CSCI 5673 : Distributed Systems - Spring 23

Assignment 2 – Marketplace
Swaminathan Sriram
18th Feb, 2022

Source Code

Github : Assignment2

Evaluation

Scenario1 : Run one instance of seller and one instance of buyer.
Scenario 2 : Run ten instances of buyers and ten instances of sellers concurrently.
Scenario 3 : Run 100 instances of buyers and 100 instances of sellers concurrently.

Scenario	Buyer(Response Time)	Seller(Response Time)	Buyer(Throughput)	Seller(Throughput)
1	0.02207022863	0.02240307235	1000 / 10.28099 = 97.266	1000 / 9.55943 = 104.608
2	0.1032068182	0.1135394945	10000 / 202.28201 = 49.4359335267	10000 / 186.83725 = 53.5225175922
3	0.9509440405	1.025748285	100000 / 4236.06194 = 23.6068313959	100000 / 4790.44102 = 20.8749047494

The above table was created based on running the system for three different scenarios. The details on how to run the system and the performance scripts are mentioned in the README.md file

Average response time:

Response time of a client-side API function is the duration of time between the time when the client invokes the function and time when a response is received from the server. To measure average response time, measure the response time for ten different runs and then take the average.

Average server throughput:

Throughput is the number of client operations completed at the server per second. To measure average throughput, measure the throughput for ten different runs and then take the average. Each run consists of each client invoking 1000 API functions.

Execution Log and Results
The logs and results for each scenario can be found in the Performance Folder”.
The response and server throughput logs are found inside the subfolder “Results”

Observation

1) As we increase the number of instances, We can see an increase/delay in the response time. This is due to the fact that the server becomes a bottleneck. All the clients are sending requests to the server in parallel, but there is only one server to process all the requests. There is added latency because of gRPC.
2) For the previous assignment, Every server was hosted as a separate process under the same host. But now we have hosted every service as a separate container in the cloud. This further causes delay because of the network and latency.
3) Also previously the client and server resided in the same host, but now server and database reside in the cloud and client run locally.
4) As we increase the number of instances, We can see there is a decrease in throughput, The reason being exactly as the previous one. When the server is bombarded with requests, The number of requests that the server can process per second is definitely going to decrease.
5) There is another reason for the dip in performance. We know that for every request that requires database operation, The server has to send a request to the database. The database connection is shared between components and also there is only a single database instance that's running, this again causes a bottleneck. Additionally, When the buyer and seller are sending requests in parallel, The database has to handle concurrent read and write. The locking mechanism can further cause a delay.
6) The server has to take care of handling connections (creating, maintaining and closing connections). When there is only one instance of a client, The process becomes very simple but when the number of instances increases, The connection handling creates an extra load at the server.