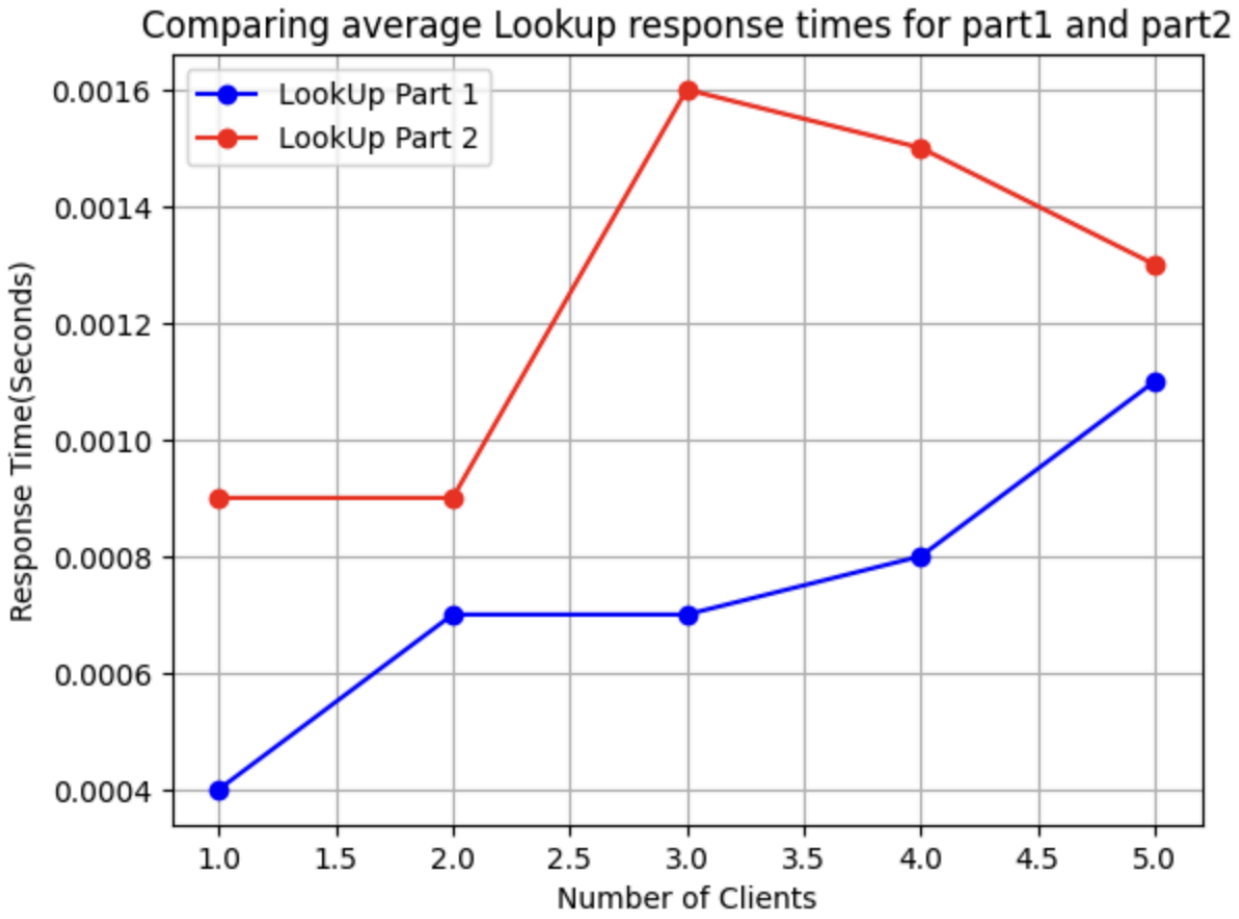
Lab 1: Stock Lookup and Trading System

Design Document

Authors: Isha Nilesh Gohel, Roshini Sanikop

Q1) Comparing latency for lookup for part1 and part2 :



Observations -

LookUp Part 1 shows a gradual increase in response time as the number of clients increases whereas LookUp Part 2 starts at a higher response time and exhibits a steeper increase, indicating that the gRPC implementation is experiencing greater latency under load.

With respect to the performance of each of these lookups - gRPC works better for fewer clients; as the number of clients increases, gRPC's custom time also grows significantly. At 5 clients, gRPC takes nearly double the response time compared to the custom thread pool. From 3-5 clients - gRPC takes longer than the custom thread pool suggesting higher overhead and contention issues.

Q2) Part 1 :

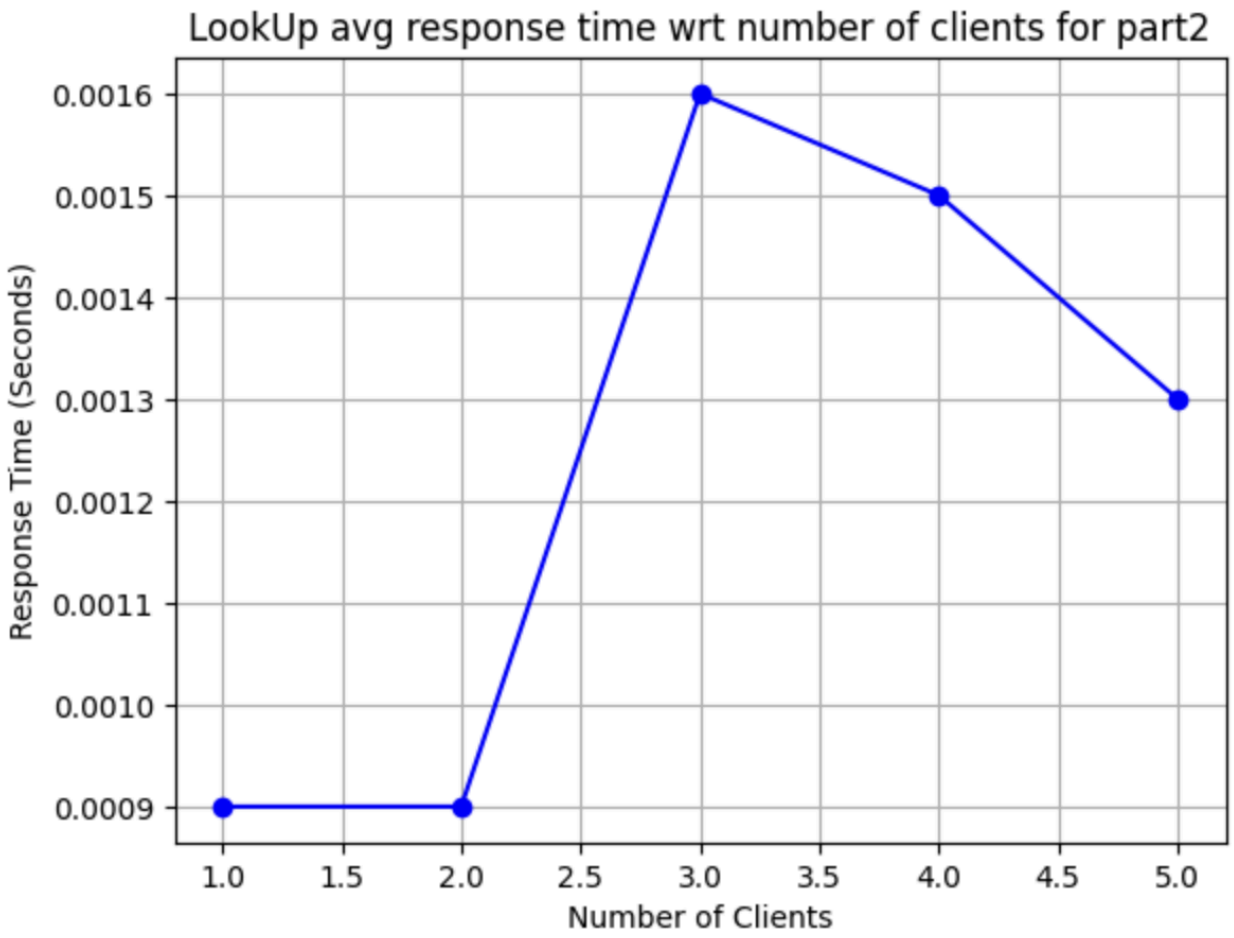
Latency when number of clients is varied for LookUp :

Observations:

1. The response time increases gradually up to 4 clients, indicating efficient handling, but rises sharply beyond 5 clients due to possible thread pool saturation.
2. A slight drop at 7 clients suggests adaptive resource allocation or an anomaly in request handling under high load.

Part 2:

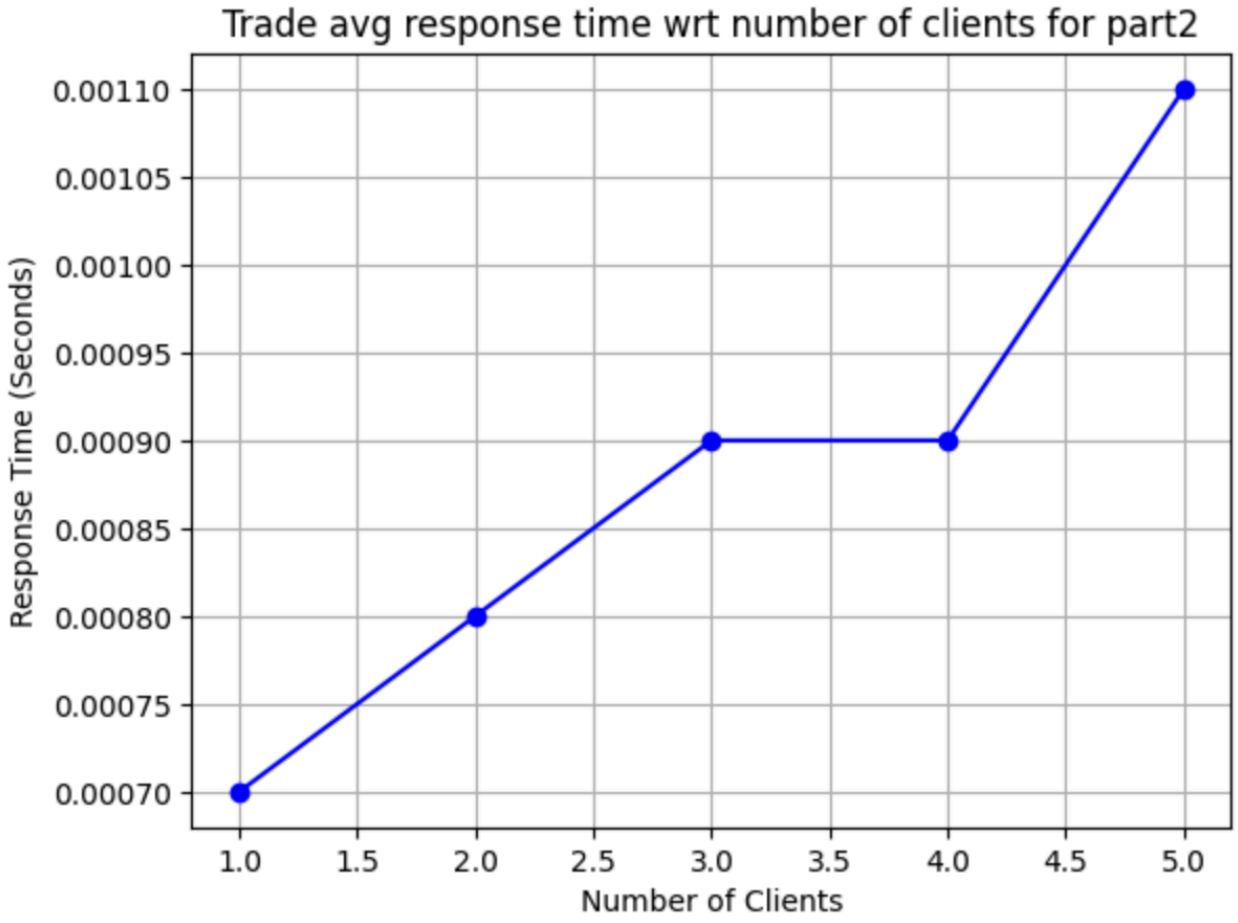
Latency when number of clients is varied for LookUp :



Observations:

The lookup response time spikes sharply at 3 clients, indicating bottleneck, and then gradually decreases as the system stabilizes, thereby suggesting an initial overload or contention, followed by adaptive resource allocation.

Latency when number of clients is varied for Trade :



Observations:

The response time increases steadily with the number of clients, showing a predictable scaling pattern, suggesting that the trade operation scales well but experiences increasing latency as concurrency rises.

Latency when number of clients is varied for Update :

Observations:

The update response time increases linearly as the number of clients grows, suggesting a consistent overhead for handling updates.

Answer -

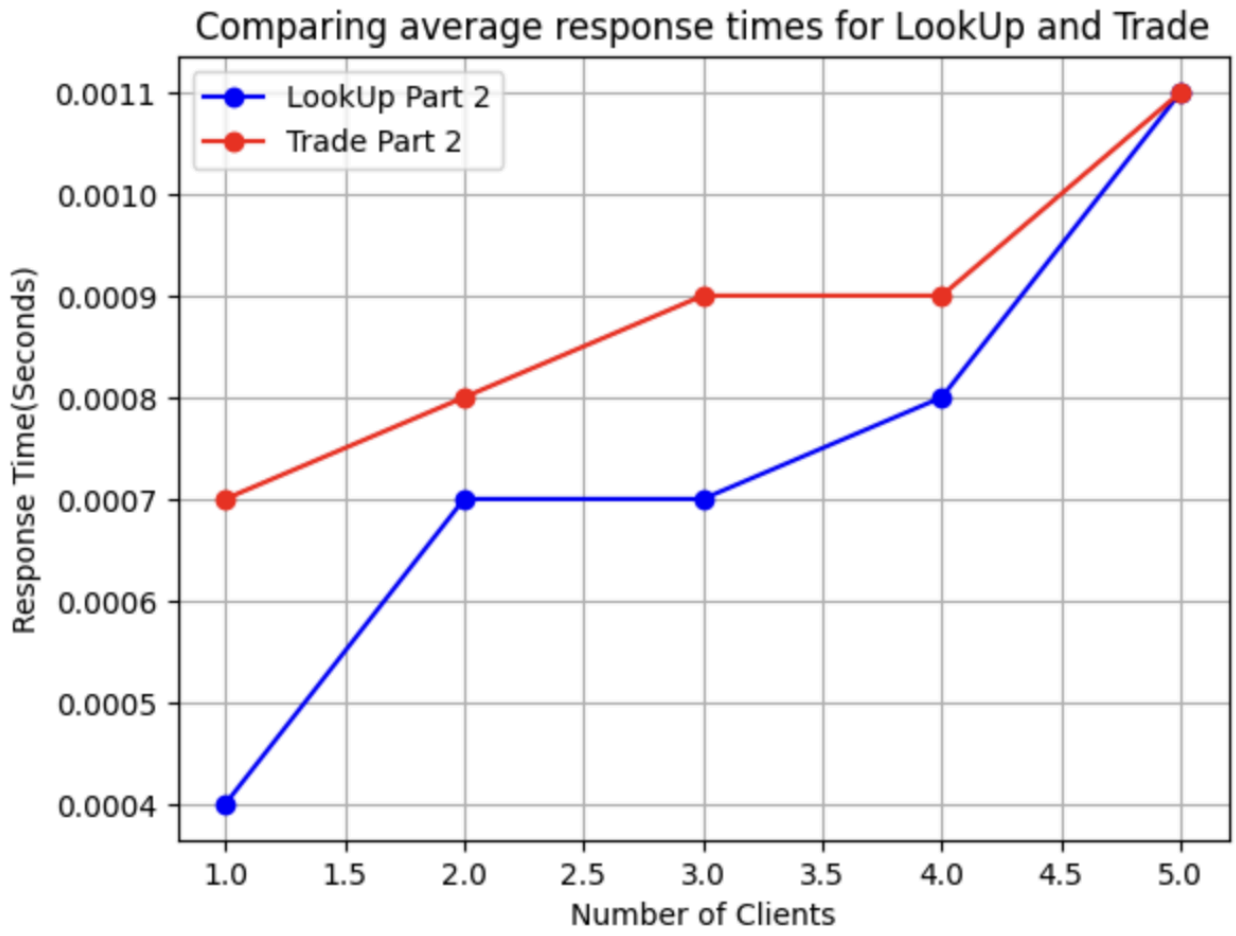
As the number of clients increases, response time generally increases across all three operations (lookup, trade, and update), but the rate of increase differs.

1. Lookup Latency -
    a. Initially stable, but spikes sharply at 3 clients, indicating a sudden increase in contention or queuing.
    b. Drops after 4 clients, possibly due to load balancing or caching effects.
2. Trade Latency -
    a. Increases steadily with load, remaining predictable up to 4 clients.

b. A sharp rise at 5 clients suggests thread pool saturation or contention for shared resources.

3. Update Latency -
    a. Shows a linear increase, indicating consistent overhead per update request.
    b. Unlike lookup, update requests involve write locks, causing higher contention.

Q3) Latency of Lookup v/s Trade (part2):



Observations:

The trade operation consistently shows higher response times than lookup, indicating the additional overhead of modifying stock data compared to a read-only operation. Lookup operations scale better due to their read-only nature. Trade operations introduce more overhead due to database writes/locking mechanisms.

While both response times increase as the number of clients grows, lookup scales better initially, but the gap narrows at higher concurrency, suggesting contention in resource handling.

Yes, synchronization impacts performance significantly, particularly for trade operations. Since lookup operations only require read access, they should execute simultaneously.

However, trade operations require write locks to modify stock volumes, bringing contention into the picture, thus slowing down concurrent requests. Since the system employs read-write locks, lookup operations scale better than trade, as multiple clients can read simultaneously, but only one can write at a time.

Q4) Number of clients is larger than the static thread pool (part1) :

Observations:

When the number of clients exceeds the static thread pool size, response time increases due to requests waiting in the queue for an available thread. Since only a fixed number of threads can process requests simultaneously, the additional requests experience delays, leading to higher response times. This is evident from the sharp rise in response time beyond 4-5 clients, indicating thread contention.