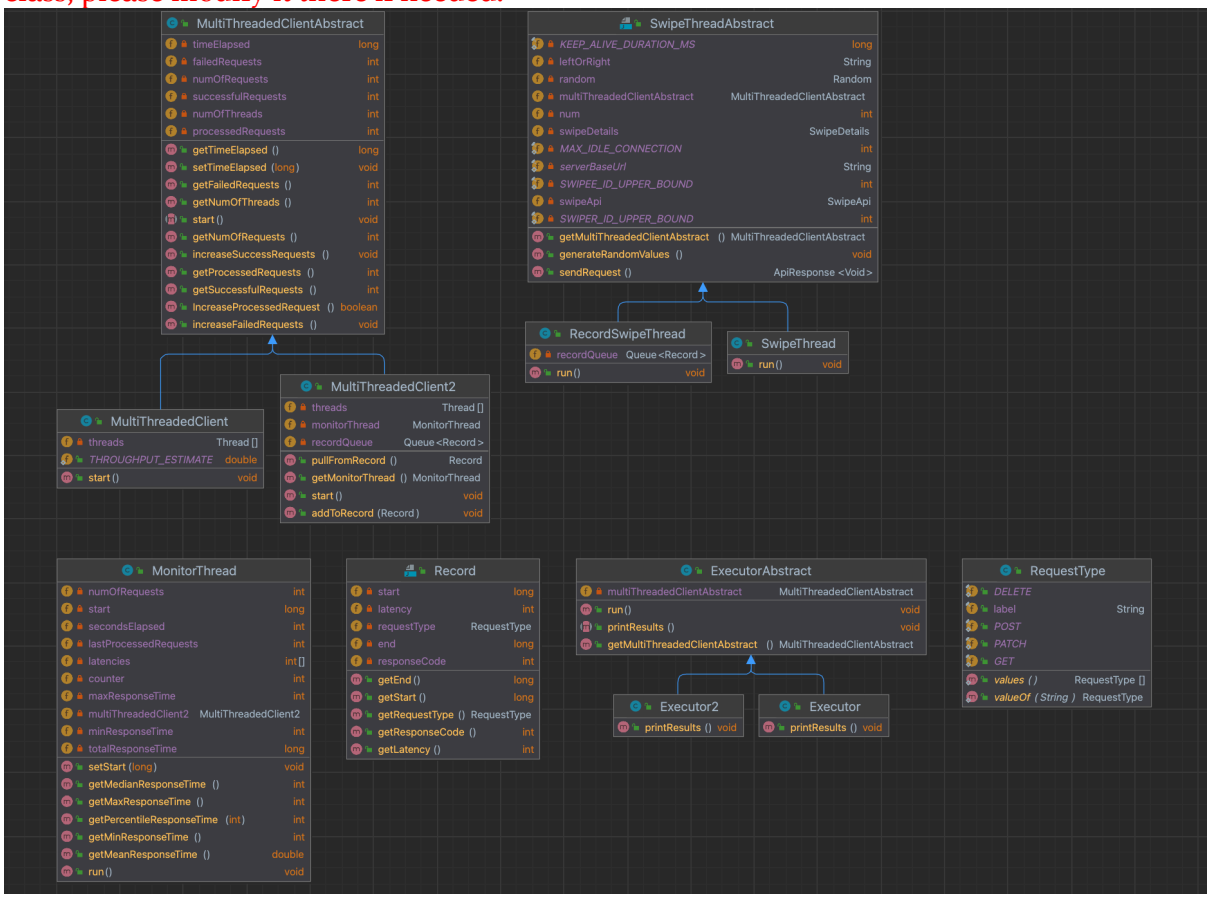
Github Repo

https://github.com/boweill/cs6650/tree/main/Assignment1

Class Description

This is a high-level overview. There are comments in code to assist with readability.
NOTE: Server base url is set up as field SERVER_BASE_URL in **SwipeThreadAbstract** class, please modify it there if needed.



MultiThreadedClientAbstract

- Abstract class;
- Configured with number of requests and number of threads;
- Responsible for stating the threads as specified and waiting for them to finish;
- Contains fields and methods shared by both clients used in Build Client Part 1 and Build Client Part 2 such as number of threads, number of requests, successful requests, failed requests etc.

MultiThreadedClient

- Client class used in Build Client Part 1;
- Configured with number of requests and number of threads;
- Responsible for stating the threads as specified and waiting for them to finish;
- The threads used by the Client are instances of SwipeThread, which is only responsible for sending the requests and updating the number of successful requests and failed requests.

MultiThreadedClient2

- Client class used in Build Client Part 2;
- Configured with number of requests and number of threads;
- Responsible for stating the threads as specified and waiting for them to finish;
- The threads used by the Client are instances of RecordSwipeThread, which is responsible for sending the requests and pushing the records for these requests onto a queue.
- The MonitorThread, which is another field of this class, is responsible for pulling from the record queue and processing the records (i.e. writing data to csv files).

SwipeThreadAbstract

- Abstract class;
- Configured with a MultiThreadedClientAbstract instance associated with the thread (since the thread needs to update values inside the client class), and a number which can be considered the name of the thread, this number is only there for debugging purpose (i.e. checking which thread the request is coming from).
- Responsible for performing a certain task and updating values inside MultiThreadedClientAbstract at the same time;
- Contains fields and methods shared by threads used in both Build Client Part 1 and Build Client Part 2, such SwipeApi, server url, and the function that generates random values for each request.

SwipeThread

- Thread class used in Build Client Part 1;
- Configured with a MultiThreadedClientAbstract instance (in this case a MultiThreadedClient) associated with the thread (since the thread needs to update values inside the client class), and a number which can be considered the name of the thread, this number is only there for debugging purpose (i.e. checking which thread the request is coming from).
- Responsible for sending swipe requests in its run() method and updates values in MultiThreadedClient such as the number of processed requests, successful requests, and failed requests.

RecordSwipeThread

- Thread class used in Build Client Part 2;
- Configured with a MultiThreadedClientAbstract instance (in this case a MultiThreadedClient2) associated with the thread (since the thread needs to update values inside the client class), and a number which can be considered the name of the thread, this number is only there for debugging purpose (i.e. checking which thread the request is coming from).
- Responsible for sending swipe requests in its run() method and updates values in MultiThreadedClient such as the number of processed requests, successful requests, and failed requests.
- Stores the start time, response time, status code etc. of a request using instances of the Record class and pushes the Records onto a queue, from which the MonitorThread will pull the records and process them.

MonitorThread

- Thread class used in Build Client Part 2;

- Configured with a MultiThreadedClientAbstract instance associated with the thread;
- Responsible for processing the Records in the record queue updated by the worker threads that send swipe requests (i.e. writing to csv files).
- Also responsible for sorting the response time after looking at all the records and finding mean, median and percentile response times.

ExecutorAbstract

- Abstract class;
- Contains methods shared by Executor classes used in both Build Client Part 1 and Build Client Part 2;
- Responsible for instantiating and starting a MultiThreadedClientAbstract, running it, and printing the results afterwards.

Executor

- Executor class used in Build Client Part 1;
- Starts a MultiThreadedClient, and prints the wall time, the number of successful and failed request, and throughput as well as throughput estimate.

Excutor2

- Executor class used in Build Client Part 2;
- Starts a MultiThreadedClient2, and prints the wall time, the number of successful and failed request, and throughput as well as throughput estimate. Also prints out statistics including mean, median, 99 percentile, min, max response time.

Record

- Class that stores the record for each thread, including start time, end time, latency, request type and status code.

RequestType

- Enum for the common request types GET, POST, PATCH and DELETE.

Client (Part 1) – Output Window

```
"/Applications/IntelliJ IDEA.app/Contents/jbr/Contents/Home/bin/java" ...
The requests took 96327 milliseconds to complete.
With 500000 successes and 0 failures.
Throughput is 5190.65
Number of threads: 90

Estimated throughput: 5462.49

Process finished with exit code 0
```

Client (Part 2) – Output Window

```
"/Applications/IntelliJ IDEA.app/Contents/jbr/Contents/Home/bin/java" ...
The requests took 93876 milliseconds to complete.
With 500000 successes and 0 failures.
Throughput: 5326.17
Mean response time: 16.88
Median response time: 16.00
99 percentile response time: 30
Minimum response time: 10
Maximum response time: 333


Process finished with exit code 0
```

Throughput Plot over Time