My game is entitled *The Rock Paper Scissors Adventure* and is a simple clicking game with adventurous influences. This game was inspired by what I have learnt from this module and from external sources like codeacademy, and combined with my interest in the fantasy genre.

**Layout**

I utilized the layout page that we have been using in class. However, I modified it to suit the adventure-horror theme I wanted for the game. I did my own research and embedded my chosen google fonts into the layout, particularly in the header, the nav div which I designated for instructions and the aside div designated for the scoring. I also changed the border and the background color of the layout. In the header, the background was also changed to a graphic of a giant preparing to attack the player to heighten the sense of fear. This graphic was sourced from an online graphic library. A fantasy background was also chosen from Google Images and inserted in the background of the CenterDiv, where the main gameplay occurs.

**Gameplay elements**

There are a number of music clips used in the game to enhance player experience and interaction. Sounds which are sourced from freesounds.org, as introduced in the class, are used as background music and to indicate when the player wins, loses or ties in a round with the giant, and also if the player loses the game. These music could also be used to build the tension while the player plays the game. To give players a sense of control over the music, I included buttons for them to toggle the background music on and off. For the music, I made good use of the setTimeout function learnt in class. This was to trigger the sounds to be played whenever the player finishes a round.

The nav and aside div were configured to change according to the player's actions by instructing the player and displaying the player's results respectively. This is done using the innerHTML property of the div. It can be nested within various functions to run everytime these functions are called. This builds the game's interactivity and gives the player a sense of 'real-time' gameplay.

**Gameplay logic**

The game utilizes clicking to compare two choices. The code was adapted and modified from the lesson in codeacademy. The online lesson used mainly prompts and I had to modify it to become more visual. This involved developing more functions and I ran into problems while developing the code.

Firstly, I used prompts to start the game. However, I found that the prompts overwrote the rest of the code that I wanted to run simultaneously, such as the display of the background. This meant that when the prompts were activated, the background image and other elements were unable to be displayed. The way around this was to create my start option as a div in the paper itself. Thereafter, I added event listeners to the buttons so that if the player clicks 'Yes', the start function would run. I also had to create a code for the 'No' option and added text to ask the player to exit the browser.

I added a restart button at the footer so that if the player reconsiders playing, he or she can easily click on the restart button to refresh the page. This was done by research on W3Schools, which indicated how to refresh a page.

If the user chooses to play, he or she will be prompted to key in the number of rounds to be played. If the player keys in an invalid number below 5, an alert will appear and the page will be refreshed for the player to key in a number again. I had to initialize the variable outside the start function, otherwise it would not work as I will use the variable in other functions. The number that the player keys in will be updated with the end of each round. The round function contains a loop that will count the number of rounds already played and will end the game when the round reaches the number that that player keyed in.

After the user keys in the number, I will have to show the elements- rock, paper and scissors previously hidden at the start page. This code was re-used from my previous assignments. I added in pictures of the player and the giant to indicate which set of pictures correspond to each. I also added event listeners for each of the elements in player's set, then created 3 corresponding functions. With each choice the player makes, the unchosen elements will be hidden and the computer will run a probability function- the math function learnt in class, and I segmented the probabilities into 3 parts, each of which will correspond to rock, paper or scissors.

Thereafter, I learnt about the switch function from W3Schools and used it to compare the player and the computer's values. If there is a match in the values, a code will be run. Then the innerHTML in the aside div will be updated to reflect the score. I also printed messages to the console to make sure that the function is working fine. I also had to ensure that the player knows which element the giant selected by hiding the rest of the elements.

I then displayed the next button and added an event listener so that the innerHTML and round number will be updated for the next round. If the player wins the round, I will increment the result variable and display it in the aside div as well. This increment 'i++' was learnt in class. This is done by adding 'i++' after each winning case in the switch function. With each round, the result will be updated to let the player know his or her progress. After each round, I had to clear the giant's choice by hidden its set of elements and showing all the player's elements again for the player to choose.

After the game has reached the number of rounds that the player has keyed in, an alert would tell the player that the game has ended. The player has to click 'OK' and the nav div will give the player more information on what to do next. I also added a surprise element when the player loses the game so that it would be consistent with the game theme.

In hindsight, the code is very long and a way to optimise the game is to develop an algorithm to process the player's choices. That way, I would not have to change individual lines of code everytime I want to edit the functions. In addition, I could have also used the appendChild to organize the elements in JavaScript, such as creating the 'Next' button in the aside div instead of within the paper itself.