COP5615 Project5: Implementation of WebSocket Interface for a Twitter-like engine

Documentation

Srinivas Koushik Kondubhatla (UFID: 69238911)
Dharani Kanchanapalli (UFID : 75351996)

Problem Statment

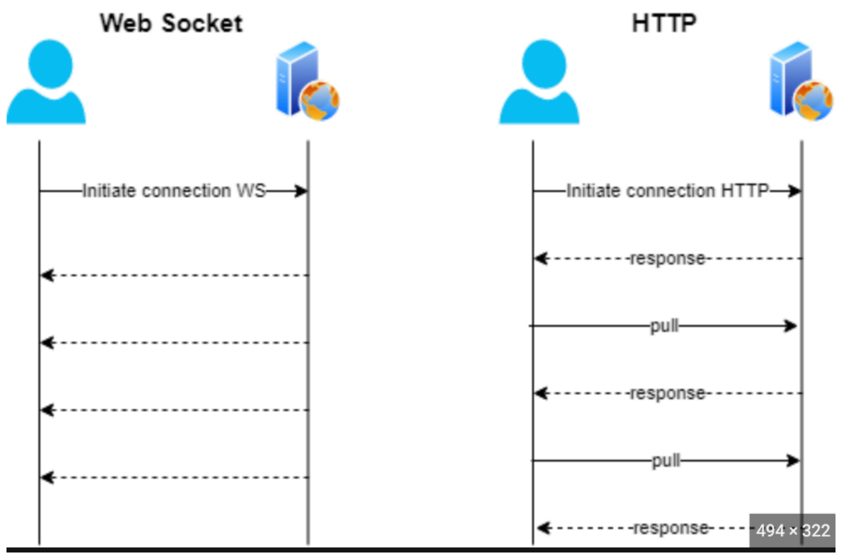
Implementation of WebSocket interface for Twitter API implemented using actor model in Erlang. The main functionalities of this engine will be

- To provide an interactive client interface
- Updating feed without client intervention.
- Client should be able to
  - Register
  - Send tweets
  - View Feed and Retweet
  - Subscribed
  - Query using hashtags, mentions & subscribed users.

Implementation

HTTP vs WebSockets

In HTTP long polling, the client has to continuously pull the updates from the server and update accordingly. If there's a continuous stream of data, the client will have to request more pull requests and wait till the server is free. It is ineffective.
In Web Sockets, a channel is created which enables the server to send a stream of data without client asking server.



Cowboy

Cowboy is a small, fast, HTTP server for Erlang/OTP. Cowboy provides a complete web stack which is supported by HTTP/1.1, HTTP/2, Websocket, REST

Let us discuss on how to create and run a cowboy application in a linux/Mac device.

- Create a directory with all lower cases
- Install erlang.mk: **wget https://erlang.mk/erlang.mk**
- Bootstrap the application : **make -f erlang.mk bootstrap bootstrap-rel**
- Run the application : **make run**
- Now, add cowboy to the existing dependencies(in Makefile)

```
PROJECT = hello_erlang

DEPS = cowboy
dep_cowboy_commit = 2.6.3

DEP_PLUGINS = cowboy

include erlang.mk
```

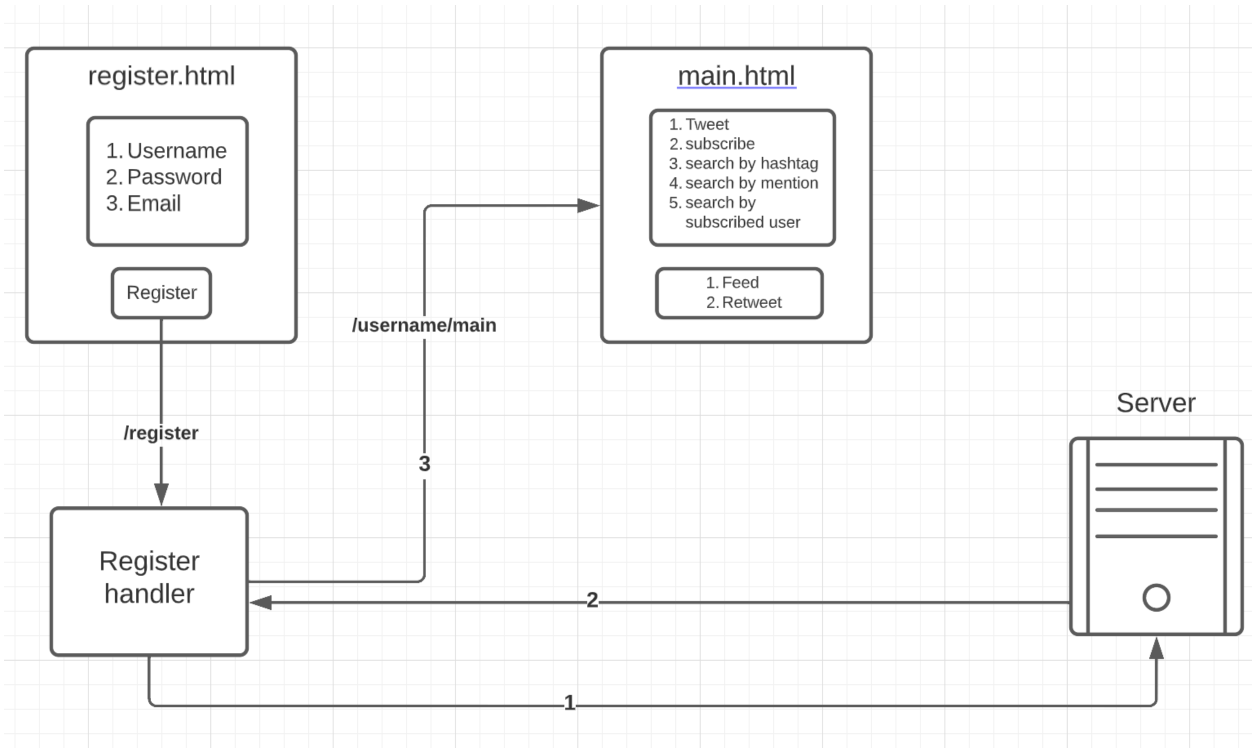
- Add Routing and listening in the <app name>_app.erl
- Run the application : make run

For more references, check out the documentation

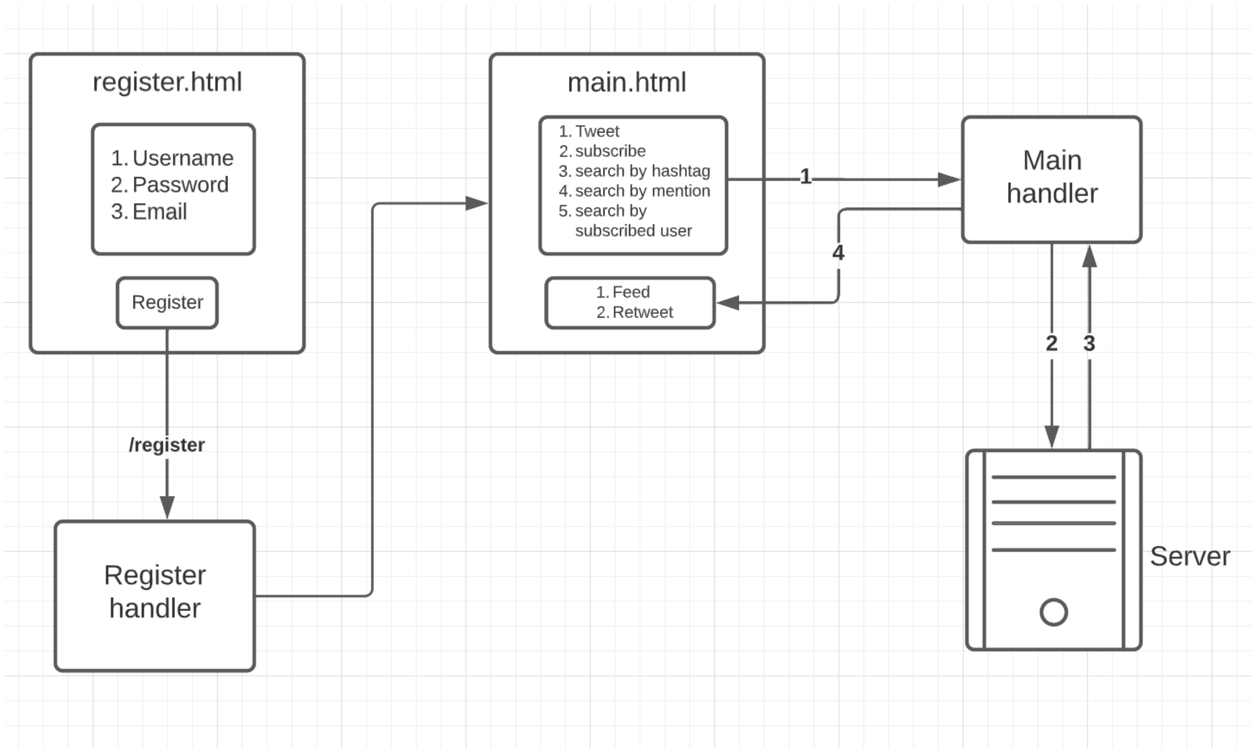
Source Code : https://github.com/ninenines/cowboy.

Architecture

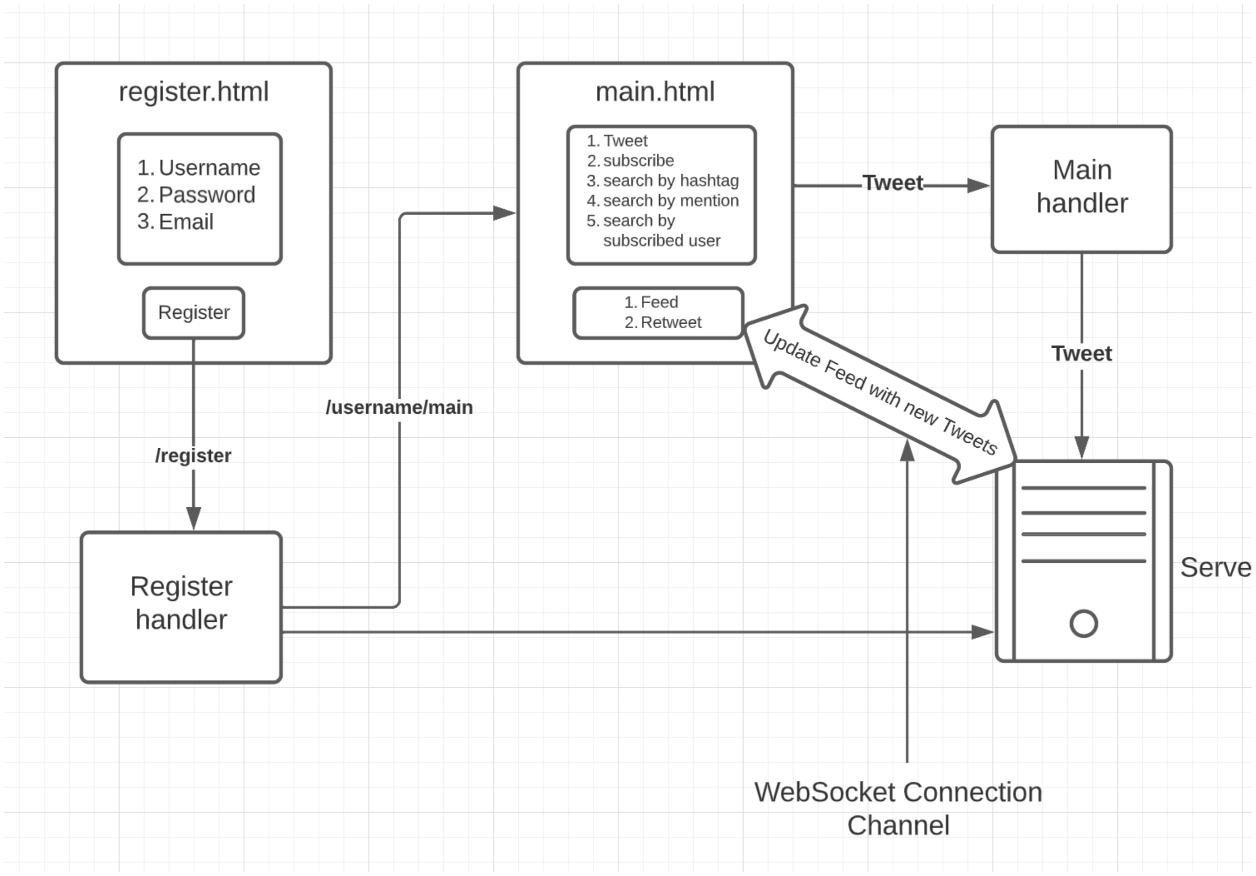
Register



Twitter Functionality

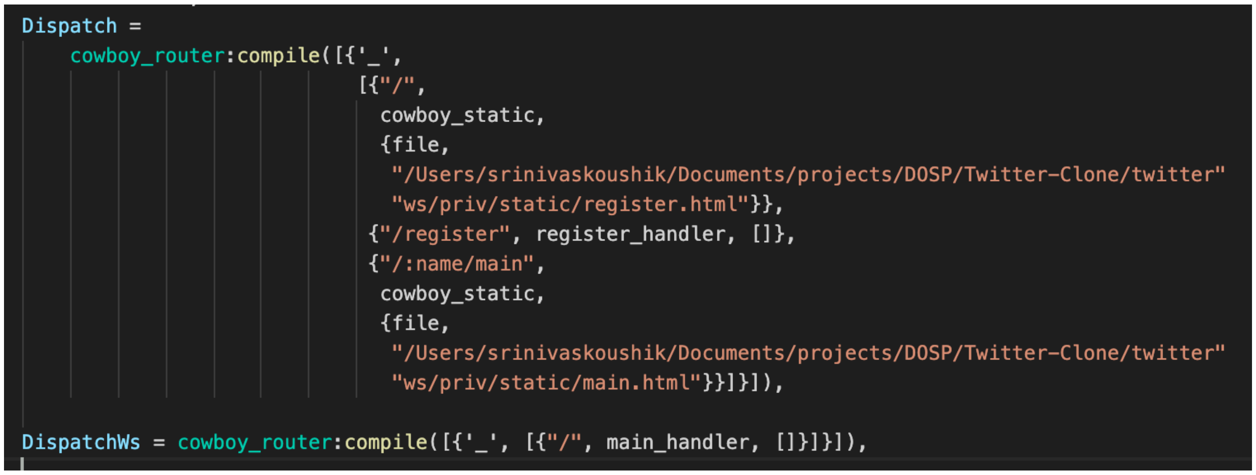


Feed Update with WebSocket



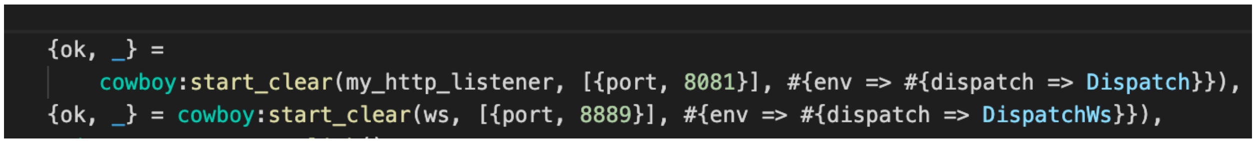
Routing & Connections

Routes



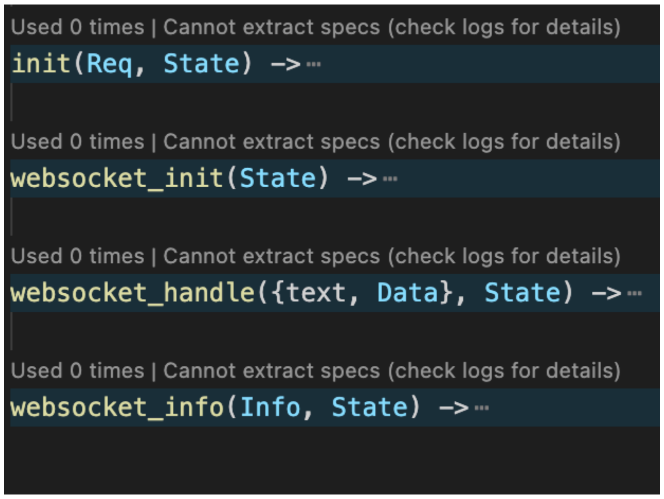
```
Dispatch =
    cowboy_router:compile([{'_',
                            [{"/",
                              cowboy_static,
                              {file,
                               "/Users/srinivaskoushik/Documents/projects/DOSP/Twitter-Clone/twitter"
                               "ws/priv/static/register.html"}},
                             {"/register", register_handler, []},
                             {"/:name/main",
                              cowboy_static,
                              {file,
                               "/Users/srinivaskoushik/Documents/projects/DOSP/Twitter-Clone/twitter"
                               "ws/priv/static/main.html"}}]}]),

DispatchWs = cowboy_router:compile([{'_', [{"/", main_handler, []}]}]),
```

Connections

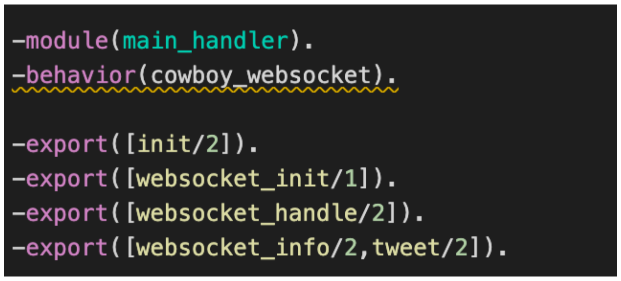


```
  {ok, _} =
      cowboy:start_clear(my_http_listener, [{port, 8081}], #{env => #{dispatch => Dispatch}}),
  {ok, _} = cowboy:start_clear(ws, [{port, 8889}], #{env => #{dispatch => DispatchWs}}),
```

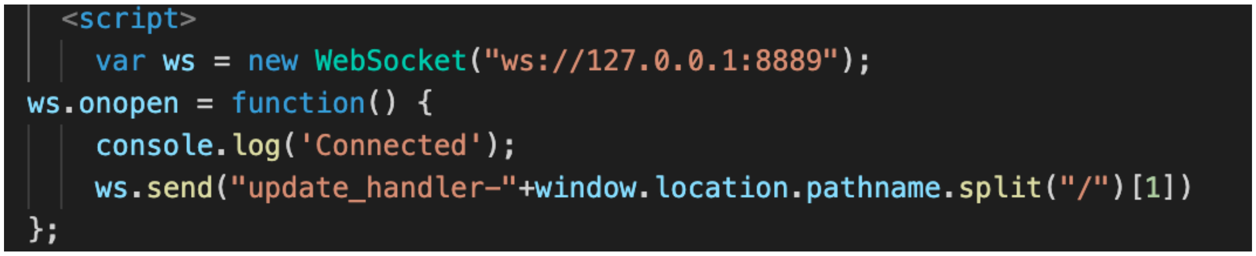
WebSockets



```
Used 0 times | Cannot extract specs (check logs for details)
init(Req, State) ->…

Used 0 times | Cannot extract specs (check logs for details)
websocket_init(State) ->…

Used 0 times | Cannot extract specs (check logs for details)
websocket_handle({text, Data}, State) ->…

Used 0 times | Cannot extract specs (check logs for details)
websocket_info(Info, State) ->…
```

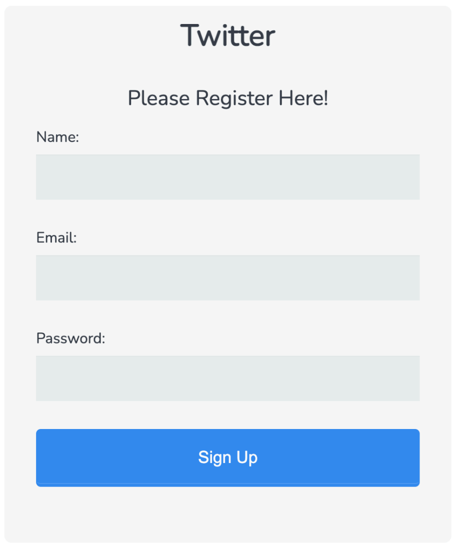


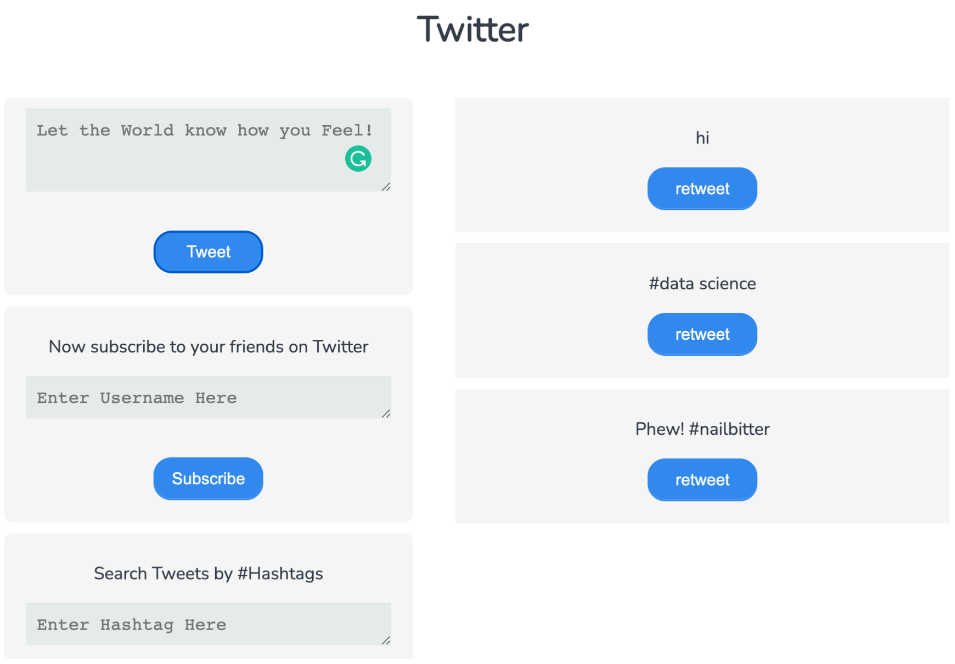
```
-module(main_handler).
-behavior(cowboy_websocket).

-export([init/2]).
-export([websocket_init/1]).
-export([websocket_handle/2]).
-export([websocket_info/2,tweet/2]).
```



```
  <script>
    var ws = new WebSocket("ws://127.0.0.1:8889");
ws.onopen = function() {
    console.log('Connected');
    ws.send("update_handler-"+window.location.pathname.split("/")[1])
};
```

Sample




Twitter
Let the World know how you Feel!
Tweet
Now subscribe to your friends on Twitter
Enter Username Here
Subscribe
Search Tweets by #Hashtags
Enter Hashtag Here
retweet
retweet
retweet

hi

#data science

Phew! #nailbitter

Run Instructions

- Run the following in terminal
  - cd twitterws
  - make run
- Open http://localhost:8081/ in your desired browsers.

Conclusions

A websocket interface for a twitter like engine is successfully implemented with an interactive user interface through which users can perform functionalities like

- Tweet
- Register
- Subscribe
- Retweet
- Query tweets by mention, hashtag & by subscribed users

A websocket connection is established after registering and redirected to /name/main. Once a user tweeted, all the subscribed users will instantly get in the feed through websockets without any user interaction.

Youtube Link: https://youtu.be/21jNTJ_o3R4