Twitter Implementation(Project-4.1)

Distributed Operating Systems

Group Members:

Chandan Chowdary (UFID-6972-9002)

Gayathri Manogna Isireddy (UFID-9124-0699)

Project Description and Approach:

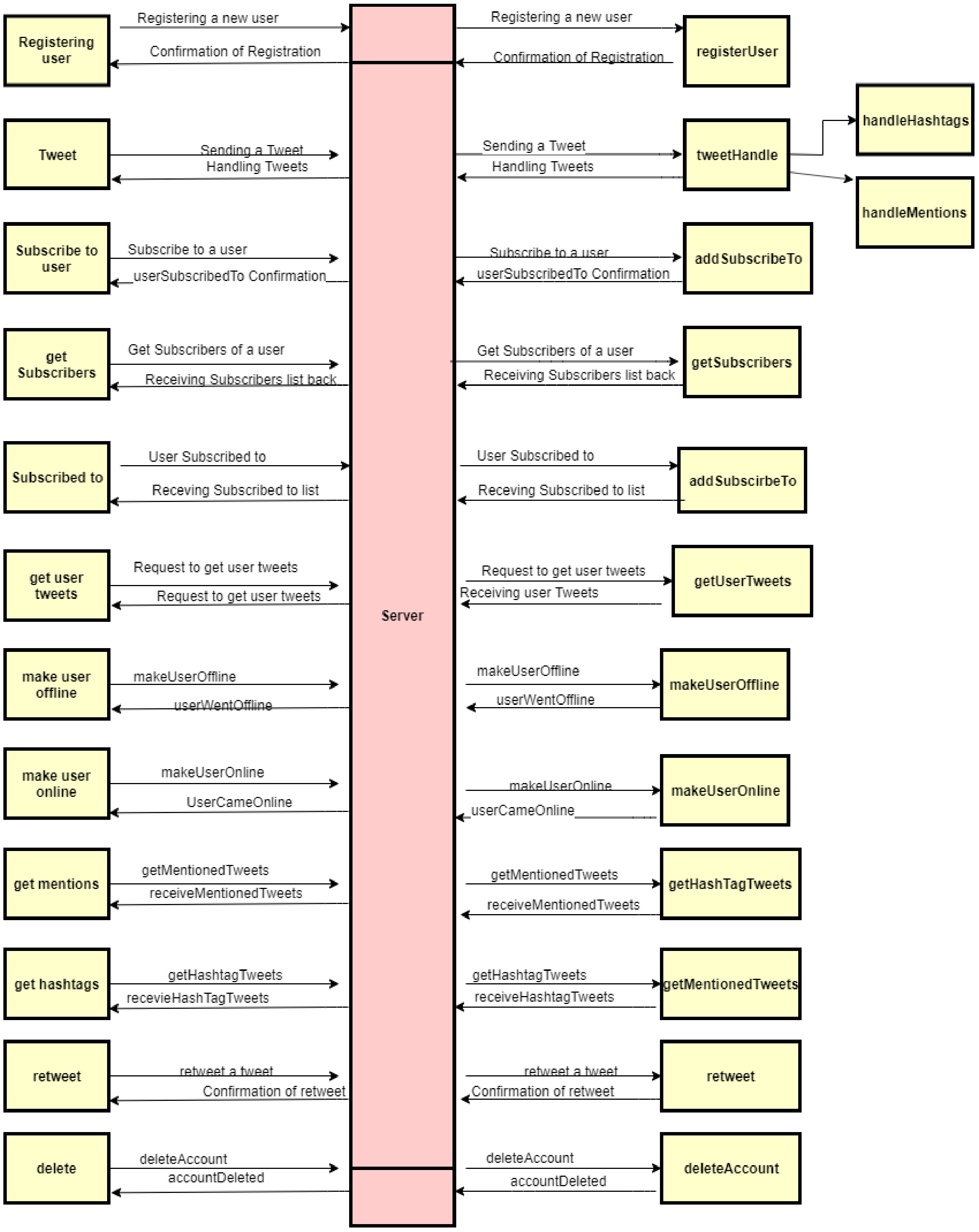
The goal of this project is to implement a Twitter Clone. We implemented a twitter like engine and this engine will be paired up with web sockets to provide full functionality in part 2.
Part 1, which is twitter like engine is implemented with following functionalities-

- Register account
- Send tweet. Tweets can have hashtags (e.g. #COP5615isgreat) and mentions (@bestuser)
- Subscribe to user's tweets
- Re-tweets (so that your subscribers get an interesting tweet you got by other means)
- Allow querying tweets subscribed to, tweets with specific hashtags, tweets in which the user is mentioned (my mentions)
- If the user is connected, deliver the above types of tweets live (without querying)

Entire project runs on the concept of simulator. We need not provide any input to the project, it runs like an application in the real world. The users gets registered in the starting and keeps on tweeting. Connecting and disconnecting the users have been implemented similar to the login and logout in the real world. The user gets tweets only if he is logged in and he can even retweet the tweet of an user that he has subscribed to. He can also request for the tweets that he has made, the tweets in which he is mentioned and also the tweets in which a specific hash tag is present. Entire process takes place through server. All the tweets are gone to the server and the server does the rest. Online and offline of users is managed by the server.

Also, implemented test cases with EXUNIT to test these functionalities. These test cases verifies the output of the project. Below are some of the performance results that we have calculated for different number of users and tweets.

Below is the figure that explains the hierarchy and the implementation that we have done in this project. Left side part represents the clients and the right side part represents the handling part that is done by the server.


Registering user
Registering a new user
Confirmation of Registration
Server
Registering a new user
Confirmation of Registration
registerUser
Tweet
Sending a Tweet
Handling Tweets
tweetHandle
handleHashtags
handleMentions
Subscribe to user
Subscribe to a user
userSubscribedTo Confirmation
addSubscribeTo
get Subscribers
Get Subscribers of a user
Receiving Subscribers list back
getSubscribers
Subscribed to
User Subscribed to
Receving Subscribed to list
addSubscirbeTo
get user tweets
Request to get user tweets
Request to get user tweets
Receiving user Tweets
getUserTweets
make user offline
makeUserOffline
userWentOffline
makeUserOffline
make user online
makeUserOnline
UserCameOnline
userCameOnline
makeUserOnline
get mentions
getMentionedTweets
receiveMentionedTweets
getHashTagTweets
get hashtags
getHashtagTweets
recevieHashTagTweets
receiveHashtagTweets
getMentionedTweets
retweet
retweet a tweet
Confirmation of retweet
retweet
delete
deleteAccount
accountDeleted
deleteAccount

Implementation Details-

Proj4.ex: This elixir file is the starting point and contains the main method. Our implementation takes 2 parameters in the order – num_user, num_msg

- num_user: number of users to simulate (eg.10)
- num_msg: Number of tweets each user can tweet.(eg.2)

Server.ex: In this file twitter server engine code is implemented. ETS tables are used for database implementation. This engine is responsible for registering user, processing tweets and distributing tweets. Engine directly communicates with the database.
Client.ex: This file consists of user related information like creating user actors which includes the information of its user id which is a numeric string (“user 1”). Also, automation process is implemented here.

Performance Results:

1. Average Time to register user (microseconds)
2. Average time taken by the user to tweet (microseconds)
3. Average time taken to get tweets with a specific hashtag (microseconds)
4. Average time taken to get mentioned tweets (microseconds)
5. Average time taken to get tweets of user (microseconds)
6. Average time taken to subscribe to an user (micsroseconds)
7. Average time taken to get subscribers of user (microseconds)
8. Average time taken to get the users that an user has subscribed to (microseconds)
9. Average time taken to retweet (microseconds)

Number of Clients	Number of Tweets	**1.**	**2.**	**3.**	**4.**	**5.**	**6.**	**7.**	**8.**	**9.**
10	1	167.2	233.834	375.0	176.0	312.5	867.0	288.8	282.4	0
50	10	320.0	313.725	3041.096	31000	2285.714	40082.474	37872.340	37219.512	900.0
100	5	150.0	297.030	2065.359	47000	2259.740	90905.087	46413.334	44025.641	1719.101
500	2	780.0	808.383	2092.807	0	2551.111	86039.418	163733.644	170074.699	3895.833
1000										