TWITTER SIMULATOR

Team Members:

- Divya Saroja Rengasamy - UFID: 3338-1372
- Riddhima Arora - UFID: 1941-1995

How to run:

Our project has to be run on two separate terminals on the same machine. Perform step 2 for both terminals.

1. Unzip project4.zip
2. Go to project4 folder
3. Run: mix escript.build
4. Run the server on one terminal with ‘0’ as the arg1 followed by number of users in the arguments like: escript ./project4 arg1 no_users eg. **escript ./project4 0 500**
5. Run the client manager on another terminal with ‘1’ as the arg1 followed by number of users in the arguments like: escript ./project4 arg1 no_users eg. **escript ./project4 1 500**

Project Features:

- Our project has a server which maintains all the ETS tables including user table, hashtags and mentions.
- We have a client manager which is responsible for starting all the actors which are essentially our users. Every time a user is created it registers with the server, where a new entry is added to the user table. After all the users are registered we generate the followers and following list with Zipf distribution for every user.
- We have a static list of tweets and hashtags to be chosen from on the client side. A message is randomly selected by a client actor from this list while performing send tweet action.
- The client manager then randomly chooses an action from send tweet, query a tweet, query a mention, and toggling the online state for a random user. This simulation is recursively called.
- When a user receives a tweet, he has an option to retweet the tweet which happens on a random basis.
- We toggled the online state of a user in intervals of 1-5 seconds randomly chosen for each user.

- The number of tweets sent by a user was made proportional to the number of followers by performing the send tweet and retweet action 100 * no of followers times. In case the user had 0 followers, number of tweets was set to 50.
- After the user has tweeted/retweeted that many times, the user is made inactive.
- We measured performance metrics by changing our code and logging the result for different number of users. We stopped all other activity and just kept send tweet and retweet as the possible action to compute tweets/second.
- The output on the client side shows a live feed of everything happening in our simulator, users sending tweets, retweeting, querying for hashtags and mentions, and also temporary time of disconnection of different users.

Performance Metrics:

- **Maximum number of users**: We tried up to 20,000 users and the system began to lag after 17,000 users. For users more than this, our system was going out of memory.
- **Tweets per second**: We computed tweets/second by keeping a constant number of total number of tweets, measuring the time after the total was tweeted by multiple users**.** We kept increasing the number of users keeping total number of tweets to be sent constant. We observed that as we increased the number from 50 to 500 to 10,000 users, the tweets/second increased from 5 to 100 to 1000 tweets/second.
- **CPU Utilization:** Our CPU utilization went up to 100% in the beginning when the users were registering themselves. After a certain amount of time the memory reached 100% and for users larger than 20,000 the system runs out of memory.
- **Casts per second:** This is a measure of how many times each client hits the server. For 500 users we measured it to be 1.6 requests/second and for 1000 users it was 1.3 requests/sec.