# Distributed Operating Systems Project – 4 (Part 2):

**Group Members:**

Raghuveer Sharma Saripalli - 5094-6752
Vangmayi Vydyula - 3549-3676

## Outline:

The main aim of this project is to implement a webSocket interface to the part 1 implementation of the project which was "the implementation of Twitter clone engine and an Actor - model based client tester/simulator".

## What is working?

- Basic Requirements - completed
- This project mainly consists of the following parts:

    - **Backend** – The functions for every activity that is supposed to take place when the user calls that function have been methodized in the backend. So, when the user tries to login or logout or get mentions or hashtags etc, he/she would press the button or enter the text depending on the activity and the button then acts according to the functions written in the server side to give the user the requested result. This was mainly covered in part 1 of the twitter project.

    - **WebSockets**:
        - These are connections that make it possible for us to establish a two-way interactive channel between the client and the server. So, when the server is running and a client logs into the server, we can see that a webSocket connection is established with the address followed by /websocket URL. We can observe this in the mainConnect module of the code.
        - Upon the establishment of the connection, handshake messaged are exchanged between the client and the server and the server receives the username from the client and this message will be stored by the server with reference to the id of the client.
        - Further, to push any updated to this current client, the corresponding actor exchanges the messages and updates through the webSocket address associated with this client.
        - On the other hand, if the client receives any messaged from the server through the webSocket, it is displayed on the live feed in the UI where the client can see it.

    - **REST APIs**:
        - A REST API (also known as RESTful API) is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services.
        - The REST implementation for this project has been done using the suave library of F#.
        - GET requests – get tweets, mentions and hashtags.

- POST requests to the server – tweet, register, logout and login requests from the user. (All of them are functions in the code)
- Upon receiving a request, the server performs the respective functions by assigning work to several actors as per the client's request.
- All the REST API calls can be found in the **routingFunctions** module of the code which starts at line 401.

## Pre-requisites:

- Visual Studio code
- .NET version 5.0.207
- NuGet Akka.NET v1.4.25

## Language used:

F#

## Steps to compile and run

1. Unzip the Project 4 – Part 2 folder to retrieve the project4.fsx and client.html files.
2. Open these files using the VSCode (Visual Studio Code) editor
3. Open terminal and execute the command "dotnet fsi project4.fsx on the server machine which starts the server.
4. Open the file client.html in any browser.
5. Perform any twitter operations on the User interface that would be displayed in the browser.
6. For more detailed explanation on how to run the project and how everything works, please find the following link for a video explaining the execution, implementation and working of the code and a tour on how the twitter functions work.

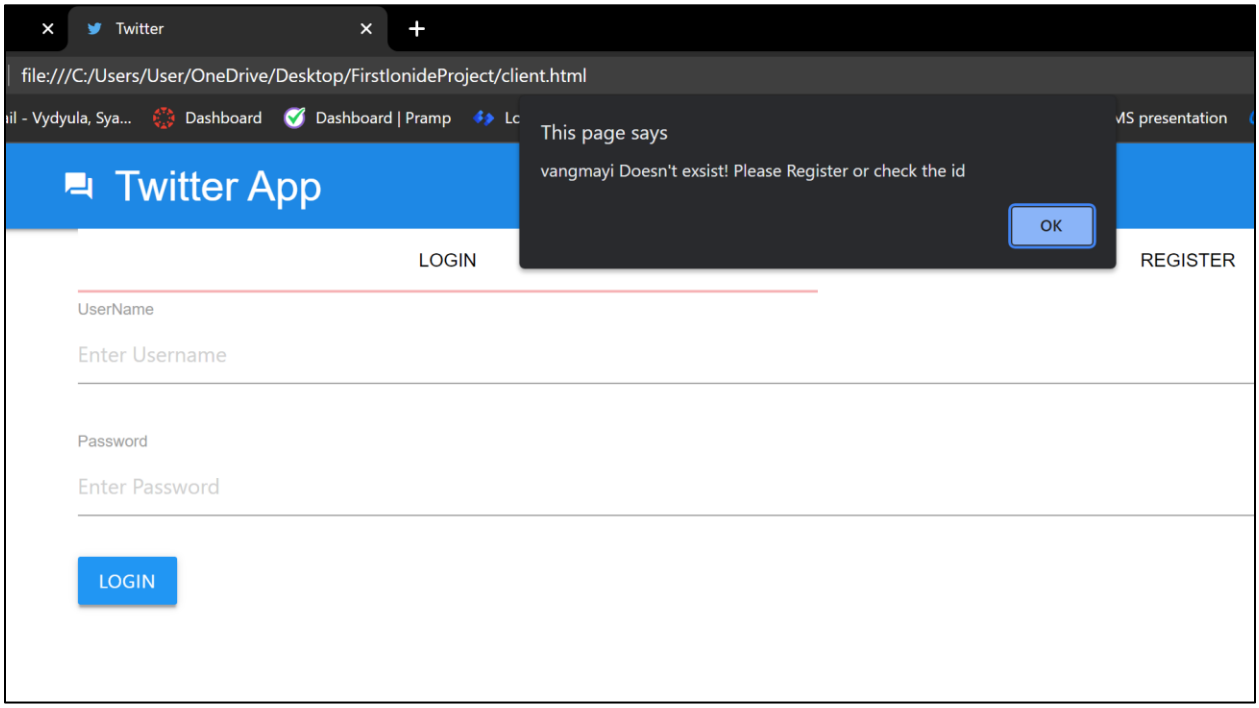
The demo video link - https://www.dropbox.com/sh/qnojv8zaxq0db4v/AACsmStPVrB0umgfcxCWBuU-a?dl=0&preview=Twitter.mp4

## Results –

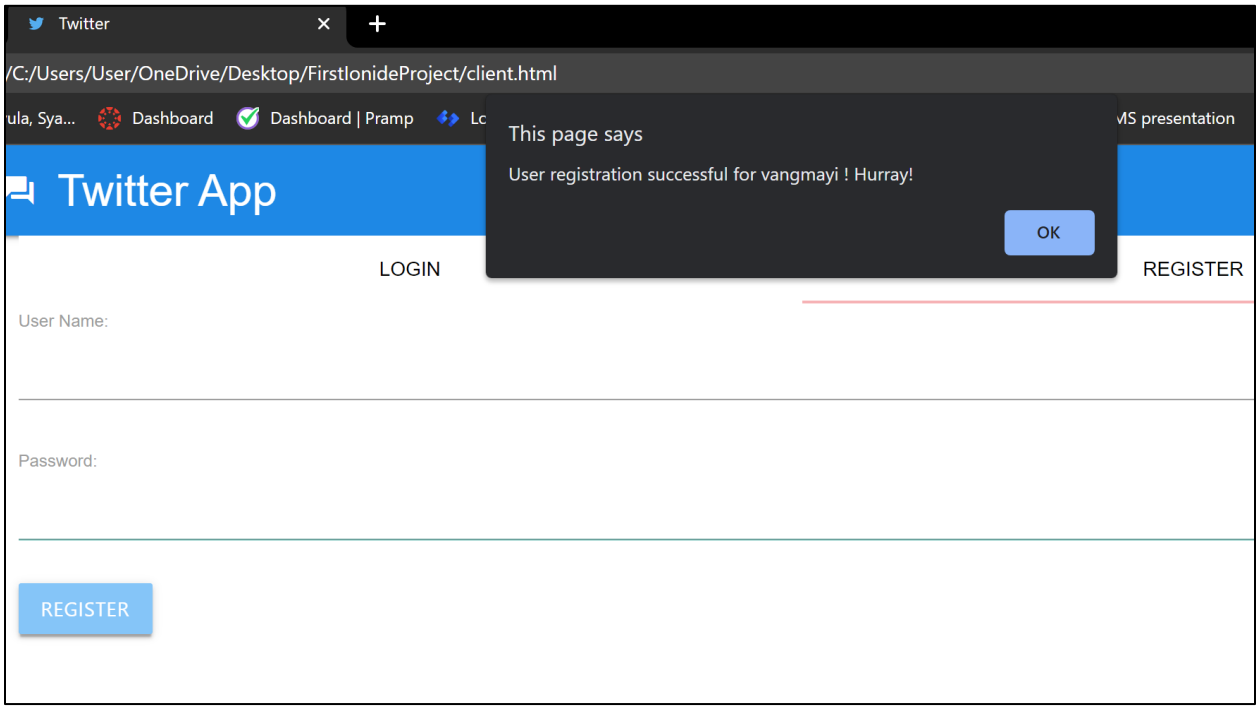
### Output examples –

### Command used – dotnet fsi project4.fsx

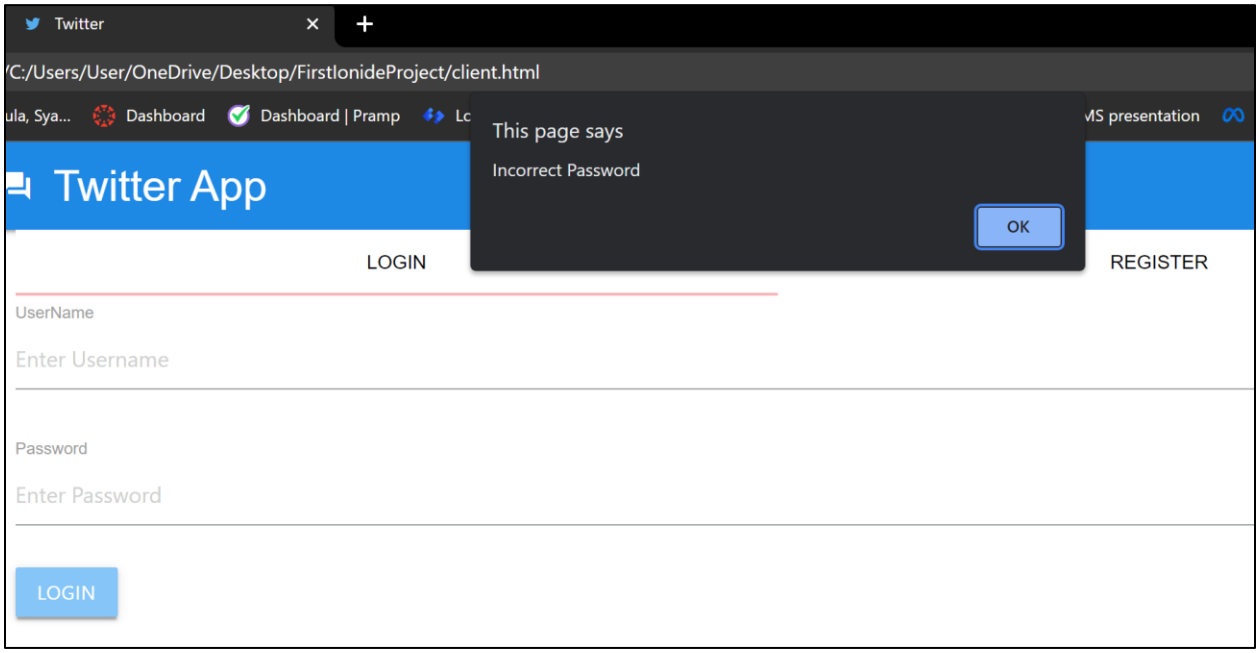
1. **Login when the user isn't registered**



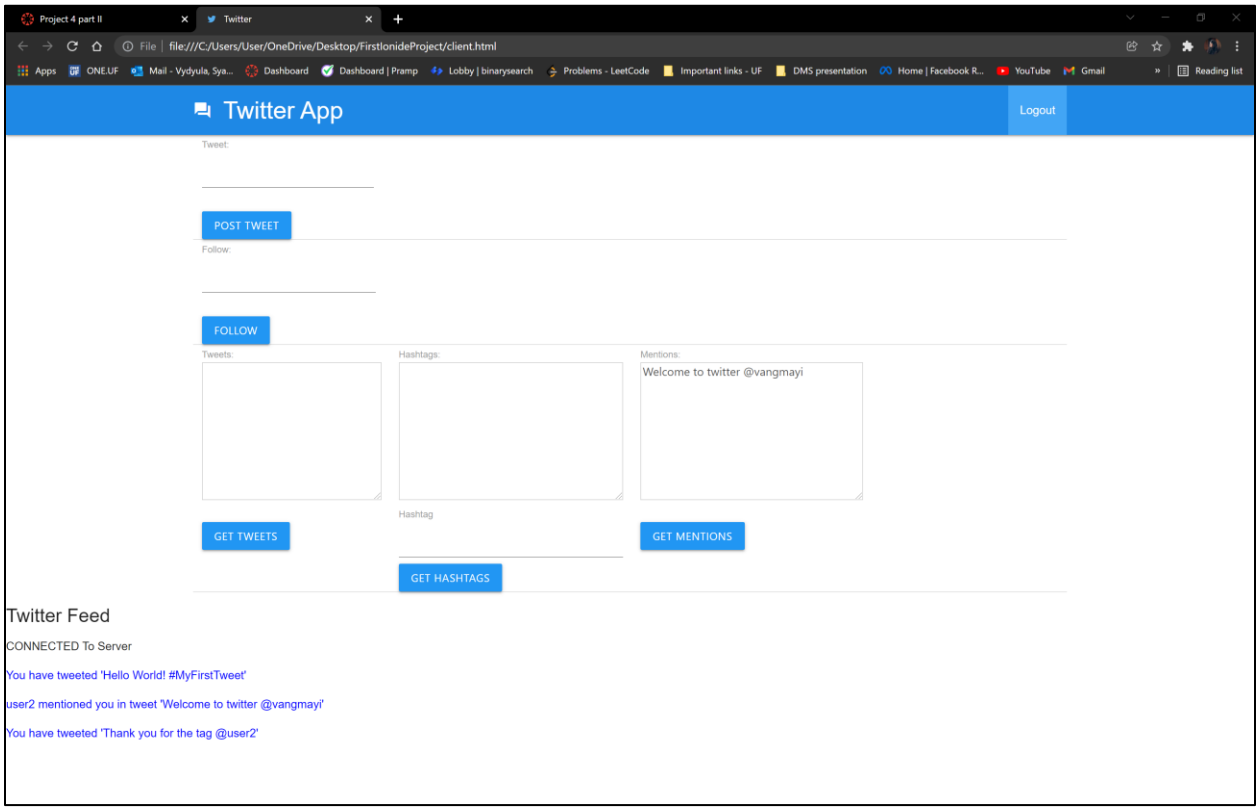

2. **Register user**



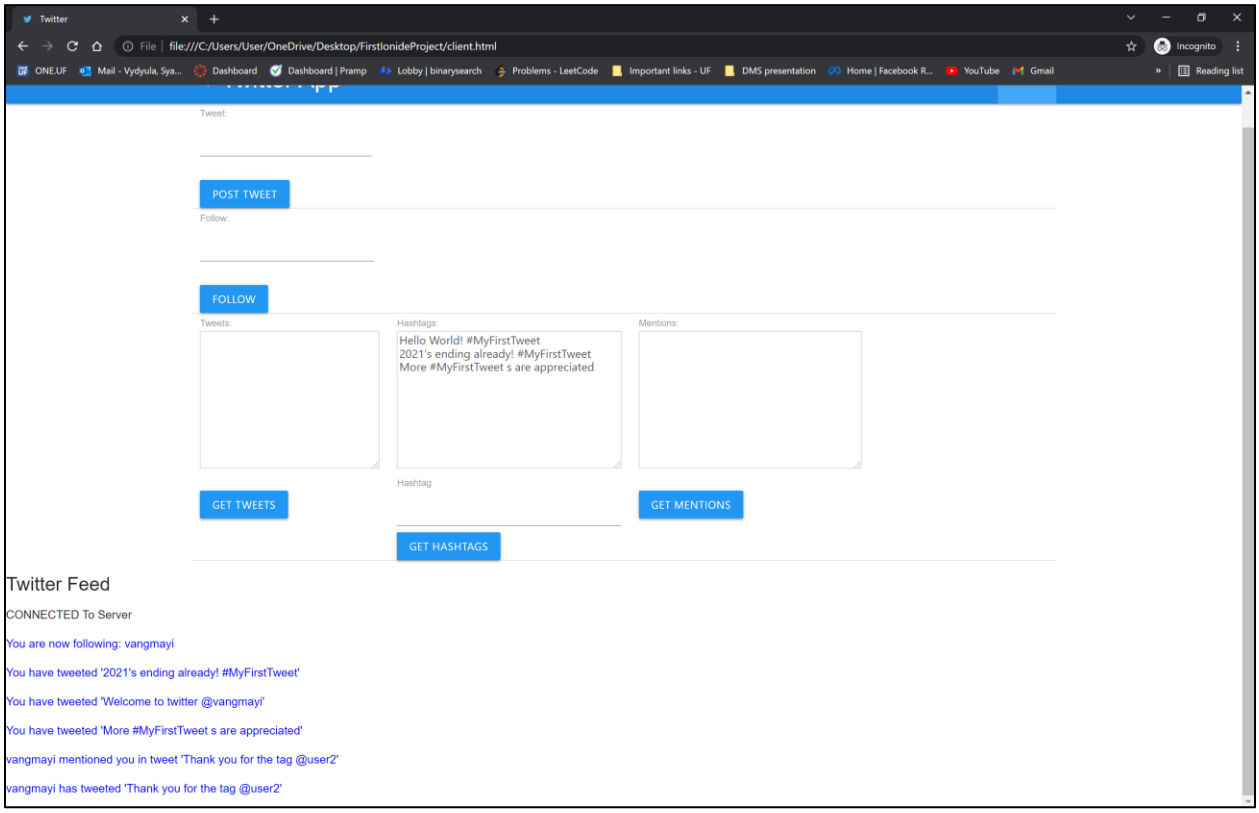
### 3. Incorrect password entry



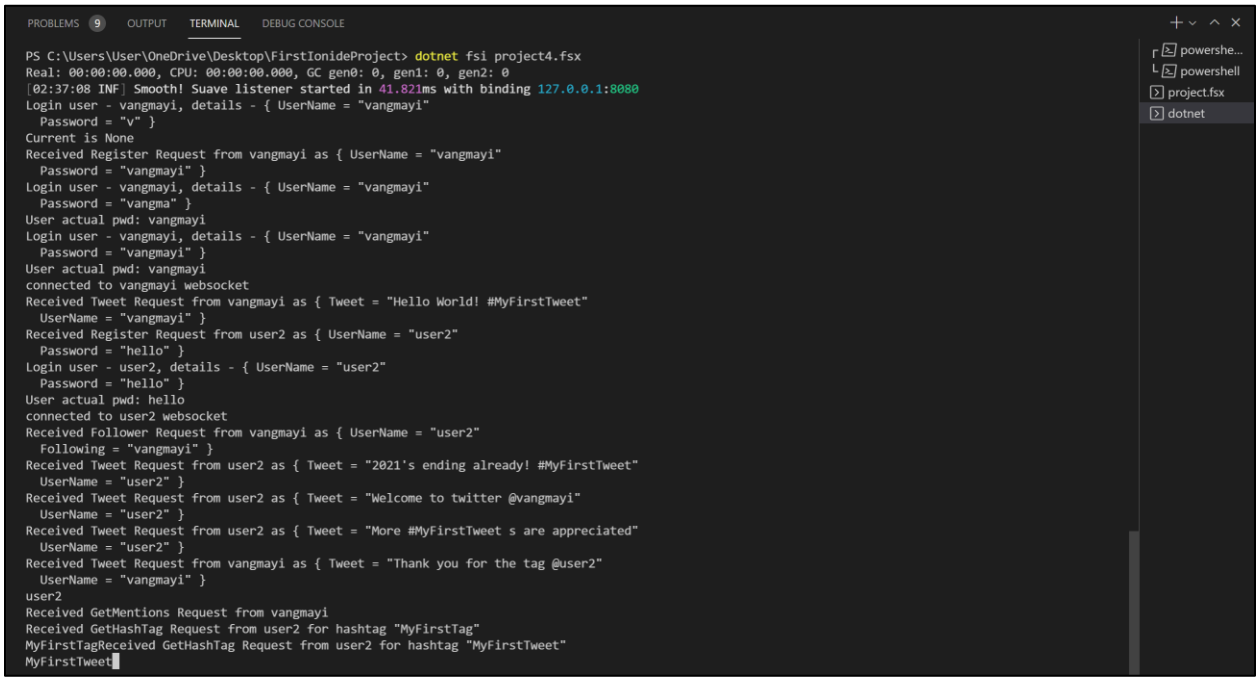

### 4. User1's home page



## 5. User2's homepage




## 6. Server side outputs



**System CPU Utilization – This machine has 4 cores (8 logical processors)**