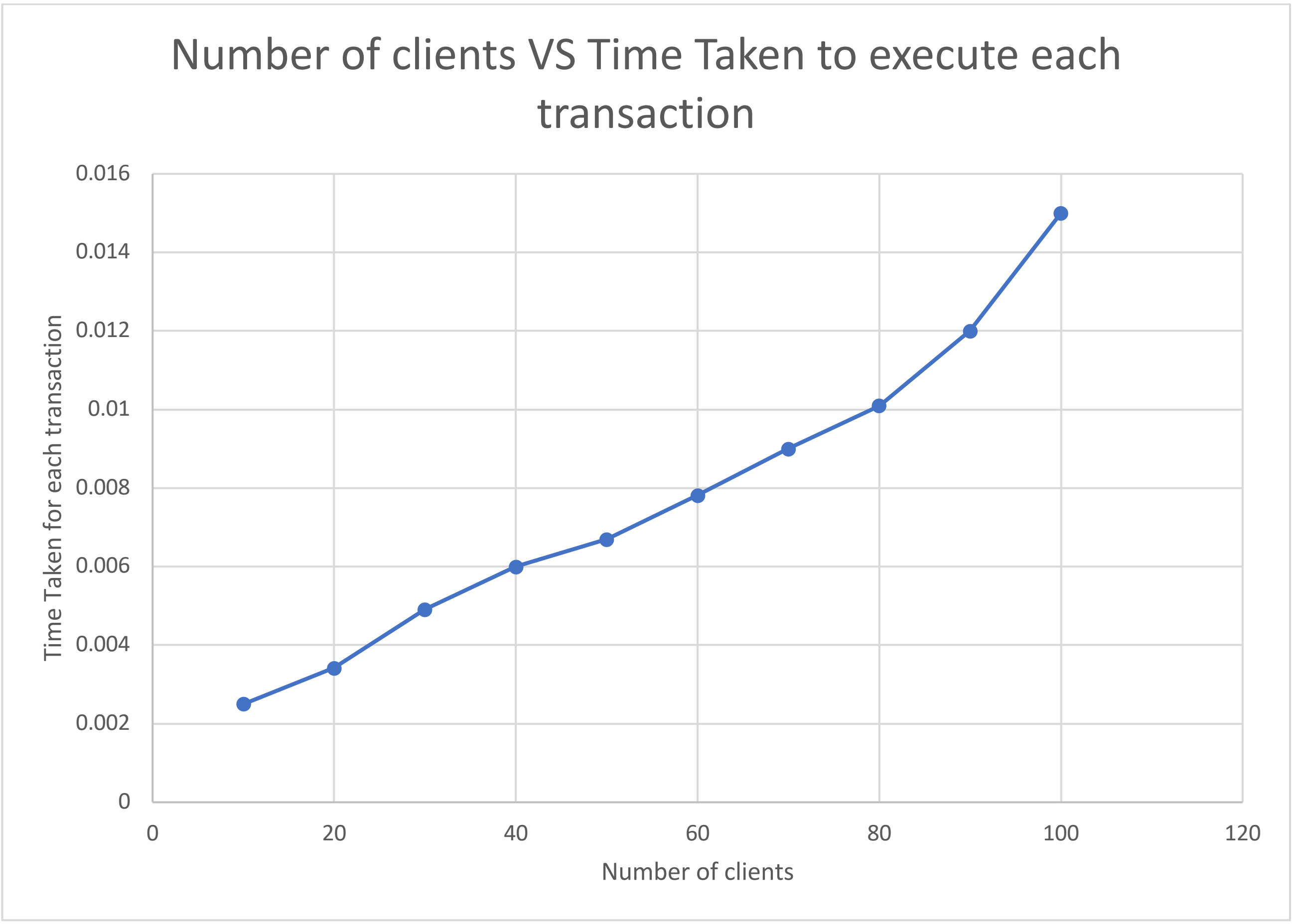
Scalability:

RAM used: 8 GB

Three machines was used for testing. Two from AWS(different regions and one from virtual box)

I've used apache ab tool for benchmarking testing which makes concurrent requests varying clients and transactions as per the requirement

1. 1000 requests varying clients from 10 to 100 each making concurrent requests and restricting clients each to 20 requests at a time.



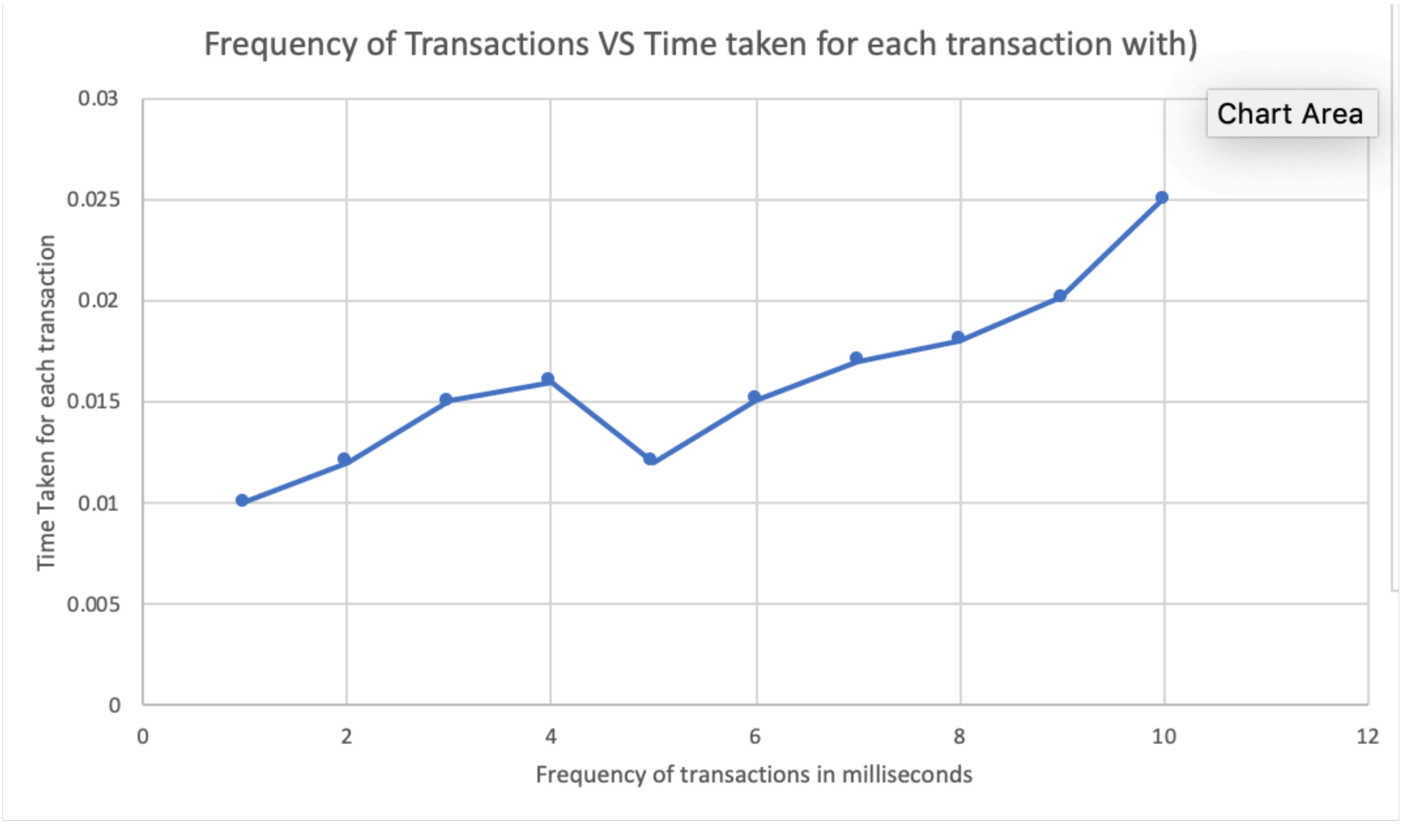
Number of clients on x-axis
Execution time of each transaction on Y-axis

Observation: As we can observer in the graph, time taken for executing the transactions increased when we increase the clients. With 100 clients and 1000 transactions per second we can say that server is reliable

Result: Server can at a time handle up to 1,000 requests per second when connected to multiple clients

2. 25 clients and varying frequency of transactions from 1 millisecond to 10 milliseconds.

frequency of Transactions on x-axis
Execution time of each transaction on y-axis



Observation: As we can observer in the graph, time taken for executing the transactions increased when we increase the frequency of transactions. However, there is a decrease when frequency of requests is 5 milliseconds. For firs 4 data points, I've executed the apache ab command in the same machine where server is running. But for the next few transactions I've executed the apache ab tool using AWS VM. That might be the possible reason. I've intentionally used three four different machines from different providers to observe the difference in network delay. When running from other VMs the response was little late when compared to running the apache ab tool in localhost.

Result: Server is able to handle threading and locks when we send requests at different times.

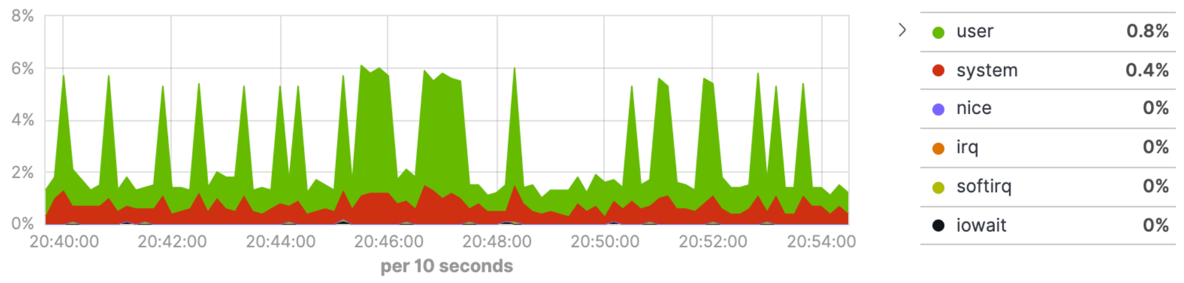
3.
Also, captured the **memory and CPU utilization while running the above transactions.**

Tools used: Elasticsearch, Metricbeat, and Kibana

Observation:

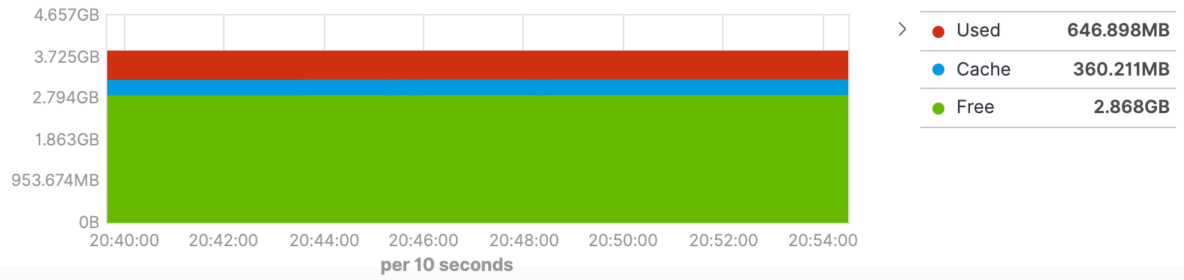
1. Though memory is almost same both times there is greater usage in CPU while running the above transactions with multiple clients. More than 6% of CPU is utilized while running the server whereas only 2% of CPU is utilized when the server is stopped.

CPU Usage [Metricbeat System] ECS



●	user	0.8%
●	system	0.4%
●	nice	0%
●	irq	0%
●	softirq	0%
●	iowait	0%

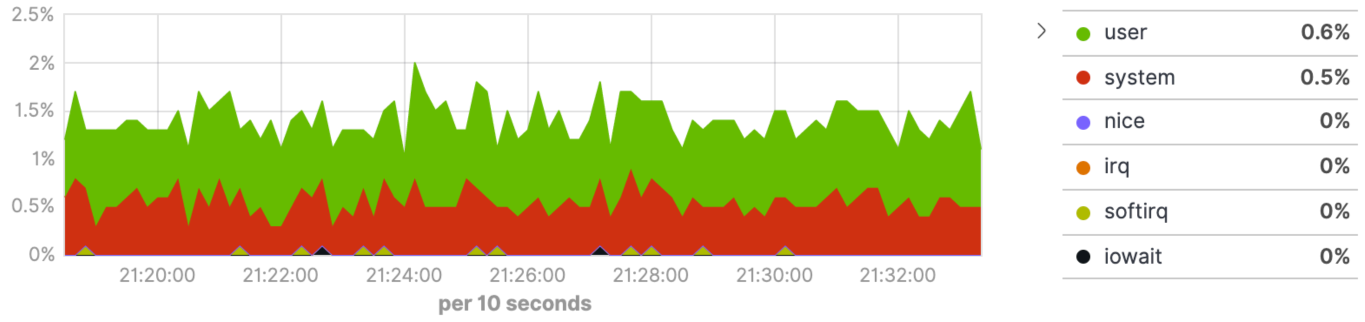
Memory Usage [Metricbeat System] ECS



●	Used	646.898MB
●	Cache	360.211MB
●	Free	2.868GB

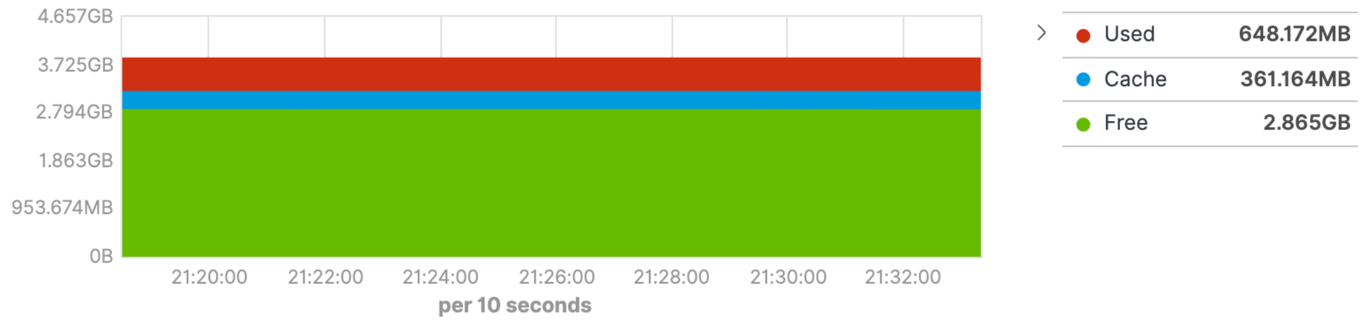
CPU and memory Without running the transactions

CPU Usage [Metricbeat System] ECS



●	user	0.6%
●	system	0.5%
●	nice	0%
●	irq	0%
●	softirq	0%
●	iowait	0%

Memory Usage [Metricbeat System] ECS



●	Used	648.172MB
●	Cache	361.164MB
●	Free	2.865GB

END