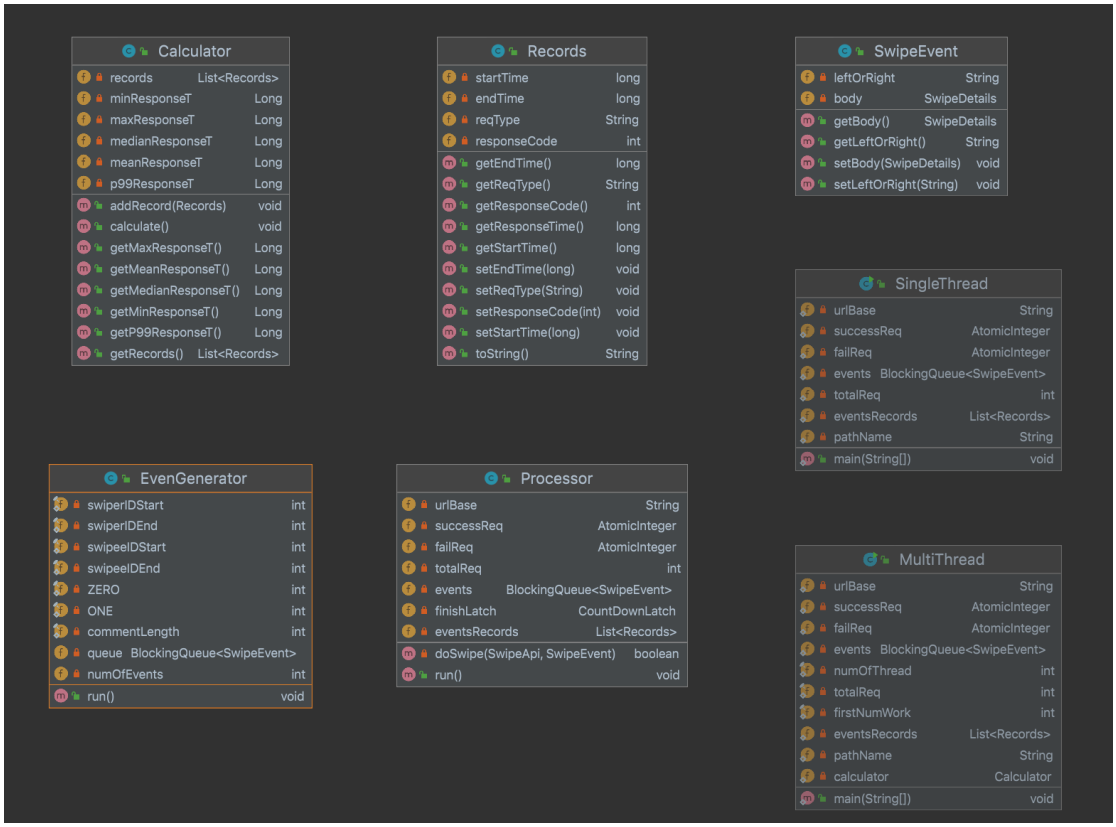
CS6650 Assignment1 Report

Part 1:

The repo URL: https://github.com/Qingzhao331/6650_Assignment.git

Part 2:

This is the UML diagram for my client.



Class 1: SwipeEvent

This is an object class for a swipe event. An swipe even has two fields: leftOrRight(String) and request body(SwipeDetails). The SwipeEvent class has its generators, getters and setters.

Class 2: Records

This is an object class for a successful event record. An record has four fields: start time(long), end time(long), request type(String) and request code(Integer). The Record class has its generators, getters and setters.

Class 3: Calculator

This class can create a calculator to get the information we want. It has a thread safe list to store the records information and has five fields to record the information we need. The main function is the calculate function, which can sort the record list by the records' response time and update the value of the five fields.

Class 4: EvenGenerator
This class can generate the specific number of events and store the events in a thread safe queue.

Class 5: Processor
This is the main class of client project. This class can create a processor, which act as a client and call the Server API.

Class 6: SingleThread
This class is to test the single thread implementation. In its main function, we create a thread to call the EvenGenerator to generate events and create another thread to call the Processor.

Class 7: MultiThread
This class is to test the multiple thread implementation. In its main function, we create a thread to call the EvenGenerator to generate events and create several threads to call the Processor.

Part 3:

Screen shot of output window for Client (Part 1)
Single Thread:

```
This thread is finished!
************************************************************
End Processing
************************************************************
Number of successful requests :10000
Number of failed requests :0
Total wall time: 176406
Throughput: 56 requests/second
```

Expected throughput using Little' s Law for 50 threads:
Expected throughput = num of threads / average response time
= 50 / 0.0179 = 2793

50 Threads:

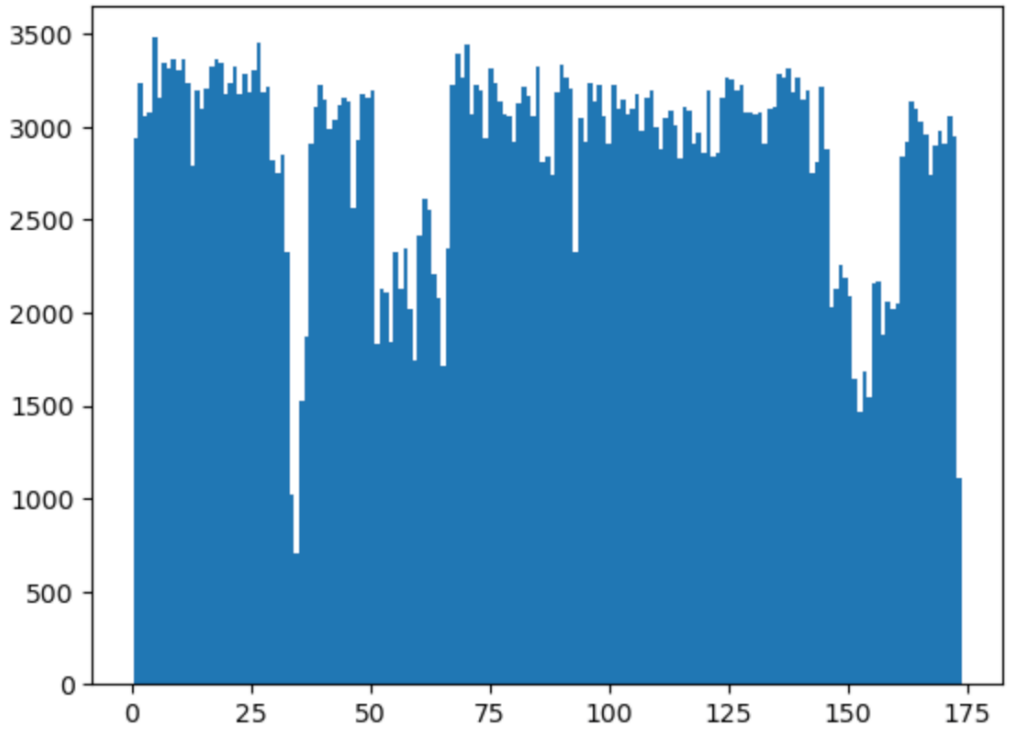
```
************************************************************
Processing Ends
************************************************************
Number of successful requests :500000
Number of failed requests :0
Total wall time: 217216
Throughput: 2304 requests/second
```

Part 4:

80 Threads:

```
Number of successful requests :500000
Number of failed requests :0
Mean response time (in milliseconds): 27
Median response time (in milliseconds): 23
Min response time (in milliseconds): 13
Max response time (in milliseconds): 413
99th percentile response time (in milliseconds): 80
Actual Throughput: 2890 requests/second
```

Part 5:


3500
3000
2500
2000
1500
1000
500
0
0
25
50
75
100
125
150
175