DISTRIBUTED OPERATING SYSTEM PRINCIPLES
PROJECT 4.2
(README)

Team Members:
- Jayavidhi Kumar
- Aditya Subramanian

Demo Video Link
https://www.youtube.com/watch?v=3z6y0e42RkE

Overview
In the first phase of the project, we created a Twitter simulator called Tweeter that includes the following features: user registration, sending and receiving tweets, user and hashtag following and retweets. For the second phase of the project, we built a REST API interface using WebSockets to implement the above-mentioned features.

Implementation:

WebSockets:
The WebSocket protocol allows two-way communication between the client and host server over a single TCP connection. In Erlang, a variety of framework libraries are available such as Cowboy, Elli, MochiWeb, SocketIO, and so on. For our project, we decided to use the Cowboy framework along with Rebar3 for project setup and the Postgres database.

Functions/Features:
- Register user: user registration with a unique id
- Send Tweets: Tweets with hashtags and other user mentions.
- Subscribe/Follow: Users can follow other users.
- Re-tweets: user gets a tweet received by the sender from different means.
- Search/Query tweets: search or filter tweets with user mention or hashtag.

Compilation and Execution:
1. Extract the zip file.
2. cd to the project4 directory.
3. Install rebar3.
4. Open the `erl` shell in this directory
5. Compile the code file:
   a. c(tweeter_server).
   b. c(tweeter_app).
   c. c(tweeter_sup).
6. CMD 1 (server):
   a. rebar3 shell –name user1@<ip_address> –setcookie <cookie_name>
   b. tweeter_server:initServer().
7. CMD2 (for user):
   a. Rebar3 shell –name user1@<ip_address> - - setcookie <cookie_name>
   b. net_adm:ping('server@<ip_address>').
   c. S=rpc:call('server@<ip_address>',erlang,whereis, [server]).
   d. S! {self(), {login, "user 1"}}.
8. To login/register:

a. S! {self(), {register/login, “handle”}}.

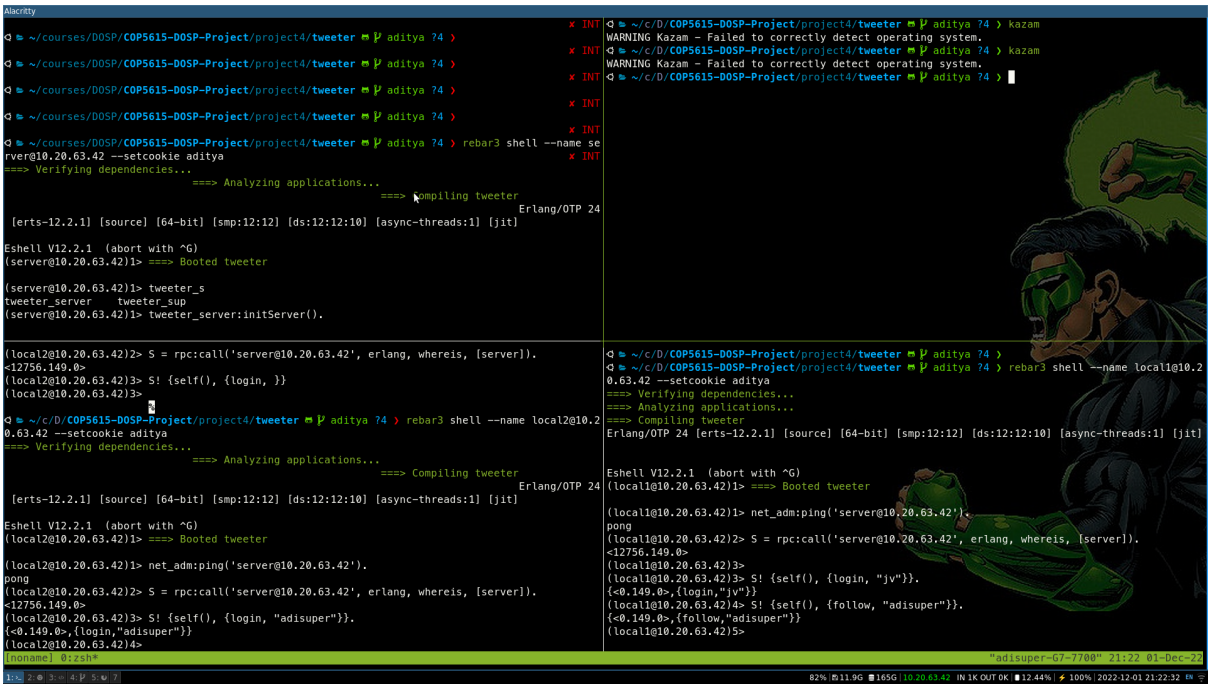
9. To tweet
    a. S! {self(), {tweet, “Write tweet here”}}.

Termination

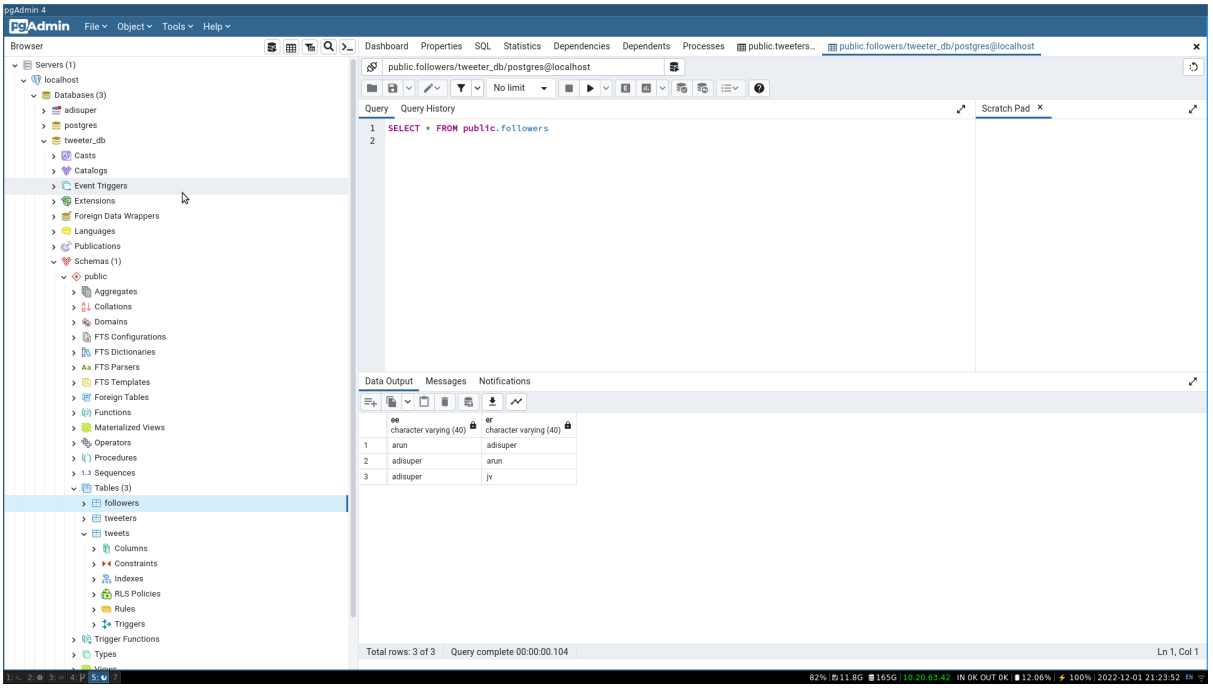
The code terminates once a threshold is reached for the number of requests or can be manually terminated.

Output:

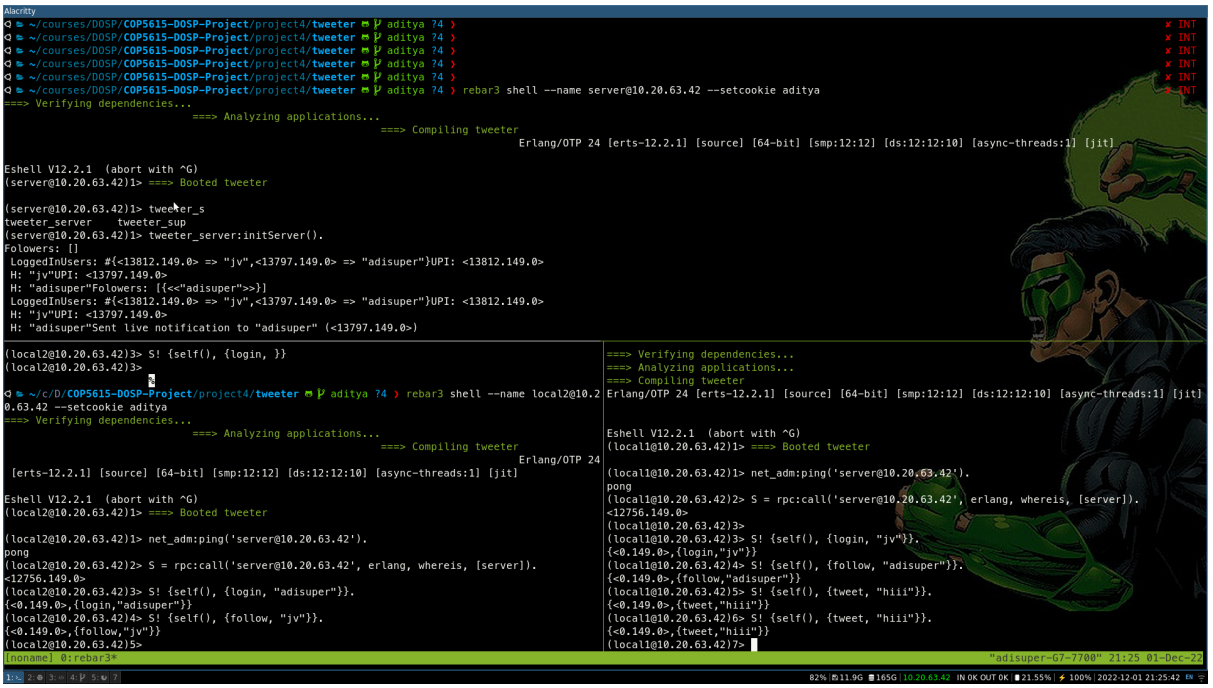
Login/Register and Follow



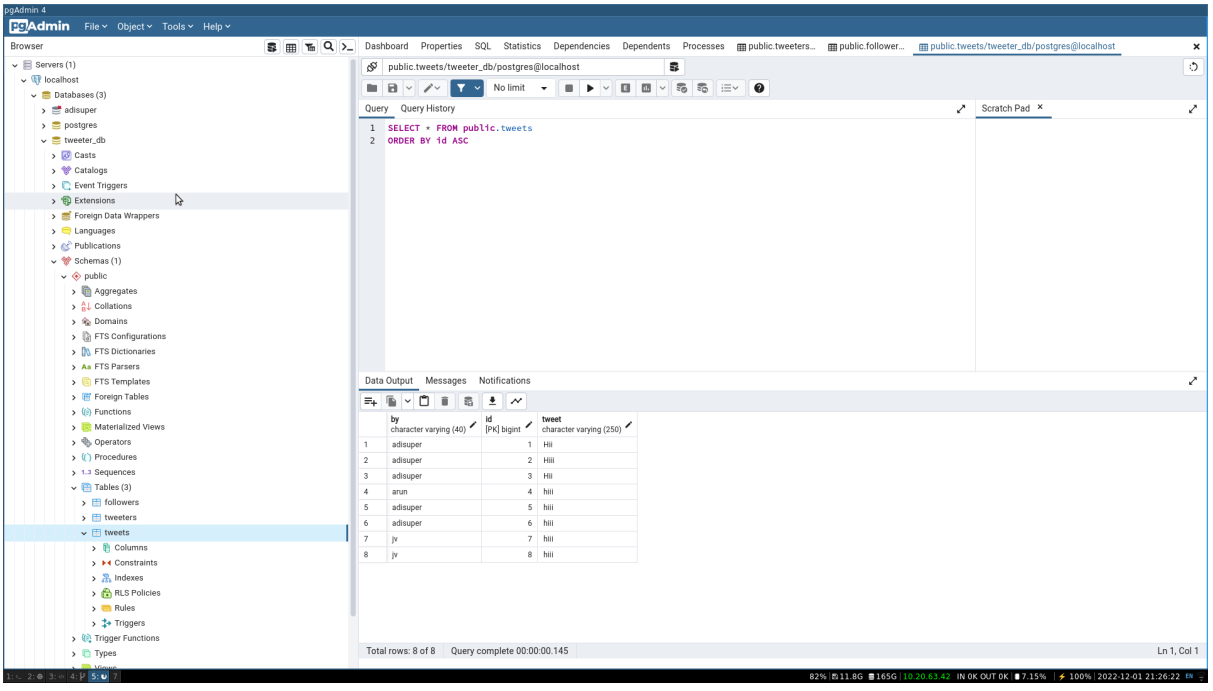
Database connection for login and follow:



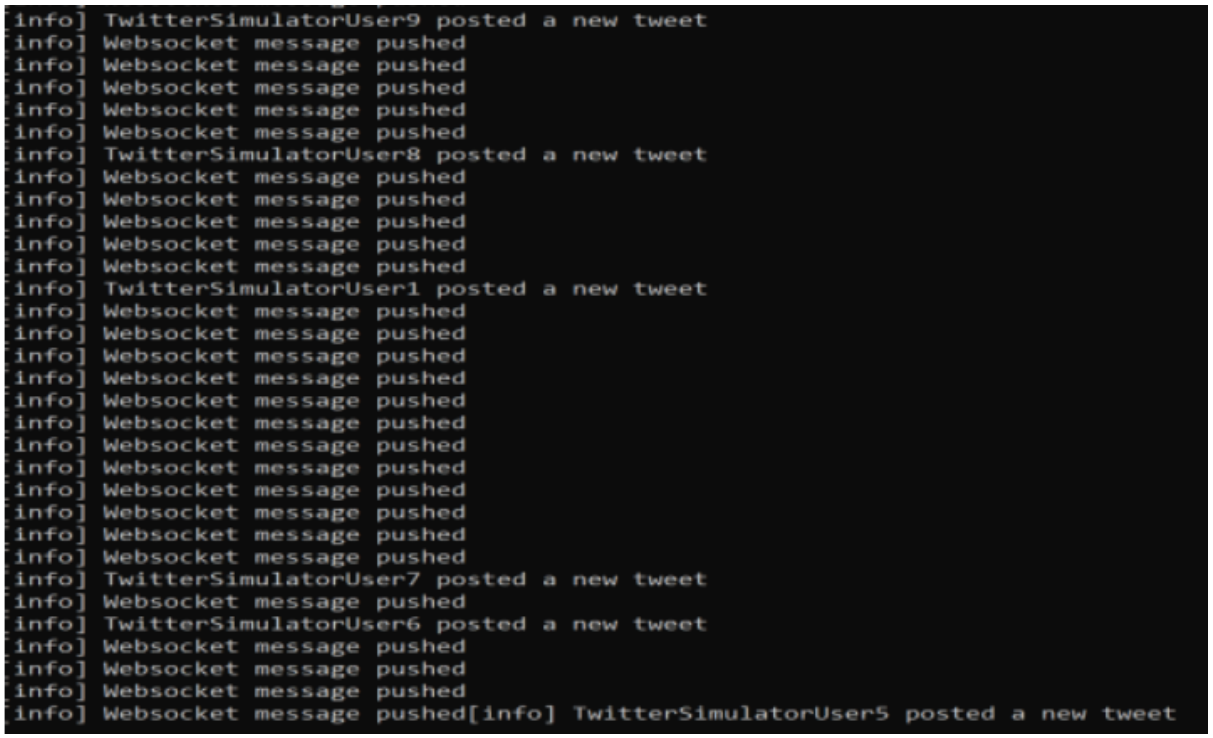
Send Tweet and notification



Database setup for tweet send and notification



Simulator:


info] TwitterSimulatorUser9 posted a new tweet
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] TwitterSimulatorUser8 posted a new tweet
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] TwitterSimulatorUser1 posted a new tweet
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] TwitterSimulatorUser7 posted a new tweet
info] Websocket message pushed
info] TwitterSimulatorUser6 posted a new tweet
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed
info] Websocket message pushed[info] TwitterSimulatorUser5 posted a new tweet