Homework1 Report for CS6650

Xianhe Zhang <zhang.xianh@northeastern.edu>

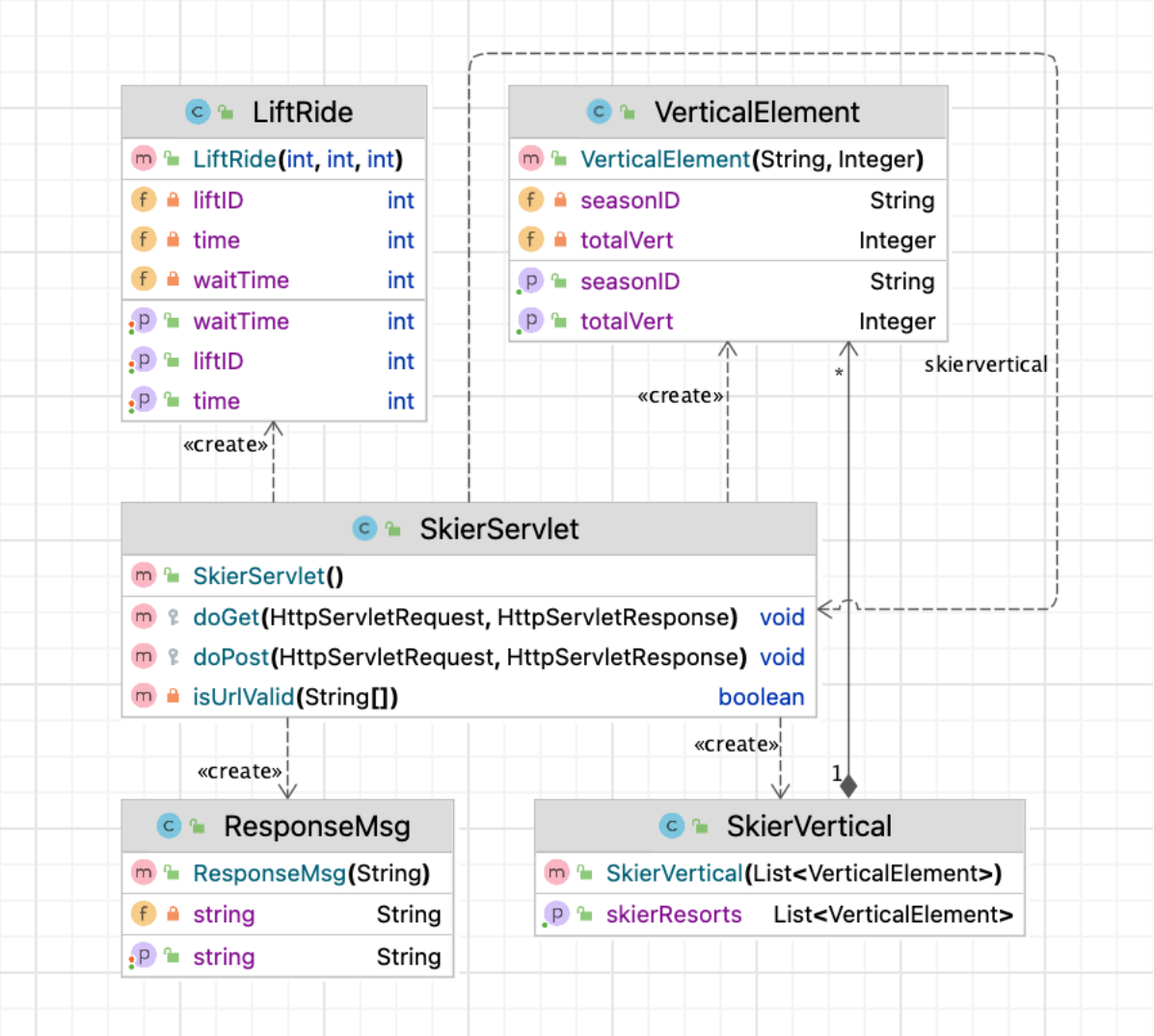
<NOTE: Sorry for late submission. Have talked with Ian about late submit after ddl. Thx!>

GitHub Repo Link

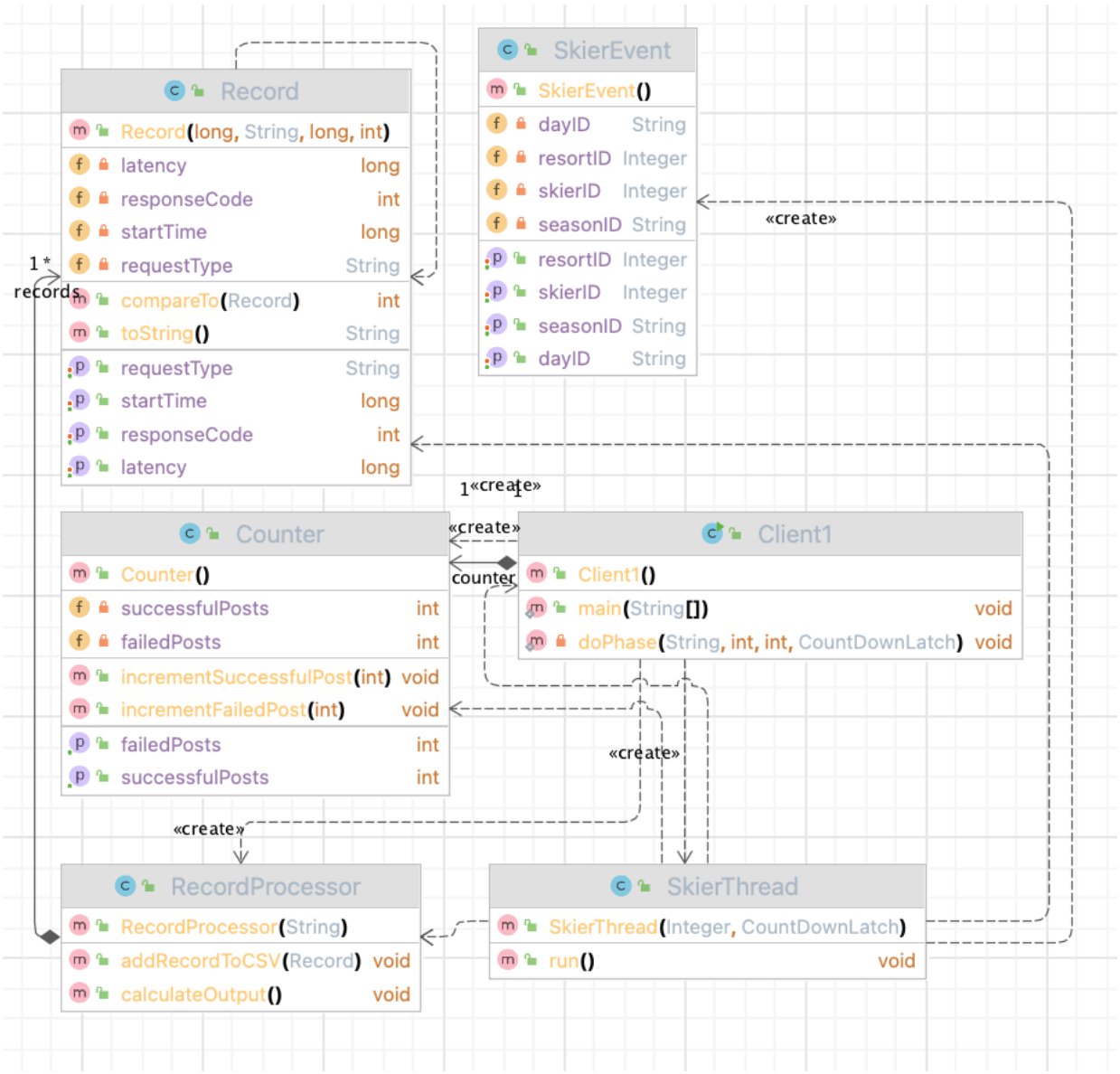
https://github.com/xianhe-zhang/CS6650-22Fall/tree/main/Assignment1

Server Overview

The server-side is implemented as Swagger describes; the following UML shows the class dependency. The server can handle GET/POST request and will return an Message object to POST request.



Client Overview



Part I: To use 32 threads to send 1000 requests each thread**. SkierThread** is responsible for sending specific times requests to server, which is determined by the variable we pass to it. In **Client1**, we will decide how many threads we need for each phase, and how many requests we will send in each thread. Also, we use **Counter** as global variable to record how many successful or failed requests we got. **SkierEvent** is mainly to generate our random data.
Part II: For Client 2, there are 2 new classes **Record** & **RecordProcessor**. Record will record data for every request. Record processor, just like another global object, will do calculation for all records.

Little Law's prediction

```
test duration: 315
phase duration: 57880
```

For **Test**: 50 requests are sent using 1 thread.
Each request takes 315/50 = 6.3 ms

For **Phase**: 200000 requests are sent using 112 threads.
Response Time = 6.3ms
Threads = 112
Little Law Prediction = 112 / 6.3 = 17.7
Actual Output = 42.42

Client 1 Statistics

```
Client Part 1 Result:
-----------------------------------------------
Number of successful requests sent: 200050
Number of unsuccessful requests: 114
The total run time(wall time): 51393 milliseconds
The total throughput per Sec: 3894
```

The result is generated by sending requests to AWS EC2 Instance.

Client 2 Statistics

```
Client Part 2 Result:
-----------------------------------------------
200050
4715032.0
Mean response time: 23.56926768307923
Median response time: 18.0
Throughput: 42.428131982985484
99th response time: 259.0
min and max response time: min: 10.0 , max: 350.0
```

The result is generated by sending requests to AWS EC2 Instance.