cache hit**, reducing the response time to **almost zero**.

✓ For cache misses, the performance is similar to the scenario without caching, suggesting that cache misses might not bring significant benefits.

- ✓ The purchase operation's response time remains consistent with and without caching, indicating that caching doesn't impact this operation because its write operation not read.

Section 2: Cache Invalidation Experiment

- I recorded 10 experiments for each part in this section:
- After orders (purchase)

#	Time Taken for Cache Invalidation	Time Taken for Purchase
1	0.00103	20.63870
2	0.00094	20.55220
3	0.00100	20.57237
4	0.00000	22.62142
5	0.00097	20.58132
6	0.00000	22.53567
7	0.00101	20.52995
8	0.00099	20.53479
9	0.00101	20.46559
10	0.00000	22.76898

Average overhead for cache invalidation (purchase) = **0.000695** seconds.
Average time taken for purchase = **21.180099** seconds.

- ✓ This experiment simulates cache invalidation after purchase operations. The overhead of cache consistency operations, in this case, is relatively low, suggesting efficient cache management. The subsequent request that sees a cache miss after cache invalidation would experience the usual cache miss response time, which I can measure separately.
- ✓ Analysis:
    - ➢ Before Caching:
      Average response time (query/buy) = **8.458489** seconds.
    - ➢ After Caching:
      Average response time (hit/miss) = **6.8763315** seconds.
    - ➢ Average overhead for cache invalidation (purchase) = **0.000695** seconds.

- Average response time (query/buy) + Average overhead for cache invalidation (purchase) = **6.8763315 + 0.000695 = 6.8770265** seconds**.**

✓ Conclusion:

- Comparing the average response times before and after caching (taking into account the overhead), we can observe that after caching, the system performs better. The average response time after caching is **lower than** the average response time before caching.
- Caching, even with the overhead of cache invalidation, leads to better performance compared to the system without caching. The overhead introduced by cache invalidation is **relatively small** compared to the overall improvement gained from caching.

- After catalog updates

#	**Time Taken for Cache Invalidation**	**Time Taken for book info after update**
1	0.00102	2.04703
2	0.00099	2.04924
3	0.00000	2.05558
4	0.00108	2.03964
5	0.00000	2.02217
6	0.00000	2.01954
7	0.00000	2.05538
8	0.00095	2.06842
9	0.00100	2.04502
10	0.00000	2.04782

Average overhead for cache invalidation (update) = **0.000504** seconds.
Average time taken for (info) after (update) = **2.044984** seconds.

✓ Analysis:

- Before Caching:
  Average response time (query: info/search) = **2.0575** seconds.

- ➢ After Caching:
  Average response time (query: hit/miss) = 2.0575 + 0.00000 / 2 = **1.02875** seconds.
- ➢ Average overhead for cache invalidation (update) = **0.000504** seconds.
- ➢ Average response time (query: hit/miss) + Average overhead for cache invalidation (update) = **1.02875 + 0.000504 = 1.029254** seconds**.**

✓ Conclusion:

- ➢ Comparing the average response times before and after caching (taking into account the overhead), you can observe that after caching, the system performs **significantly better**. The average response time after caching is **much lower** than the average response time before caching.
- ➢ Caching with cache invalidation (update) improves the system's performance, and the overhead introduced by cache invalidation is **relatively small** compared to the overall improvement gained from caching
- ➢ After the cache is initially populated (cache hit), subsequent requests that see a cache miss experience **lower latency** compared to requests without caching.
- ➢ After a cache invalidation (update), there might be a **small increase** in latency for subsequent requests that miss the cache due to the overhead of cache invalidation operations.

This analysis suggests that the caching mechanism, even with cache invalidation overhead, is beneficial for the system's overall performance.