Zachary Kuchar CardGames.org 2-27-19

Over the course of my life, I have always had an interest in playing video games. It has been one of my favorite hobbies to do in my down time, which has inspired me to create them. This project allows me to achieve that. It involves a website full of various card games. Users to create an account on my website. From there they can participate in several different card games. They can play games against the computer, or against other players online. Once they win a game, either their score, or the number of moves will be recorded in a leaderboard. Users can view these leaderboards and compare their score to those of others who have used the site. This project drew my interest because I have a desire to be a video game designer. One hobby that I adore doing in my free time is playing videogames. I love to play videogames, especially with friends. I feel like I could have just as much fun creating games as I would by playing them.

I decided on the main look of the project so that it could look more like an actual website. A title at the top as well as tabs make the project look more presentable. There will be a tab for creating a login, one for each game, and one for loading up any particular leaderboard. The setup not only looks nice, but it is very easy for a user to maneuver around. Everything is clear to the user how to move around the webpage.

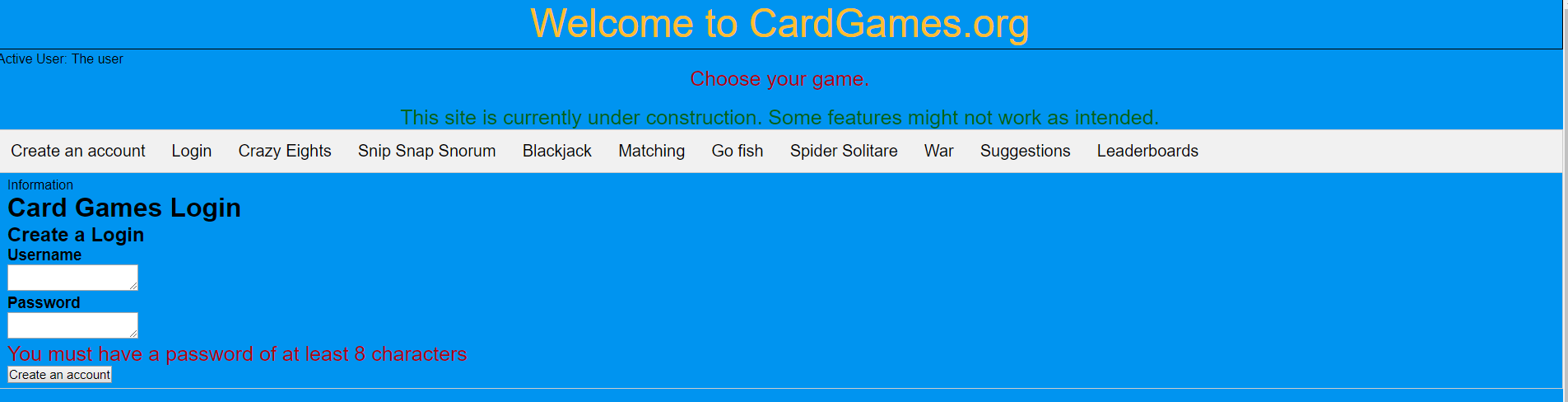


Fig 1 Project look

The graphical style of each card game implemented was based off a Crazy Eights project that was done in a previous class. The look of the cards was taken from the project, while the positioning and the functionality were adjusted in order to fit the game that was being implemented. The only exception to this was a match game that was implemented. The features for this game were based on code that was borrowed from an online source. For messages that needed to be displayed, changes were made so that they would be displayed in the message bar of the respective game. I wanted to make sure that there were no alert messages were being displayed, even if it was just for the purpose of testing.

As for the functionality of each game, Crazy Eights maintained similar functionality, with the addition of an improved computer. Other games were given adjustments so that they could work properly but kept a similar setup to that of Crazy Eights, with a different class for each of the elements of the game. Crazy Eights did not undergo much change. Upgrades were made to the computer player so that it would perform better and be a tougher opponent for any user that played against it. Other games had a similar setup to Crazy Eights, but had their internal methods changed to reflect how the game was played. For games such as Go Fish and Snip Snap Snorum, the initial look did not need to change much. However, the internal workings of the game resulted in a dramatic change in how it works. Games such as Spider Solitaire had to have a complete overhaul, resulting in a change in both look and features. Games such as war and blackjack required changes, but these were much simpler to implement. The main feature involved pressing a button to have cards go where they needed to go.

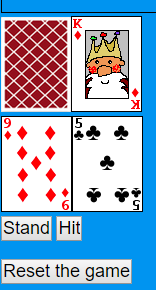


Fig 2 Blackjack(you need to hit buttons to use the game)

For war, I would put out the cards, and then determine which card has the higher value to determine who got the point. Blackjack was similar in a way. I had buttons to have the user determine if he wanted to put down a card or not. However, I also had to keep track of the overall value of the cards in both players hands. This allowed me to determine how to see who won the game. One additional feature that I thought that would be a nice feature to have was an ability to reset a game that was being played. This allowed players to just start a new game at any time if they so choose. Where it was appropriate, I also disabled some features when using them would cause problems.

Challenges/Problems

Working on this project did not come without its fair share if difficulties. It was not easy to calculate the values of the cards in games like Blackjack. One coding error that I encountered was the way the values were being concatenated together. They would be put together as one big number instead of being added together separately. This problem was eventually solved through a trick I discovered online. I would simply put a + sign in front of the values to be added up. This would allow them to be added up instead of being put together as one giant number.



Fig 3 code snipit. Using + sign to add numbers instead of concatenation

A technical problem that arose while working on Blackjack was how to properly determine how to incorporate both values of an ace. An ace in blackjack can count as the value of either a 1 or an 11. It proved challenging to effectively switch between the two values. As I tested different methods. I eventually settled on going with a value of 11, and then subtracting 10 if the total hand value went over 21. This method appeared to have worked. Another technical problem was that I was having trouble with was setting up a way for play to be disabled once a winner was determined. Once we determine a winner, we do not want to allow the player to interact with the game anymore. At first, I was using an image to click on, but when problems arose with disabling the ability to click the image, I instead went to use buttons. These were much easier to disable, since all I needed to do was set the disabled property to true.

A few snags that developed while working with Go Fish involved determining what game action to perform, as well as making sure the user could not put down more cards than necessary. Eventually, I managed to work out the problem. I set up a Boolean variable to determine if the player needed to ask for a card or give a card to the computer. This allowed for game to be able to determine what action for the player to take as intended. I was also able to have the player only be able to put down the correct card or hit go fish.

Spider Solitaire also presented some problems since it required a complete overhaul in both the look and functionality. Through making the card’s positions absolute, I was able to set up the cards in vertical rows, instead of having them in horizontal rows. I set up cards to be moved by clicking on them, and then clicking on the section that they were be moved to. This feature took some time to figure out how to do correctly. In the end, I opted to note each of the cards that were to be moved, and then determine if they can be moved to the new location. If they can be, the cards are removed from their old row, and placed into their new one. I also had to make sure that the player did not try to move a face down card or a card that would make an empty row.

With the main issues fixed, I began to work on the other features of the game. I set up the ability to move a king into an empty row. I also set up the ability to deal out new cards to each row. That way a player would not be stuck if they ran out of moves. I also set up a way to remove a whole row of cards from a pile in order to add a point for the user. This required looking at the row we just moved cards to and seeing if we had all the necessary cards placed down in the correct order and then remove those cards if they were.

Week two by setting up a GitHub account to put the code for my project out on. I also worked to set up a Heroku account, to run my project as a normal website.

Setting up the server

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.