Team Members:
Aditi Page (UFID: 0427-9968)
Kartik Rode (UFID: 8971-1791)

Instructions to run the file:

- 1. Unzip the file
- 2. Run the project using:
  - **dotnet fsi --langversion:preview simulator.fsx no_of_users**
- if your system is running a language version lower than 5.0

What is working:

- Users are getting registered initially.
- Users log in where their status is changed to 'Online'.
- Subscribe: Followers are simulated for each of the users using Zipf distribution.
- Send Tweets: The online users tweet which is then sent to their followers.
- Re-tweets: User is able to re-tweet the tweets that are originally generated by its followees (users whom the current user follows).
- Live Tweets: When a user tweets, this tweet appears directly on the Home timeline of its online followers.
- Querying of Tweets: This is implemented as the search functionality where the user can search for hashtags, mentions, or the keywords in the tweets.
- We have used Datatables as our in-memory database for storing User Info, tweets info and retweet info

Largest Network Tested:
Number of Users: 2500 (Approximately more than 14000 tweets)

Performance Metrics:

- Time taken to deliver 14000 tweets: 253 seconds
- Time taken for performing Hashtag query on all (14000 tweets) the tweets: 0.1001 seconds
- Time taken for performing Mention query on all (14000 tweets) the tweets: 0.100146 seconds
- Time taken for performing Keyword query on all (14000 tweets) the tweets: 0.0405 seconds
- We have added Sleep mechanism to display the results which means that the total time is less than the provided time which includes the Sleep of 100 milliseconds.

Client Distribution
We have used Zipf distribution to assign followers to the users and we have set a threshold value (20% of total actors) for number of followers. If, using the Zipf

distribution, the number of followers of a user come out to be more than the threshold value, then that user is deemed as a celebrity.
The clients are distributed in two categories depending on the number of followers:

1. General User
2. Celebrity

Functionalities Implemented

1. **Registration:** All the new users are able to register on the server and these entries are added to the user datatable.

2. **Login :** User can login to the system by providing the credentials. Server than authenticates the user can logs the user to the system.

3. **Logout:** Logout functionality to logout the user from the system**.**

4. **Simulating Followers (Subscribers) :** Each user is assigned followers randomly and the number of followers for each user is determined using the using the Zipf distribution. A map is created where the key is username of the user and the lost of followers is stored corresponding to this key.

5. **Simulating Followees**: Followees are the users whom the current user follows. A map is created, similar to the followers map, to store the list of users the current user is following. The number of other users a particular user follows is determined by the Zipf distribution. The key in the following map is also username corresponding to which list of followees is stored.

6. **Home Timeline**: It is a compilation of all the tweets and retweets by the followees of the current user. Tweets will be displayed in the format “First name, Last name, Tweet” and retweets will be displayed in the format “First name of original user, Last name of original user, First name of user who retweeted, Last name of user who retweeted, Tweet”.

7. **Add New Follower**: New follower is added to the list of followers.

8. **New Tweet:** New tweets are generated for user with 4 categories- tweet without hashtag or mentions, tweet with hashtag, tweet with mention, tweet with both hashtag and mention. For this, we have chosen the case randomly, and generated the tweet accordingly. Once the tweet is generated, the tweet is added to the tweet datatable. For the generation of sentences we have used a word list from which we are randomly selecting a word to form a sentence of 6 words.

9. **New Retweet:** Retweets are the tweets fetched from the Home of the user. The first name and last name of the user who originally tweeted the tweet is fetched, and the retweet saved in a 5 tuple format- First name of original user, Last name of original user, First name of user who retweeted, Last name of user who retweeted, Tweet. For this method, we need a tweetID and the user who is performing the retweet.

10. **Search for Hashtag:** All hashtags are stored in a global map where the key is the particular hashtag and the value is the set of tweetIDs containing the hashtag. When a certain hashtag is searched, the list corresponding to the key (here: hashtag) is fetched. The tweets containing the hashtag are displayed in the format- First name, Last name, Tweet.

11. **Search for Mention:** All mentions are also stored in a global map where the key is username and the value is the set of all tweets where the username was mentioned. When a user searches its mentions, the details of these tweets are fetched and displayed in the format: First name, Last name, Tweet.

12. **Search for a Keyword:** When a keyword is searched, all the tweets and retweets containing the keywords are searched and displayed.

13. **Live Delivery of Tweets:** If a user tweets/retweets, then it will appear directly on the Home timeline of its followers. Server checks if the followers are online and the tweets/retweets appear in the Home timeline of only the online followers.

14. **Delivery of tweets to offline user:** When a user tweets/retweets, the server checks which among all of its followers are offline. The new tweets/retweets will be added to the pending map of the offline users. When these offline users log back in, they check their pending map and their Home timeline is updated using the pending map. The pending map is then emptied.

Overall Working

In the working of our project, we have 1 user corresponding to every actor. Initially, all the users are registered which means that they are added to the user datatable. Using Zipf distribution, followers for each of the users are simulated. A threshold value is set and if the number of followers exceeds this threshold value, then the user is deemed a ‘Celebrity’. Next, all users except 3 users are logged in where their status flag changes to ‘True’. All the logged in users generate tweets. Each user generates 5 tweets and each celebrity user generates 10 tweets, out of which 5 are re-tweets. Each tweet is given a unique tweetID and these entries are stored in the Tweets datatable.

For all the users who are online, home timeline Map will be updated directly without querying. In this way, we are making sure that online users do not have to query the data table for retreiving new tweets. Whenever we want to print the home timeline of a user, we will show it in a reverse order(latest 5 tweets maximum). If the user is offline, then we are not directly updating the Home Map, instead we are updating a pending Map first which will be first checked by the user whenever he/she logs into the system. If there is some data, the user will update its home Map with it and empty the pending Map.

After this fan out process, we have demonstrated the use of Add New follower method where we are adding a new user as a follower for an existing user.

We have chosen one of the original offline user to come online into the system and updated its home map and printed its original pending map for verification. This is visible as “Original Pending map” and also the updated Home Map.
After this, we have demonstrated the retweet process and have displayed the result of the Home Map where we can see the new retweet as the first tweet. This is displayed as “Home Map after Retweeting”.

Home timeline can be displayed for each user. All tweets and retweets by the followees (other users the current user follows) are displayed on the Home timeline in the *f*ollowing format:

- *If it is an original tweet:*
  firstname lastname
  Tweet
- *If it is a re-tweet:*
  firstname lastname (of new user who retweeted)
  f*irstname lastname (of old user who originally tweeted)*
  Tweet

For Hashtag Searching, we have picked one of the words present in the list of the words and Performed a search query and displayed 5 results.

For Mentioned Searching, we have displayed the tweets where this particular user has been mentioned.

For keyword searching, we have chosen a word to be used by the method, and after performing the query, displayed 5 latest results from the users this particular user is following. This means that keyword search is performed on the tweets that the user has subscribed to.

In order to display the results on the terminal in a proper way, we have used Sleep method multiple times which is slowing down the execution. This will be rectified in Project 4.2.

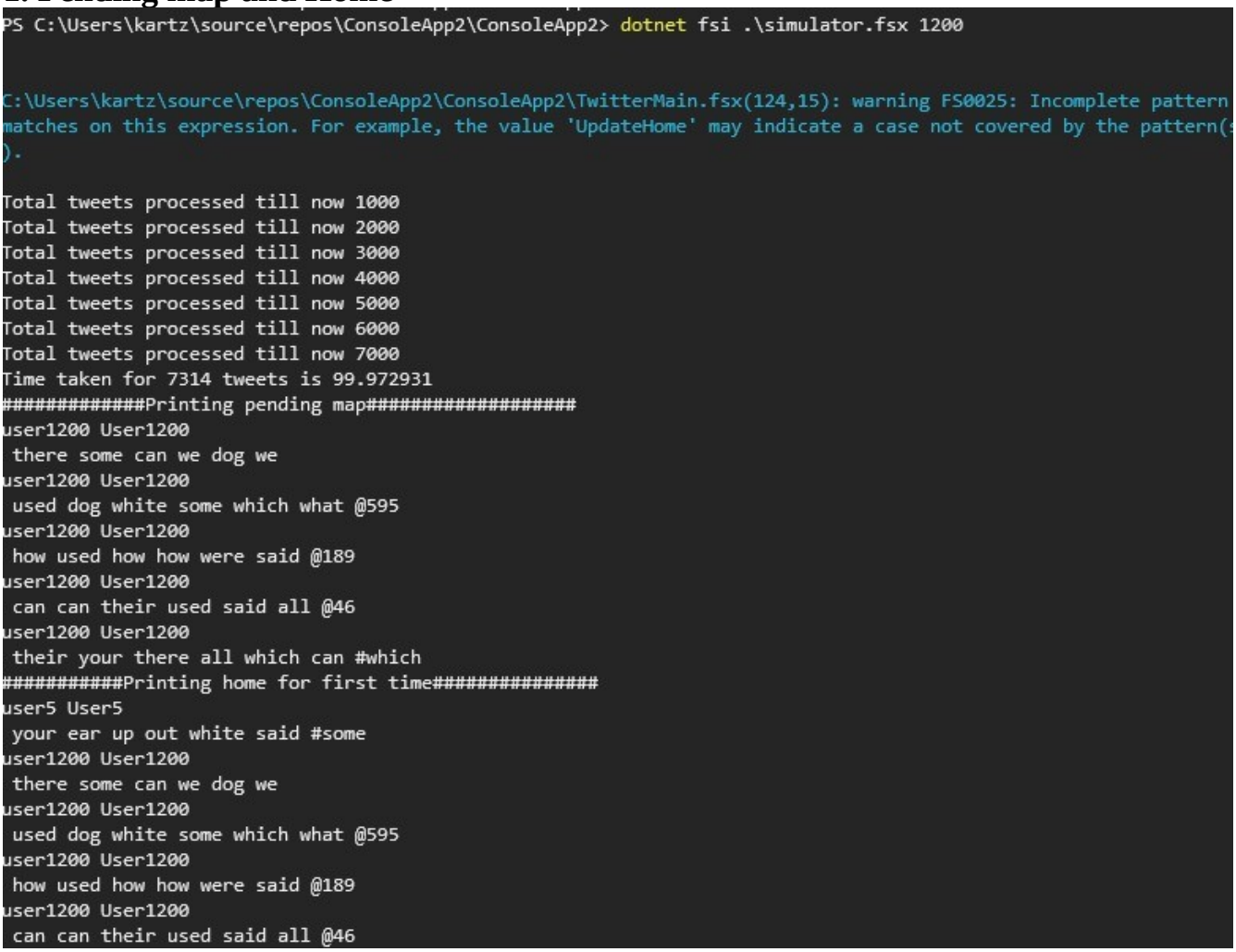
Test Cases Implemented:

1. Registering Users: All users must have status as ‘Offline’ – **Passed**
2. User Login: All users must have status as ‘Online’ – **Passed**
3. User Logout: All users must have status as ‘Offline’ – **Passed**
4. Home timeline must update when user logs in – **Passed**
5. Pending Map must update when user is ‘Offline’ but when one of its followees tweets/retweets – **Passed**
6. Home Map must not update when user is ‘Offline’ and when one of its followees tweets/retweets – **Passed**
7. When the user logs back in, the pending map should be emptied as Home is updated – **Passed**
8. When one of the followees tweets and the user is online, the tweet is not added to the Pending Map – **Passed**
9. New follower is added for the current user – **Passed**
10. All tweets and retweets containing the searched hashtag displayed – **Passed**
11. All tweets and retweets containing the user’s own mention displayed – **Passed**
12. All tweets and retweets containing the searched keyword displayed – **Passed**

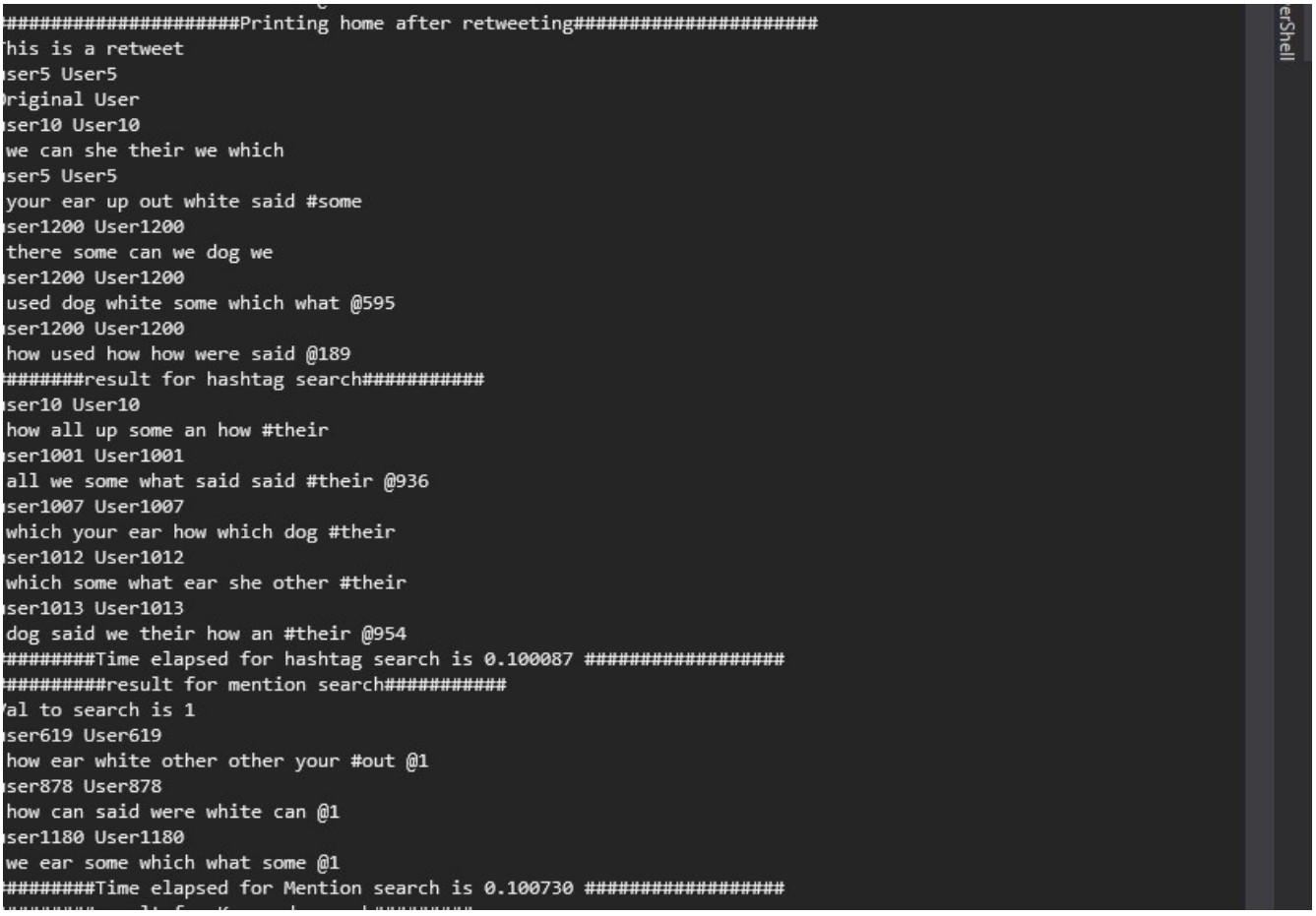
Screenshots

Note: All the times are in seconds.

1. Pending map and Home



2. Home displayed after Re-tweet and results of Hashtag search



3. Results of mention and keyword search

```
#########Time elapsed for hashtag search is 0.100087 ##################
#########result for mention search###########
Val to search is 1
User619 User619
 how ear white other other your #out @1
User878 User878
 how can said were white can @1
User1180 User1180
 we ear some which what some @1
########Time elapsed for Mention search is 0.100730 ##################
########result for Keyword search#########
User5 User5
 which all white some there all
User11 User11
 out an up ear there dog #white @564
User11 User11
 said white can your what all #other
User11 User11
 white all out their some all #can
User5 User5
 all how white said there said #we @462
Search completedS
########Time elapsed for Keyword search for white is 0.010770 ##################
PS C:\Users\kartz\source\repos\ConsoleApp2\ConsoleApp2>
```

4. Zipf Distribution code snippet

```
let array = Array.create nActors 0
let z = new Zipf(1.0,nClients-1)
z.Samples(array)

let words = ["what";"some";"we";"can";"out";"other";"were";"all";"there";"white";"up";"used";"your";"how";"said";"an";"ear";"she";"which";"dog";"their";"time"]
let rand = System.Random()
let mutable IdsList = []


let signinTesting(userID) =
```