Ho Yin Chan

[Email address]

term 4 digital technologies written reflection

We learned JavaScript in Digital Technologies this term. The objective of our assessment was to create a web based card game to assess what we have learned in the entire term.

**Section 1: Card Draw Game**

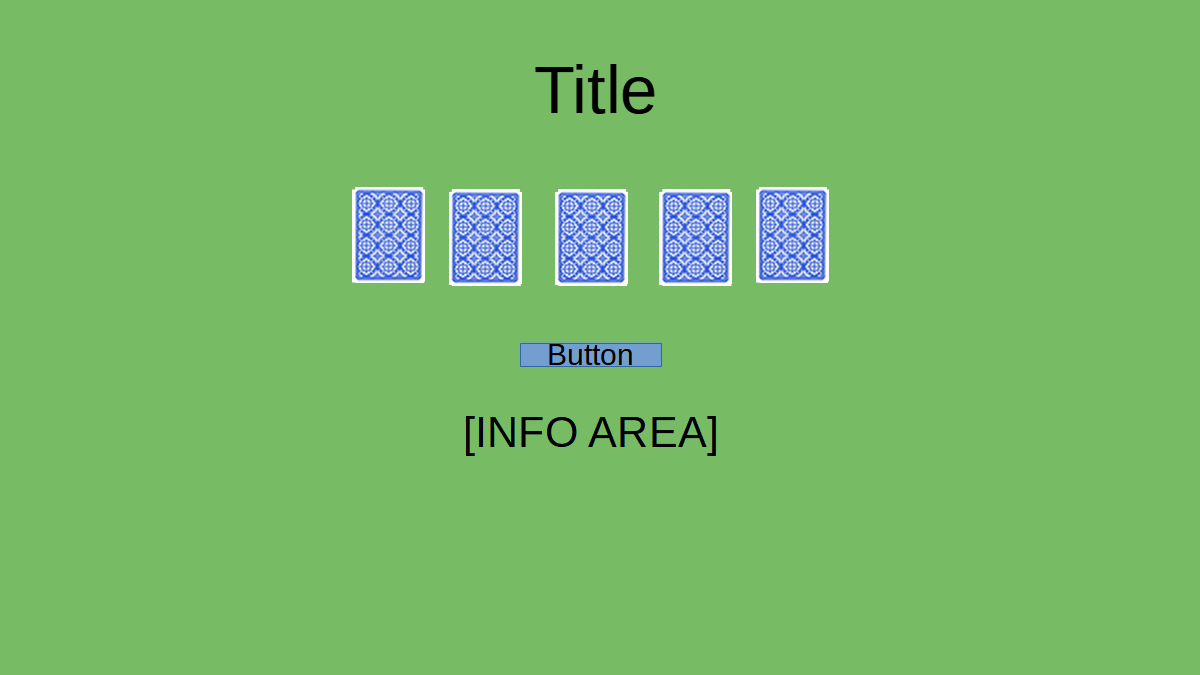


This is the first of the two web games I have created for the assignment. It is a simple card draw game that draws 5 cards, and then displays the combined score and ranking.



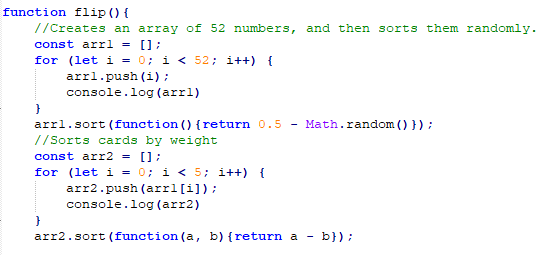
Here are two examples of the game.

First, I planned out how I wanted the site to look.



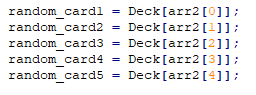
Then I began work on the game.

First, I made the code that shuffles the cards.



The code creates an array with 52 numbers and then shuffles them. The reason I used arrays for the random number generation is that it is the easiest way to ensure that all of the numbers generated will be unique.

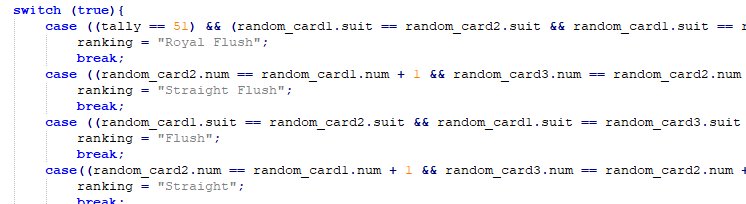
Then, I assigned the generated numbers to 5 variables that will correspond to the five cards on screen.



After that, I wrote the code that adds up the value of the cards.



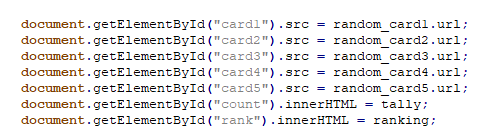
Then, I had to write the part of the code that checks the player’s hand and ranks it. For flushes, it was pretty straightforward.



However, for (x) of a kind/pair hands, I had to write code that check for duplicates, and code that checks for duplicates of duplicates.

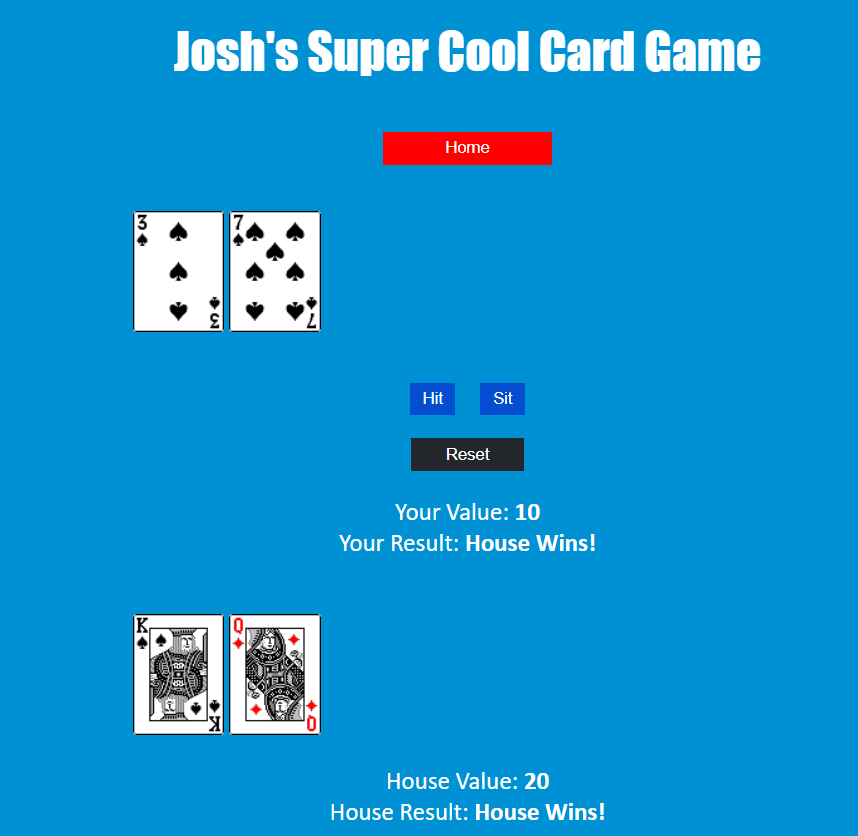
Graphical user interface, text, application

Description automatically generated



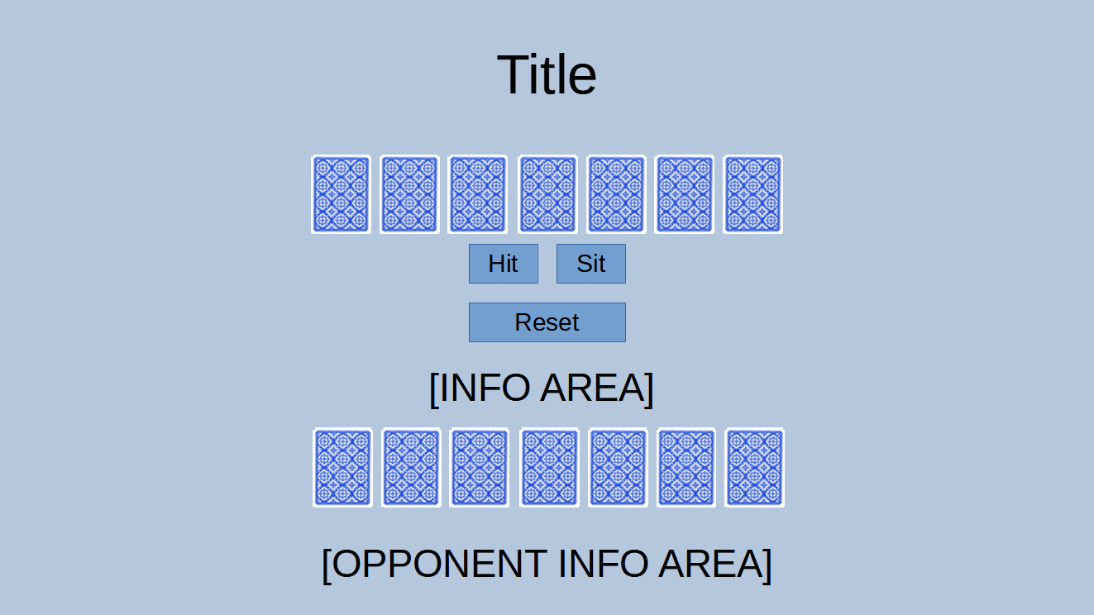
This is the final part of the code, which just assigns the javascript variables with the corresponding HTML elements.

**Section 2: Blackjack**



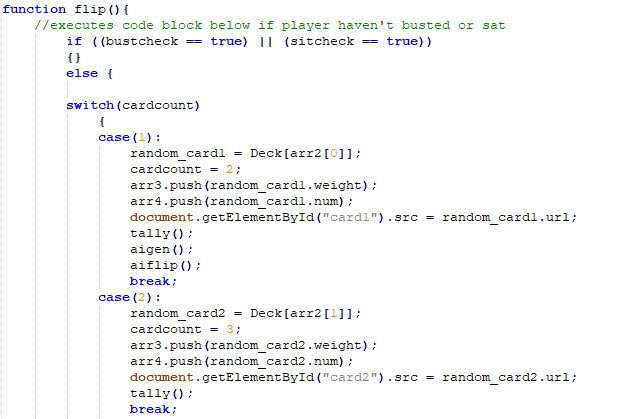
This is the second web game I’ve created. Blackjack was a massive headache to work on but in the end I learned a lot about JavaScript and coding in general. The objective of the game is to get a high score than the house without going bust (getting a value over 21). Blackjack was a lot more complex than the card draw game as I needed to include a computer opponent.

First, I planned out the UI.



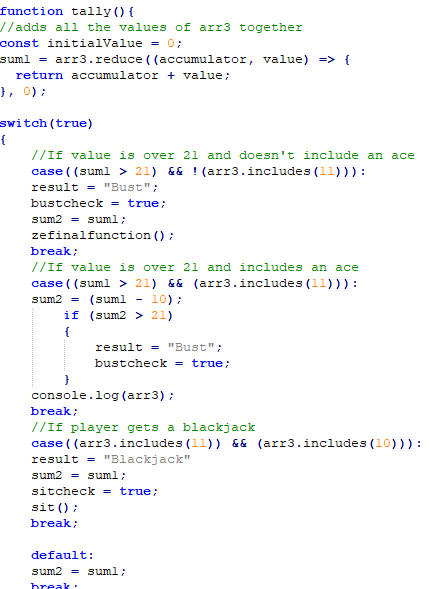
I used the layout from the first game as a template, then I added a button that allows the player to “sit”. After that, I added a button to reset everything after the game is done. Then I added the area that displays information for the computer opponent.

I reused code from the card draw game as a foundation for blackjack, and then I created a card counter to keep track of how many cards the player has.



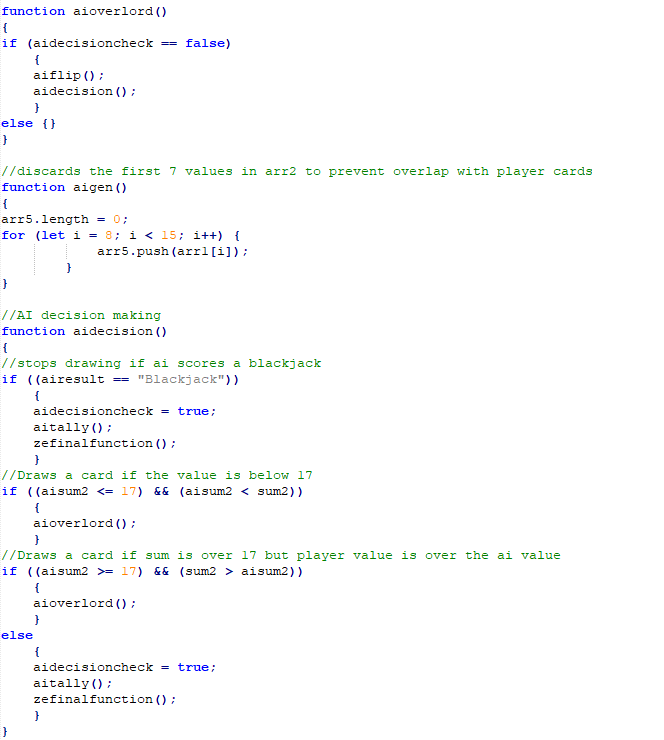
Every time the player presses the hit button, a card will be displayed on screen and the card counter and the value will be updated. Then, the combined score will be calculated. If it is the first time that the player is pressing the hit button, the house will also draw a card.

Then, I wrote the code that calculates the total score and decides if the player has gone bust. I changed the simple code that just adds up all the combined values of the variables in the first game. The reason is that I wanted to avoid problems with calling up variables before they are defined.

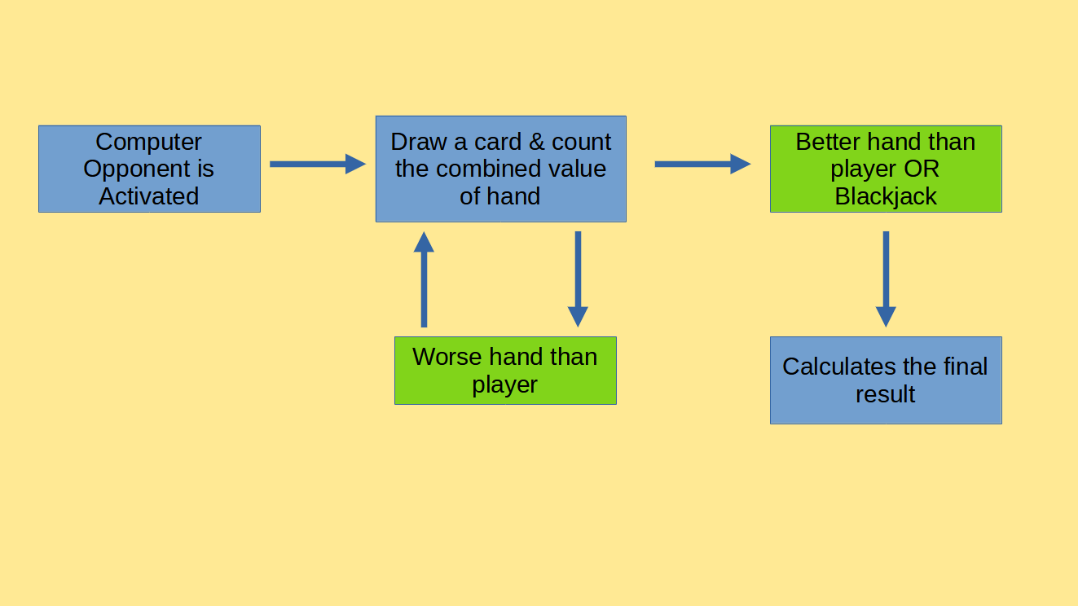


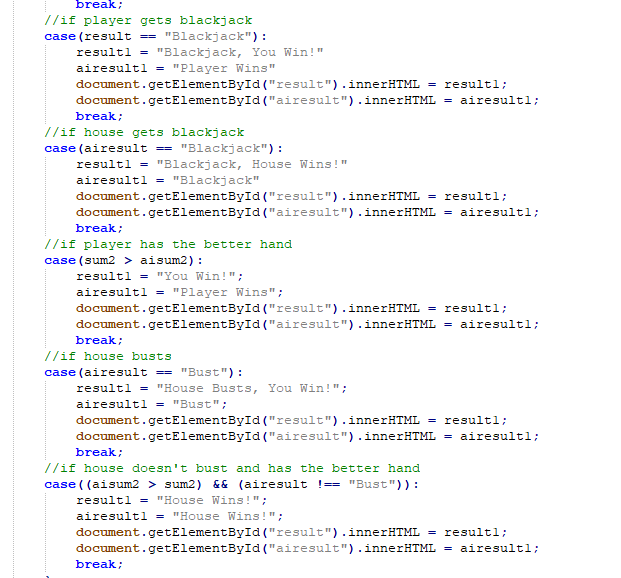
After the scores are calculated, the code checks checks if the player has gone bust, gotten an ace or blackjack (an ace and a 10 value card). The ace in blackjack can be either a 1 or 11, which means I had to write a block of code to make sure the player doesn’t go bust when they pick up an ace. The code is pretty simple. If the player has an ace and the combined value is over 21, the ace’s default value of 11 gets reduced to 1.

After that, I began work on the computer opponent, which is activated when the player presses the sit button.



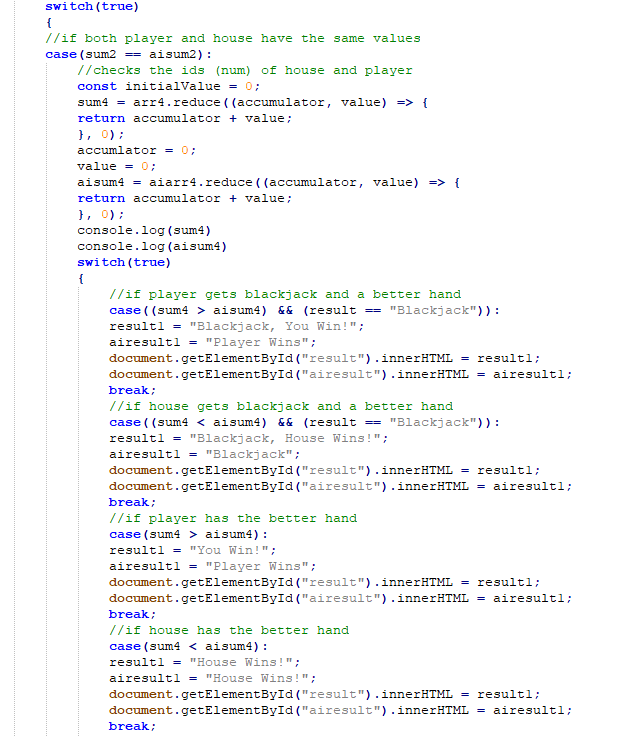
Here is how the computer opponent works. It will draw a card and then count the combined value of it’s hand. If the computer has a hand that is worse than the player’s, it will draw another card. However, if it has a blackjack or a better hand than the player then it will “sit”. Which executes the final part of the code.





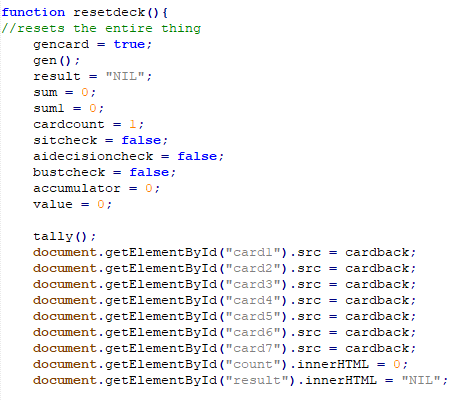
This block of code compares the player’s hand with the computer’s hand. If the computer has a better hand or has gone bust, the player wins, vise versa.

But what if both the player and the computer has the same value? I then had to write code to deal with that specific problem.

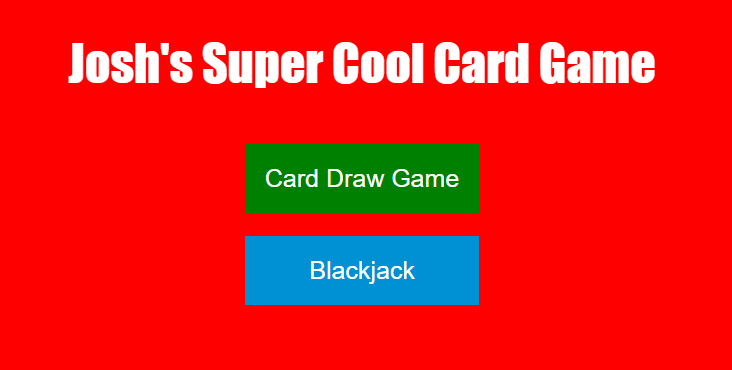


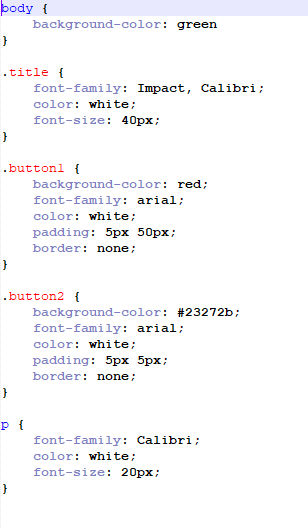
This block of code checks for 10 value cards in both the player and the computer’s hand and compares them. For example, if both player and computer ends up with a value of 17, and the player had a King and the computer had a Jack (which is below the King), then the player would win.

After that, I programmed the reset button, which just resets everything back to the initial values.



After I had finished the headache-inducing part of the assignment, I decided to make the UI look pretty and also make a home page.





The home page links to the two games and the CSS sheets are for cosmetic purposes.

And that’s it! I’m finally done!

The biggest challenge in this assignment is my own carelessness, as I often make simple mistakes that breaks the code. For example:

If (cardcount == 5)

This if statements check if the number the cards the player has is equal to 5. However, when I first wrote this statement, I wrote it like this:

If (cardcount = 5)

At first glance, they look like they do the same thing. However, “=” assigns value to variables and “==” means “equal to”. The incorrect statement I wrote broke the entire code because it assigned the variable “cardcount” to the value 5 instead of checking if it is the value 5.

This assignment taught me a lot about javascript, and also how tiny mistakes can have huge consequences. It also taught me to comment in my code, as it makes it a lot easier to figure out what went wrong if everything is labeled.