Project 4b Implementation Report:

Note: *The project was implemented using a Client User Interface, Postgres database, Phoenix JSON APIs, Erlang message passing and WebSockets.*

Youtube Link(Demo): https://youtu.be/ZkuGb5b8hng

Project Members: 1. Shubham Shukla (4124-3903) 2. Srishti Mishra (8933-7966)

```
To start your Phoenix app:

  * Download the zip file twitter_clone.zip
  * Unzip the zip file by UI or go to the terminal and type

        `unzip twitter_clone.zip`

  * Since the project is run under umbrella, first enter the twitter_clone folder

        `cd twitter_clone`

  * Enter the folder apps
        `cd apps`

  * Enter the folder twitter server. This is where the required mix file is
        `cd twitter_server`

  * Install dependencies with
        `mix deps.get`

  * There is a UI and Postgres database in the project, so first create and migrate
your database with

        `mix ecto.create && mix ecto.migrate`

  * Install Node.js dependencies if required with

        `npm install`

  * Start Phoenix endpoint with `iex -S mix phoenix.server`

Now you can visit [`localhost:4000`](http://localhost:4000) from your browser.
```

Structure of the Project

  * There is a registration module built on JSON APIs, that lets the user create an identity and store in the DB. Only then he/she can use the application.

     url: `/register`
     files: `controllers/UserController.ex, model/User.ex`

  * Upon registration, the user can then sign in to the application with the email that he/she has provided. Or can also come back to the app and sign in any time.

     url: `/signin`
    files: `controllers/UserController.ex`

  * Upon Signing in, the application creates a session for the user and which is then authenticated with the session. The session needs to be authenticated for every request and the token is a sha256 string which is stored on the server for 1 second.

    files: `controllers/auth_token.ex, controllers/auth.ex, channels/timeline_channel.ex`

  * The user can delete the session by signing out.

    url: `/signout`
    files: `controllers/UserController.ex`

  * The UI consists of a timeline where a User can tweet, retweet, search for hashtags, see his followers and also follow any active username in the server.

    url: `/timeline`
    file: `templates/timeline/index.html`

  * The Client interacts with the server using JSON APIs and all the data to and from the client and server is with JSON and Phoenix Elixir maps. Once the server receives the request, it interacts with the Internal GenServers implemented in the first part of this project through native Erlang message passing. And once a required operation is completed, the data is passed over the socket back to the client.

  * To help the instructor/reviewer easily view the operations in the real time, we have implemented enough code to display results and changes for all the necessary functions.

* We have used socket's message passing methods such as broadcast and push, to allow data transfer over the socket. The implementation is done in our module TimelineChannel.

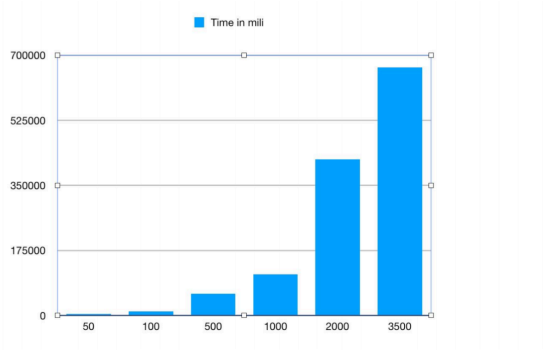
    file: `channel/timelinechannel.ex`

* All the client code is written in a javascript file, that is loaded with the base layout.

    file: `static/js/app.s`

* Since, we have implemented the project using a User Interface, our metrics for performance was hard to compute given the limitations that come up with using UI on browser. Our simulation was done for a maximum of 6 users on separate sessions in different browsers. Everything worked perfectly and there was no real-time lag. But here are some performance metrics that can are still valid on this project, given if we simulate the clients from GenServers using the zipf distribution from Project 4a.

Performance Measures (from Elixir Client-Server message passing)



Number of Users	Time (milli seconds)
50	3221
100	10032
500	58119
1000	110229
2000	420065
3500	666981
10000	1692121

* Future plan is design a better User Interface and publish the project on Heroku for a live server in production.

* Conclusion: All the project requirements were fulfilled to the best of extent, including the bonus. Thank you for the opportunity and please mail us if any questions or concerns regarding the project:

    Email: `shubh77@ufl.edu`, `srishtipmishra@gmail.com`

Thank you.