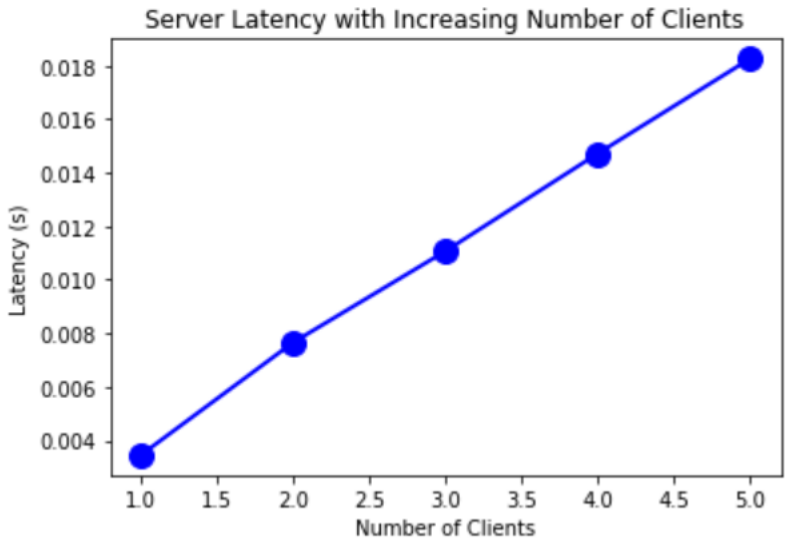
Evaluation Document

Part 1: Implementation with Socket Connection and Handwritten Thread Pool

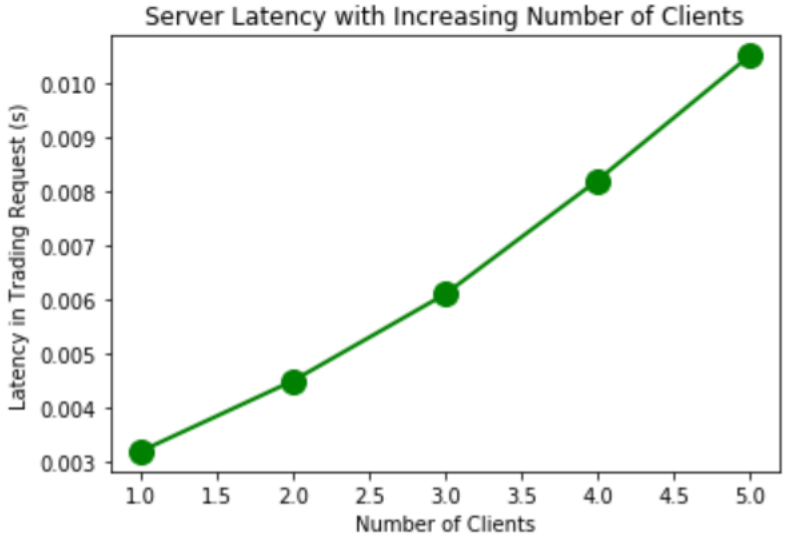


Graph 1 : Server Latency in Socket Programming.

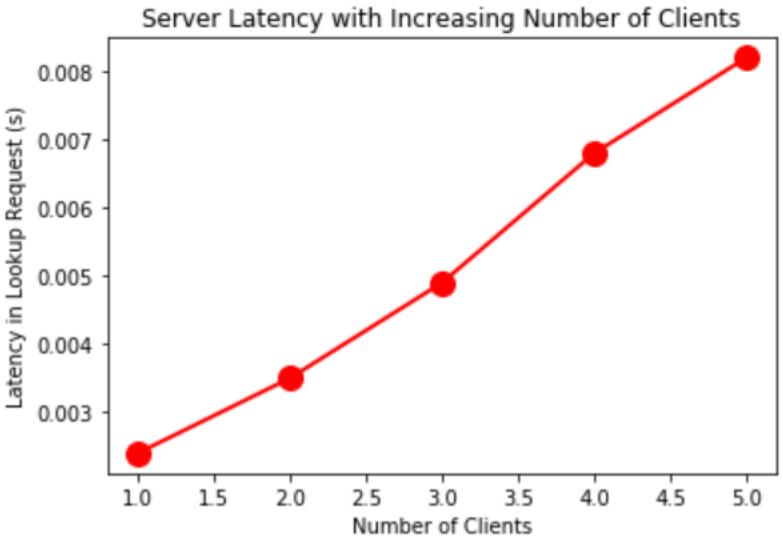
To evaluate the performance of the server, we measured the response time for different numbers of concurrent clients. For each number of clients, we ran a for-loop in the client code that issued sequential query requests to the server. We used the time module in Python to measure the time it took for the client to receive a response from the server. We can see from the plot that the response time increased as the number of concurrent clients increase. This is expected as the server has a fixed thread pool size of 2 that means it can handle up to 2 client requests concurrently. When the number of concurrent clients exceeds the thread pool size, the server has to wait for an available thread in the pool, which increases the response time.

We also observed that the response time was more stable when the number of concurrent clients was less than or equal to the thread pool size. This is because the server was able to handle all client requests concurrently without having to wait for an available thread in the pool.

Part 2: Implementation with gRPC and Built in Thread Pool



Graph 2 : Server Latency in Trading Requests in gRPC .



Graph 2 : Server Lookup in Trading Requests in gRPC .

Here we have compared the Server's latency during two separate tasks 1) During Trading and 2) During Lookup. We can observe that during the first scenario there is a constant increase in the latency period as the number of clients increase. Compared to the Lookup graph, the slope is steeper and the max latency in the first scenario is more than the second scenario. This is because, the Trading process is more demanding than the Lookup process. In Trading the Server must perform more tasks and hence the larger latency period is justified.

Part 3: Evaluation and Performance Measurement

Ans 1. The latency of Lookup varies significantly from Part 1 to Part 2. As "gRPC" is much more efficient than "Socket Programming" we can see that the latency in Graph 1 is increasing significantly as compared to Graph 3 as more clients are being added. Also, the final value of latency in Graph 1 is more than twice the final latency value in Graph 3.

Ans 2. As the number of clients (load) increase, so does the latency time. A load increase significantly impacts response time in both cases.

Ans 3. The latency period increases consistently in the first scenario as the number of clients increases. The slope of the Trading graph is steeper compared to the Lookup graph, and the maximum latency in the Trading scenario is greater than the maximum latency in the Lookup scenario. This can be attributed to the fact that Trading involves more demanding tasks for the server, which justifies the longer latency period.

Ans 4. As the number of clients increase and grow bigger than the thread pool, the latency period increases dramatically. This is expected as the server has a fixed thread pool size of 2 that means it can handle up to 2 client requests concurrently. When the number of concurrent clients exceeds the thread pool size, the server must wait for an available thread in the pool, which increases the response time.