Tweeter Project 4 - Part 1

-By Anurag Chakraborty and Jigar Patel

How to execute:

Steps to run the simulation:

1. Compile with ***c(tweeter).***
2. ***tweeter:start_simulation(<STARTING_USER_NUMBER>,<ENDING_USER_NUMBER>>,<TOTAL_NUMBER_OF_REQUESTS>).***

   where,

   · STARTING_USER_NUMBER signify the starting number of the range that is used to create usernames such as "user1" ("user"+STARTING_USER_NUMBER)

   · ENDING_USER_NUMBER signify the ending number of the range that is used to create usernames such as "user100". Therefore, total users created for starting_user_number = 1 and ending_user_number = 100 will be 100 users. (user1, user2, user3 … etc).

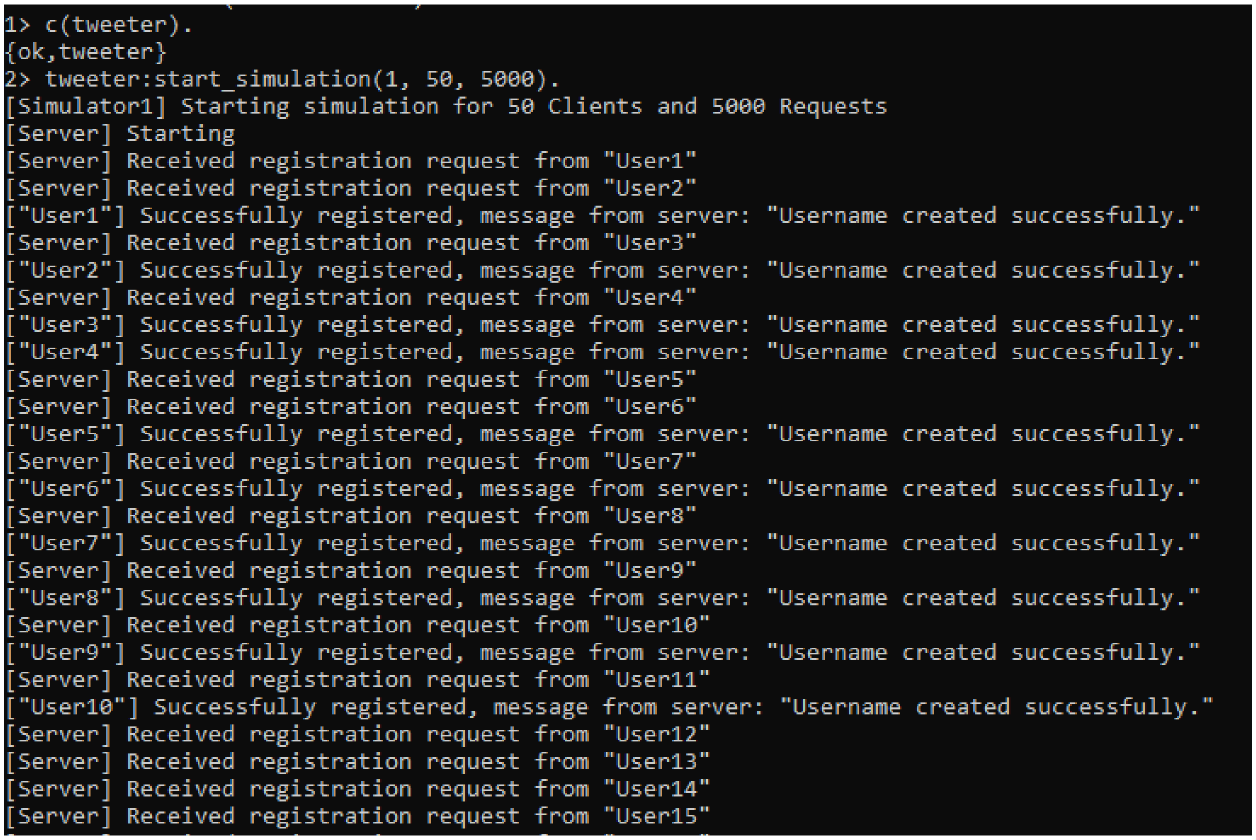
   · TOTAL_NUMBER_OF_REQUESTS signifies the total number of requests in the simulation before it terminates itself

Below are the Input/Output screenshots:

1. Inputs:

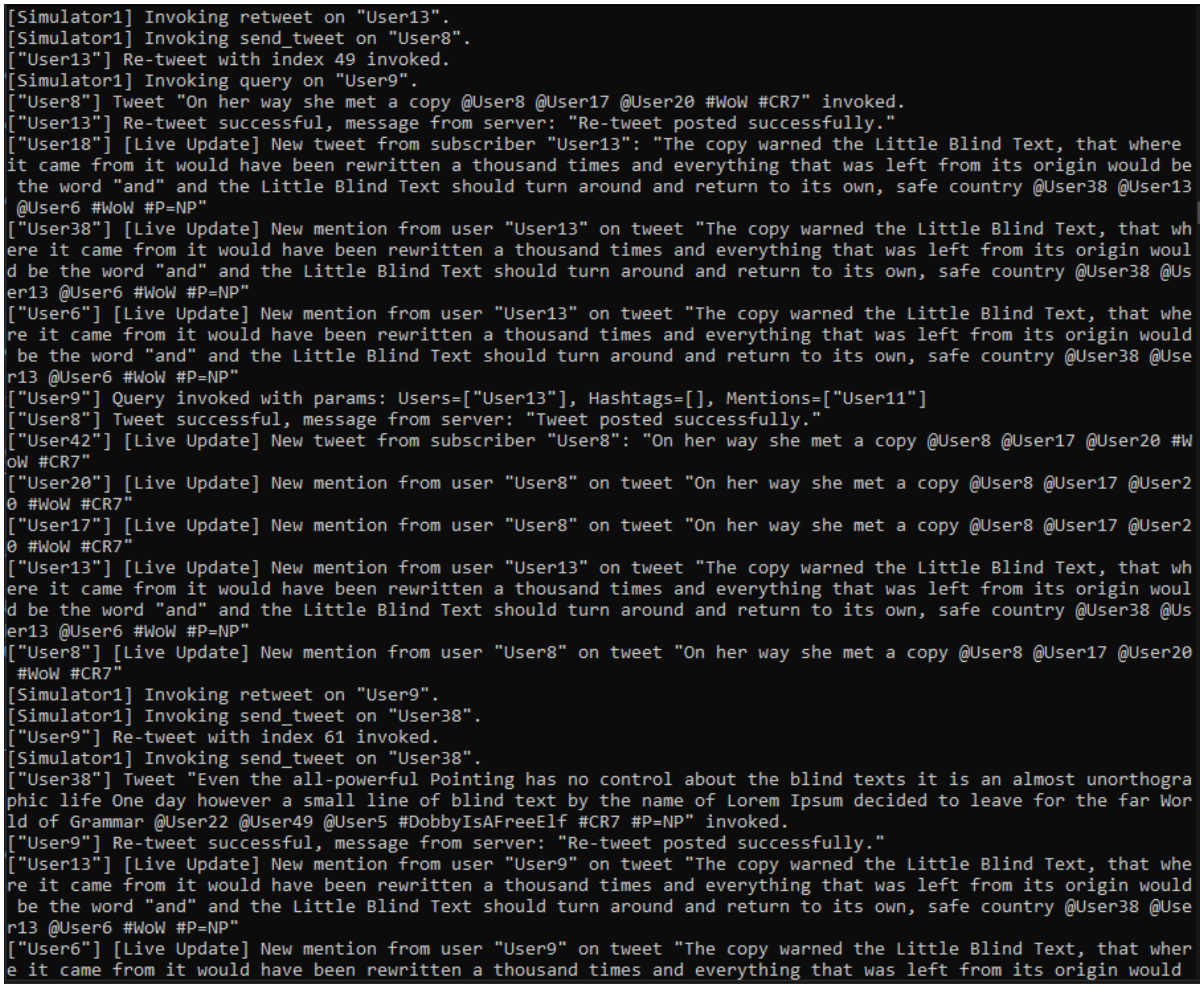
   · STARTING_USER_NUMBER => 1
   ENDING_USER_NUMBER => 50
   TOTAL_NUMBER_OF_REQUESTS => 5000

   Outputs:



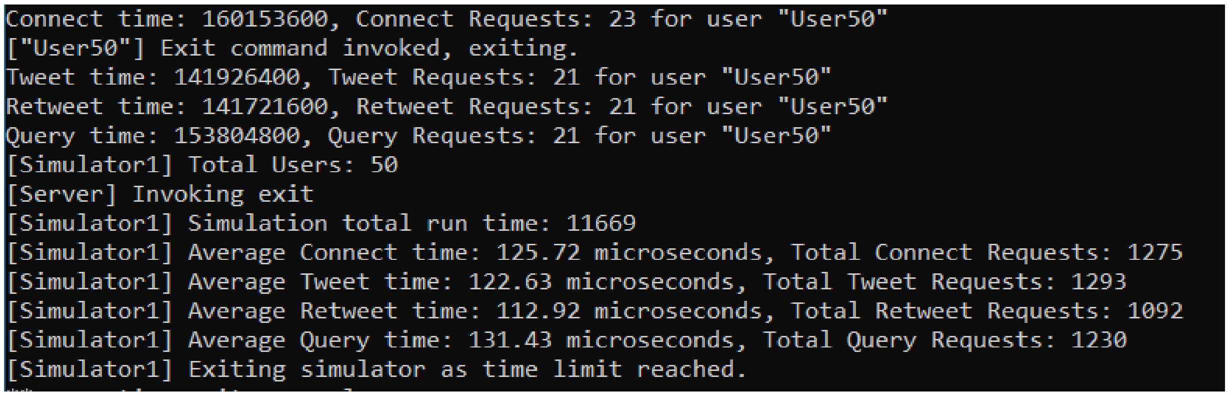
```
1> c(tweeter).
{ok,tweeter}
2> tweeter:start_simulation(1, 50, 5000).
[Simulator1] Starting simulation for 50 Clients and 5000 Requests
[Server] Starting
[Server] Received registration request from "User1"
[Server] Received registration request from "User2"
["User1"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User3"
["User2"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User4"
["User3"] Successfully registered, message from server: "Username created successfully."
["User4"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User5"
[Server] Received registration request from "User6"
["User5"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User7"
["User6"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User8"
["User7"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User9"
["User8"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User10"
["User9"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User11"
["User10"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User12"
[Server] Received registration request from "User13"
[Server] Received registration request from "User14"
[Server] Received registration request from "User15"
```

Fig 1.a) Screenshot of input with 50 users, 5000 total requests. – Start



```
[Simulator1] Invoking retweet on "User13".
[Simulator1] Invoking send_tweet on "User8".
["User13"] Re-tweet with index 49 invoked.
[Simulator1] Invoking query on "User9".
["User8"] Tweet "On her way she met a copy @User8 @User17 @User20 #WoW #CR7" invoked.
["User13"] Re-tweet successful, message from server: "Re-tweet posted successfully."
["User18"] [Live Update] New tweet from subscriber "User13": "The copy warned the Little Blind Text, that where
it came from it would have been rewritten a thousand times and everything that was left from its origin would be
 the word "and" and the Little Blind Text should turn around and return to its own, safe country @User38 @User13
 @User6 #WoW #P=NP"
["User38"] [Live Update] New mention from user "User13" on tweet "The copy warned the Little Blind Text, that wh
ere it came from it would have been rewritten a thousand times and everything that was left from its origin woul
d be the word "and" and the Little Blind Text should turn around and return to its own, safe country @User38 @Us
er13 @User6 #WoW #P=NP"
["User6"] [Live Update] New mention from user "User13" on tweet "The copy warned the Little Blind Text, that whe
re it came from it would have been rewritten a thousand times and everything that was left from its origin would
 be the word "and" and the Little Blind Text should turn around and return to its own, safe country @User38 @Use
r13 @User6 #WoW #P=NP"
["User9"] Query invoked with params: Users=["User13"], Hashtags=[], Mentions=["User11"]
["User8"] Tweet successful, message from server: "Tweet posted successfully."
["User42"] [Live Update] New tweet from subscriber "User8": "On her way she met a copy @User8 @User17 @User20 #W
oW #CR7"
["User20"] [Live Update] New mention from user "User8" on tweet "On her way she met a copy @User8 @User17 @User2
0 #WoW #CR7"
["User17"] [Live Update] New mention from user "User8" on tweet "On her way she met a copy @User8 @User17 @User2
0 #WoW #CR7"
["User13"] [Live Update] New mention from user "User13" on tweet "The copy warned the Little Blind Text, that wh
ere it came from it would have been rewritten a thousand times and everything that was left from its origin woul
d be the word "and" and the Little Blind Text should turn around and return to its own, safe country @User38 @Us
er13 @User6 #WoW #P=NP"
["User8"] [Live Update] New mention from user "User8" on tweet "On her way she met a copy @User8 @User17 @User20
 #WoW #CR7"
[Simulator1] Invoking retweet on "User9".
[Simulator1] Invoking send_tweet on "User38".
["User9"] Re-tweet with index 61 invoked.
[Simulator1] Invoking send_tweet on "User38".
["User38"] Tweet "Even the all-powerful Pointing has no control about the blind texts it is an almost unorthogra
phic life One day however a small line of blind text by the name of Lorem Ipsum decided to leave for the far Wor
ld of Grammar @User22 @User49 @User5 #DobbyIsAFreeElf #CR7 #P=NP" invoked.
["User9"] Re-tweet successful, message from server: "Re-tweet posted successfully."
["User13"] [Live Update] New mention from user "User9" on tweet "The copy warned the Little Blind Text, that whe
re it came from it would have been rewritten a thousand times and everything that was left from its origin would
 be the word "and" and the Little Blind Text should turn around and return to its own, safe country @User38 @Use
r13 @User6 #WoW #P=NP"
["User6"] [Live Update] New mention from user "User9" on tweet "The copy warned the Little Blind Text, that wher
e it came from it would have been rewritten a thousand times and everything that was left from its origin would
```

Fig 1.b) Screenshot of input with 50 users, 5000 total requests. – Middle



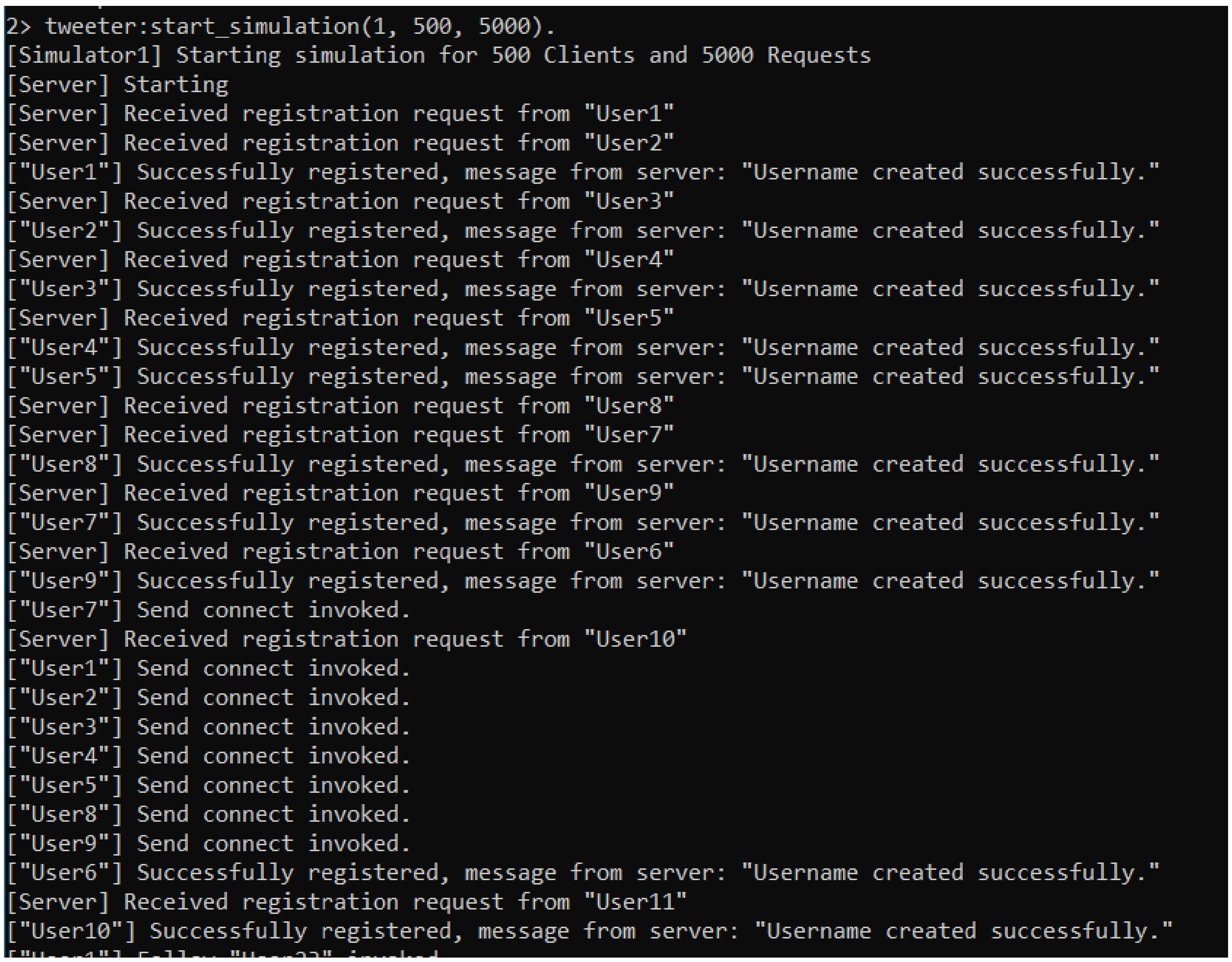
```
Connect time: 160153600, Connect Requests: 23 for user "User50"
["User50"] Exit command invoked, exiting.
Tweet time: 141926400, Tweet Requests: 21 for user "User50"
Retweet time: 141721600, Retweet Requests: 21 for user "User50"
Query time: 153804800, Query Requests: 21 for user "User50"
[Simulator1] Total Users: 50
[Server] Invoking exit
[Simulator1] Simulation total run time: 11669
[Simulator1] Average Connect time: 125.72 microseconds, Total Connect Requests: 1275
[Simulator1] Average Tweet time: 122.63 microseconds, Total Tweet Requests: 1293
[Simulator1] Average Retweet time: 112.92 microseconds, Total Retweet Requests: 1092
[Simulator1] Average Query time: 131.43 microseconds, Total Query Requests: 1230
[Simulator1] Exiting simulator as time limit reached.
```

Fig 1.c) Screenshot of input with 50 users, 5000 total requests. – End

2. Inputs:

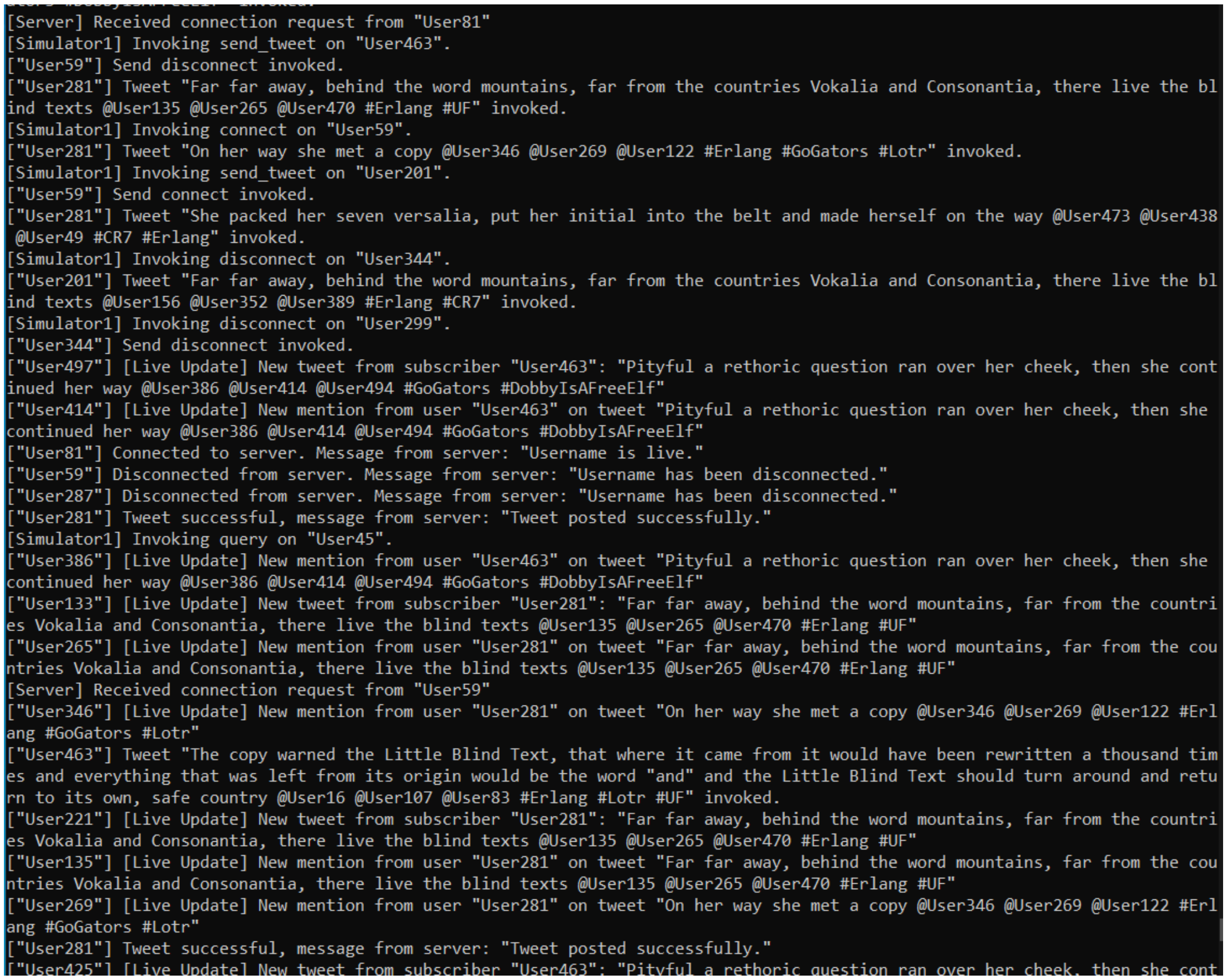
STARTING_USER_NUMBER => 1
ENDING_USER_NUMBER => 500
TOTAL_NUMBER_OF_REQUESTS => 5000

Outputs:



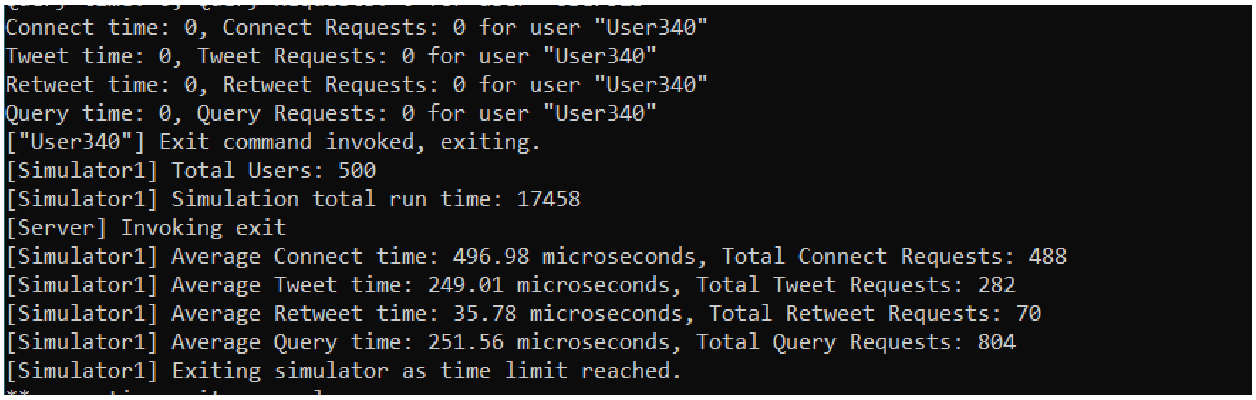
```
2> tweeter:start_simulation(1, 500, 5000).
[Simulator1] Starting simulation for 500 Clients and 5000 Requests
[Server] Starting
[Server] Received registration request from "User1"
[Server] Received registration request from "User2"
["User1"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User3"
["User2"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User4"
["User3"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User5"
["User4"] Successfully registered, message from server: "Username created successfully."
["User5"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User8"
[Server] Received registration request from "User7"
["User8"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User9"
["User7"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User6"
["User9"] Successfully registered, message from server: "Username created successfully."
["User7"] Send connect invoked.
[Server] Received registration request from "User10"
["User1"] Send connect invoked.
["User2"] Send connect invoked.
["User3"] Send connect invoked.
["User4"] Send connect invoked.
["User5"] Send connect invoked.
["User8"] Send connect invoked.
["User9"] Send connect invoked.
["User6"] Successfully registered, message from server: "Username created successfully."
[Server] Received registration request from "User11"
["User10"] Successfully registered, message from server: "Username created successfully."
```

Fig 2.a) Screenshot of input with 500 users, 5000 total requests. – Start



```
[Server] Received connection request from "User81"
[Simulator1] Invoking send_tweet on "User463".
["User59"] Send disconnect invoked.
["User281"] Tweet "Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the bl
ind texts @User135 @User265 @User470 #Erlang #UF" invoked.
[Simulator1] Invoking connect on "User59".
["User281"] Tweet "On her way she met a copy @User346 @User269 @User122 #Erlang #GoGators #Lotr" invoked.
[Simulator1] Invoking send_tweet on "User201".
["User59"] Send connect invoked.
["User281"] Tweet "She packed her seven versalia, put her initial into the belt and made herself on the way @User473 @User438
 @User49 #CR7 #Erlang" invoked.
[Simulator1] Invoking disconnect on "User344".
["User201"] Tweet "Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the bl
ind texts @User156 @User352 @User389 #Erlang #CR7" invoked.
[Simulator1] Invoking disconnect on "User299".
["User344"] Send disconnect invoked.
["User497"] [Live Update] New tweet from subscriber "User463": "Pityful a rethoric question ran over her cheek, then she cont
inued her way @User386 @User414 @User494 #GoGators #DobbyIsAFreeElf"
["User414"] [Live Update] New mention from user "User463" on tweet "Pityful a rethoric question ran over her cheek, then she
continued her way @User386 @User414 @User494 #GoGators #DobbyIsAFreeElf"
["User81"] Connected to server. Message from server: "Username is live."
["User59"] Disconnected from server. Message from server: "Username has been disconnected."
["User287"] Disconnected from server. Message from server: "Username has been disconnected."
["User281"] Tweet successful, message from server: "Tweet posted successfully."
[Simulator1] Invoking query on "User45".
["User386"] [Live Update] New mention from user "User463" on tweet "Pityful a rethoric question ran over her cheek, then she
continued her way @User386 @User414 @User494 #GoGators #DobbyIsAFreeElf"
["User133"] [Live Update] New tweet from subscriber "User281": "Far far away, behind the word mountains, far from the countri
es Vokalia and Consonantia, there live the blind texts @User135 @User265 @User470 #Erlang #UF"
["User265"] [Live Update] New mention from user "User281" on tweet "Far far away, behind the word mountains, far from the cou
ntries Vokalia and Consonantia, there live the blind texts @User135 @User265 @User470 #Erlang #UF"
[Server] Received connection request from "User59"
["User346"] [Live Update] New mention from user "User281" on tweet "On her way she met a copy @User346 @User269 @User122 #Erl
ang #GoGators #Lotr"
["User463"] Tweet "The copy warned the Little Blind Text, that where it came from it would have been rewritten a thousand tim
es and everything that was left from its origin would be the word "and" and the Little Blind Text should turn around and retu
rn to its own, safe country @User16 @User107 @User83 #Erlang #Lotr #UF" invoked.
["User221"] [Live Update] New tweet from subscriber "User281": "Far far away, behind the word mountains, far from the countri
es Vokalia and Consonantia, there live the blind texts @User135 @User265 @User470 #Erlang #UF"
["User135"] [Live Update] New mention from user "User281" on tweet "Far far away, behind the word mountains, far from the cou
ntries Vokalia and Consonantia, there live the blind texts @User135 @User265 @User470 #Erlang #UF"
["User269"] [Live Update] New mention from user "User281" on tweet "On her way she met a copy @User346 @User269 @User122 #Erl
ang #GoGators #Lotr"
["User281"] Tweet successful, message from server: "Tweet posted successfully."
["User425"] [Live Update] New tweet from subscriber "User463": "Pityful a rethoric question ran over her cheek, then she cont
```

Fig 2.b) Screenshot of input with 500 users, 5000 total requests. – Middle



```
Connect time: 0, Connect Requests: 0 for user "User340"
Tweet time: 0, Tweet Requests: 0 for user "User340"
Retweet time: 0, Retweet Requests: 0 for user "User340"
Query time: 0, Query Requests: 0 for user "User340"
["User340"] Exit command invoked, exiting.
[Simulator1] Total Users: 500
[Simulator1] Simulation total run time: 17458
[Server] Invoking exit
[Simulator1] Average Connect time: 496.98 microseconds, Total Connect Requests: 488
[Simulator1] Average Tweet time: 249.01 microseconds, Total Tweet Requests: 282
[Simulator1] Average Retweet time: 35.78 microseconds, Total Retweet Requests: 70
[Simulator1] Average Query time: 251.56 microseconds, Total Query Requests: 804
[Simulator1] Exiting simulator as time limit reached.
```

Fig 2.c) Screenshot of input with 500 users, 5000 total requests. – End

Implementation Details:

Engine:

We have created a tweeter engine that can perform the following functionalities:

- Register Account: We maintain an in-memory data structure to store the unique usernames for the users in the system. Every first time user must provide a unique username.

- Connect: Users can choose to connect to the server (engine) to receive live updates.

- Disconnect: Users can choose to disconnect from the server. Disconnected users will no longer receive live updates.

- Send Tweets: Every tweet made by users is stored in an in-memory data structure and the index of this list is used for referencing a particular tweet within the system. Each tweet can have multiple hashtags and mentions of other users as well.

- Subscribing/following users: A user can follow/subscribe to other users, to receive live updates of tweets made by them whenever the user is connected to the server.

- Query: Users can query tweets by providing either a list usernames or a list of hashtags or a list of mentions.
  We store the hashtags in a in-memory map, where key is the hashtag and the value is a list of indexes. These indexes refer to the indexes of the list that stores all tweets, thus giving us a efficient way to lookup tweets that have particular hashtags.

  We also store the mentions in a in-memory map, where key is the mentioned username and the value is a list of indexes. These indexes refer to the indexes of the list that stores all tweets, thus giving us a efficient way to lookup tweets that have particular usernames mentioned.

- Re-tweets: A user can re-tweet by performing a query to get the particular tweet that is to be re-tweeted.

- Live_Users: Every user that is connected to the server is considered a live user. We maintain the list of live users in an in-memory data structure and every such user will receive live notifications, if they have been mentioned in a tweet/re-tweet or if any of the users they follow sends a tweet/re-tweet.

Simulator:

We have also implemented a simulator, to test all the functionality and behaviour of the system.

The simulator starts with spawning the server (engine) and then spawning client which then register users with unique usernames.

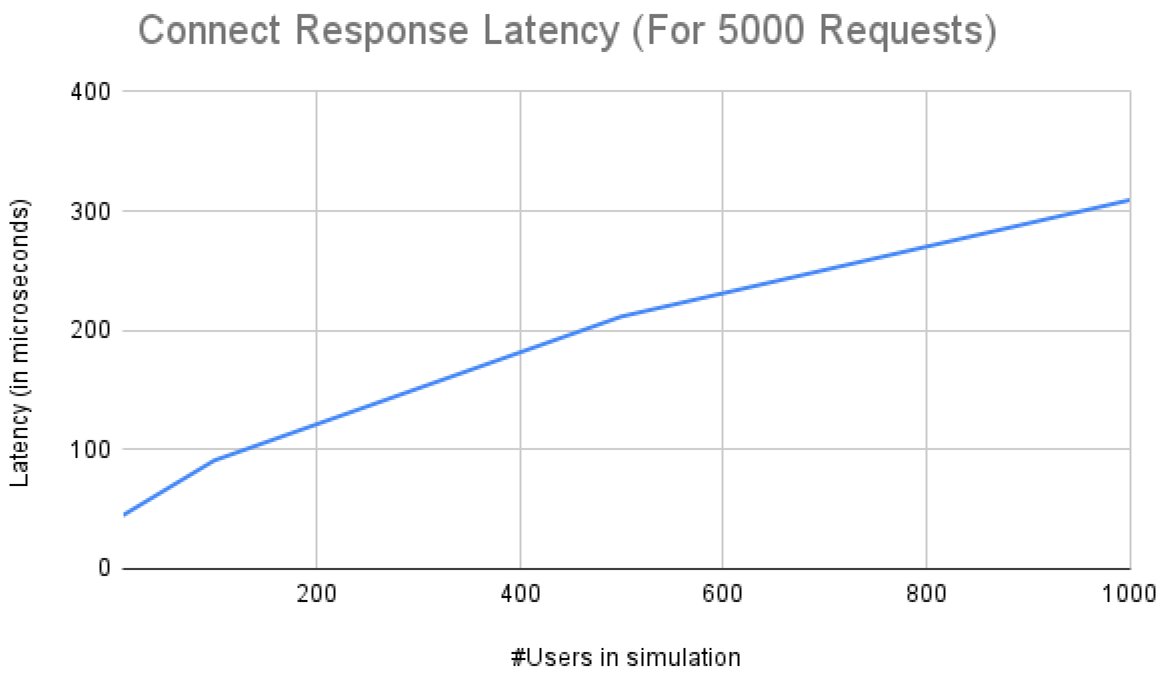
Input parameters which are required are starting_user_number, ending_user_number and total_number_of_requests. The simulator will create usernames with the string "user" + starting_user_number till ending_user_number and the simulation runs until the number of requests in the system is equal to total_number_of_requests.

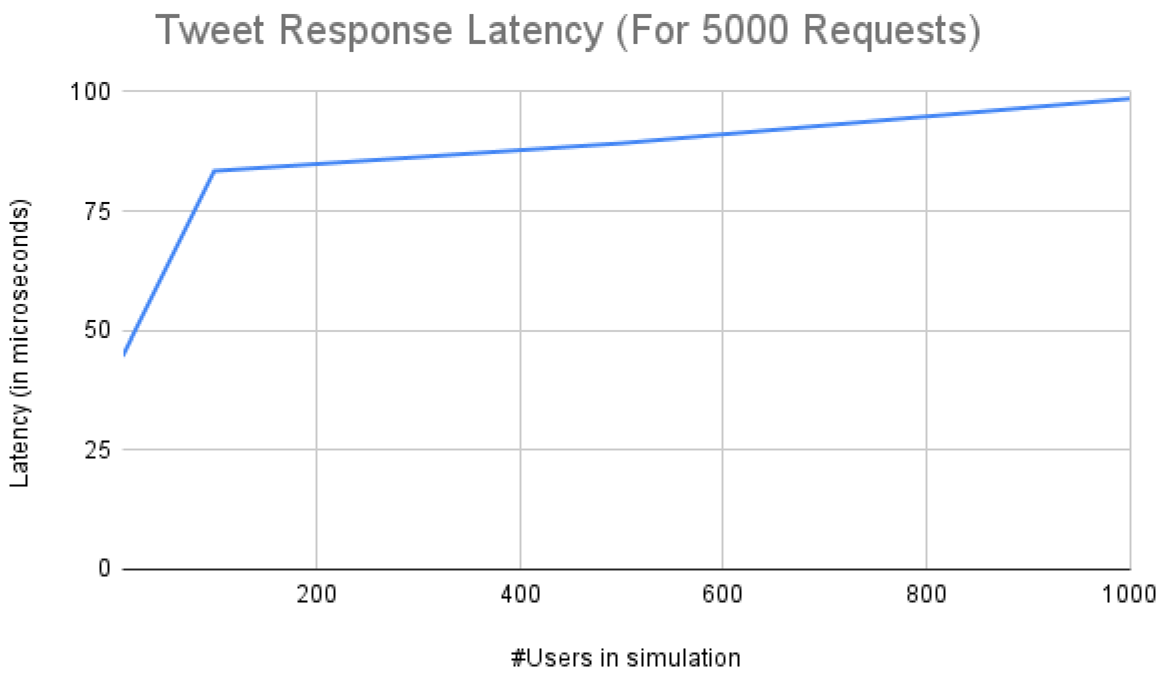
The simulator then initializes subscribers of a user with a zipf distribution. Therefore, a user with large number of followers will have more number of tweets.

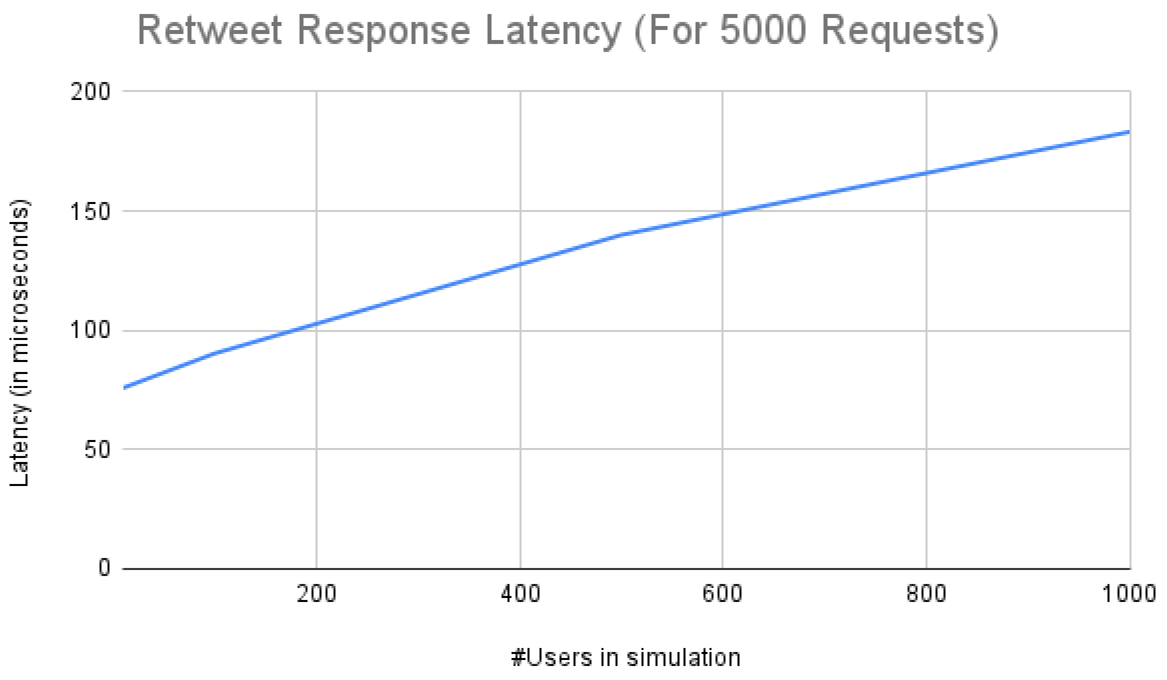
The simulator randomly performs either one of the following functionalities for a user: (connect, disconnect, query, tweet, re-tweet) until the total number of requests provided in the input is reached.

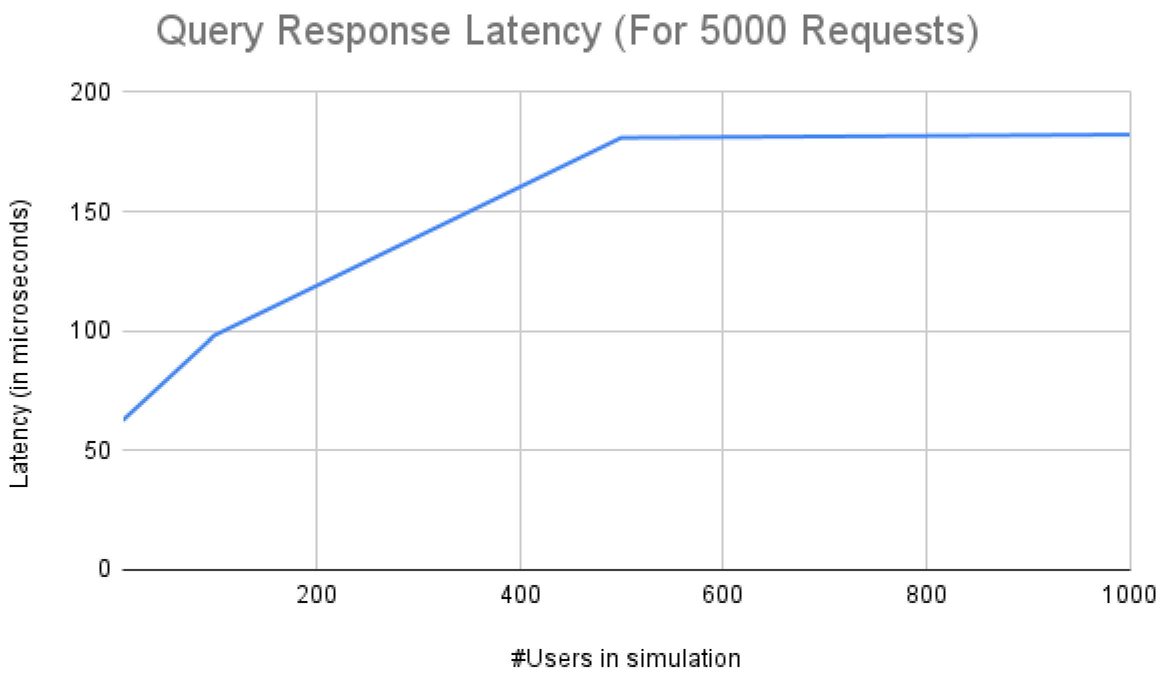
Observations:

- Here we can observe from the below graphs a steady increase in the latency for each of the graphs as the number of users in the simulation increases.
- To rectify this and further improve performance, we could spawn a separate process which would specifically handle the responses.
- This would provide the server (engine) more bandwidth to handle all other requests and thus reduce overall latency.


Connect Response Latency (For 5000 Requests)
Latency (in microseconds)
400
300
200
100
0
200
400
600
800
1000
#Users in simulation


Tweet Response Latency (For 5000 Requests)
Latency (in microseconds)
100
75
50
25
0
200
400
600
800
1000
#Users in simulation


Retweet Response Latency (For 5000 Requests)
200
150
100
50
0
200
400
600
800
1000
Latency (in microseconds)
#Users in simulation


Query Response Latency (For 5000 Requests)
200
150
100
50
0
200
400
600
800
1000
Latency (in microseconds)
#Users in simulation