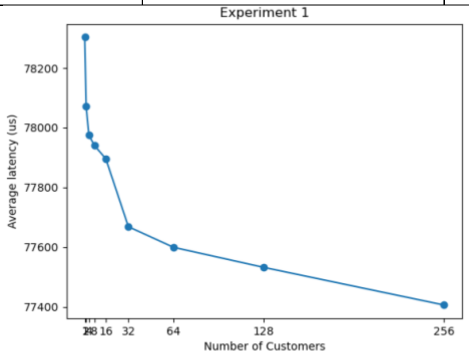
Server on linux-080 and client on linux-077:

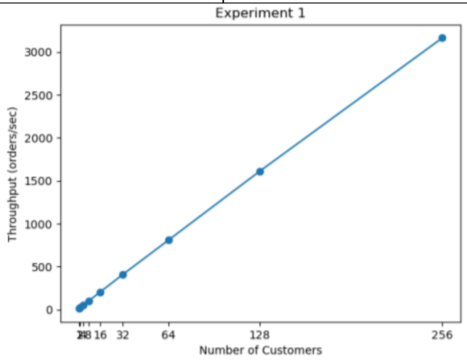
Experiment 1 – Impact of Increasing Customers

Use the regular laptop type and vary the number of customers: 1, 2, 4, 8, 16, 32, 64, 128, and 256.

1. ./server 12345
2. bash test.sh 10.200.125.80 12345 0

customer	avg latency(us)	min latency(us)	max latency(us)
1	78303.1	481.33	83435
2	78072.12	399	83256.33
4	77974.45	363.66	83449.33
8	77939.75	361	97598.33
16	77895.68	343.66	83862
32	77668.25	337.66	83976.33
64	77599.35	302	92635.33
128	77532.16	291.66	85102.33
256	77405.98	296.66	84905.33





Avg Latency: Exhibits a logarithmic rather than linear decrease as customer numbers grows, indicating improved efficiency up to a point as the system scales.

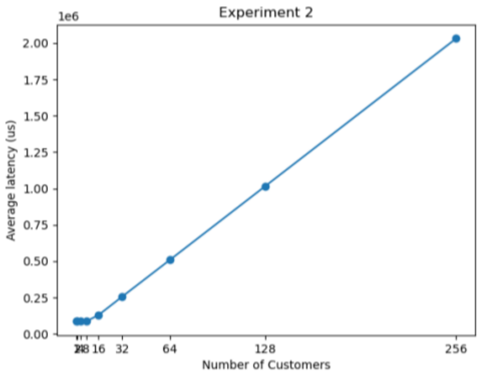
Throughput: Shows a linear increase with more customers, highlighting the system's capacity to handle higher loads effectively.

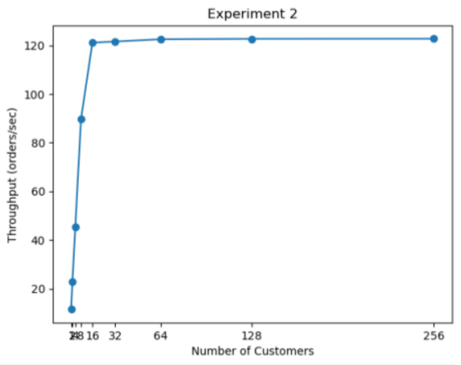
Experiment 2 – Effect of a Single Expert

Use the custom laptop type and set the number of expert engineer to 1. Vary the number of customers: 1, 2, 4, 8, 16, 32, 64, 128, and 256.

1. ./server 12345 1
2. bash test.sh 10.200.125.80 12345 1

customer	avg latency(us)	min latency(us)	max latency(us)
1	86683.83	8737	91271.33
2	86924.8	8741	91280.66
4	87393.56	8680	92371
8	87624.38	8666.66	138763
16	128949.98	8625.66	203353
32	256645.75	8553	334984
64	509219.95	9647	592919.33
128	1016637.92	8532.66	1118127
256	2031682.12	8692.33	2135569



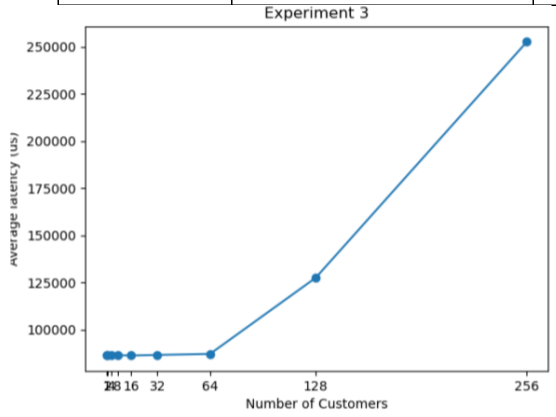


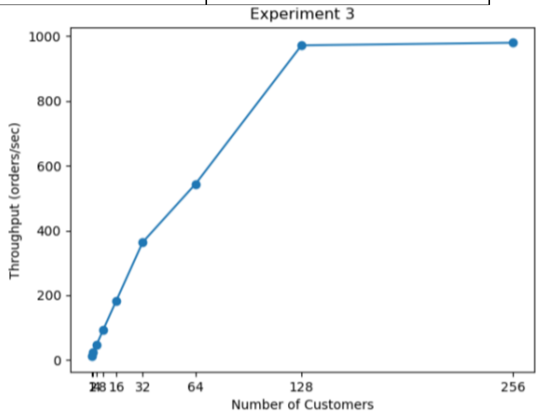
Avg Latency: Remains consistent for customer counts below 8, then linearly increases due to the bottleneck created by a single expert engine leading to order queuing.
Throughput: Decreases linearly for customer counts below 8, stabilizes when customers reach 8, reflecting the bottleneck effect of the single expert engine and the order is queued.

Experiment 3 - Scaling with Customers Beyond Expert Capacity
Use the custom laptop type and fix the number of expert engineer to 8. Vary the number of customers: 1, 2, 4, 8, 16, 32, 64, 128, and 256.

1. ./server 12345 8
2. bash test.sh 10.200.125.80 12345 1

customer	**avg latency(us)**	**min latency(us)**	**max latency(us)**
1	86746.51	8721.66	91241
2	86661.5	8533	91507.33
4	86734.66	8439.33	92371
8	86563.11	8426.33	91622
16	86414.18	8410.33	103446.66
32	86704.96	8413	94685.66
64	87256.92	8406.33	118854.33
128	127571.47	8430.66	211073
256	252575.05	8427.33	340526.33





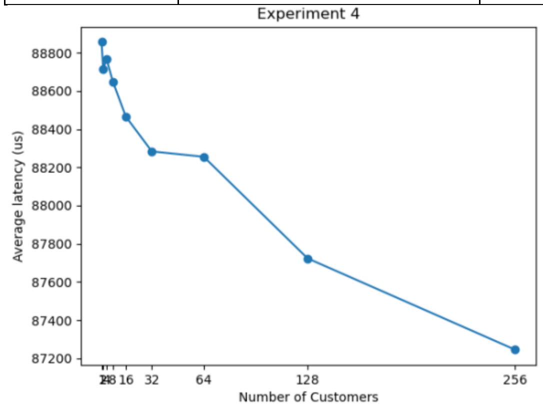
Avg Latency: Stays nearly unchanged for customer counts below 64, then increases linearly, mirroring Experiment 2's bottleneck effect that the single expert just can handle 8 customers at the same time.
Throughput: For customer counts under 64, throughput linearly decreases and then the throughput is same as the number of customers at 64 customers, showing the system's limit with one expert handling up to 8 customers simultaneously.

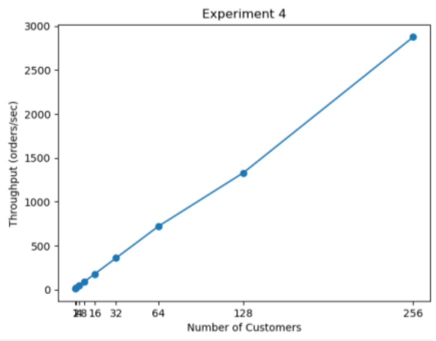
Experiment 4: Matching Experts with Customer Counts

Use the custom laptop type and set the number of expert engineer to be the same number as the customers. Vary the number of customers and expert engineers: 1, 2, 4, 8, 16, 32, 64, 128, and 256.

1. ./server 34521 1
2. bash test_ep4.sh 10.200.125.80 34521 1 1
3. repeat with 2 4, 8, 16, 32, 64, 128, 256

customer	**avg latency(us)**	**min latency(us)**	**max latency(us)**
1	88856.58	8952	91984.33
2	88714.32	8694.66	91219
4	88768.49	8554	91295.33
8	88648.06	8459.33	92151.66
16	88465.17	8494	92378.66
32	88283.49	8406.33	103745.66
64	88255.42	8384.66	94053.33
128	87723.41	8394.33	93891
256	87244.86	8394	92403.33





Avg Latency: Decreases as the number of expert engines increases, suggesting diminishing returns on latency improvement with additional experts.
Throughput: Demonstrates a linear increase as both expert engines and customer counts rise, indicating that eliminating expert engine bottlenecks allows for consistent throughput gains.