

LENGUAJE *DE MARCAS*

ADRIAN *VICENTE LOPEZ*

1º *DAW*

JUEGO *DE LAS SIETE Y MEDIA*

-CONTENIDO-

[1.- EXPLICACION Y REGLAS DEL JUEGO 2](#_Toc190800909)

[2.- ANALISIS Y ENFOQUE DEL PROCESO 2](#_Toc190800910)

[3.- DESARROLLO DEL PROCESO DE PROGRAMACIÓN Y EXPLICACIÓN DEL MISMO 3](#_Toc190800911)

[3.1- DISEÑO DE LA BARAJA DE CARTAS Y ELMENTOS GRAFICOS 3](#_Toc190800912)

[3](#_Toc190800913)

[4.- INTERFAZ Y EXPERIENCIA DEL USUARIO 5](#_Toc190800914)

[5.- CONCLUSION 6](#_Toc190800915)

[-BIBLIOGRAFIA- 12](#_Toc190800916)

# 1.- EXPLICACION Y REGLAS DEL JUEGO

El objetivo de este juego es el de sumar un valor total **lo más cercano posible a 7,5 puntos** o exacto a ese valor, sumando todas las cartas que tiene el jugador en la mano.

**Las cartas suman un total de 40 carta**s en la baraja (**4 palos de 10 cartas cada uno**), de las cuales van desde el **1 al 7** con sus respectivas puntuaciones, ósea el 1 = 1 punto, el 2 = 2 puntos, así sucesivamente hasta el 7, después para las últimas tres utilizaremos las **tres figuras las cuales serían el 10, 11 y 12** (quitaríamos el 8 y el 9), para estas tres figuras estableceremos una puntuación paralela **de 0,5 punto**s por cada una.

En el desarrollo del juego obtenemos **primero la jugada el usuario** y la cual obtendrá una **primera carta boca arriba**, el cual decidirá si con su puntuación **sigue** (hasta intentar sumar 7,5) o por el contrario decide **plantarse** y quedarse con su puntuación (según su estrategia).

En caso de que el usuario obtenga una puntuación **superior a 7,5 puntos directamente quedaría la partida como derrota** frente a la máquina, pero si queda **igual o por debajo** de la puntuación antes mencionada entonces quedaría esperar al turno de la máquina.

Una vez le llega el turno a la maquina igual que el usuario podrá **seguir o plantarse** con su puntuación, pero esta vez, el contrincante del usuario juega con la ventaja de que **ya sabe la puntuación del usuario** por tanto podría plantarse o intentar seguir si el jugador ha sacado la puntuación más alta.

**Si la maquina iguala la puntuación sea cual sea la del usuario, el usuario pierde**, aunque sea la puntuación más alta, ósea 7,5. Pero si por el contrario la maquina **decide seguir y se pasa el jugador resultaría vencedor.**

# 2.- ANALISIS Y ENFOQUE DEL PROCESO

**En primer lugar,** nos vamos a centrar en **obtener las cartas con su forma visual**, ósea en **diseñar las 40 cartas** (4 palos diferentes entre sí), **también diseñaremos el fondo o tapiz** que elegiremos para nuestro juego y **así como los botones (en mi caso un botón verde para seguir y un botón rojo para plantarse)** e intentaremos meter también un **sonido de fondo o en caso de perder o ganar.**

Después ya iremos a la programación de las diferentes funciones que en este punto desglosaremos, **en segundo lugar,** tendremos que **unir dichas cartas graficas a un valo**r, esta función la haremos con **dos arrays paralelos**, uno con las imágenes y otro con los valores los cuales irán asociados a su vez a una función de **generar un valor aleatorio** para que **no salgan en orden de inserción**.

En tercer lugar, queremos que las cartas **no se repitan**, esto quiere decir que una vez que salga una carta **ya no vuelva a salir en la partida, ni al usuario ni a la máquina**.

En cuarto lugar, tras la implementación de los botones de seguir dando cartas (en este punto aún no he nombrado al botón, además serán verde o rojo) y el de plantarse, queremos **que el botón de seguir se bloquee y no deje seguir** si detecta que la puntuación del usuario o la maquina ya se ha pasado de 7,5 puntos.

En quinto lugar, programaremos **el cálculo de las puntuaciones** del usuario y la maquina después, para por último determinar quién se pasa o no llegar o con otras palabras **quien es el ganador de la partida.**

En sexta posición la programación de los turnos para que **primero sea el del usuario** y una vez el turno termine **empiece el turno de la máquina**, que hará su jugada para por último punto determinar y verificar las puntuaciones y proclamar al vencedor de la partida.

# 3.- DESARROLLO DEL PROCESO DE PROGRAMACIÓN Y EXPLICACIÓN DEL MISMO

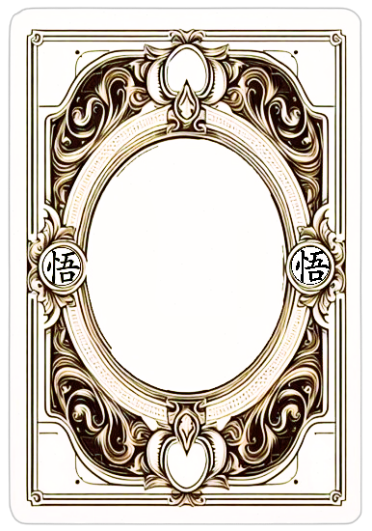
Siguiendo con la misma tónica iremos explicando gráficamente todo y siguiendo más o menos a modo de guía el proceso del análisis anterior.

## 3.1.- DISEÑO DE LA BARAJA DE CARTAS Y ELMENTOS GRAFICOS

En las siguientes imágenes mostramos los diseños de los diferentes elementos que contendrá el juego, cartas, fondos, botones y un teaser a modo demostración de producto final (puede que difiera del resultado final o eso espero…).



# 

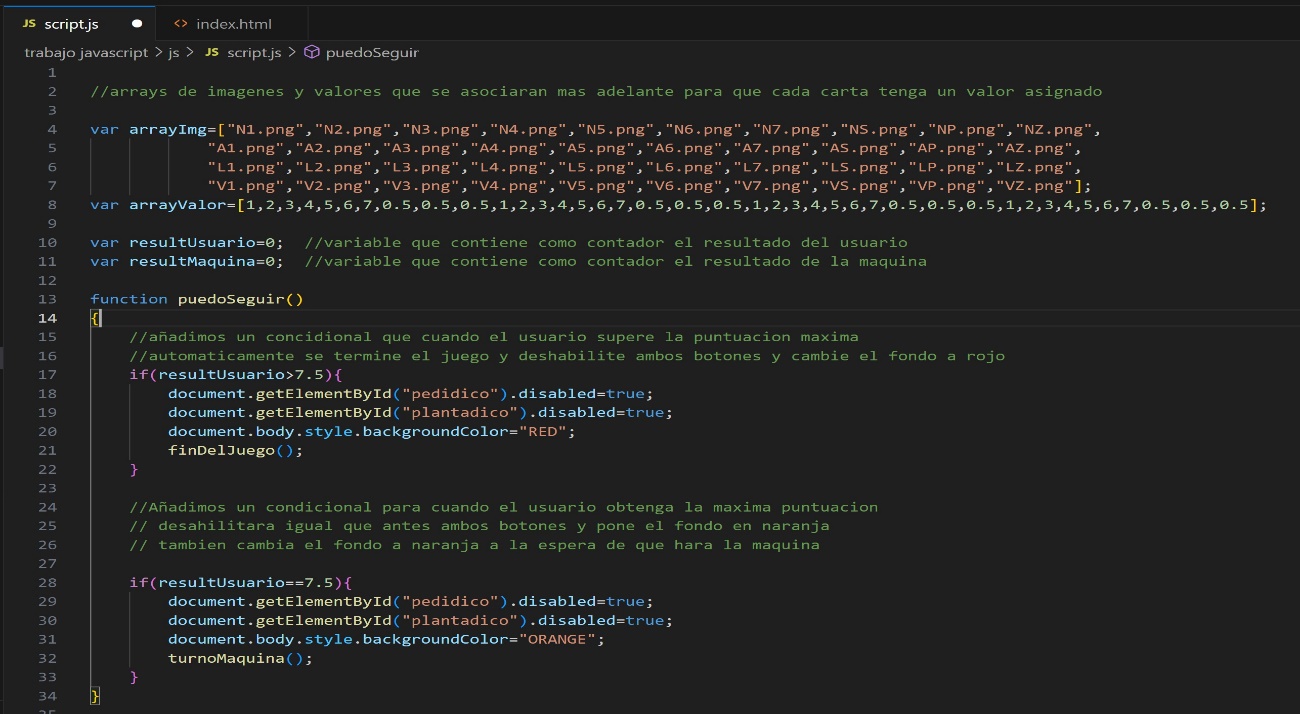


## 3.2.- IMPLEMENTACION DE IMÁGENES CON ARRAYS Y VALORES

En esta imagen en principio hay más cosas, pero nos fijamos sobre todo en que hemos creado dos arrays que irán unidos en las siguientes funciones, como vemos tenemos un array de imágenes que hacen referencia al archivo con extensión png y abajo un array de valores los cuales de forma ordenada se corresponden con los valores dados a las imágenes.

Mas abajo podemos observar como hemos creado dos variables donde se guardarán las puntuaciones de ambos jugadores, así como la función del botón para deshabilitarlo en caso de que el jugador se pase con un correspondiente cambio de color en el fondo a rojo (mas adelante lo intentaremos cambiar por un gif y una melodía de derrota, preciosa, por cierto).

Aprovechando que tenemos esta imagen explicamos también la función de pedir carta la cual ira asociada al botón de pedir carta y la cual vemos que tiene una variable que obtiene un numero aleatorio (ósea una carta aleatoria) otra variable que guarda la ruta de donde se encuentran las imágenes y asocia a todos los archivos del array de imágenes, también vemos que se modifica mas abajo con la función “splice” el numero de cartas que quedan ósea resta una a la baraja, mas concretamente las que van saliendo, tanto a las imágenes como a los valores. También vemos como nos muestra en el voy las cartas y el resultado y metemos dentro la función antes mencionada “puedo seguir” que deshabilita los botones para cuando te pasas. También como podemos ver hemos desarrollado otro condicional el cual nos hace lo mismo que el anterior para la puntuación igual a la más alta, pero en vez de terminar el juego este pasaría al turno de la máquina. Y pone el fondo en naranja como para dar énfasis a la espera de la maquina y ver si ganas o no. PD; ambos fondos pretendo sustituirlos en el futuro por gifs para cuando ganas o pierdes, y por tanto el resultado definitivo no sería ese.



## 3.3.- FUNCIONAMIENTO DE LOS BOTONES PEDIR Y PLANTARSE

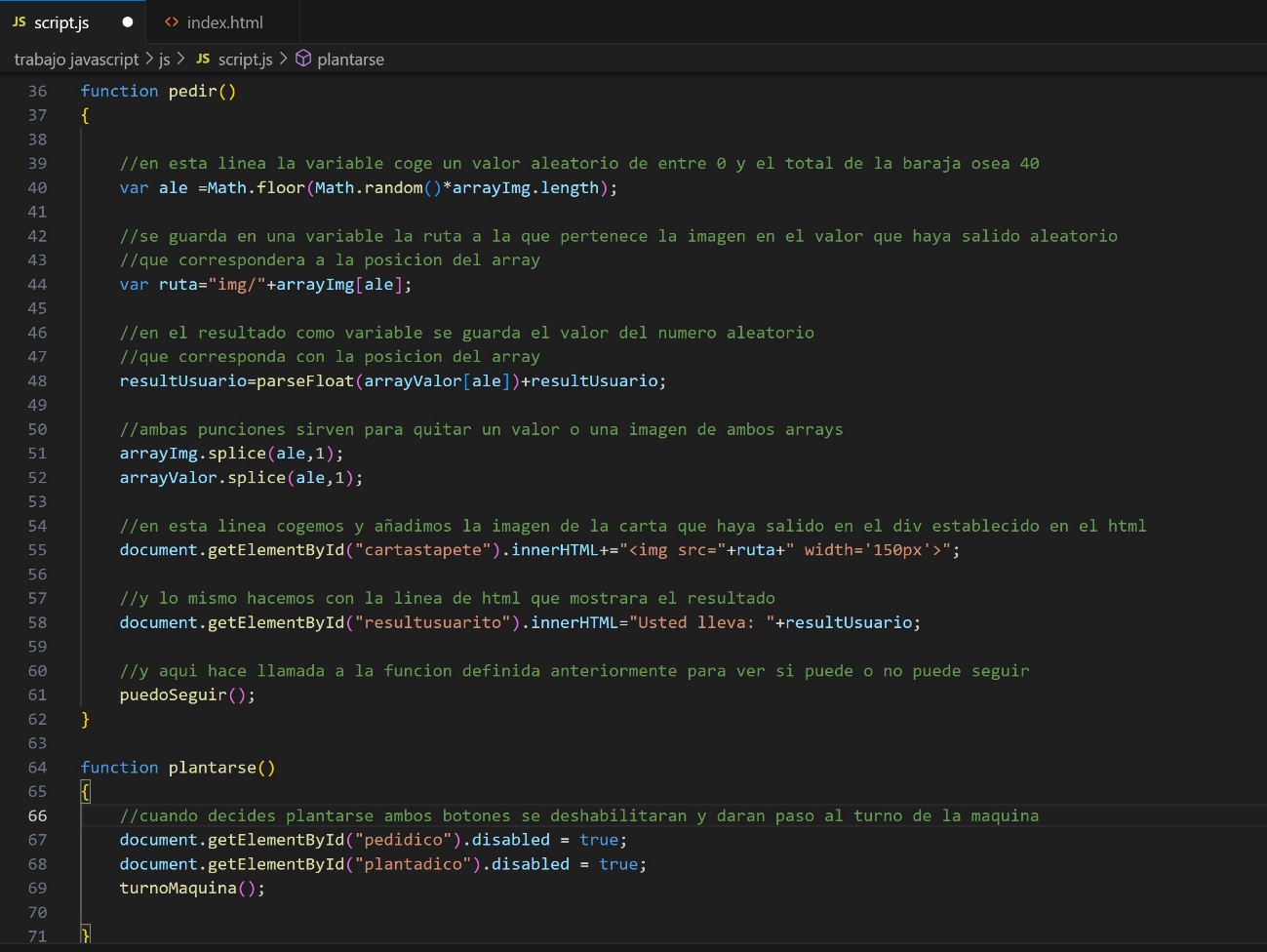
A continuación, en las imágenes mostraremos las funciones que se llevan a cabo para los botones de plantarse y pedir carta. La función de pedir y la del turno de la maquina son casi idénticas, es mas la del turno de maquina me base en la función pedir para realizarla. El funcionamiento del botón pedir comienza por generar un numero aleatorio el cual almacena en una variable, después en la siguiente variable “ruta” se almacena la ruta a la cual se le ha asignado el valor que salió en la variable “ale” ósea, que si sale el tres, pues se le asignara la ruta de la carta que ocupe la posición 3 (en este caso sería la carta número 3, aunque está mal en la imagen porque las cartas van desde el numero 1 al 40 dado que el math.floor redondea a 1 las posiciones y de esa forma coinciden posiciones, cartas y valores ).

Después actualizamos el resultado con la variable “resultUsuario” y vamos sumando los valores asignados a las diferentes cartas que nos van saliendo, a continuación, quitaremos con la función “splice” las cartas de la baraja que ya hayan salido de manera que las eliminamos del array y no vuelven a salir para que no se repitan.

En la siguiente línea vemos como mostramos las cartas en el <div> donde van las cartas del usuario y mas abajo mostramos la puntuación del usuario.

Por último, terminamos con la función “puedo seguir” antes explicada que llamara a la función para verificar si la puntuación es correcta o para proseguir dando cartas.

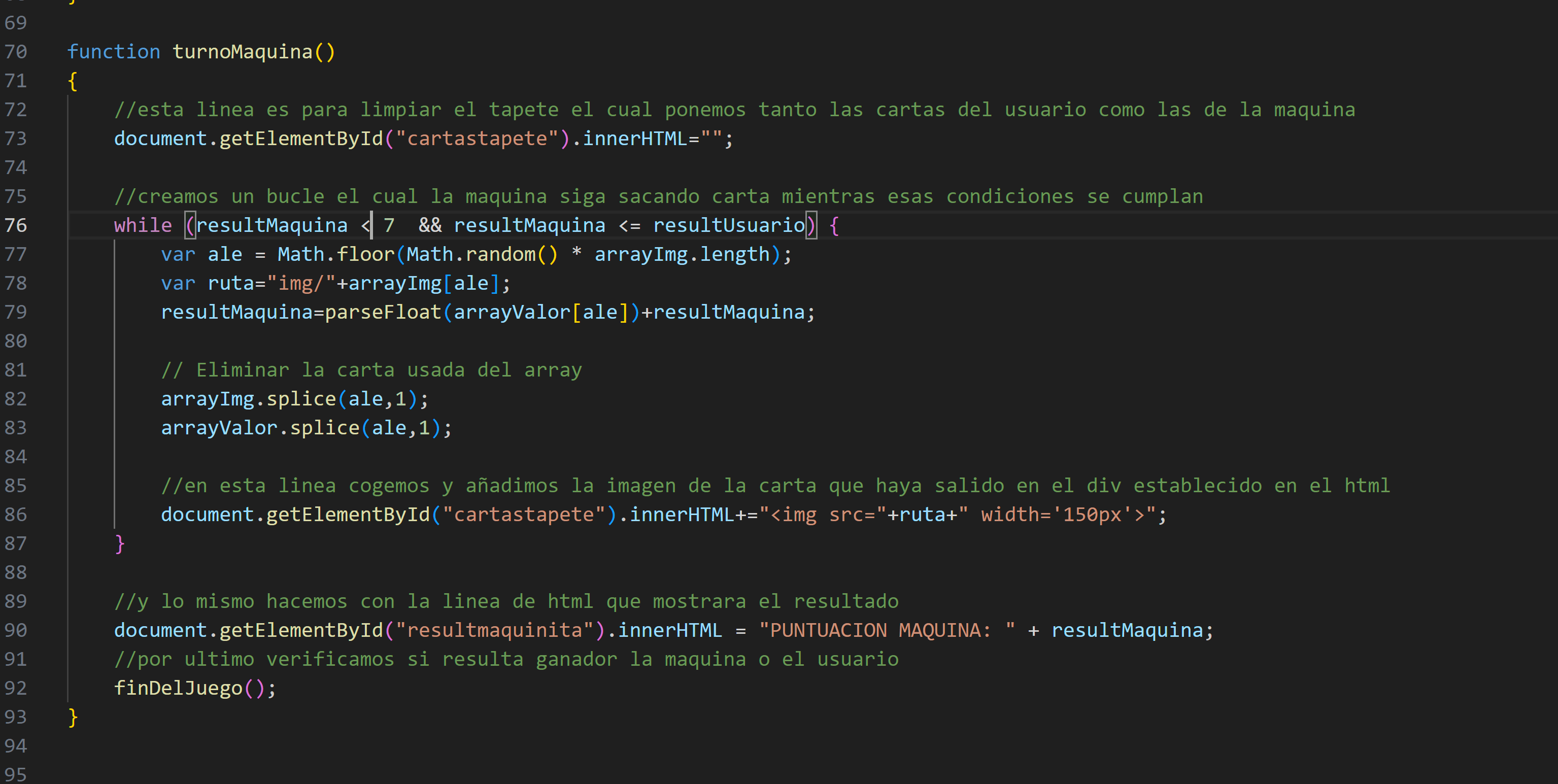
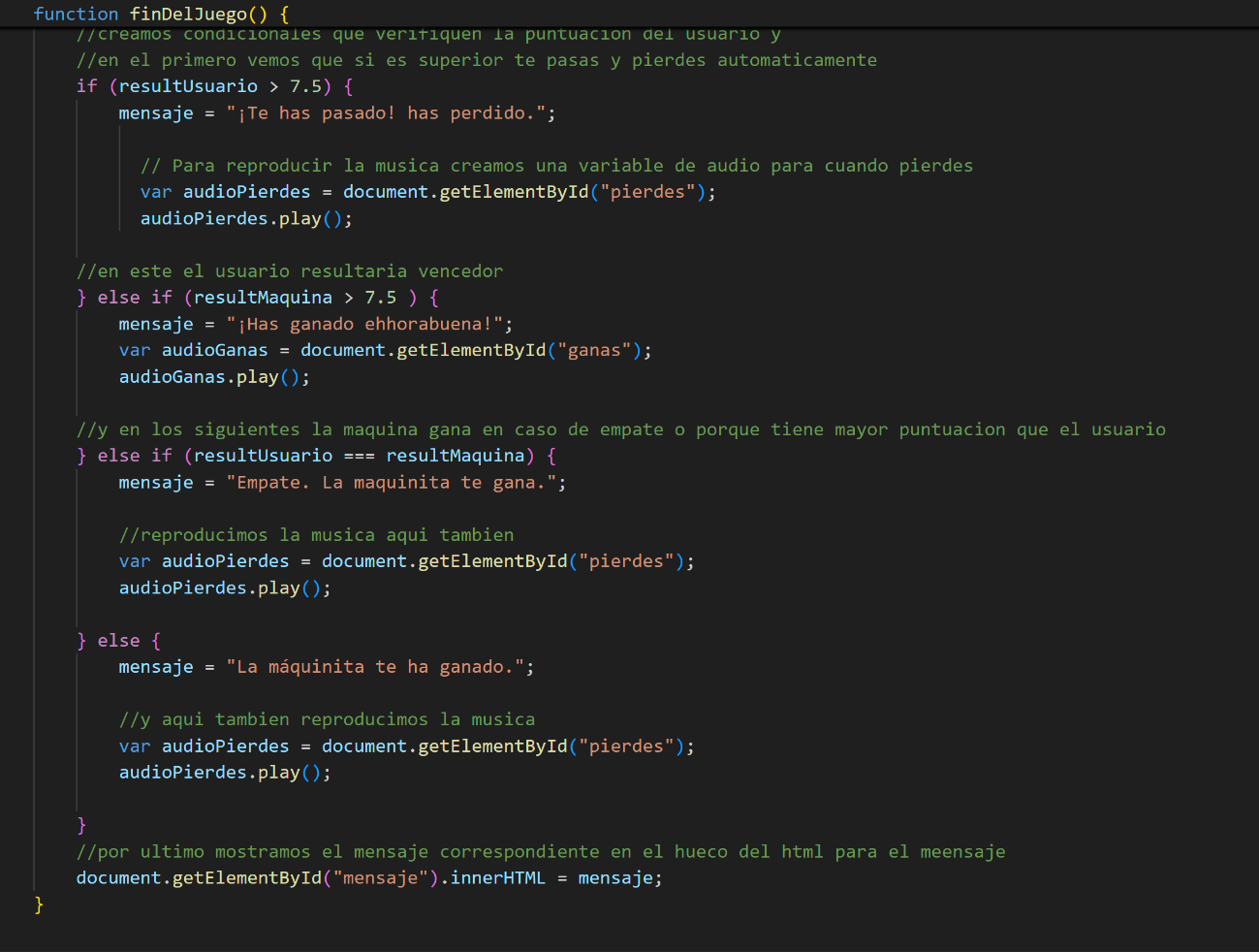
En la siguiente función la de plantarse como vemos es muy parecida a la de puedo seguir lo cual somo deshabilitamos ambos botones y hacemos la llamada a la función “turnoMaquina” que desglosaremos más adelante y que da paso a la máquina.



## 3.4.- FUNCIONAMIENTO DE LAS FUNCIONES TURNO DE MAQUINA Y FIN DE JUEGO

Por ultimo explicamos el funcionamiento de las ultimas funciones que anteriormente avanzamos que son las del “turnoMaquina” y las de “finDelJuego”, con la del turno de maquina vemos que al comienzo limpiamos el <div> donde se mostraran después las cartas, he cogido el mismo div para que sea más homogéneo y no ocupen las cartas dos hileras y quede todo más estético, después de eso vemos como he implementado un bucle que como condiciones tiene que si es menor a 7 o menor e igual a la puntuación del usuario siga tirando, ósea que salga otra carta aleatoria y la muestre en pantalla, el resto de variables ya serian iguales a la función de pedir pero con la salvedad que esta función (al ser el último turno y acabar el juego si o si ) termia con la llamada a la función “finDelJuego”.

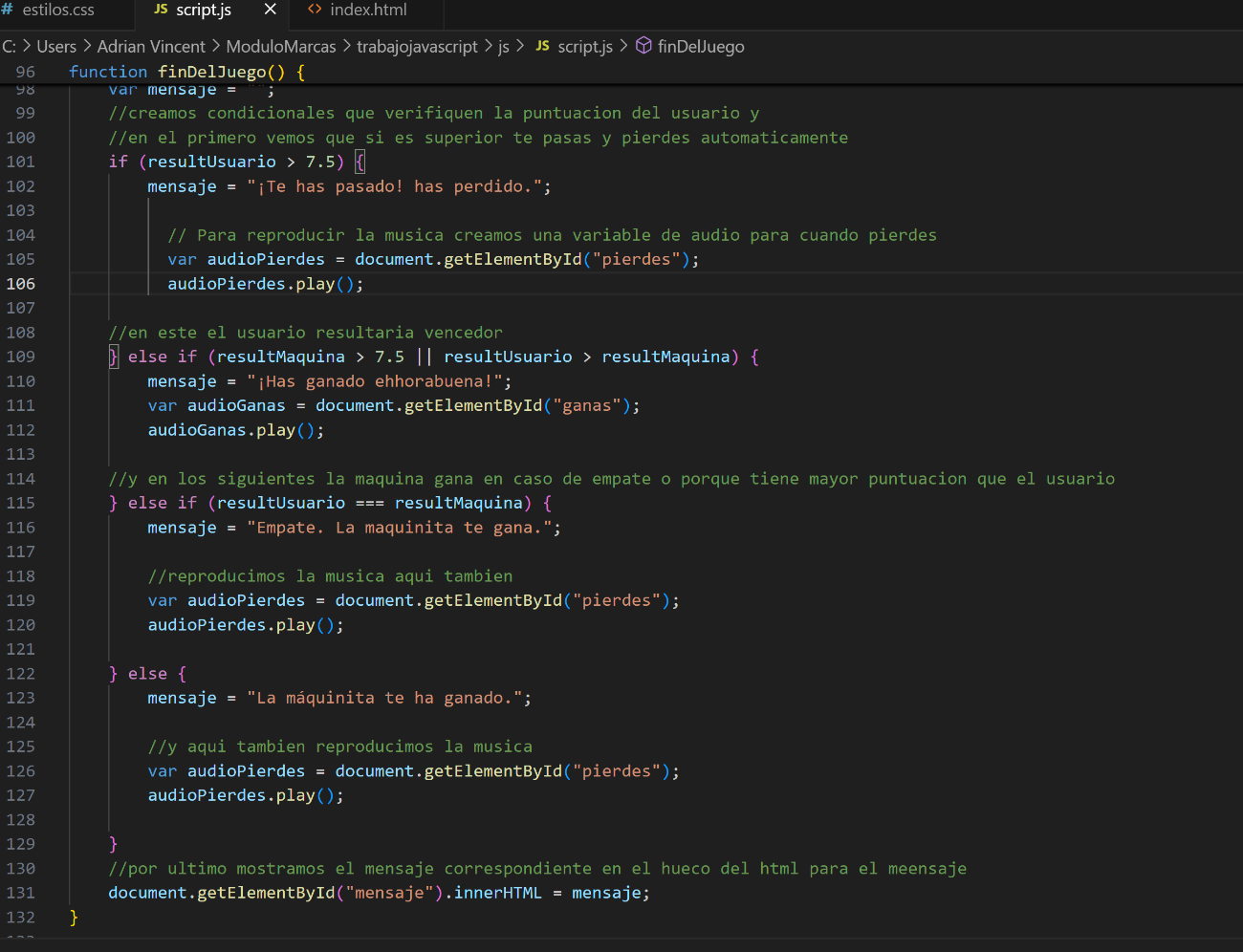
Mas adelante se pretende modificar ciertas funciones y añadir música en caso de ganar, perder o como inicio del juego y a su vez implementar diferentes gifs en caso de que se gane o se pierda.



## 

## 3.5.- IMPLEMENTACION DE LOS SONIDOS DE DERROTA Y VICTORIA

Como queríamos anteriormente implementamos el audio en la función de “finDeJuego” de manera que si pierdes en todas las posibilidades que tienes de perder automáticamente se reproducirá la música de derrota y por el contrario cuando ganas se reproducirá la de victoria. Dejo una captura de cómo se quedó finalmente el código y como lo implementamos.



# 4.- INTERFAZ Y EXPERIENCIA DEL USUARIO

# 5.- CONCLUSION

# -BIBLIOGRAFIA-

(Universidad Europea)

<https://universidadeuropea.com/blog/programacion-orientada-objetos/#:~:text=%C2%BFCu%C3%A1les%20son%20los%20principales%20lenguajes,%2C%20Smalltalk%2C%20PHP%20o%20Python>.

(Ionos)

<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/programacion-imperativa/>

(Universidad de Oviedo)

<https://www.atc.uniovi.es/telematica/2ac/Transparencias/T02-Programacion-Procedural.pdf>

(Hiberus)

<https://www.atc.uniovi.es/telematica/2ac/Transparencias/T02-Programacion-Procedural.pdf>

(Lucidchart)

<https://www.lucidchart.com/pages/es/que-es-un-modelo-de-base-de-datos>

(OpenAI (chatGPT))

<https://chat.openai.com/>

(Google Gemini)

<https://gemini.google.com/?hl=es-ES>

(Microsoft Copilot)

<https://copilot.microsoft.com/>

(*Todo el trabajo lo he llevado a cabo a través de la síntesis entendimiento y contraste entre no solo las diversas fuentes de información si no también con la ayuda de las varias IAs, siempre con mi comprensión de la información y aplicando mi criterio sobre el resultado de dicha información*)