## **Assignment 1**

CS 6650, Fall 2023, Ximing Liang

### **GitHub Repository URL**

https://github.com/ximing0116/CS6650-Assignment1

### **Client Design**

The SimpleClient is designed to call an online server, sending multiple requests to save album details and retrieve them. Using tools like HttpURLConnection to connect and ObjectMapper to interpret server replies, it first conducts a series of "test" requests and then dispatches bulk requests in batches, with a pause between each batch. The program keeps track of its operation's duration and provides optional debug messages detailing its actions throughout the process.

#### **Major classes**

- **SimpleClient:** This is the main class that contains the core logic of the client operation. It's responsible for sending requests to a server, managing threads for those requests, and measuring the execution time.
- AlbumResponse: Nested within the SimpleClient class, it's a model class that like a data holder. It
  represents the response received from the server, capturing details like the album's ID and the
  size of its image.

#### Methods

- main method: This is the program starts and makes sure you give it the right number of
  instructions (4 arguments, threadGroupSize = N, numThreadGroups = N, delay = N (seconds), and
  IPAddr = server URL). It then sends out some "test" requests. After that, it sends out many more
  requests in groups and waits a bit between each group. At the end, it tells you when it started and
  stopped and how long it took.
- sendRequests & sendPairRequest methods: These are the "threads" that make server requests. One fetches a single album and repeatedly requests its details. The other sequentially retrieves multiple albums and their data one by one.
- sendPostRequest & sendGetRequest methods: These are the "protocols" that communicate with
  the server. One sends data to save an album, while the other retrieves album details from the
  server.
- debugPrintln method: This is a helper tool that displays program messages while it's active.

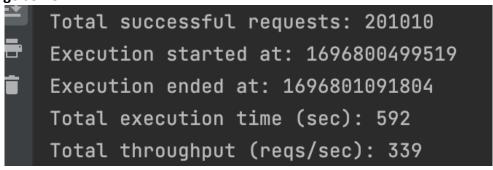
#### **Special Points**

- ExecutorService: It control the threads, directing when they start and stop.
- HttpURLConnection: It's the connection method uses to communicate with the server.
- ObjectMapper: It translates server responses to a JSON string.

### **Client Part 1 Results**

Scenarios	GO Throughput (reqs/sec)	Java Throughput (reqs/sec)
threadGroupSize = 10,		
numThreadGroups = 10,	339	356
delay = 2		
threadGroupSize = 10,		
numThreadGroups = 20,	335	349
delay = 2		
threadGroupSize = 10,		
numThreadGroups = 30,	281	338
delay = 2		

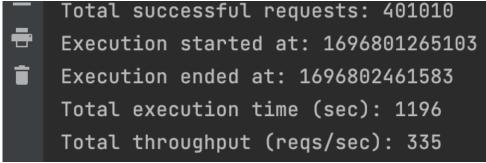
# threadGroupSize = 10, numThreadGroups = 10, delay = 2 go server:



#### iava server:

```
Total successful requests: 201010
Execution started at: 1696893881733
Execution ended at: 1696894445571
Total execution time (sec): 563
Total throughput (reqs/sec): 356
```

threadGroupSize = 10, numThreadGroups = 20, delay = 2 go server:



#### iava server:

Total successful requests: 401010

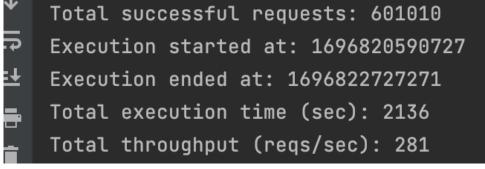
Execution started at: 1696880192460

Execution ended at: 1696881340631

Total execution time (sec): 1148

Total throughput (reqs/sec): 349

threadGroupSize = 10, numThreadGroups = 30, delay = 2 go server:



#### iava server:

Total successful requests: 601010

Execution started at: 1696829147638

Execution ended at: 1696830925705

Total execution time (sec): 1778

Total throughput (reqs/sec): 338

## **Client Part 2 Results**

Scenarios	GO Throughput (reqs/sec)	Java Throughput (reqs/sec)
threadGroupSize = 10,		
numThreadGroups = 10,	313	342
delay = 2		
threadGroupSize = 10,		
numThreadGroups = 20,	321	342
delay = 2		
threadGroupSize = 10,		
numThreadGroups = 30,	281	337
delay = 2		

# threadGroupSize = 10, numThreadGroups = 10, delay = 2 go server:

```
Total successful requests: 201010
Execution started at: 1696812624593
Execution ended at: 1696813265012
Total execution time (sec): 640
Total throughput (reqs/sec): 313
POST Stats:
Mean response time (ms): 31.95769423057694
Median response time (ms): 29.0
p99 response time (ms): 88
Min response time (ms): 16
Max response time (ms): 546
GET Stats:
Mean response time (ms): 31.620158415841583
Median response time (ms): 29.0
p99 response time (ms): 88
Min response time (ms): 16
Max response time (ms): 670
```

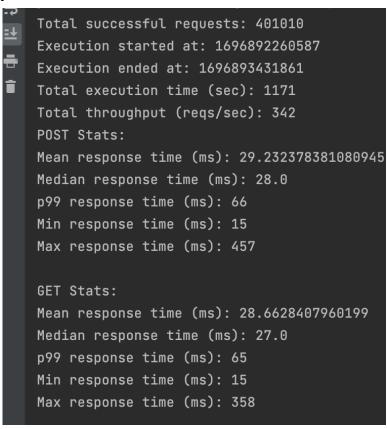
#### java server:

```
Total successful requests: 201010
Execution started at: 1696891360167
Scroll to End ended at: 1696891946573
Total execution time (sec): 586
Total throughput (regs/sec): 342
POST Stats:
Mean response time (ms): 29.234756524347564
Median response time (ms): 28.0
p99 response time (ms): 64
Min response time (ms): 15
Max response time (ms): 404
GET Stats:
Mean response time (ms): 28.970465346534652
Median response time (ms): 27.0
p99 response time (ms): 65
Min response time (ms): 15
Max response time (ms): 403
```

# threadGroupSize = 10, numThreadGroups = 20, delay = 2 go server:

```
Total successful requests: 401010
Execution started at: 1696816998291
⇒ Execution ended at: 1696818243971
Total execution time (sec): 1245
   Total throughput (reqs/sec): 321
    POST Stats:
    Mean response time (ms): 30.529973501324935
    Median response time (ms): 28.0
    p99 response time (ms): 68
    Min response time (ms): 16
    Max response time (ms): 1365
    GET Stats:
    Mean response time (ms): 30.346432835820895
    Median response time (ms): 28.0
    p99 response time (ms): 69
    Min response time (ms): 15
    Max response time (ms): 1358
```

#### java server:



# threadGroupSize = 10, numThreadGroups = 30, delay = 2 go server:

```
Total successful requests: 601010
    Execution started at: 1696820590727
<u>∓</u>
   Execution ended at: 1696822727271
   Total execution time (sec): 2136
   Total throughput (regs/sec): 281
    POST Stats:
    Mean response time (ms): 34.071270957634745
    Median response time (ms): 29.0
    p99 response time (ms): 72
    Min response time (ms): 15
    Max response time (ms): 2311
    GET Stats:
    Mean response time (ms): 33.335591362126245
    Median response time (ms): 28.0
    p99 response time (ms): 71
    Min response time (ms): 14
    Max response time (ms): 2310
```

#### iava server:

```
➡ Total successful requests: 601010
   Execution started at: 1696827087778
   Execution ended at: 1696828870408
    Total execution time (sec): 1782
    Total throughput (reqs/sec): 337
    POST Stats:
    Mean response time (ms): 29.441875270824305
    Median response time (ms): 28.0
    p99 response time (ms): 66
    Min response time (ms): 15
    Max response time (ms): 371
    GET Stats:
    Mean response time (ms): 28.92500996677741
    Median response time (ms): 27.0
    p99 response time (ms): 65
    Min response time (ms): 15
    Max response time (ms): 532
```

## Throughput over time

