CIS-197

Javascript

Nihar Patil

Final Project Proposal

Andrew Merczynski-Hait

November 4, 2019

---

1 Overview

I will build a front-end/client interface to access data from servers running websocketd (`http://websocketd.com/`). The use case is as follows: I have a server, or servers, that are running programs that write to STDOUT. I want to easily be able to view that output over the web (in something like a read only terminal) without writing complex code to send the data to a webpage or logging in over SSH. Amazingly, I find that websocketd let's me magically send my STDOUT to a websocket connection! Yay!

But then I realize, that all I can do out of the box is use my browser to locally view my STDOUT, or I need to write a specific web page to handle the websocket from my application and display it. What I'd really like to is have a webapp that I can use to easily view the STDOUT of any servers I have running websocketd and, switch between them, just as easily as I can use the many available web based SSH clients.

So, this project assumes a server running websocketd, accessible over the internet, and provides an interface to login to an account, manage connected servers (websocket connections), and view the output of those servers.

2 Feature List

- User login to client area when they can see a list of connected servers.

- Can add a server (that is running websocketd).
- Can remove servers
- Can click on added server to open the terminal interface
  - This interface shows the output, i.e. the websocket connection
- The application stores a configurable number of lines of text from each connection as it's running and these are repopulated in the view the next time its opened.
- (Maybe) notify user when text matching certain expressions comes in.

3 Technologies Needed

One major design decision I need to make is whether my server should act as the websocket client that connects to websocketd, and then passes the text to the web client to display OR whether my server should just pass the connection information to the web client, and the client opens the websocket connection to websocketd and displays the text. Using my server as the websocket client would allow connections to stay alive even when the user exits, and would thus allow me to store lines of text that come in even when not logged into the system, but this may not be feasible anyway as it could lead to a huge number of simultaneous websocket connections being kept alive....which may not be desirable from what I've read.

- Websockets client
- Database for users, added servers, and stored output lines.
- Express for handling requests and the server itself

4 Milestone Breakdown

4.1 Milestone 1

- Make design decision on where to open websockets client from.
- Get data flow from websocketd to displaying on a webpage correct.
- add UI to add websocketd servers, remove servers, and show websockets text depending on the server selected for a single user

4.2 Milestone 2

- Get login flow working
- Move stored websocketd servers logic to database for persistence across users
- Add capability to store previously received text

4.3 Milestone 3

- Make UI pretty
- finalize all goals above, and debug as needed