Project 4 ReadMe

Isabel Laurenceau 7393-5064
Anshika Saxena 9530-5566

Explain what functionality is implemented

Six functionalities were requested for this project. The first functionality is to register the account and delete the account. The second is to send tweets, the third is to subscribe to other users. Re-tweeting, querying and delivering tweets are the fourth, fifth and sixth functionality respectively. All of these have been implemented and their mechanism as well as test are described in detail below. The bonus section is included as a zip folder in the main directory.

How to run the simulation

To run the simulation in the main directory (/LaurenceauSaxena) use:

"elixir pro_j4.ex [num_users] [num_msg]"

This simulation will create the [num_users] specified from the command line. Each user will randomly subscribe to other user's and post [num_msg] tweets. Note, because the simulation runs with predefined inputs it requires no interaction and has no outputs. Normal prompts such as asking for a password and confirming deletion of accounts are suppressed.

How to run the tests

Unit Tests were created for the functionalities described below. To run the test simply use:

"mix test --seed 0"

Note : Mix test runs on pro_j4.ex in the **/LaurenceauSaxena/lib** directory.

The pro_j4.ex file in the /LaurenceauSaxena/lib is identical to the in the main /LaurenceauSaxena directory (with just the input argument statements commented out at the end).The tests must be run in a specific order so that decencies can be established such as a retweet test is not run before any users are subscribed.We would also like to note that some tests do print to the command line such as "testing if tweet is empty it should not pass." These tests were chosen to print as they perform special functionality.

Functionality

1a) Register

To register the main program asks if the user would like to log in or register. When the user types "register" or "Register" they are asked for their username. It is first checked whether the username is already taken or not. If the username is free, they are asked to enter their new password twice. If the passwords match the dynamic supervisor and engine are started and the username and password are saved. The main menu options are then presented to the user: delete account, send tweet, subscribe to user, re-tweet, query, check feed (it's commented out for the simulation).

Tests for Register

We have four tests for register these are:

1. Register a new user with people(3) in the database
2. The registered users (4 in number) have corresponding Client processes (CSA)
3. Can't Register with an already registered username
4. The registered users have corresponding Engine processes

1b) Delete Account

If the user selected delete account from menu ask user if they are sure they want to delete the account. If no; show main options again but if yes; ask user for password. If the user passed password check: ask again if they want to delete. If they respond yes; delete account from the supervisor but if no; go back to main menu.

Tests for Delete

1. The engine has removed the process from its list
2. The Client process for the deleted user has been deleted

2) Send Tweet

If the user selected tweet, the menu asks user "What's on your mind?" It checks that the tweeted input is not empty, > 0 characters and < 280 characters, if not it prompts the user to edit their tweet. Once the tweet fits the character constraints it is sent to the CSA GenServer which calls the Tweet module. The Tweet module calls the Engine GenServer which adds it to the user's tweets list. The Engine then distributes the tweets to all of the user's followers.

Test For Send Tweet

We have five tests for send tweet these are:

1. test if user tweets it is in their tweets list
2. test if tweet is too long it should not pass

3. test if tweet is empty it should not pass
4. test if user tweets it is in their followers feed
5. test if user tweets it is not in a non-followers feed

3) Subscribe to user's tweets

If the user selected subscribe, the menu asks user “Who do you want to subscribe to?” It checks to make sure that user exists in our twitter universe (the Engine). Once the user to subscribe to is confirmed (it’s not the user itself and does exist in twitter), the current user’s subscription list is updated with the new user. The new user’s tweets are added to the current user’s feed. The current user is also added to the new subscriber’s followers list.

Test For Subscribe

1. The subscriber's name is in the follower's and vice versa

4) Re-Tweet

The process for re-tweeting is similar to Tweet. First the menu Retweet module is called. This module shows the user’s feed so they can select which tweet to retweet. The user selects the tweet by index and is asked if they would like to add anything. Unlike tweet they do not have to provide an input and can retweet the original tweet. Similar to tweet the input is checked to make sure it is < 280 characters. If so the new re-tweet becomes just like an original tweet and uses the same processes. It is sent to the CSA GenServer which calls the Tweet module. The Tweet module calls the Engine GenServer which adds it to the user’s tweets list. The Engine then distributes the tweets to all of the user’s followers.

Test For ReTweet

We have five tests for send re-tweet these are:

1. test if user retweets it is in their tweets list
2. test if user retweets it is in their followers feed
3. test if user retweets it is not in a non-followers feed

5) Querying

There are three different types of query implemented for this program. The first is to query for a general word or phrase that is not hashtagged. This query will only query the users feed (people they are subscribed to.) If the user selects this type of query, the Query module is called. From here a check is made to make sure the user does not query an empty string. Their feed is searched from this module for the query. The second type of query is for a hashtag. If the user selects this type of query the Query module is called. From here the Engine is used to search for all the possible tweets with this hashtag. Similarly for mentions. If the user queries for themselves or another user the Query module is called. From here the Engine is used to search for all the tweets from or to this user.

Test For Query

We have four tests for query

1. test if query for normal word: puppies
2. test if query is empty it should not pass
3. test if query for hashtag
4. test if query for person

6) If the user is connected, deliver the above types of tweets live

The last functionality we were asked to implement is to deliver tweets immediately. The Engine first takes all followers of the current user and checks to see if they are alive. If so a message is passed with the new tweet which is immediately displayed to the user’s command line. For functionality testing we implemented a show feed function that allows us to see the user’s feed any time it is called, not just when a message arrives.

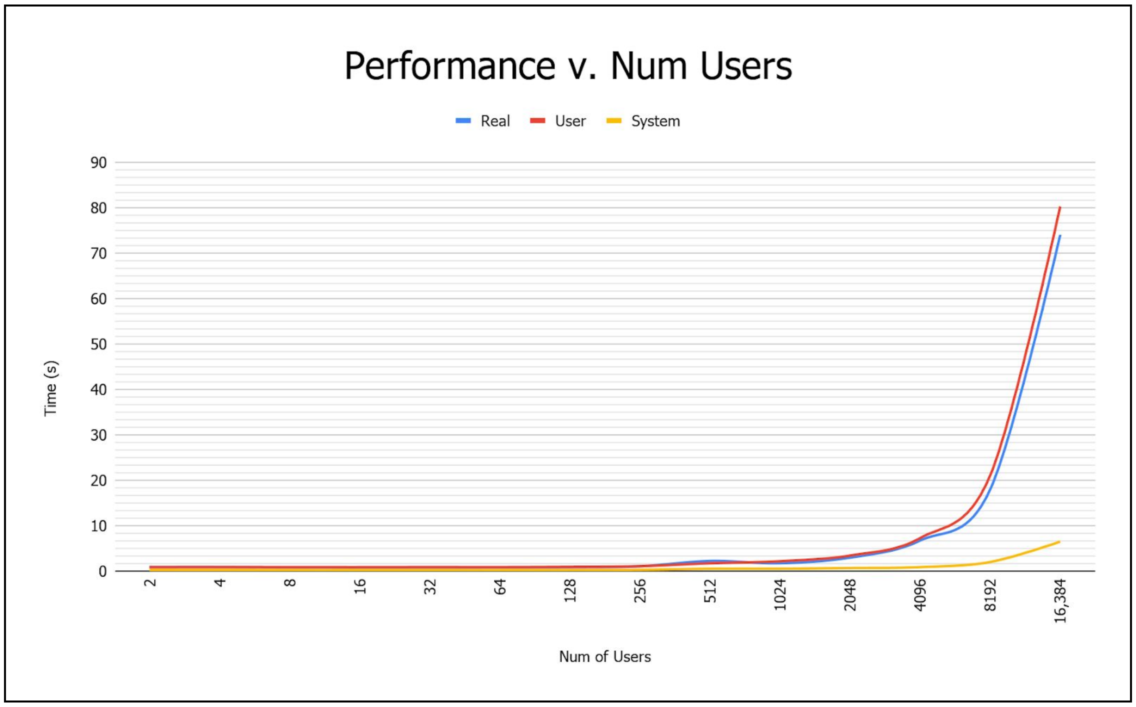
Test For Deliver

We have one test for deliver:

1. test "show testUser's feed" do

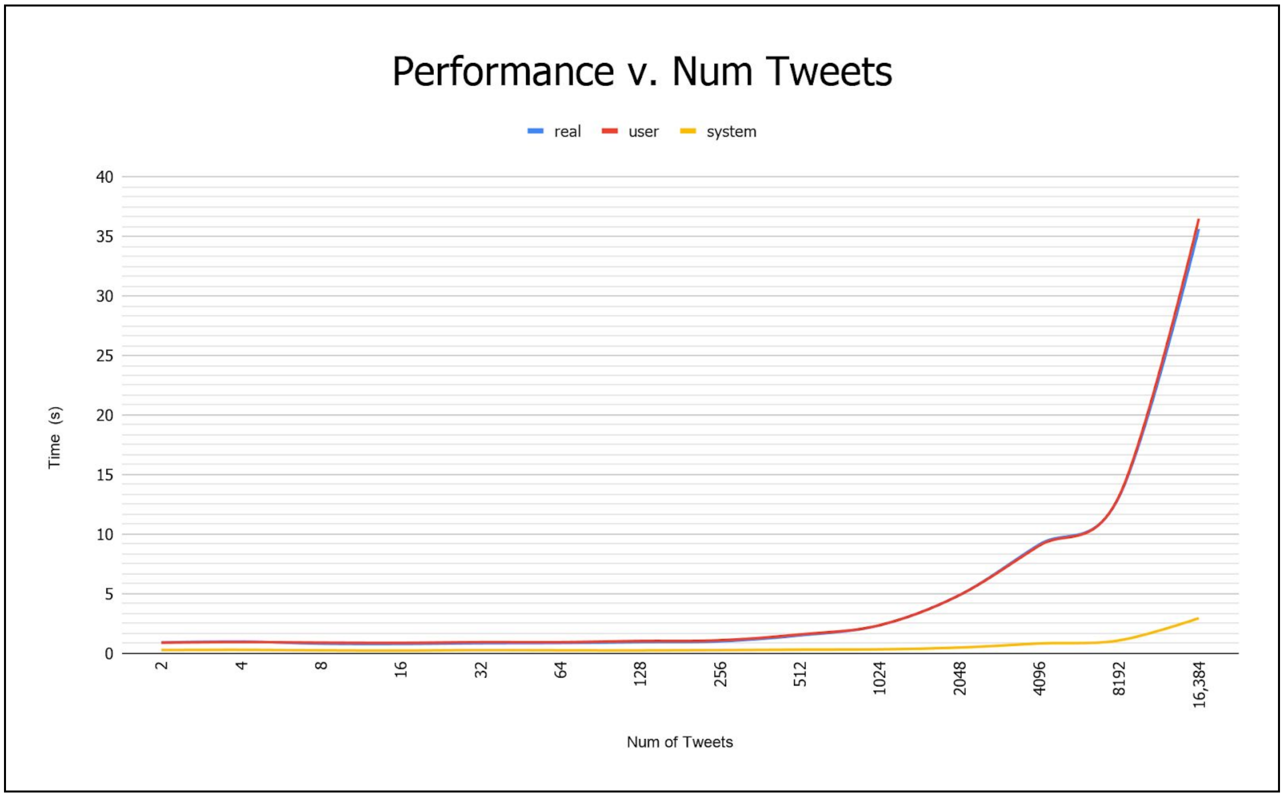
Performance Analysis

To perform our first analysis we kept the number of tweets being sent per user constant and exponentially increased the number of users. The graph showing the real, user and system time is below as well as a chart.



Num of Users	Real	User	System
2	0m0.911s	0m0.895s	0m0.281s
4	0m0.900s	0m0.911s	0m0.277s
8	0m0.821s	0m0.897s	0m0.246s
16	0m0.793s	0m0.886s	0m0.244s
32	0m0.786s	0m0.909s	0m0.245s
64	0m0.807s	0m0.899s	0m0.243s
128	0m0.850s	0m0.968s	0m0.260s
256	0m1.067s	0m1.094s	0m0.308s
512	0m2.252s	0m1.760s	0m0.536s
1024	0m1.778s	0m2.196s	0m0.555s
2048	0m2.988s	0m3.443s	0m0.707s
4096	0m6.747s	0m7.286s	0m0.895s
8192	0m17.982s	0m21.073s	0m2.022s
16,384	1m14.099s	1m20.336s	0m6.513s

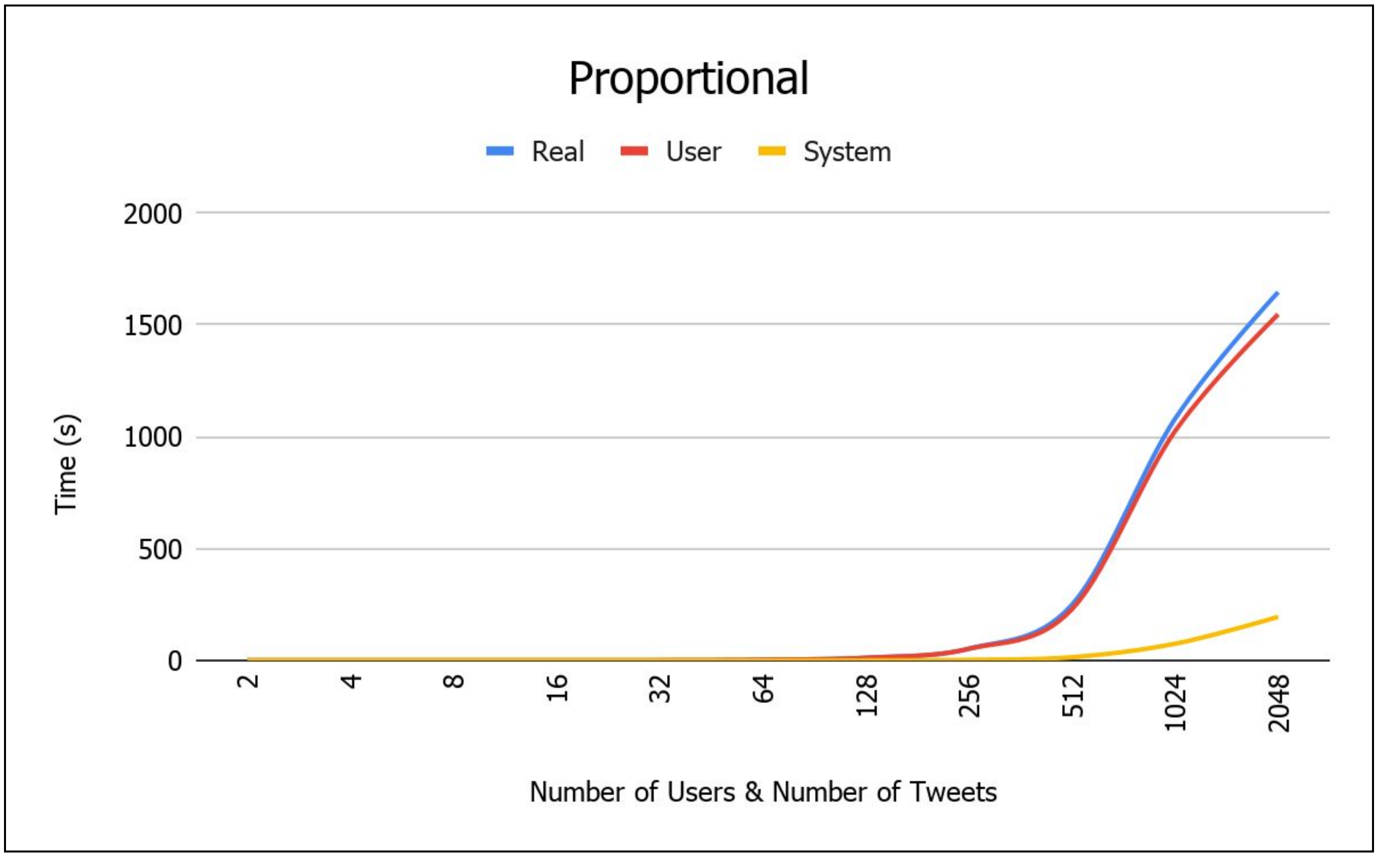
To perform our second analysis we kept the number of users sending tweets constant and exponentially increased the number of tweets they sent. The graph showing the real(blue), user (red) and system time (yellow) is below as well as a chart.



Num of Tweets	Real	User	System
2	0m0.911s	0m0.895s	0m0.281s
4	0m0.987s	0m0.947s	0m0.294s
8	0m0.811s	0m0.902s	0m0.252s
16	0m0.790s	0m0.883s	0m0.233s
32	0m0.849s	0m0.941s	0m0.270s
64	0m0.871s	0m0.940s	0m0.252s
128	0m0.934s	0m1.038s	0m0.245s
256	0m1.004s	0m1.095s	0m0.273s
512	0m1.502s	0m1.585s	0m0.308s
1024	0m2.355s	0m2.360s	0m0.338s
2048	0m4.885s	0m4.881s	0m0.484s
4096	0m9.134s	0m9.035s	0m0.823s
8192	0m13.111s	0m13.176s	0m1.082s

16,384	0m35.628s	0m36.479s	0m2.947s

For our last tests we did a proportionality test for number of users and number of tweets. This means that if there are two users, they each send two tweets; four users, they each send four tweets; eight users, they each send eight tweets and so on.



Proportional	Real	User	System
2	0m0.911s	0m0.895s	0m0.281s
4	0m1.081s	0m0.987s	0m0.346s
8	0m0.977s	0m0.924s	0m0.281s
16	0m0.939s	0m1.000s	0m0.279s
32	0m1.288s	0m1.439s	0m0.317s
64	0m3.380s	0m3.494s	0m0.461s
128	0m12.387s	0m11.986s	0m0.782s
256	0m53.153s	0m52.084s	0m3.055s
512	4m9.463s	3m49.585s	0m14.751s
1024	18m1.001s	17m5.601s	1m14.938s
2048	27m26.786s	25m47.461s	3m14.977s