Zachary Kuchar Senior Project Paper 2 2-27-19

My Senior Project has been coming along through the first three weeks. Most game functionality has been completed. These next few weeks will involve finishing up the functionality that I have yet to program, as well as implement an online mode for games that could do so.

This portion of the project involved setting up the project in Node.js for use of online functionality. I started by setting up a test document to use for Heroku, and then managed to convert the project form PHP to Node.js. I was able to set up a WebSocket server, as well as have the page load and render, along with any necessary CSS or JavaScript files. This was not without its fair share of difficulties, however. While I was familiar with setting up WebSocket servers with Node.js, there were some things involving running a node server under Heroku that I was not familiar with. I was disappointed that the process of learning about the necessary requirements and getting everything working took as long as it did. I had to use express to load the webpage, and then do some finagling in order to load any extra files that were necessary. When this step was solved, I then needed to learn how to properly set up a WebSocket server so that Heroku could interact with it. I figured out that there was not too much of a difference, however. The only real change was with changing how the WebSocket on the client end finds the server. I needed to use a fancy function such as the one down below, so that the webpage could connect to my server and send messages properly.



Figure 1. WebSocket client code

Once I set this up, I was able to test to see if the server was working correctly with some simple communication scenarios. I set up the login and create a login functions to send a message to the server, and then have the server send a message back. It seemed like there was an error with the correct message being called, but this turned out to be a simple error with if statement Booleans. Once it was fixed the server appeared to send and receive messages correctly.

Online mode

With the main server now up and running, I could begin to introduce an online mode for games. I decided not to do one for all games, since games such as Blackjack and War I felt were too simple to feature two players, and games like matching and spider solitaire could not have a second player. This meant that only Crazy Eights, Snip Snap Snorum, and Go Fish would be given online modes.

Crazy Eights would not be the hardest game to implement with an online mode, since I have previously worked on a similar project for a class. One main issue would be how to properly set up the change to an online mode. I originally wanted to go with a separate window that would open with the online version of the game, however, I opted to go with just porting everything in the current window into an online format. I would delete any traces of the current game and then set up a new online game. I would also change the current presenter that ran the game into a presenter of the online version. During this process, I had some troubles with removing event listeners from the offline mode, and then putting on event listeners of the online one. I was having trouble using the function removeEventListener(). In the end, I found a way that worked by cloning the object. By creating a clone of the object, and then reinserting it into the correct spot in the webpage, I was able to effectively remove the event listeners that were in place. While working on the game itself, I found a problem that prevented more than two people go online and be a part of multiple games. Since early attempts to fix this problem were unsuccessful, I opted to put it aside for the time being, and then revisit it later when all online games were working.

From there, I moved into work on the actual online game functionality. The game would tell the server whether it was going to play a card or draw a card from the deck. The server would then send a message back with the essential information to update the game. With how my project was currently set up I had to send the information through the web page itself, since that was where the main WebSocket connection lied. The message was sent to the webpage, and then the webpage used a part of the message to determine which game to send the update message to. From there, the respective update function was called to update any information on the respective client’s end. Lastly, I set up the ability to switch back to offline mode. By clicking the button to toggle online mode again, the current user can switch back to a normal version of the game. You have to make sure that you want to, however. Switching back will also inform the other player that their opponent has left the game, and that they have won. That player will then need to switch back to offline mode.

Finally, I began making the setup for Snip Snap Snorum. I managed to setup the ability for the server to be able to talk to the server. However, time ran short for this week, so I was unable to get any further work done with this part of the project. As time comes, I hope to be able to finish work on Snip Snap Snorum and Go Fish for online.