README

Project 4.1 & 4.2: Twitter Engine

Project 4.2:

Team:

Dipen Jain (UFID: 15521903)

Anushka Sudhir Linge (UFID: 77530821)

The goal of part 2 of project 4 is to use the Phoenix framework of Elixir to build a web interface and to use web sockets to enable communication between the client processes. We implemented the simulation for at least a hundred users. The simulation can be visualised through the web interface which was built using html, css and javascript.

Procedure to run the project:

Go to path "Linge_Jain\twitter"

Run the command "mix phx.server"

Navigate to url localhost:4000 in the browser

Youtube URL:

The video has been uploaded to Youtube and can be found here:

https://www.youtube.com/watch?v=CwZ4lCqm_jY&feature=youtu.be

Implementation:

The main component of this application is the channel. We created a channel using the "mix phx.gen.channel twitter" command. This channel is used for communication among the different users, i.e. to subscribe to other users and to send tweets to one another. Each user is uniquely represented by the channel when sending tweets.

Functionalities Implemented:

1. A user can register, login and logout.
2. In the simulation for 100 users, every user is assigned followers randomly. Further, every user sends out two tweets which are received by his respective followers.
3. In order to subscribe to a user, the follower has to click on the “Follow” button after entering the username of the user he wants to follow, in the textbox provided on screen. For example, if user2 wants to follow user1, user2 enters user1 in the textbox and clicks on the “Follow’ button.
4. A user tweets by entering his tweet in the textbox and clicking the Tweet button. This tweet is then received by his followers only if they are active i.e. they are connected. The offline/inactive followers receive the tweet only after they login.
5. A tweet can contain hashtags as well as user mentions. Hashtags are of the form #lifeisgood, #computerscience i.e. strings that start with a #. User mentions are simply the usernames of other users. For example, user1 could tweet and mention user2 in his tweet like “this is for @user2”. The user mentions begin with a @ symbol.
6. A user can navigate to the “Query” page and query tweets to find the tweets that contain the desired hashtag or user mention. He has to enter this desired hashtag or user mention in the search box and click the button. This will return all the tweets that contain the search parameter.
7. When followers receive a tweet from their subscribers, they have the option to retweet this tweet by using the “Retweet” button. This button appears next to every tweet that they receive.
8. A userwill receive the tweet irrespective of whether he is a follower of the user (who tweeted) or not, if he is mentioned in the tweet.
9. The Twitter Server is the component that stores all the information related to the clients such as the clients’ login credentials, the tweets they send, the subscriber relationships etc.

Client simulation distribution:

In the simulation that we run for 100 users, every user follows 5 other users. Every user then sends out two tweets.

Project 4.1

The goal of this project is to develop a Twitter-like engine and provide the following functionalities to the clients/users:

1. Allowing them to create an account on the Twitter server
2. Allowing them to delete their accounts from the server
3. Users can send short messages called tweets. These tweets are sent to other users through the server.
4. A user can ‘subscribe’ to another user. Thus, when a particular user tweets, his subscribers/followers receive the tweet. For example, says there are two users A and B. If B is a subscriber of A, B receives all tweets that A sends out.
5. There is another functionality called retweeting. This allows a user to forward the tweets that he received (from another user) to his followers. For example, say there are three users, A, B and C. B is a subscriber of A and C is a subscriber of B. When A sends a tweet, B receives it. B could now retweet the same, so that his subscriber C receives it.
6. The tweets that are sent could have hashtags (e.g. #thisisahashtag) and user mentions too (@userJake).
7. A user can query the tweets that he receives, to find the desired one. He can query tweets using his username or specific hashtags as the query parameter.
8. A user receives tweets only if he has logged in, i.e. he is connected to the server.

We have developed a server module called TwitterServer which is of type GenServer. The server needs to store huge amounts of information such as all the user credentials required for login, the relationships among the different users, such as who is following whom, users that are active at any given time, a record of the sent and received tweets etc. For this purpose, we have used ets tables.

Procedure to run the file:

Go to path “Linge_Jain\project4”

Run command “mix run project4.exs numUsers numTweets”

What is Working

Depending on the numUsers parameter, the number of client processes are created. Depending on the numTweets parameter, the number of tweets are sent by every user.

1. Registering and deleting user account

A user first registers with the Twitter server by creating a username and password. There is a validation check performed on the server to ensure that no user with this username exists already. This check is done to prevent duplicate usernames. If the username does not exist on the server, then a user account is created with the username and password.

A user is also provided with the facility to delete his account from the server. If he is a valid user i.e. one who has previously registered with the server, his account is simple deleted from the server. This deleted user would no longer be able to log in unless he creates a new user account.

2. Logging in and Logging out

A user logs in to the server by passing his username and password to the server. A user is allowed to log in only if he has previously registered with the server. If his user credentials exist on the server and his password matches with the one stored on the server, he is logged in and his status now changes to an active user.

When a user wishes to log out, he provides his username to the server and is logged out. His connection status now changes to inactive.

3. Subscribing to other users

A user can subscribe to any other user. He sends a request to the server with the name of the user he wants to subscribe to. The server now stores this information to correctly disseminate tweets.

For example, if a number of users B, C, D, E have subscribed to user A, when A tweets, all active users among these should receive the tweets.

4. Tweeting

When a user tweets, a unique tweetid is generated for his tweet. The tweet and the tweetid are stored on the server. A partition among this user's followers is made, on the basis of their current connection status. This tweet is stored to later disseminate it to the user's followers. The tweet is also parsed to check if it contains any user mentions or any hashtags. The server makes a note of the tweets and the hashtags that it found in the tweets. It also records the user mentions and the tweets in which it found these user mentions. This information is later used to provide notifications to a user.

5. Retweeting

The same process as in the case of tweeting is followed on the server. Retweeting is essentially similar to tweeting hence it proceeds in a similar manner.

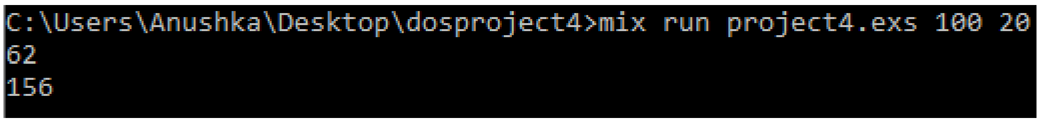
6. Querying hashtags and user mentions

A tweet can be queried to check if it contains any specific hashtags or user mentions. The hashtag is a short string starting with the '#' character. Similarly, a user mention is a username which is preceded by the '@' symbol in a tweet.

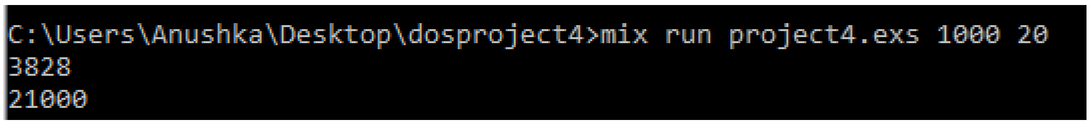
7. Delivery of tweet to active and inactive users

A separation is made among the followers of a user depending on their connection status i.e. whether they are currently logged in to the TwitterServer and are hence active or not. When a user tweets a message, his active followers are immediately delivered the tweet. The followers who are currently inactive are delivered the tweet only when their connection status becomes active.

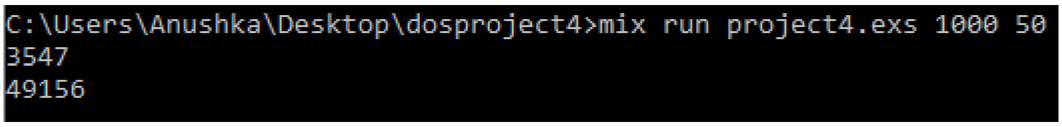
Program output:



```
C:\Users\Anushka\Desktop\dosproject4>mix run project4.exs 100 20
62
156
```



```
C:\Users\Anushka\Desktop\dosproject4>mix run project4.exs 1000 20
3828
21000
```



```
C:\Users\Anushka\Desktop\dosproject4>mix run project4.exs 1000 50
3547
49156
```

Maximum number of users tested was 1000 and maximum number of tweets per user was 50.

The output displayed is the time required in milliseconds to create numUsers/2 followers and the time required to send the tweets.

Functionalities implemented:

Registration:

A user registers on the server for the first time by providing a username and password. Once a user account is created, he uses the same credentials to login.

Login:

A user can login to the server using his user credentials. His connection status changes to active after logging in.

Logout:

A user can logout whenever he wants to. His connection status changes from active to inactive on the Twitter server.

Delete account:

A user an delete his existing account from the server.

Subscribing to a user:

A user can subscribe to the tweets of another user. He will now receive the tweets whenever they are tweeted by the followed user.

Sending tweets:

A user can send out a short message called a tweet. These tweets can contain hashtags or user mentions. The followers of a user receive these tweets. The server also checks the tweets for hashtags and user mentions. It sends the tweet to the mentioned user irrespective of whether he is a follower or not.

Retweet:

It is possible for a user to retweet a tweet that he received from another user. This retweet is sent out as if originating from the second user himself.

Querying using hashtag:

A user can query a tweet to check it for hashtags. He provides the query parameter as the hashtag he wants to search for.

Querying using user mentions:

A user can check the tweets for user mentions by passing the user mention as a query parameter

Live delivery of tweets:

The server delivers the tweets to the followers when they are active i.e. when they are connected to the server. If not currently connected, the server stores the tweets to deliver later to the followers, when they have active status.

Test Cases:

1. **Registering a new user**

Input	**Description**	**Output**
Registration for three users namely, user1, user2 and user3	This testcase tests whether user accounts are successfully created for these three users	**true**

2. **Deleting a user account**

Input	**Description**	**Output**
Deleting user account for user user7	This testcase tests whether a user account is successfully deleted when requested	**true**

3. **Login of a user**

Input	**Description**	**Output**
Log in of user5	This testcase verifies that user login happens correctly i.e. the connection status of a user changes to ‘active’ after he logs in	**true**

4. **Incorrect login of a user**

Input	**Description**	**Output**
Incorrect login of user5	This testcase tests if a user who has provided incorrect login credentials is allowed to login or not.	**true**

5. **Logout of a user**

Input	**Description**	**Output**
Logout of user8	This testcase checks whether a user is logged out correctly. His connection status should change to inactive on logging out.	**true**

6. **Subscribing to a user**

Input	**Description**	**Output**
user9 subscribing to user10	This testcase whether this subscriber relationship between user9 and user10 is successfully generated and stored on the server	**true**

7. **Sending tweet**

Input	**Description**	**Output**
user11 sends three tweets	This testcase checks whether the server records these tweets and then send to the followers of user11	**true**

8. **Querying hashtags**

Input	**Description**	**Output**
user12 sends three tweets each containing a hashtag	This testcase checks whether the hashtags are recognised and extracted from the tweets so that they can be queried	**true**

9. User mentions query

Input	Description	Output
user13 sends two tweets both containing user mention of user14	This testcase checks whether these tweets are successfully delivered to user14 since his username was mentioned in the tweets	**true**

10. Delivery of tweet to live followers

Input	Description	Output
user16 is an active/live follower of user15. User15 tweets three messages	This testcase checks whether tweets are successfully delivered to a live follower of a user	**true**

11. Delivery of tweet to offline followers

Input	Description	Output
user18 follows user17. user17 tweets a message. user18 is currently offline	This testcase checks to ensure that a tweet is delivered to the followers only when he is logged in i.e. only when his connection status is active	**true**

12.Retweeting

Input	**Description**	**Output**
There are three users, user19, user20 and user21. user20 is a subscriber of user19 and user21 is a subscriber of user20. user19 sends two tweets.	This testcase checks whether retweeting is performed correctly. user19 sends two tweets which are received by user20. user20 now randomly selects one of these tweets and retweets it. The testcase verifies whether the follower of user20, user21 receives the retweeted message.	**true**