CS6650 Project 3

Stefani Sindarto

GitHub Repository:

Server - https://github.com/stefanisindarto/Twinder3_Server

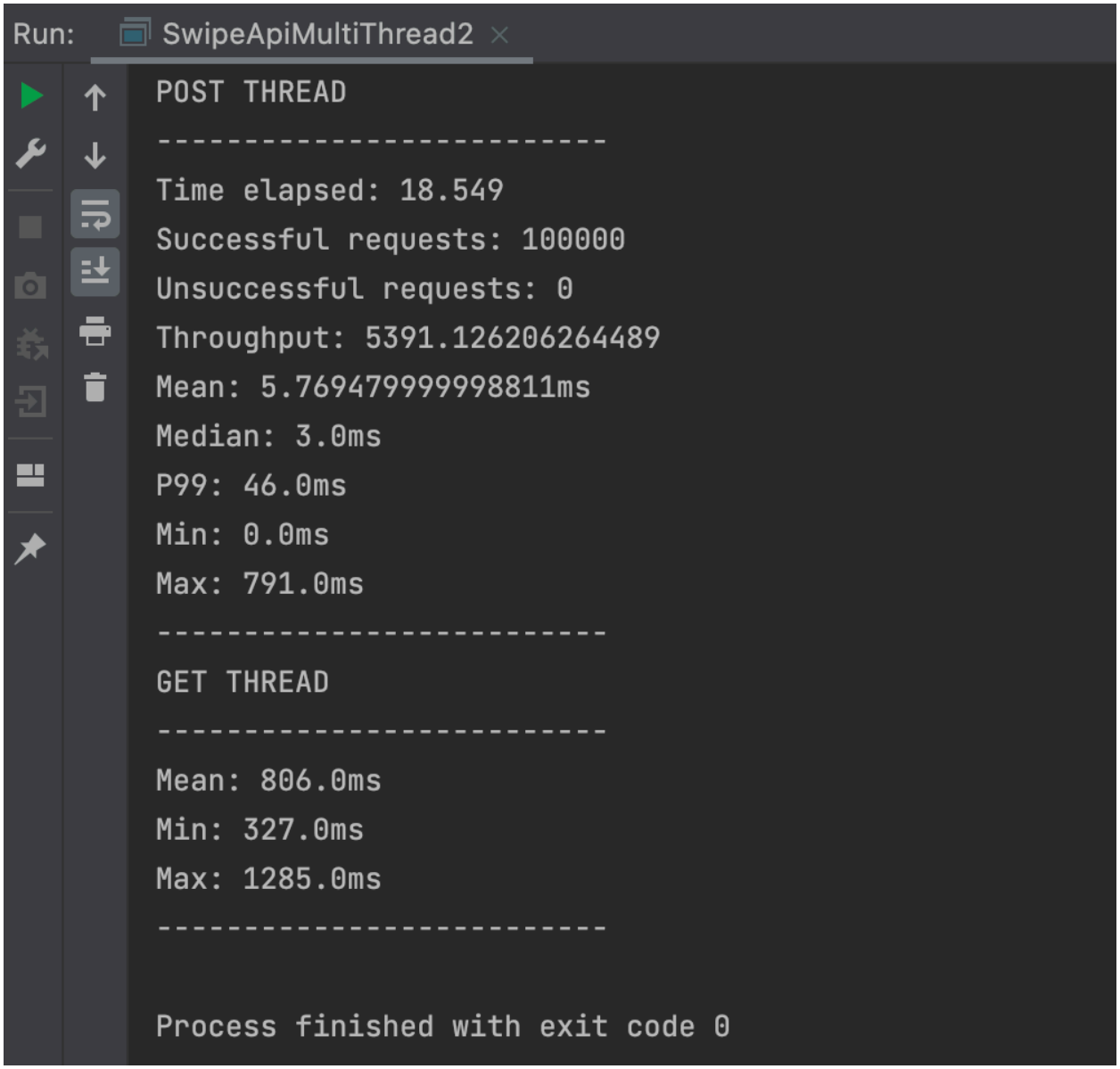
Consumers - https://github.com/stefanisindarto/Twinder3_Consumer

Client - https://github.com/stefanisindarto/Twinder3_client

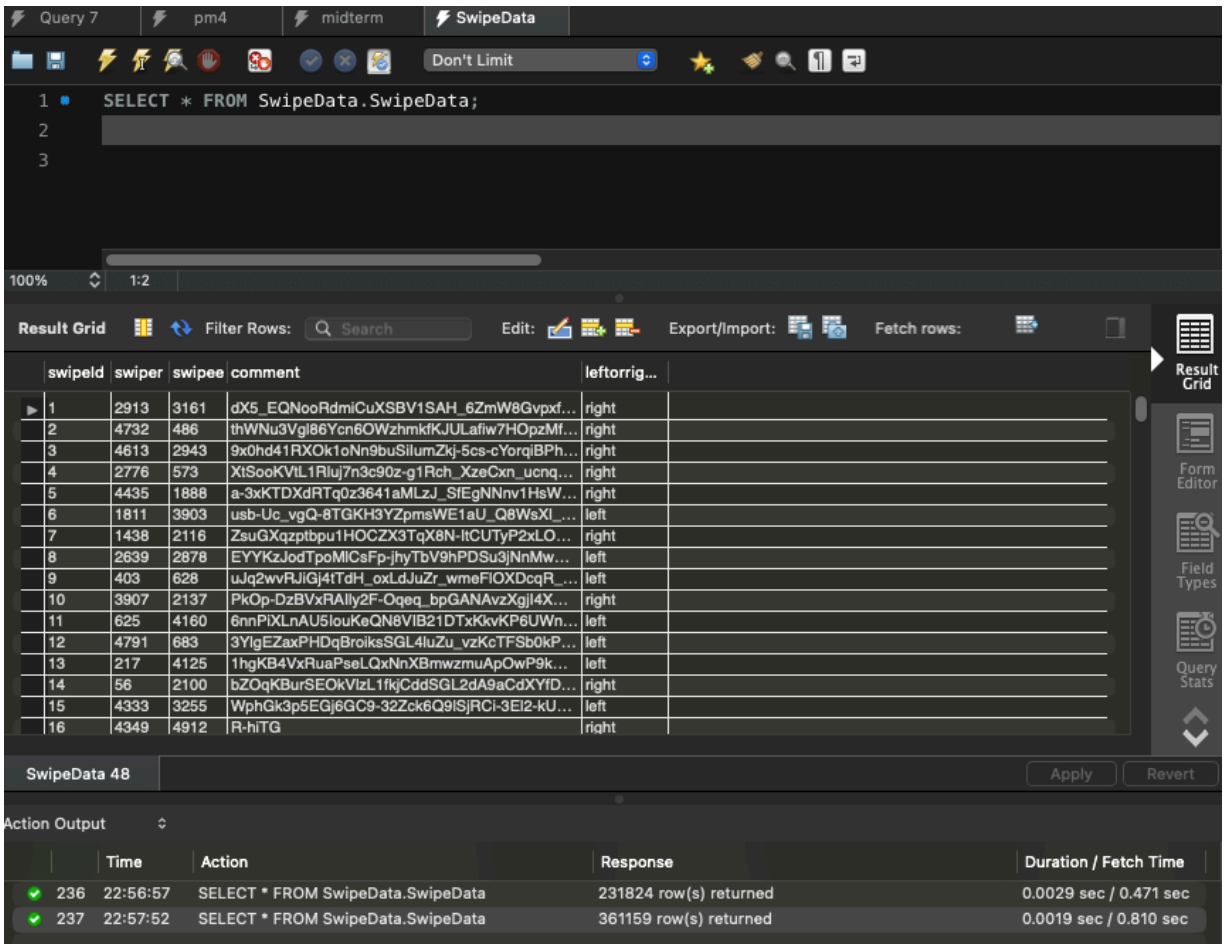
Result From Project 2

```
Time elapsed: 25.79
Successful requests: 100000
Unsuccessful requests: 0
Throughput: 3877.4718883288097
Mean: 4.945341176471323ms
Median: 3.0ms
P99: 44.0ms
Min: 0.0ms
Max: 1491.0ms
```

Result From Project 3



Database - MySQL



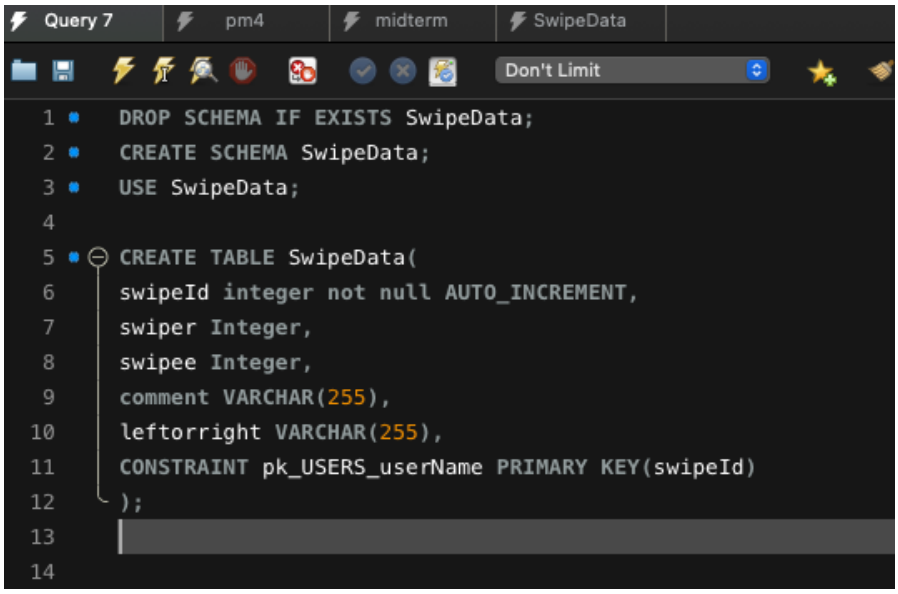
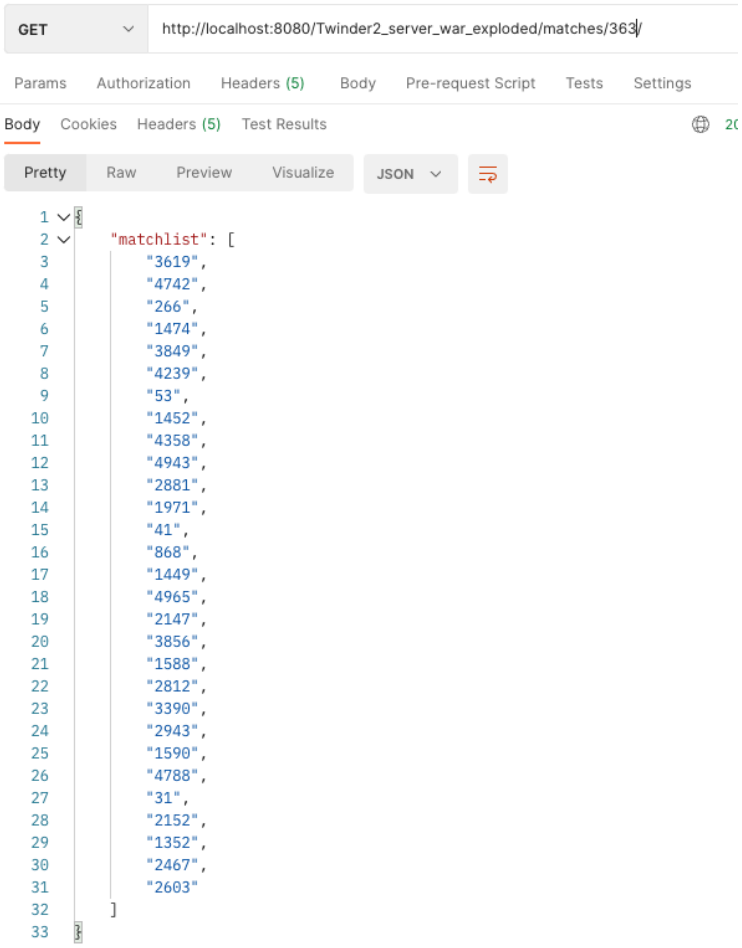
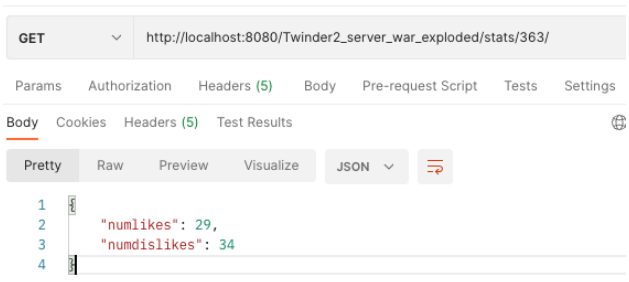


Table Creation & Populated Table

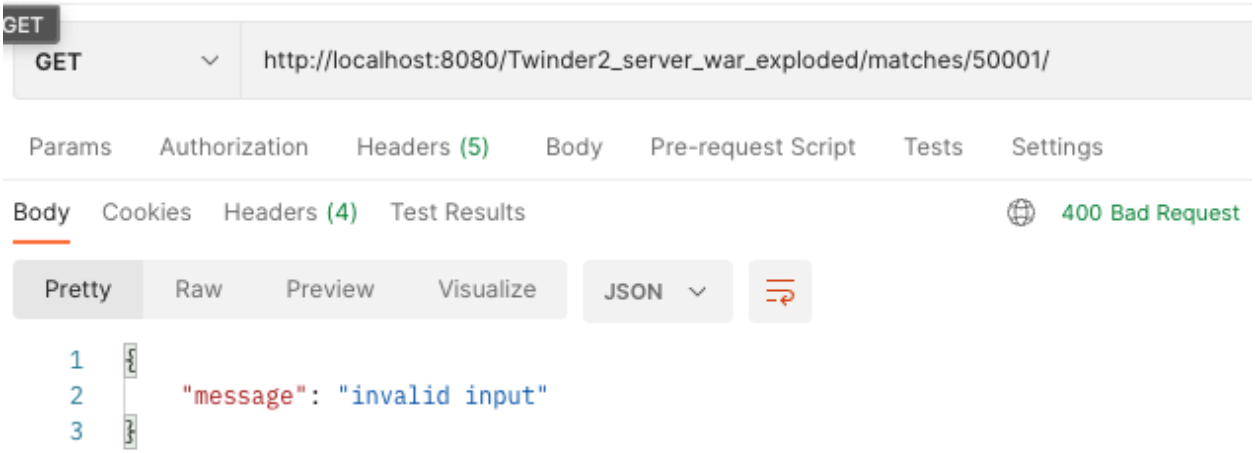
Get Request Successful Output - /matches



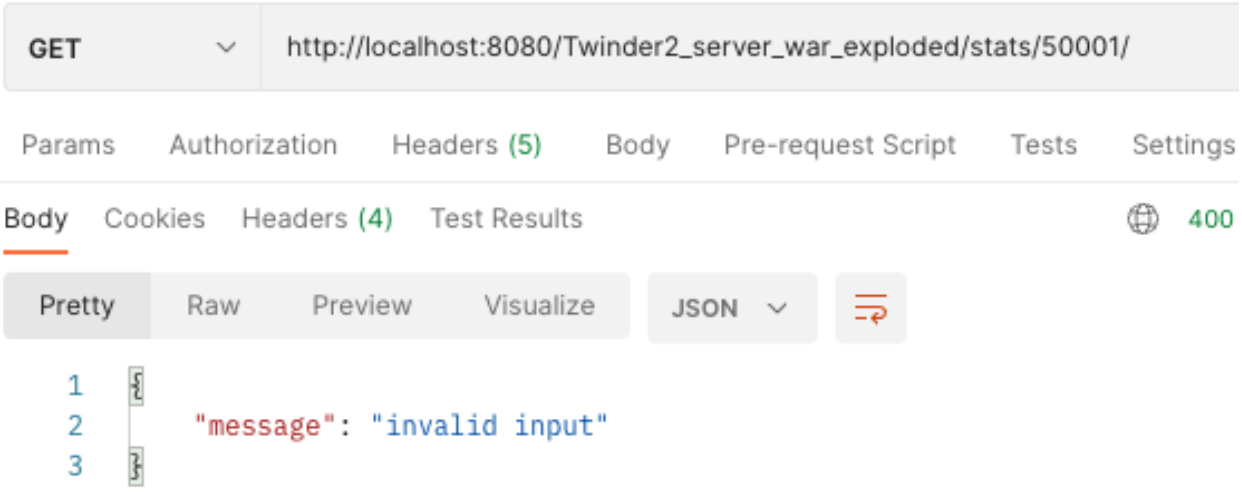
Get Request Successful Output - /stats



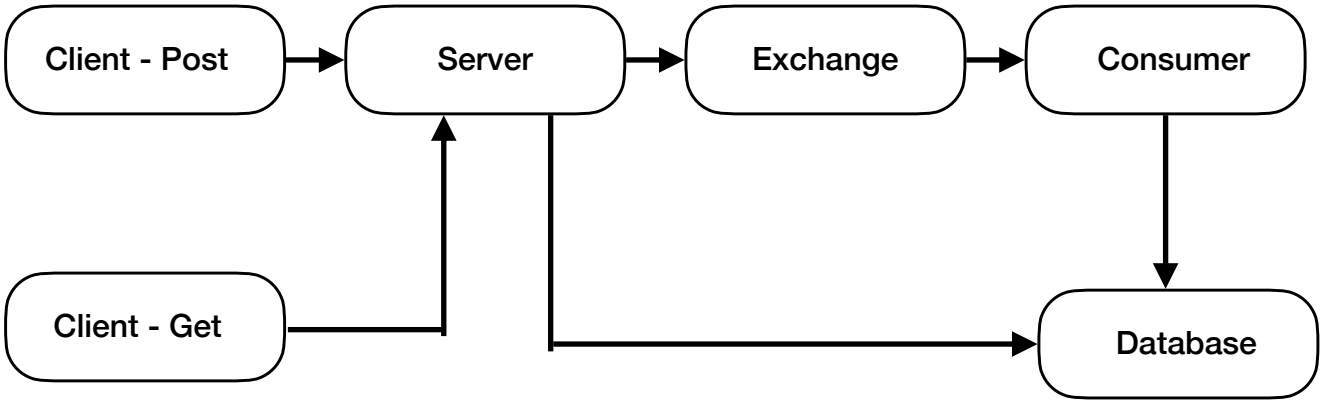
Get Request Unsuccessful Output - /matches



Get Request Successful Output - /stats



Design



- **3 separate projects are created for the purpose of this assignment as listed below:**
  - Twinder_Client (Updated from the previous assignment)
  - Twinder2_Server (Updated from the previous assignment)
  - Twinder2_Consumer ((Updated from the previous assignment)

- **High Level Design**

The Twinder_Client project contains of **Multithreaded class** which create random body and send post requests to the server(Twinder server). When the thread start, it calls another thread (**GetThread**) to send 5 get requests to both Match Servlet and Stats servlet every 5 seconds and it will stop once the last post threads terminates.

There is another separate project which contains the servlets files and DAO, Twinder2_server. In the **Swipe servlet**, the url and the body sent from the client is validated before it is published to a RabbitMQ exchange to be consumed by the consumer. The server class initiate an exchange which is used to send the messages to the consumer queues in the doPost method. There is a **DBCPDataSource class** which establishes connection to the MySQL database and a **SwipeData DAO** which contains method to get list of matches and to get a match statistics for the user. **Match servlet** calls the DAO layer to get the list of matches for the user and return the response in JSON format. **Stats servlet** calls the DAO layer to get the like and dislike counts for the user and return the response in JSON format as well.

The Twinder2_Consumer project contains a Swipe class, 3 consumer classes, a **DBCPDataSource class** which establishes connection to the MySQL database and a **SwipeData DAO**. Consumer1 and Consumer2 are from the previous project and lastly **Consumer3** which is specified for this project.

Consumer 1 and Consumer2 classes will bind the exchange to their respective persistent queues and will consume the message to be converted as a Swipe object before processing the object into their respective hashmaps. Consumer 1 owns a hashmap which stores the user id/swiper as the key and takes in a list of 2 integers. The list represents the count of like and dislike for each swiper. Consumer 2 owns another hashmap which stores the user id/swiper as the key and takes in a list of the swipees' id which the swiper swiped. Consumer3 takes in the messages from the **persistent queue** and calls the SwipeData DAO layer to create a new Swipe object and store it in the database.

- **Deployment Topology - AWS**
  - 1 instance for Server
  - 1 instance for Client
  - 1 instance for Consumer
  - Load balancer
  - AWS RDS - MySQL