Assignment 1 Client Design

STUDENT NAME: WEN ZHAO

Major classes and Relation Introduction:

- MarketServlet: This MarketServlet class is used to process two APIs in the A1 project, which does server implementation work.

- Parameter: This Parameter class is used to parse and validate the arguments from the command line. In this class, I choose the map to store the arguments, the key is the name of the argument, value is the corresponding data. What's more, this class meets the requirements of parameters about the default value and valid range.

In Part1 Package, ClientPart1 class and StoreThread1 are included.

- StoreThread1: This StoreThread1 class implements Runnable and play as a single thread for every store to complete sending requests task. What's more, the methods to randomly select a custID and itemIDs for the order are implemented in this class by employing the PurchaseItems class and Purchase class from the Swagger client API package.

- ClientPart1: This ClientPart1 class accepts a set of arguments from the command line at startup. In the main function, it will generate a Parameter object to help parse arguments. Then it will generate max threads based on the Parameter.maxStores, then run in three phases with the help of CountDownLarch to set up barriers, so that it could simulate the staggered opening times of stores across 3 timezones.

In Part2 Package, ClientPart2 class and StoreThread2 are included.

- StoreThread2 and ClientPar2: It is basically the same as Part 1, but there is a little changes to have deeper insights into the performance of the system. It adds timestamps to calculate the latency and write records to CSV file for performance analysis.

Good Design practices used:

- To follow good design practice, inheritance is adopted in the StoreThread1 class so that we could use Runnable object to create a thread and it is more flexible.
- To enhance thread-safe, StoreThread2 employs CopyOnWriteArrayLis instead of ArrayList to store the record data of each thread, which is especially useful in multithreaded code when reads frequent.

Accept command-line options:

—maxStores 32 --ipAddress http://localhost:8080/A1_war_exploded

--maxStores 64 --ipAddress http://localhost:8080/A1_war_exploded

--maxStores 128 --ipAddress http://localhost:8080/A1_war_exploded

--maxStores 256 --ipAddress http://localhost:8080/A1_war_exploded

Client 1 output window for each run in EC2:

```
[ec2-user@ip-172-31-91-114 ~]$ java -jar A1_client1.jar --maxStores 32 --ipAddress http://52.90.135.31:8080/A1_war_exploded/
--Client 1 start running--
Total threads: 32
Start at: 2021/02/10 09:44:49
--Store opens in east--
Wait for central store to open...
--Store opens in central--
Wait for western store to open...
--Store opens in west--
Wait for all threads to complete...
All threads complete at: 2021/02/10 09:44:56

--------------------

Total number of successful requests sent: 17280
Total number of unsuccessful requests sent: 0
Wall Time: 6.394s
Throughput is: 2s
```

```
[ec2-user@ip-172-31-91-114 ~]$ java -jar A1_client1.jar --maxStores 64 --ipAddress http://52.90.135.31:8080/A1_war_exploded/
--Client 1 start running--
Total threads: 64
Start at: 2021/02/10 09:46:02
--Store opens in east--
Wait for central store to open...
--Store opens in central--
Wait for western store to open...
--Store opens in west--
Wait for all threads to complete...
All threads complete at: 2021/02/10 09:46:12

--------------------

Total number of successful requests sent: 34560
Total number of unsuccessful requests sent: 0
Wall Time: 10.235s
Throughput is: 3s
```

```
[ec2-user@ip-172-31-91-114 ~]$ java -jar A1_client1.jar --maxStores 128 --ipAddress http://52.90.135.31:8080/A1_war_exploded/
--Client 1 start running--
Total threads: 128
Start at: 2021/02/10 09:46:36
--Store opens in east--
Wait for central store to open...
--Store opens in central--
Wait for western store to open...
--Store opens in west--
Wait for all threads to complete...
All threads complete at: 2021/02/10 09:46:54

--------------------

Total number of successful requests sent: 69120
Total number of unsuccessful requests sent: 0
Wall Time: 17.491s
Throughput is: 3s
```

```
[ec2-user@ip-172-31-91-114 ~]$ java -jar A1_client1.jar --maxStores 256 --ipAddress http://52.90.135.31:8080/A1_war_exploded/
--Client 1 start running--
Total threads: 256
Start at: 2021/02/10 09:47:34
--Store opens in east--
Wait for central store to open...
--Store opens in central--
Wait for western store to open...
--Store opens in west--
Wait for all threads to complete...
All threads complete at: 2021/02/10 09:48:06

--------------------

Total number of successful requests sent: 138240
Total number of unsuccessful requests sent: 0
Wall Time: 32.221s
Throughput is: 4s
```

Client 2 output window for each run in EC2:

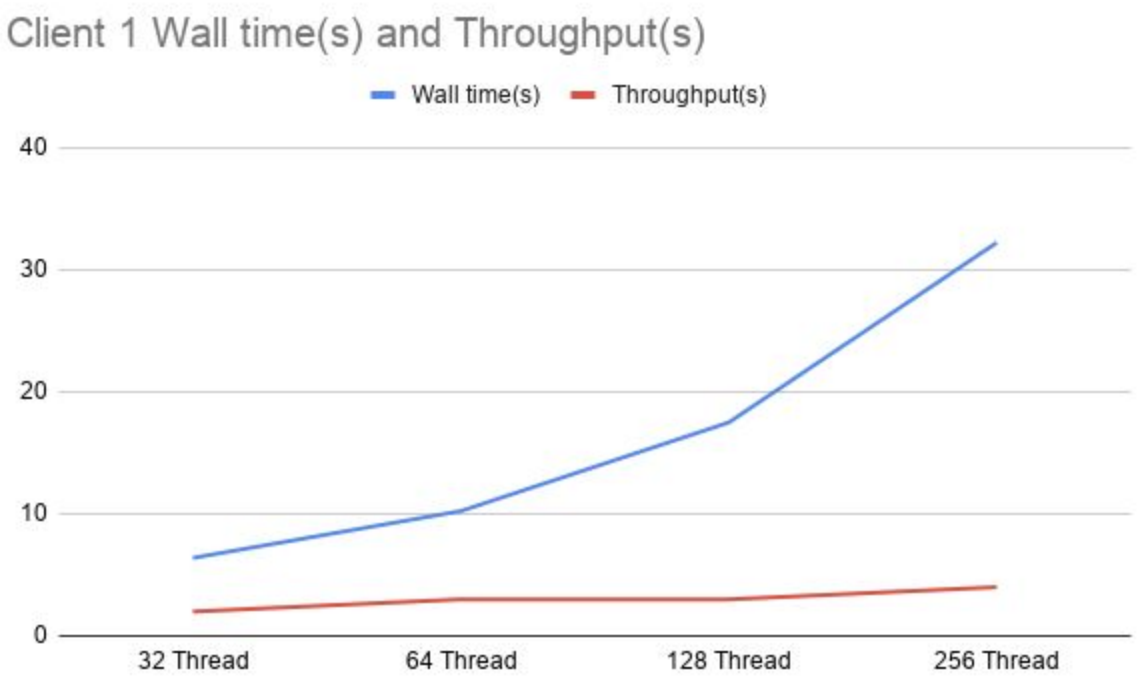
```
[ec2-user@ip-172-31-91-114 ~]$ java -jar A1_client2.jar --maxStores 32 --ipAddress http://52.90.135.31:8080/A1_war_exploded/
--Client 2 starting--
Total threads: 32
Start at: 2021/02/10 10:08:22
--Store opens in east--
Wait for central store to open...
--Store opens in central--
Wait for western store to open...
--Store opens in west--
Wait for all threads to complete...
All threads complete at: 2021/02/10 10:08:29

--------------------

Total number of successful requests sent: 17280
Total number of unsuccessful requests sent: 0
Wall Time: 6.761s
Throughput is: 2.555834935660405s
Mean response time for POSTs:4.125 ms
Median response time for POSTs:3.0 ms
p99 (99th percentile) response time for POSTs:9 ms
Max response time for POSTs:9 ms
```

```
[ec2-user@ip-172-31-91-114 ~]$ java -jar A1_client2.jar --maxStores 64 --ipAddress http://52.90.135.31:8080/A1_war_exploded/
--Client 2 starting--
Total threads: 64
Start at: 2021/02/10 10:06:55
--Store opens in east--
Wait for central store to open...
--Store opens in central--
Wait for western store to open...
--Store opens in west--
Wait for all threads to complete...
All threads complete at: 2021/02/10 10:07:06

--------------------

Total number of successful requests sent: 34560
Total number of unsuccessful requests sent: 0
Wall Time: 10.33s
Throughput is: 3.345595353339787s
Mean response time for POSTs:6.75 ms
Median response time for POSTs:5.0 ms
p99 (99th percentile) response time for POSTs:18 ms
Max response time for POSTs:18 ms
```

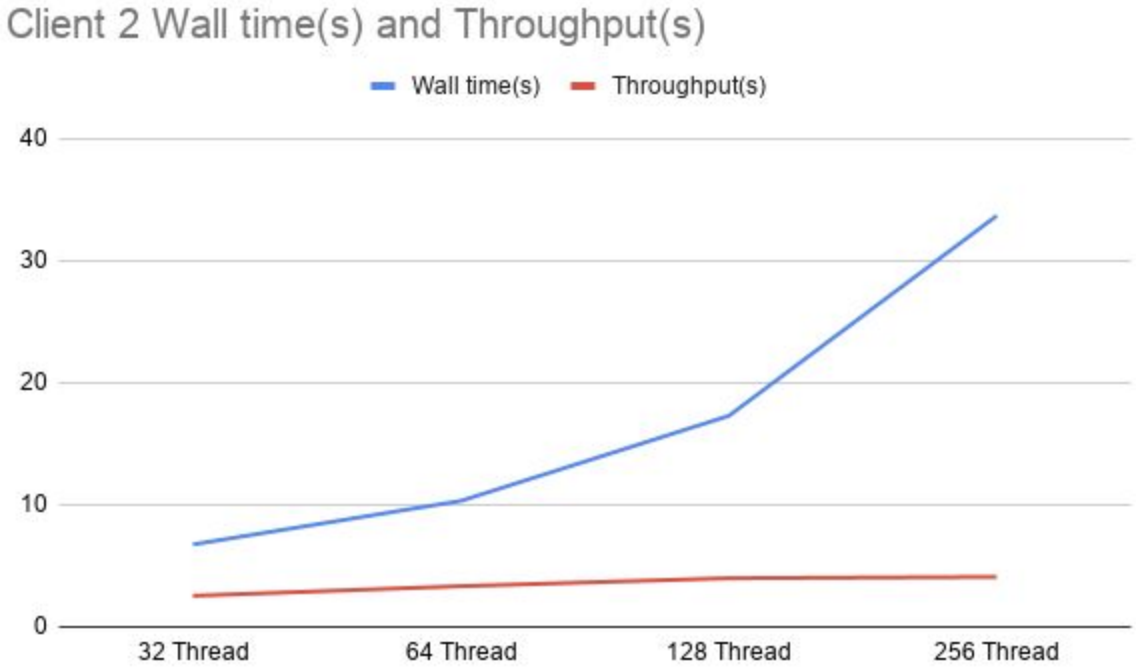
```
[ec2-user@ip-172-31-91-114 ~]$ java -jar A1_client2.jar --maxStores 128 --ipAddress http://52.90.135.31:8080/A1_war_exploded/
--Client 2 starting--
Total threads: 128
Start at: 2021/02/10 10:05:19
--Store opens in east--
Wait for central store to open...
--Store opens in central--
Wait for western store to open...
--Store opens in west--
Wait for all threads to complete...
All threads complete at: 2021/02/10 10:05:36

--------------------

Total number of successful requests sent: 69120
Total number of unsuccessful requests sent: 0
Wall Time: 17.296s
Throughput is: 3.996299722479186s
Mean response time for POSTs:11.265625 ms
Median response time for POSTs:9.0 ms
p99 (99th percentile) response time for POSTs:37 ms
Max response time for POSTs:54 ms
```

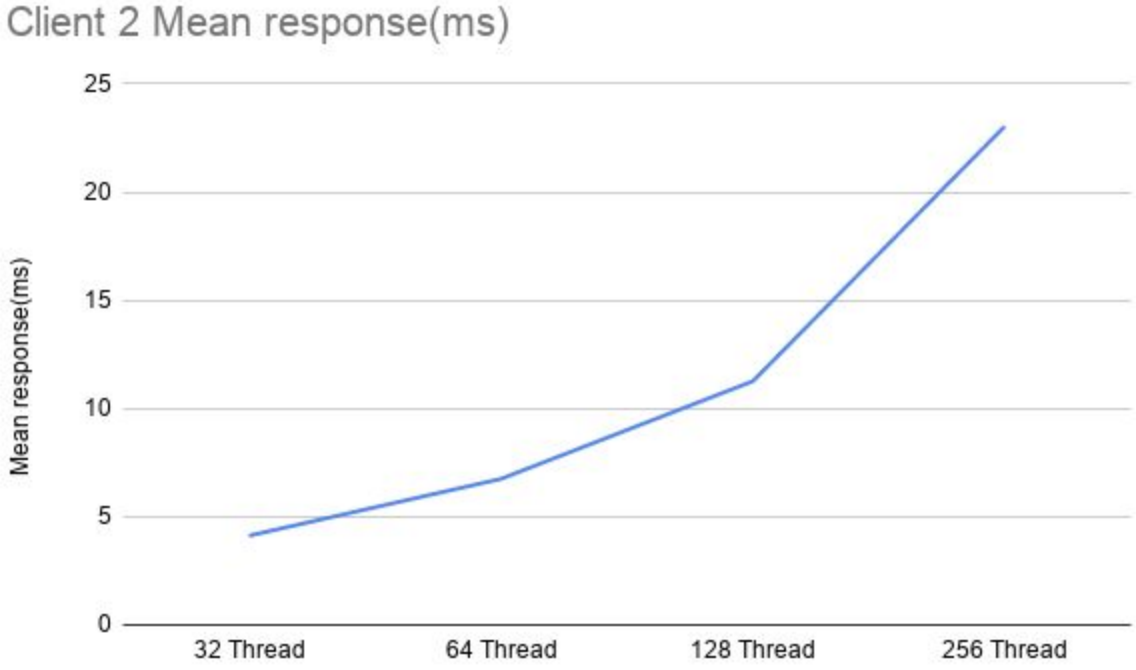
```
[ec2-user@ip-172-31-91-114 ~]$ java -jar A1_client2.jar --maxStores 256 --ipAddress http://52.90.135.31:8080/A1_war_exploded/
--Client 2 starting--
Total threads: 256
Start at: 2021/02/10 09:58:52
--Store opens in east--
Wait for central store to open...
--Store opens in central--
Wait for western store to open...
--Store opens in west--
Wait for all threads to complete...
All threads complete at: 2021/02/10 09:59:25

--------------------

Total number of successful requests sent: 138240
Total number of unsuccessful requests sent: 0
Wall Time: 33.689s
Throughput is: 4.103416545459942s
Mean response time for POSTs:23.04296875 ms
Median response time for POSTs:18.5 ms
p99 (99th percentile) response time for POSTs:73 ms
Max response time for POSTs:87 ms
```

Client 1 Plot of the wall time and throughput by the number of threads



Client 2 Plot of the chart showing the wall time, throughput and mean response time by the number of threads


Client 2 Wall time(s) and Throughput(s)
Wall time(s)
Throughput(s)
40
30
20
10
0
32 Thread
64 Thread
128 Thread
256 Thread


Client 2 Mean response(ms)
Mean response(ms)
25
20
15
10
5
0
32 Thread
64 Thread
128 Thread
256 Thread