DESARROLLO DE APLICACIONES EN INTERNET

LABORATORIO N° 3

JAVASCRIPT

CODIGO DEL CURSO:

Alumno(s)	*Nota*
Ccama Apaza Gabriel Anderson	

Grupo	*D*
Ciclo	*III*
Fecha de entrega	*21/09/2023*

I.- OBJETIVOS:

- Conocer los aspectos básicos para el uso de JavaScript.
- Utilizar JavaScript en una página HTML.
- Entender la estructura de Javascript.

II.- SEGURIDAD:

	**Advertencia:** **En este laboratorio está prohibida la manipulación del hardware, conexiones eléctricas o de red; así como la ingestión de alimentos o bebidas.**

III.- FUNDAMENTO TEÓRICO:

Revise sus diapositivas del tema antes del desarrollo del laboratorio.

IV.- NORMAS EMPLEADAS:

No aplica

V.- RECURSOS:

- En este laboratorio cada alumno trabará con un equipo con Windows
- Visual Studio Code
- Node JS
- Extensiones:
    - Prettier
    - Quokka.js
    - LiveServer

VI.- METODOLOGÍA PARA EL DESARROLLO DE LA TAREA:

- El desarrollo del laboratorio es individual.

VII.- PROCEDIMIENTO:

MARCO TEORICO

JavaScript es un lenguaje de programación interpretado. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas aunque existe una forma de JavaScript del lado del servidor. Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

JavaScript se diseñó con una sintaxis similar al C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo Java y JavaScript no están relacionados y tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

Tradicionalmente se venía utilizando en páginas web HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. JavaScript se interpreta en el agente de usuario, al mismo tiempo que las sentencias van descargándose junto con el código HTML.

ACTIVIDADES

USO EN NODE JS

Seguir las instrucciones del docente e iniciar con el desarrollo del laboratorio.

1. **Tipos de Datos**

```
/*
Ejemplos de
tipos de datos
en JavaScript
*/
//Tipo de dato string
var nombre = "Carlos";
console.log(nombre);

//Tipo de dato numerico
var numero = 1000;
console.log(numero);

//Tipo de dato object
var objecto = {
    nombre : "Juan",
    apellido : "Perez",
    telefono : "55443322"
}
console.log(objecto);
```

```
Reveal in value explorer
Carlos at nombre quokka.js:7:1

Reveal in value explorer
1000 at numero quokka.js:11:1

Reveal in value explorer
{ nombre: 'Juan', apellido: 'Perez', telefono: '55443322' }
  at objeto quokka.js:19:1
```

2. **Tipos de Datos(parte 2)**

```js
//Tipo de dato boolean (true, false)
var bandera = false;
console.log(typeof bandera);

//Tipo de dato function
function miFuncion(){}
console.log(typeof miFuncion);

//Tipo de dato Symbol
var simbolo = Symbol("mi simbolo");
console.log(typeof simbolo);

//Tipo clase es una function
class Persona{
    constructor(nombre, apellido){
        this.nombre = nombre;
        this.apellido = apellido;
    }
}
console.log(typeof Persona);

//Tipo undefined
var x;
console.log(typeof x);

x = undefined;
console.log(typeof x);

//null = ausencia de valor
var y = null;
console.log(typeof y);
```

```
Reveal in value explorer
boolean at typeof bandera quokka.js:3:1

Reveal in value explorer
function at typeof miFuncion quokka.js:7:1

Reveal in value explorer
symbol at typeof simbolo quokka.js:11:1

Reveal in value explorer
function at typeof Persona quokka.js:20:1

Reveal in value explorer
undefined at typeof x quokka.js:24:1

Reveal in value explorer
undefined at typeof x quokka.js:27:1

Reveal in value explorer
object at typeof y quokka.js:31:1
```

3. Arreglos y Datos Vacios

```
var y = null;
console.log(typeof y);

//arreglo en JavaScript
var autos = ['BMW','Audi','Volvo'];
console.log(autos);
console.log(typeof autos);

//Cadena vacia (empty string)
var z = '';
console.log(z);
console.log(typeof z);
```

```
Reveal in value explorer
object at typeof y quokka.js:2:1

Reveal in value explorer
[ 'BMW', 'Audi', 'Volvo' ]
  at autos quokka.js:6:1

Reveal in value explorer
object at typeof autos quokka.js:7:1

Reveal in value explorer
[empty string] at z quokka.js:11:1

Reveal in value explorer
string at typeof z quokka.js:12:1
```

4. **Concatenar Caracteres**

```
var nombre = 'Juan';
var apellido = 'Perez';

var nombreCompleto = nombre + ' ' + apellido;
console.log(nombreCompleto);

var nombreCompleto2 = 'Carlos' + ' ' + 'Lara';
console.log(nombreCompleto2);

var x = nombre + 2 + 4;
console.log(x);

x = nombre + (2 + 4);
console.log(x);

x = 2 + 4 + nombre;
console.log(x);
```

```
Reveal in value explorer
Juan Perez at nombreCompleto quokka.js:5:1

Reveal in value explorer
Carlos Lara at nombreCompleto2 quokka.js:8:1

Reveal in value explorer
Juan24 at x quokka.js:11:1

Reveal in value explorer
Juan6 at x quokka.js:14:1

Reveal in value explorer
6Juan at x quokka.js:17:1
```

5. **Concatenar Strings**

```
let nombre;
nombre = "Juan";
console.log( nombre );

const apellido = "Perez";
//apellido = "Lara"; Error no se puede cambiar el valor de una constante
```

```
Assignment to constant variable.
  at quokka.js:6:1

Reveal in value explorer
Juan at nombre quokka.js:3:1
```

6. **Reglas de declaración de Variables.**

```
let nombreCompleto = "Juan Perez";
let nombrecompleto = "Carlos Lara";
console.log( nombreCompleto );
console.log( nombrecompleto );

let a1nombreCompleto;
let _nombreCompleto;
let $nombreCompleto;
//let 1nombreCompleto; no está permitido iniciar el nombre de una variable con numeros

let ruptura = 10;
```

```
Failed to instrument quokka.js
   7 | let _nombreCompleto;
   8 | let $nombreCompleto;
>  9 | let 1nombreCompleto; //no está permitido iniciar el nombre de una variable con
numeros
     |       ^ SyntaxError: Identifier directly after number (9:6)
  10 |
  11 | let ruptura = 10;
```

7. **Operadores Aritmeticos**

```
let a = 3, b = 2;
let z = a + b;
console.log("Resultado de la suma: " + z );

z = a - b;
console.log("Resultado de la resta: " + z);

z = a * b;
console.log( "Resultado de la mult:" + z);

z = a / b;
console.log( "Resultado de la division:" + z);

z = a % b;//residuo de la division
console.log( "Resultado de operacion modulo (residuo):" + z);
z = a ** b;
console.log( "Resultado de operador exponente:" + z);
```

```
Reveal in value explorer
Resultado de la suma: 5
  at 'Resultado de la suma: ' + z quokka.js:3:1

Reveal in value explorer
Resultado de la resta: 1
  at 'Resultado de la resta: ' + z quokka.js:6:1

Reveal in value explorer
Resultado de la mult: 6
  at 'Resultado de la mult: ' + z quokka.js:9:1

Reveal in value explorer
Resultado de la division: 1.5
  at 'Resultado de la division: ' + z quokka.js:12:1

Reveal in value explorer
Resultado de operacion modulo (residuo): 1
  at 'Resultado de operacion modulo (residuo)... quokka.js:15:1

Reveal in value explorer
Resultado de operador exponente: 9
  at 'Resultado de operador exponente: ' + z quokka.js:18:1
```

8. Incremento y Decremento de variables

```
let a = 3, b = 2;
let z = a + b;

//Incremento
//Pre-incremento (el operador ++ antes de la variable)
z = ++a;
console.log(a);
console.log(z);

//Post-incremento (el operador ++ despues de la variable)
z = b++;
console.log(b);
console.log(z);

//Decremento
//Predecremento
z = --a;
console.log(a);
console.log(z);

//Postdecremento
z = b--;
console.log(b);
console.log(z);
```

```
Reveal in value explorer
4 at a quokka.js:7:1

Reveal in value explorer
4 at z quokka.js:8:1

Reveal in value explorer
3 at b quokka.js:12:1

Reveal in value explorer
2 at z quokka.js:13:1

Reveal in value explorer
3 at a quokka.js:18:1

Reveal in value explorer
3 at z quokka.js:19:1

Reveal in value explorer
2 at b quokka.js:23:1

Reveal in value explorer
3 at z quokka.js:24:1
```

9. Precedencia de Operadores

```
let a = 3, b = 2, c = 1, d = 4;

let z = a * b + c / d;
console.log(z);

z = c + a * b / d;
console.log( z );

z = (c + a) * b / c;
console.log(z);
```

```
Reveal in value explorer
6.25 at z quokka.js:4:1

Reveal in value explorer
2.5 at z quokka.js:7:1

Reveal in value explorer
8 at z quokka.js:10:1
```

10. Operadores de Comparación

```
let a = 3, b = 2, c = "3";

let z = a == c; // se revisa el valor sin importar el tipo
console.log(z);

z = a === c;// revisa los valores pero tambien los tipos
console.log(z);
```

```
Reveal in value explorer
true at z quokka.js:4:1

Reveal in value explorer
false at z quokka.js:7:1
```

11. Operadores de Diferencia

```
let a = 3, b = 2, c = 3;

let z = a != c; // se revisa el valor sin importar el tipo
console.log(z);

z = a !== c;// revisa los valores pero tambien los tipos
console.log(z);
```

```
Reveal in value explorer
false at z quokka.js:4:1

Reveal in value explorer
false at z quokka.js:7:1
```

12. Operador AND

```
let a = 15;
let valMin = 0, valMax = 10;

if( a >= valMin && a <= valMax ){
    console.log("Dentro de rango");
}
else{
    console.log("Fuera de rango");
}
```

```
Reveal in value explorer
Fuera de rango at quokka.js:8:5
```

13. Operador OR

```
//Ejemplo OR (||), regresa true si cualquier operando es true
let vacaciones = false, diaDescanso = true;
if( vacaciones || diaDescanso ){
    console.log("Padre puede asistir al juego del hijo");
}
else{
    console.log("El padre está ocupado");
}
```

```
Reveal in value explorer
Padre puede asistir al juego del hijo
  at quokka.js:4:5
```

14. Función NaN

```
let miNumero = "17";

let edad = Number(miNumero);
console.log( edad );

if( isNaN(edad)){
    console.log("No es un número");
}
else{
    if(edad >= 18){
        console.log("Puede votar");
    }
    else{
        console.log("Muy joven para votar");
    }
}

if( isNaN(edad)){
    console.log("No es un número");
}
else{
    let resultado = (edad >= 18)? "Puede votar" : "Muy joven para votar";
    console.log( resultado );
}
```

```
Reveal in value explorer
17 at edad quokka.js:4:1

Reveal in value explorer
Muy joven para votar at quokka.js:14:9

Reveal in value explorer
Muy joven para votar at resultado quokka.js:23:5
```

15. Condicional IF

```
let numero = 2;

if( numero == 1 ){
    console.log("Número uno");
}
else if( numero == 2 ){
    console.log("Número dos");
}
else if( numero == 3 ){
    console.log("Número tres");
}
else if( numero == 4 ){
    console.log("Número cuatro");
}
else{
    console.log("Número desconocido");
}
```

```
Reveal in value explorer
Número dos at quokka.js:7:5
```

16. Condiciona Switch

```
let numero = 1;

let numeroTexto = 'Valor desconocido';

switch( numero ){
    case 1:
        numeroTexto = 'Número uno';
        break;
    case 2:
        numeroTexto = 'Número dos';
        break;
    case 3:
        numeroTexto = 'Número tres';
        break;
    case 4:
        numeroTexto = 'Número cuatro';
        break;
    default:
        numeroTexto = 'Caso no encontrado';
}

console.log(numeroTexto);
```

```
Reveal in value explorer
Número uno at numeroTexto quokka.js:21:1
```

17. Ciclo While

```
let contador = 0;

while( contador < 3 ){
    console.log(contador);
    contador++;
}
console.log("Fin ciclo while");
```

```
Reveal in value explorer
0 at contador quokka.js:4:5

Reveal in value explorer
1 at contador quokka.js:4:5

Reveal in value explorer
2 at contador quokka.js:4:5

Reveal in value explorer
Fin ciclo while at quokka.js:7:1
```

18. Ciclo For

```
for(let contador = 0; contador < 3 ; contador++ ){
    console.log(contador);
}
console.log("Fin ciclo for");
```

```
Reveal in value explorer
0 at contador quokka.js:2:5

Reveal in value explorer
1 at contador quokka.js:2:5

Reveal in value explorer
2 at contador quokka.js:2:5

Reveal in value explorer
Fin ciclo for at quokka.js:4:1
```

19. Arreglos en Javascript

```
//let autos = new Array('BMW','Mercedes Benz','Volvo');
const autos = ['BMW','Mercedes Benz','Volvo'];
console.log(autos);
```

```
Reveal in value explorer
[ 'BWM', 'Mercedes Benz', 'Volvo' ]
  at autos1 quokka.js:2:1

Reveal in value explorer
[ 'BWM', 'Mercedes Benz', 'Volvo' ]
  at autos2 quokka.js:4:1
```

20. Accediendo a elemento de un Array

```
//let autos = new Array('BMW','Mercedes Benz','Volvo');
const autos = ['BMW','Mercedes Benz','Volvo'];
console.log(autos);

console.log(autos[0]);
console.log(autos[2]);

for(let i = 0; i < autos.length; i++){
    console.log(i + ' : ' + autos[i] );
}
```

```
Reveal in value explorer
[ 'BWM', 'Mercedes Benz', 'Volvo' ]
  at autos quokka.js:2:1

Reveal in value explorer
BWM at autos[0] quokka.js:4:1

Reveal in value explorer
Volvo at autos[2] quokka.js:5:1

Reveal in value explorer
0 : BWM at i + ' : ' + autos[i] quokka.js:8:5

Reveal in value explorer
1 : Mercedes Benz at i + ' : ' + autos[i] quokka.js:8:5

Reveal in value explorer
2 : Volvo at i + ' : ' + autos[i] quokka.js:8:5
```

21. Modificando Valores de un Array

```
//let autos = new Array('BMW','Mercedes Benz','Volvo');
const autos = ['BMW','Mercedes Benz','Volvo'];
console.log(autos);

console.log(autos[0]);
console.log(autos[2]);

for(let i = 0; i < autos.length; i++){
    console.log(i + ' : ' + autos[i] );
}

autos[1] = 'MerecedesBenz';
console.log(autos[1]);

autos.push('Audi');
console.log(autos);
```

```
Reveal in value explorer
[ 'BWM', 'Mercedes Benz', 'Volvo' ]
  at autos quokka.js:3:1

Reveal in value explorer
BWM at autos[0] quokka.js:5:1

Reveal in value explorer
Volvo at autos[2] quokka.js:6:1

Reveal in value explorer
0 : BWM at i + ' : ' + autos[i] quokka.js:9:5

Reveal in value explorer
1 : Mercedes Benz at i + ' : ' + autos[i] quokka.js:9:5

Reveal in value explorer
2 : Volvo at i + ' : ' + autos[i] quokka.js:9:5

Reveal in value explorer
MercedesBenz at autos[1] quokka.js:13:1

Reveal in value explorer
[ 'BWM', 'MercedesBenz', 'Volvo', 'Audi' ]
  at autos quokka.js:16:1
```

22. Agregando Elementos a un array

```
//let autos = new Array('BMW','Mercedes Benz','Volvo');
const autos = ['BMW','Mercedes Benz','Volvo'];
console.log(autos);

console.log(autos[0]);
console.log(autos[2]);

for(let i = 0; i < autos.length; i++){
    console.log(i + ' : ' + autos[i] );
}

autos[1] = 'MerecedesBenz';
console.log(autos[1]);

autos.push('Audi');
console.log(autos);

console.log(autos.length);
autos[autos.length] = 'Cadillac';

console.log(autos);

autos[6] = 'Porshe';
console.log(autos);
```

```
Reveal in value explorer
[ 'BWM', 'Mercedes Benz', 'Volvo' ]
  at autos quokka.js:3:1

Reveal in value explorer
BWM at autos[0] quokka.js:5:1

Reveal in value explorer
Volvo at autos[2] quokka.js:6:1

Reveal in value explorer
0 : BWM at i + ' : ' + autos[i] quokka.js:9:5

Reveal in value explorer
1 : Mercedes Benz at i + ' : ' + autos[i] quokka.js:9:5

Reveal in value explorer
2 : Volvo at i + ' : ' + autos[i] quokka.js:9:5

Reveal in value explorer
MercedesBenz at autos[1] quokka.js:13:1

Reveal in value explorer
[ 'BWM', 'MercedesBenz', 'Volvo', 'Audi' ]
  at autos quokka.js:16:1

Reveal in value explorer
4 at autos.length quokka.js:18:1

Reveal in value explorer
[ 'BWM', 'MercedesBenz', 'Volvo', 'Audi', 'Cadillac' ]
  at autos quokka.js:20:1

Reveal in value explorer
[ 'BWM', 'MercedesBenz', 'Volvo', 'Audi', 'Cadillac', , 'Porshe' ]
  at autos quokka.js:23:1
```

23. Preguntar si es un arreglo

```
//let autos = new Array('BMW','Mercedes Benz','Volvo');
const autos = ['BMW','Mercedes Benz','Volvo'];
console.log(autos);

console.log(autos[0]);
console.log(autos[2]);

for(let i = 0; i < autos.length; i++){
    console.log(i + ' : ' + autos[i] );
}

autos[1] = 'MerecedesBenz';
console.log(autos[1]);

autos.push('Audi');
console.log(autos);

console.log(autos.length);
autos[autos.length] = 'Cadillac';

console.log(autos);

autos[6] = 'Porshe';
console.log(autos);

console.log(typeof autos);

console.log( Array.isArray(autos) );

console.log( autos instanceof Array);
```

```
Reveal in value explorer
object at typeof autos quokka.js:25:1

Reveal in value explorer
true at Array.isArray(autos) quokka.js:27:1

Reveal in value explorer
true at autos instanceof Array quokka.js:29:1
```

24. Funciones en Javascipt

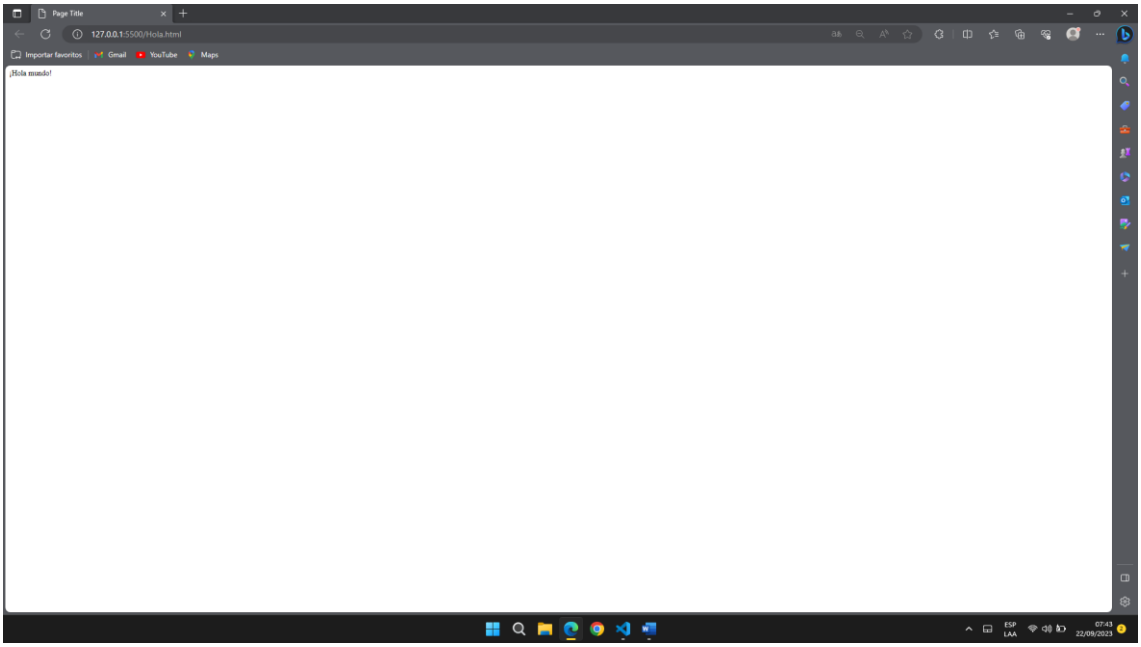
```
miFuncion(4, 2);

//Declaración de la función
function miFuncion(a, b){
    console.log("Suma: " + (a + b));
}

//Llamando a la función
miFuncion(2, 3);
```

```
Reveal in value explorer
Suma: 6 at 'Suma: ' + (a + b) quokka.js:5:5

Reveal in value explorer
Suma: 5 at 'Suma: ' + (a + b) quokka.js:5:5
```

USO EN HTML

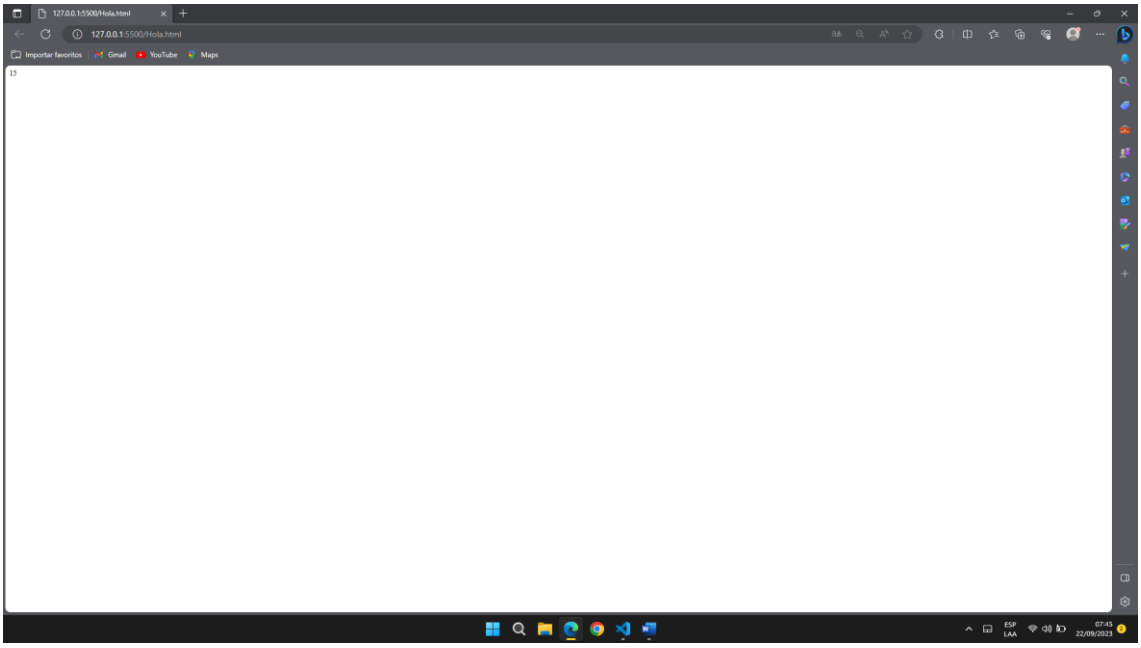
1. **En el editor de su preferencia, escriba en el archivo Hola.html el siguiente código y comente:**

```html
<!DOCTYPE html>
<html>

  <head>
    <meta charset="utf-8" />
    <title>Page Title</title>
    <script>
      document.write('¡Hola mundo!');
    </script>
  </head>

  <body>
  </body>

</html>
```



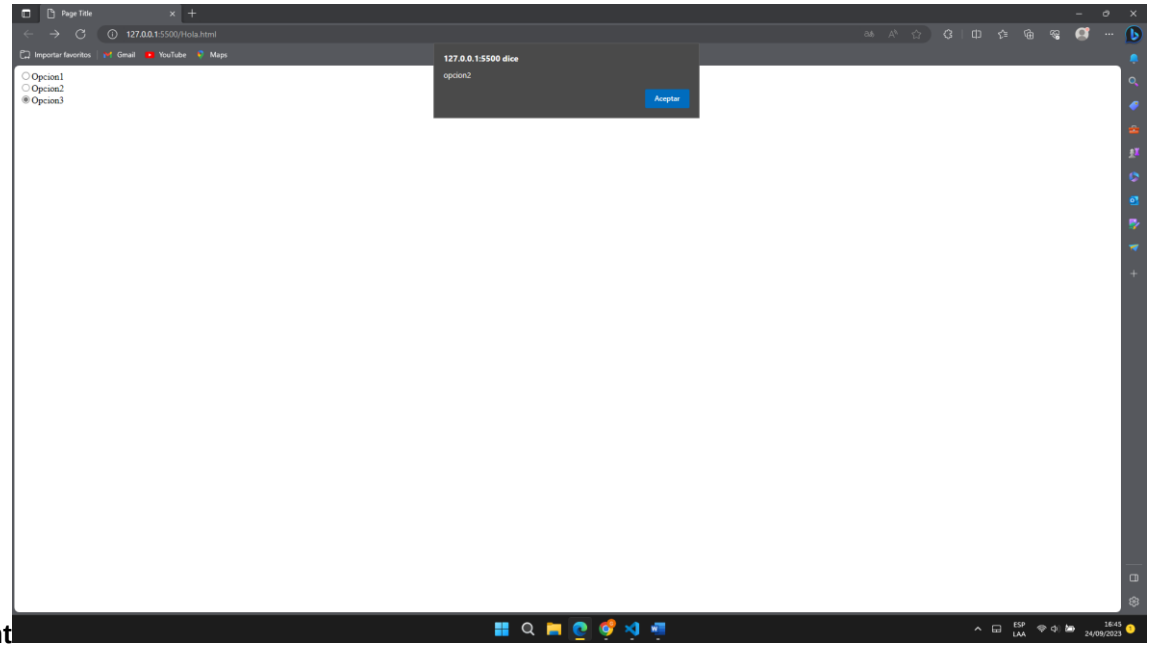
2. **Crea una función que se cargue al inicio y que muestre la suma de dos números.**

```html
<!DOCTYPE html>
<html>

  <head>
    <meta charset="utf-8" />
    <title>Page Title</title>
    <script>
      function sumaNumeros() {
        let num1 = 5;
        let num2 = 10;

        document.write(num1 + num2);
      }
    </script>
  </head>

  <body onload="sumaNumeros()">
  </body>

</html>
```



3. **Crea dos radiobuttons, cuando uno de ellos se seleccione que muestre un mensaje diciendo que opción se seleccionó.**

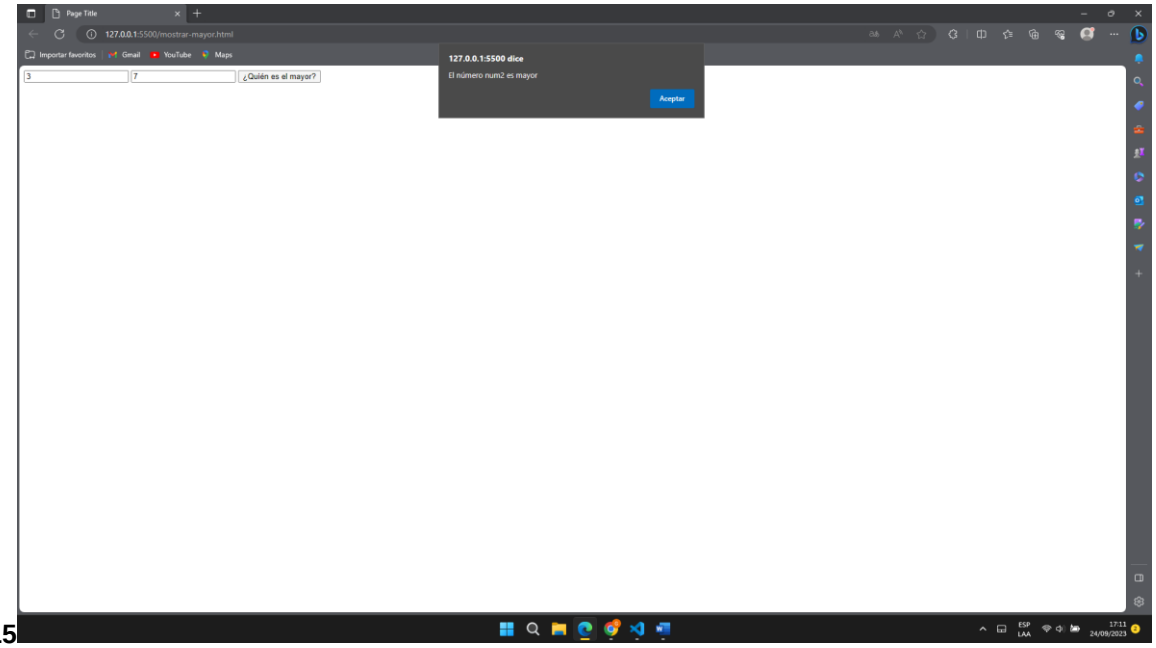
```html
<!DOCTYPE html>
<html>

  <head>
    <meta charset="utf-8" />
    <title>Page Title</title>
    <script>
      function muestraOpcion() {
        var opciones = document.getElementsByName("eleccion");
        console.log(opciones);
        for (let i = 0; i < opciones.length; i++) {
          if (opciones[i].checked) {
            alert(opciones[i].value);
          }
        }
      }
    </script>
  </head>

  <body>
    <input type="radio" name="eleccion" value="opcion1" checked onclick="muestraOpcion()">Opcion1<br/>
    <input type="radio" name="eleccion" value="opcion2" onclick="muestraOpcion()">Opcion2<br/>
    <input type="radio" name="eleccion" value="opcion3" onclick="muestraOpcion()">Opcion3<br/>
  </body>

</html>
```

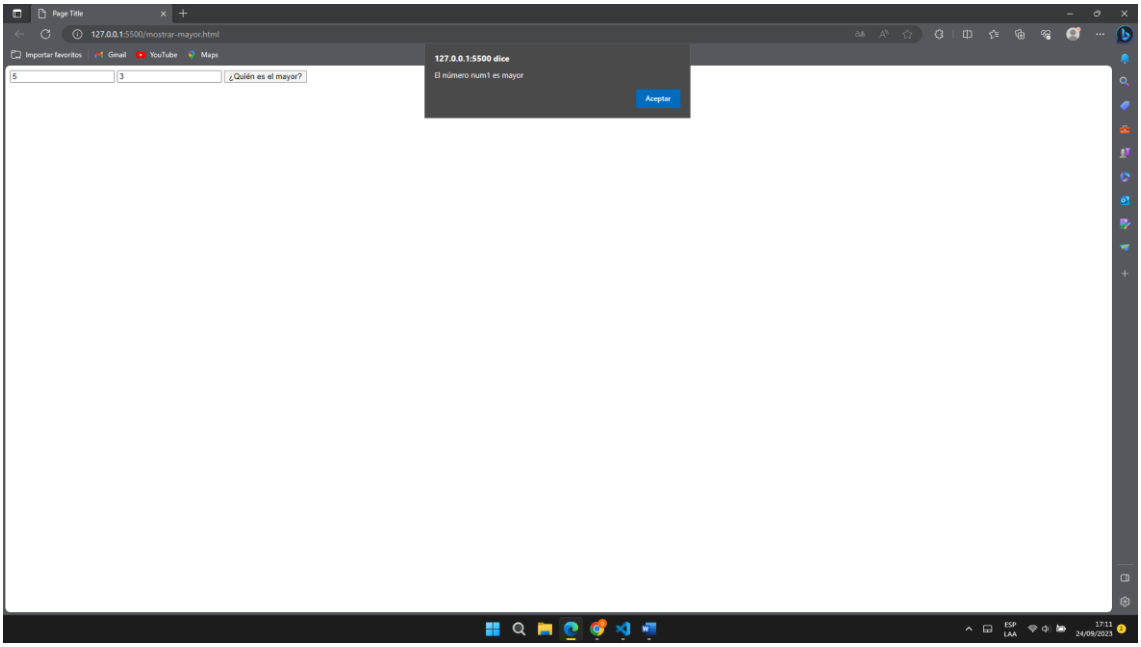


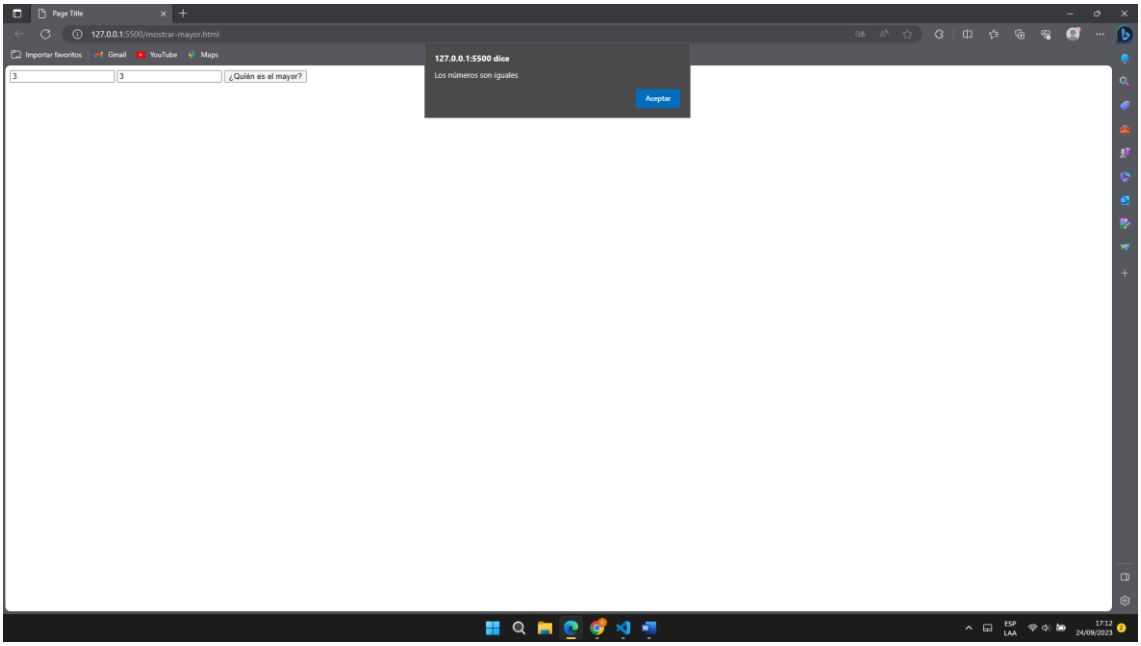
ht

4. **Crear dos input de números y un botón, al pulsar el botón, mostrar en un alert quien es el mayor de los dos:**

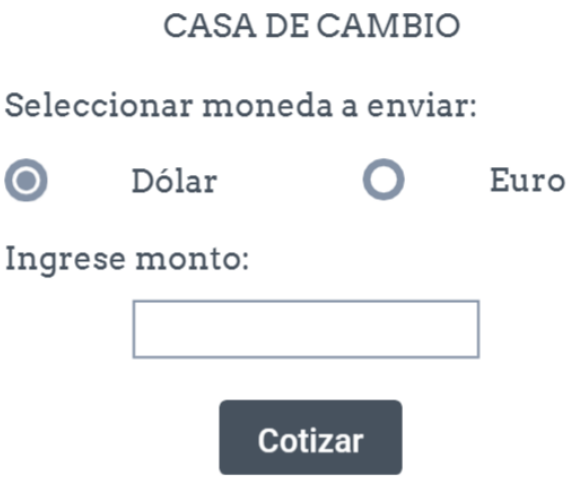
```html
<!DOCTYPE html>
<html>

  <head>
    <meta charset="utf-8" />
    <title>Page Title</title>
    <script>
      function mayorNum() {
        var num1 = parseInt(document.getElementById("num1").value);
        var num2 = parseInt(document.getElementById("num2").value);
        if (num1 && num2) {
          if (num1 >= num2) {
            if (num1 == num2) {
              alert("Los numeros son iguales")
            } else {
              alert("El num1 es mayor")
            }
          } else {
            alert("El num2 es mayor")
          }
        } else {
          alert("Uno de los numeros no se relleno")
        }
      }
    </script>
  </head>
  <body>
    <form>
      <input type="text" id="num1" />
      <input type="text" id="num2" />
      <button type="button" onclick="mayorNum()">¿Quien es el mayor?</button>
    </form>
  </body>
</html>
```



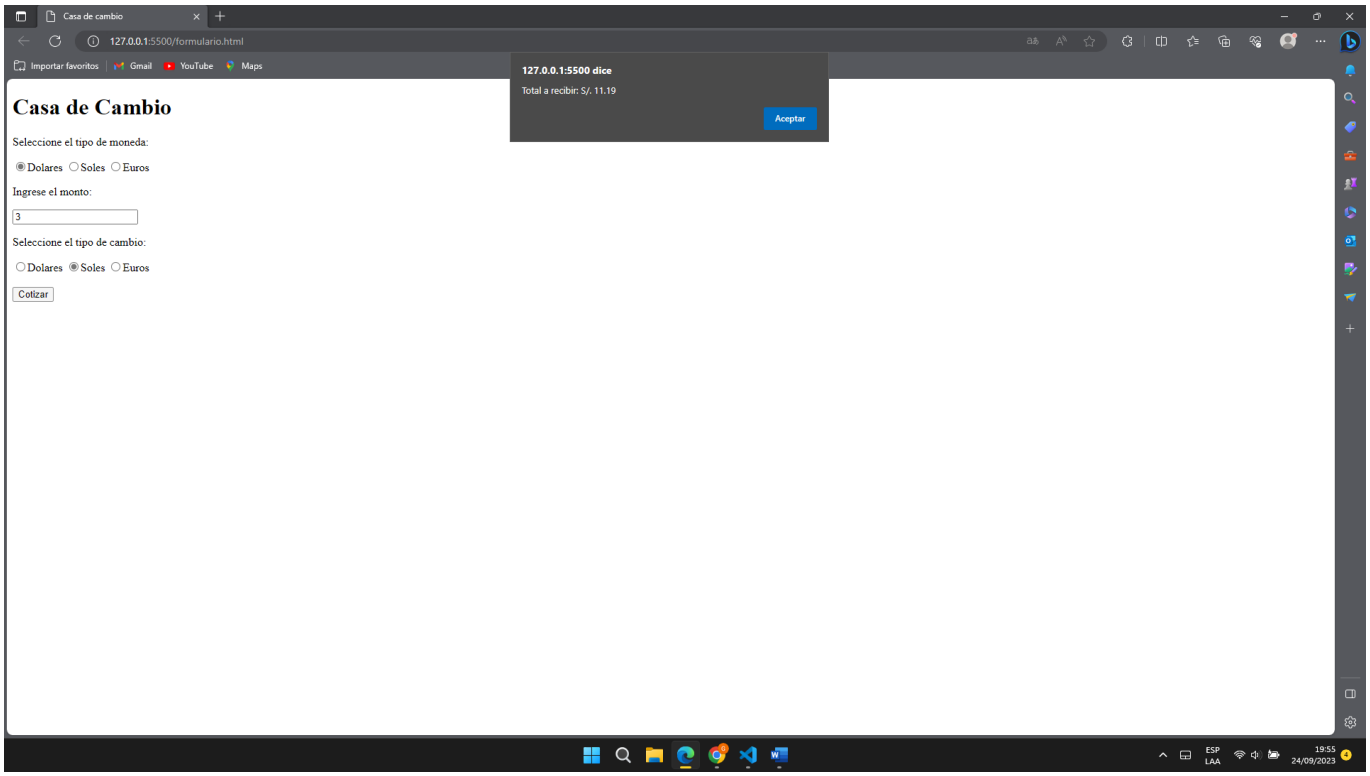


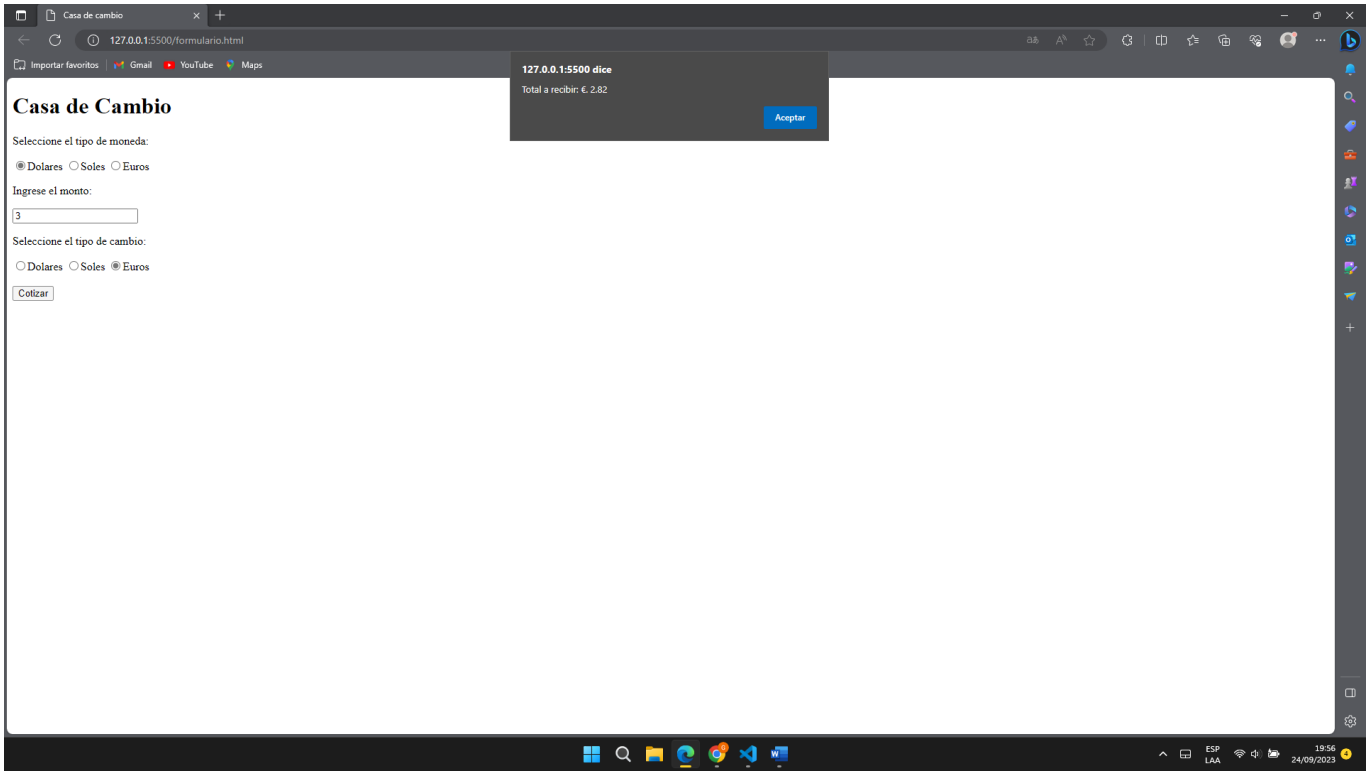


5. **Crear un formulario para una casa de cambio que me permita seleccionar el tipo de moneda (dólar, soles, euros) a cambiar a (dólar, soles, euros), acompañado de un botón que al ser seleccionado muestre una alerta con el total a recibir.**

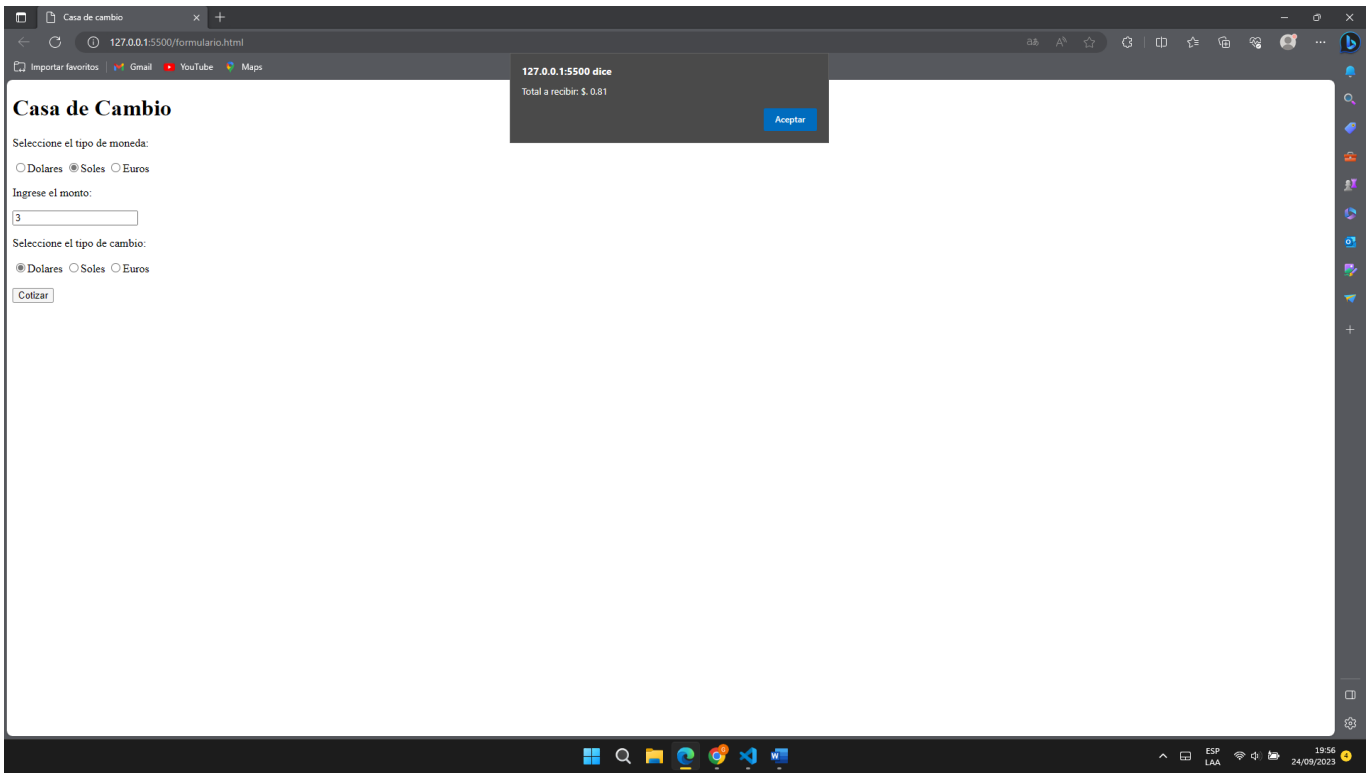


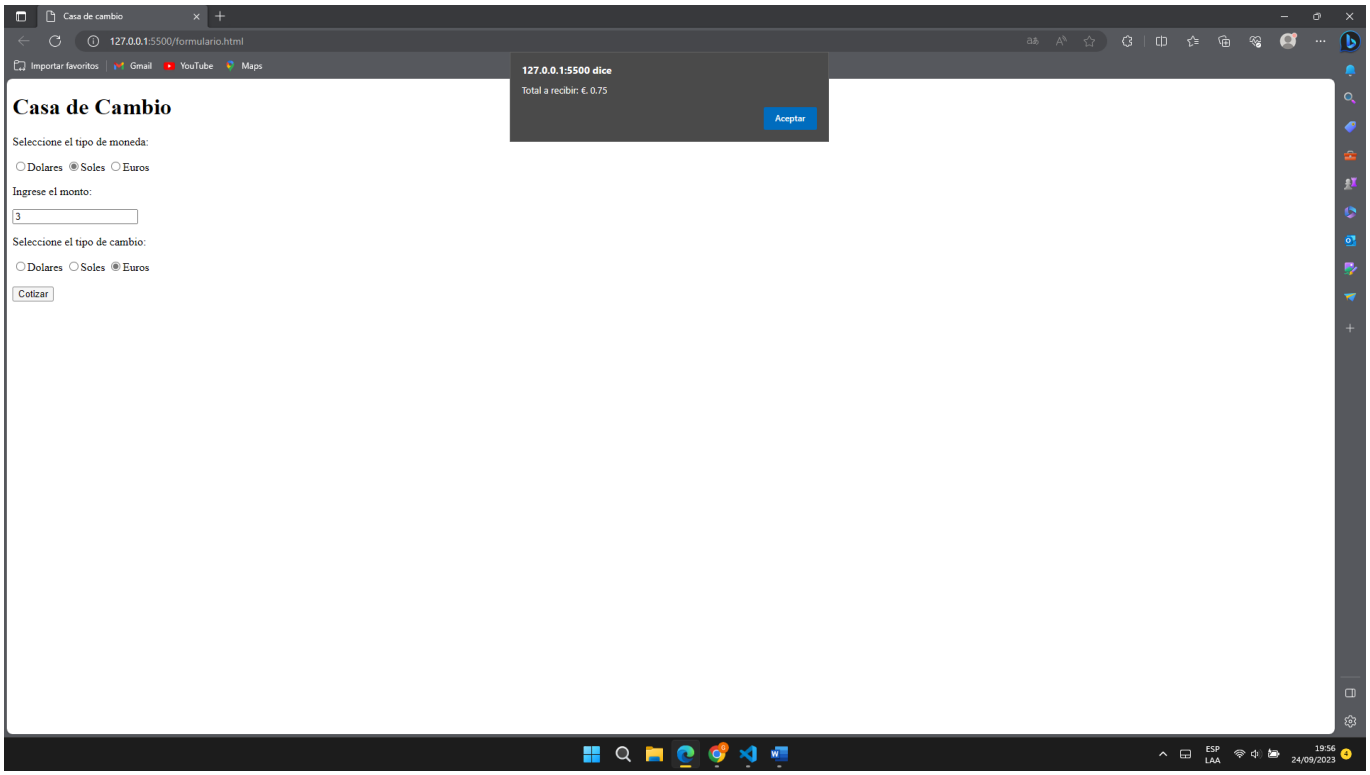
Para la moneda en dólares



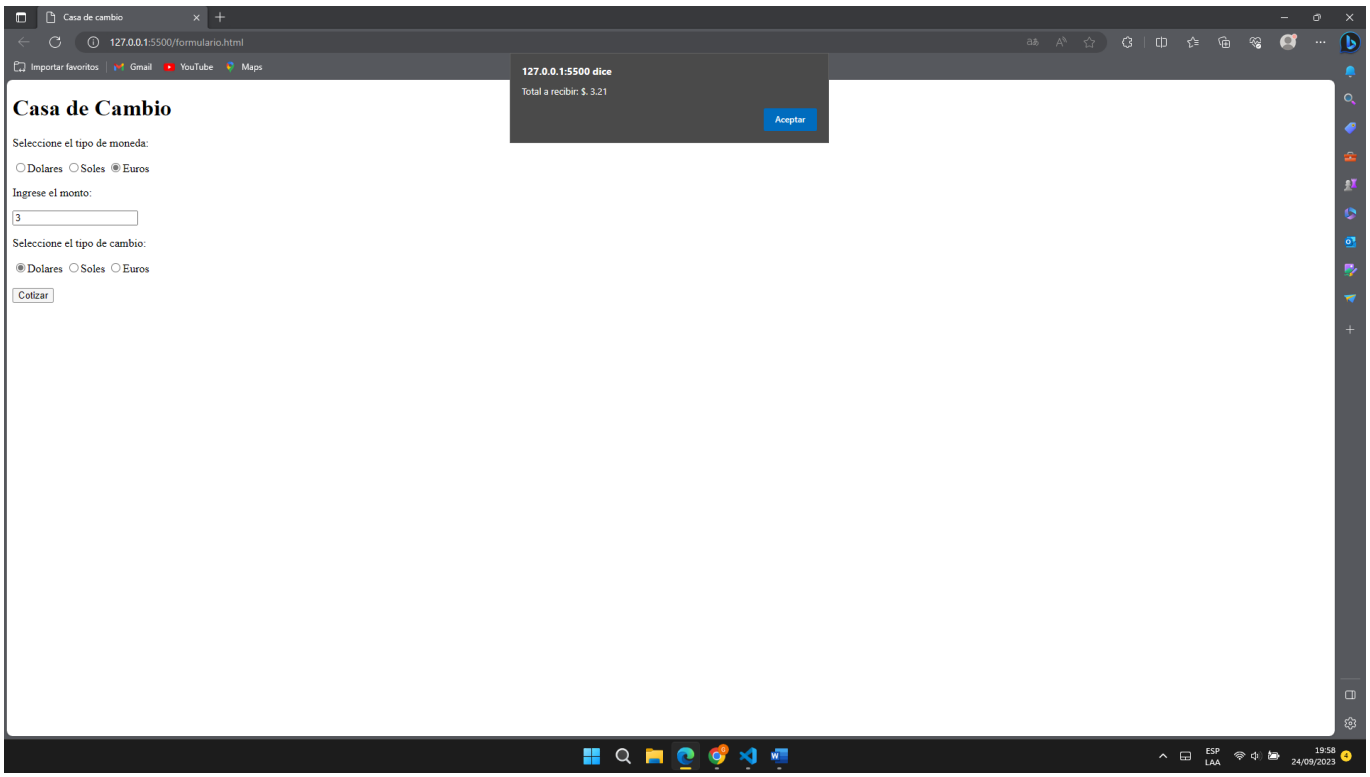


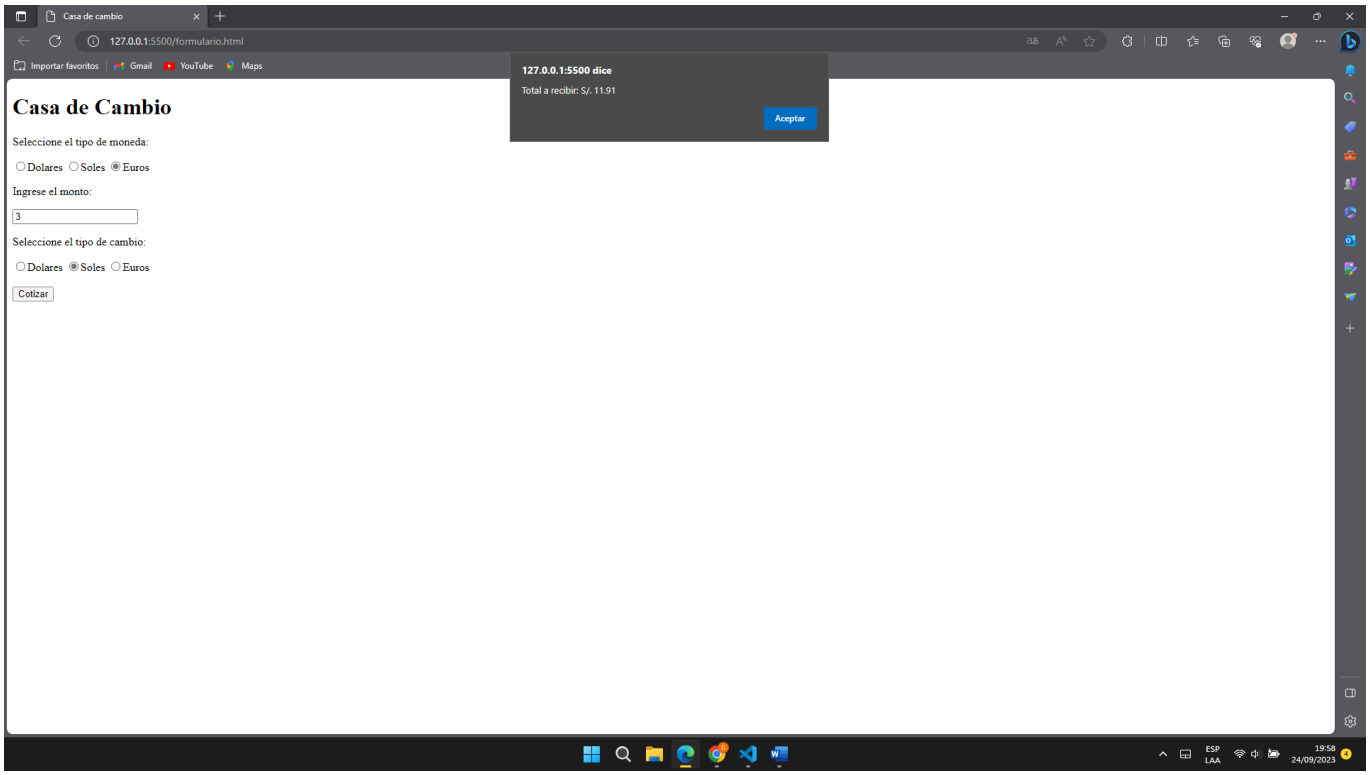
Para la moneda en soles



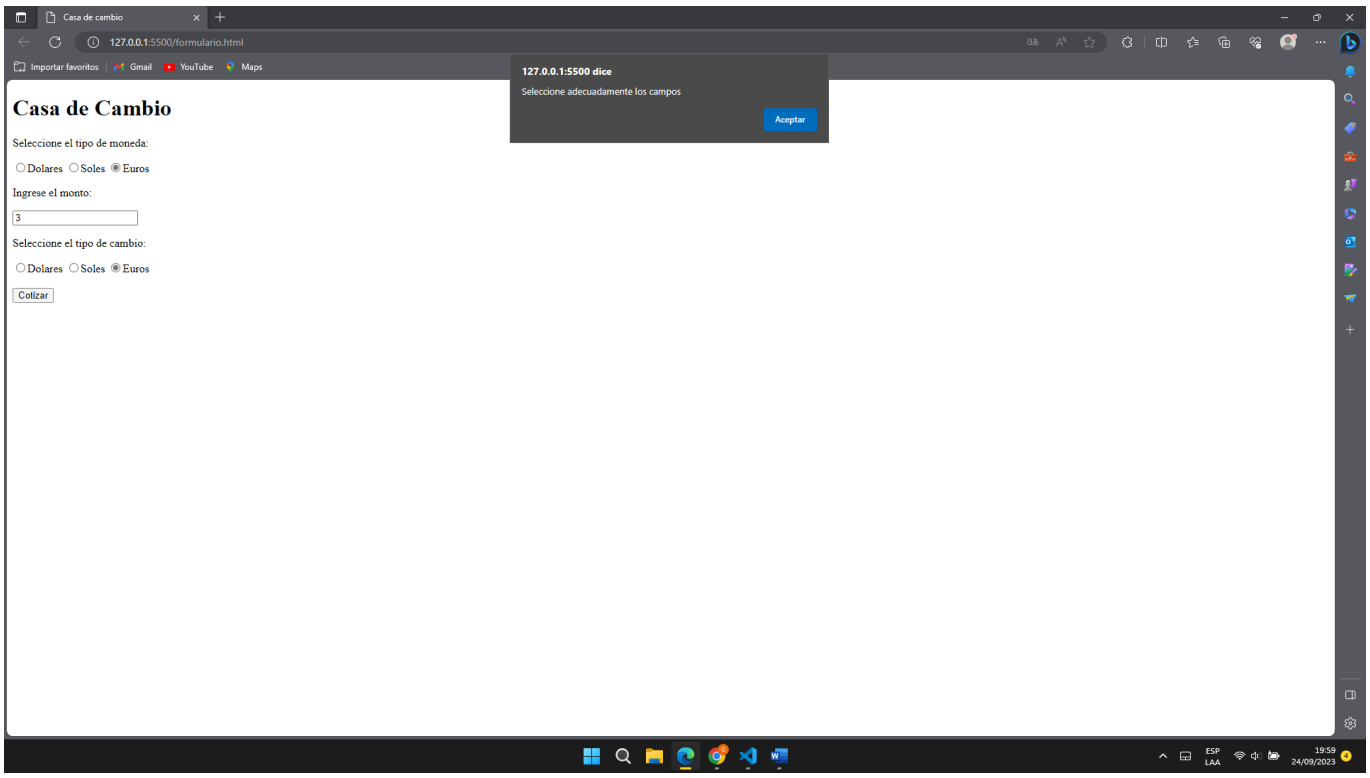


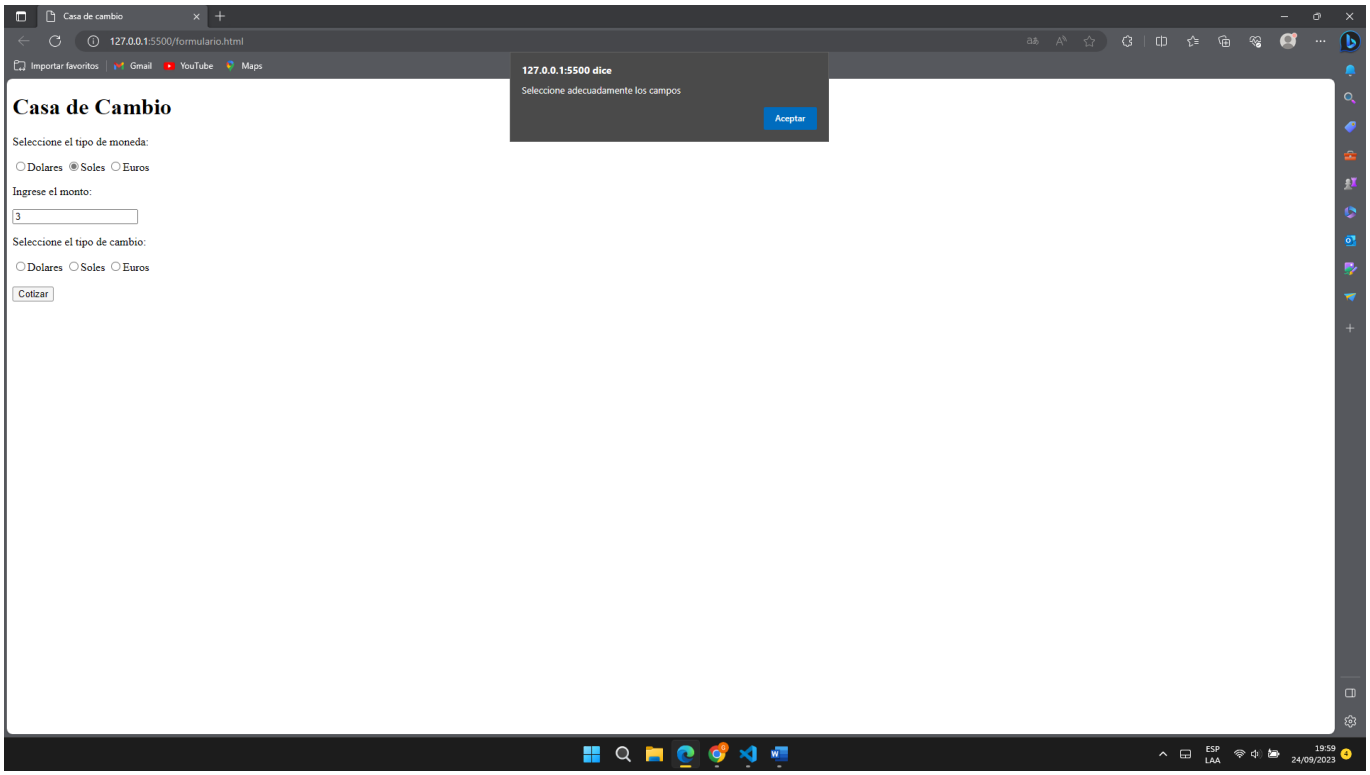
Para la moneda en euros





Cuando se selecciona el mismo tipo de Moneda y Cambio o cuando no se selecciona un campo





TRABAJO PROPUESTO

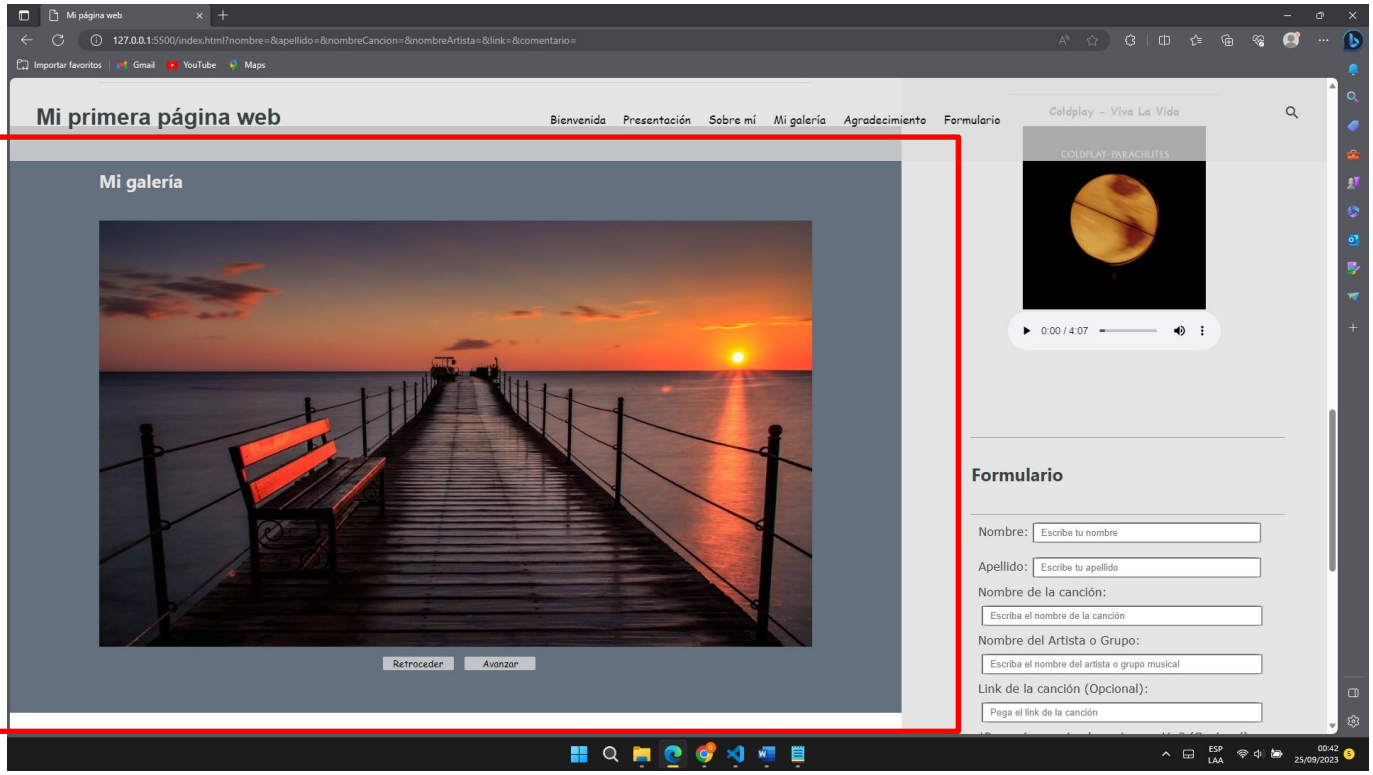
Aplicar lo aprendido el día de hoy, a su presentación personal realizada en la semana 1 y 2.

Se agregó código en JavaScript para crear un Slider de imágenes de manera automática y manual

index.html | main.js | style.css

main.js > enviar

```
// SLIDER
window.addEventListener('load', function(){
    var botonRetroceder = document.querySelector('#retroceder');
    var botonAvanzar = document.querySelector('#avanzar')
    var imagen = document.querySelector('#slider')

    var imagenes = [];
    imagenes[0] = 'img/imagen1.jpg';
    imagenes[1] = 'img/imagen2.jpg';
    imagenes[2] = 'img/imagen3.jpg';
    imagenes[3] = 'img/imagen4.jpg';
    imagenes[4] = 'img/imagen5.jpg';
    imagenes[5] = 'img/imagen6.jpg';
    imagenes[6] = 'img/imagen7.jpg';

    var posicionActualAuto = 0;
    var posicionActualManual = 0;
    var intervalo;

    function cambiarImagenes(){
        imagen.src = imagenes[posicionActualAuto];

        if (posicionActualAuto < imagenes.length - 1){
            posicionActualAuto++;
        }
        else{
            posicionActualAuto = 0;
        }
    }
    // iniciamos el intervalo de tiempo automáticamente
    intervalo = setInterval(cambiarImagenes, 3000);

    function anvanzarImagen(){
        // detenemos el intervalo de tiempo antes de cambiar a manual
        clearInterval(intervalo)

        if (posicionActualManual >= imagenes.length - 1){
            posicionActualManual = 0;
        } else{
            posicionActualManual++;
        }
        imagen.src = imagenes[posicionActualManual];

        // reiniciamos el intervalo después del cambio manual
        intervalo = setInterval(cambiarImagenes, 3000);
    }

    function retrocederImagen(){
        // detenemos el intervalo de tiempo antes de cambiar a manual
        clearInterval(intervalo)

        if(posicionActualManual <= 0){
            posicionActualManual = imagenes.length - 1;
        } else{
            posicionActualManual--;
        }
        imagen.src = imagenes[posicionActualManual];

        // detenemos el intervalo de tiempo antes de cambiar a manual
        intervalo = setInterval(cambiarImagenes, 3000);
    }

    botonAvanzar.addEventListener('click', anvanzarImagen);
    botonRetroceder.addEventListener('click', retrocederImagen);

});
```

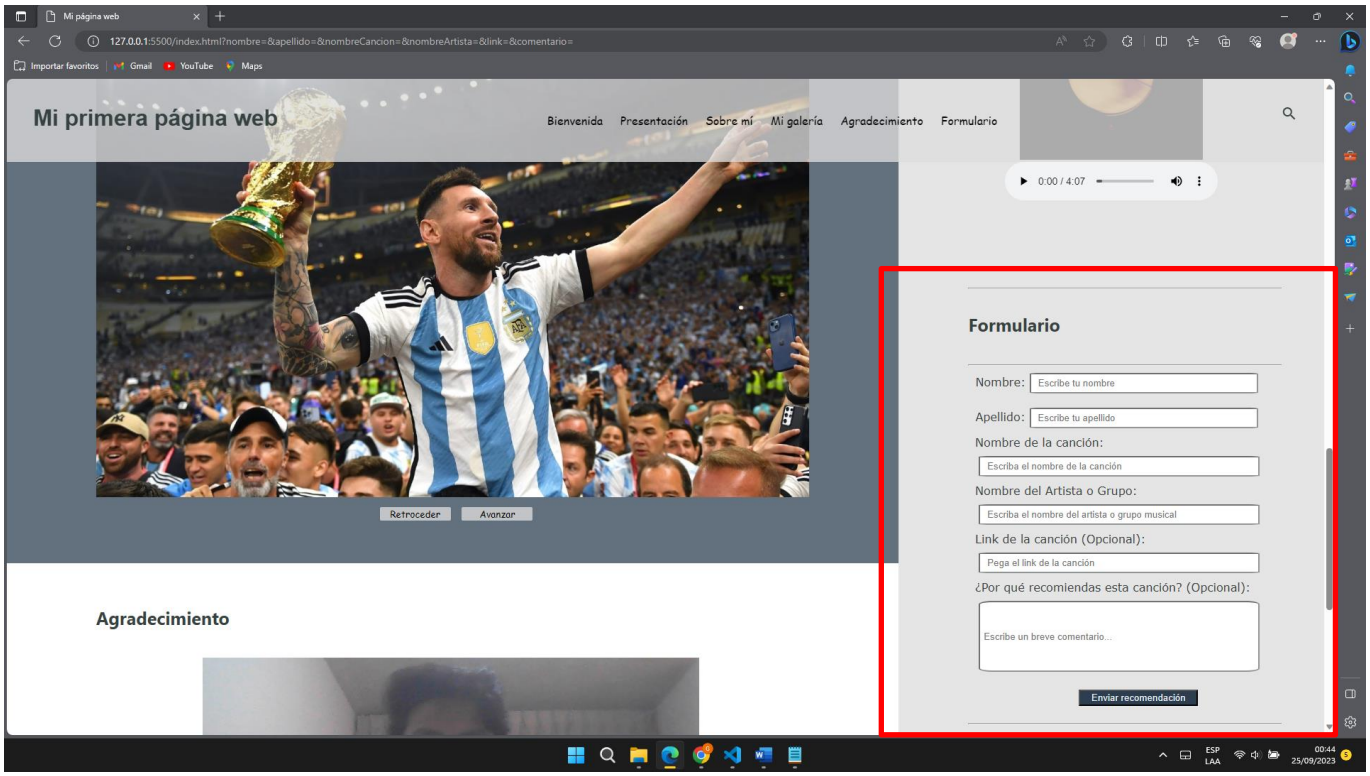


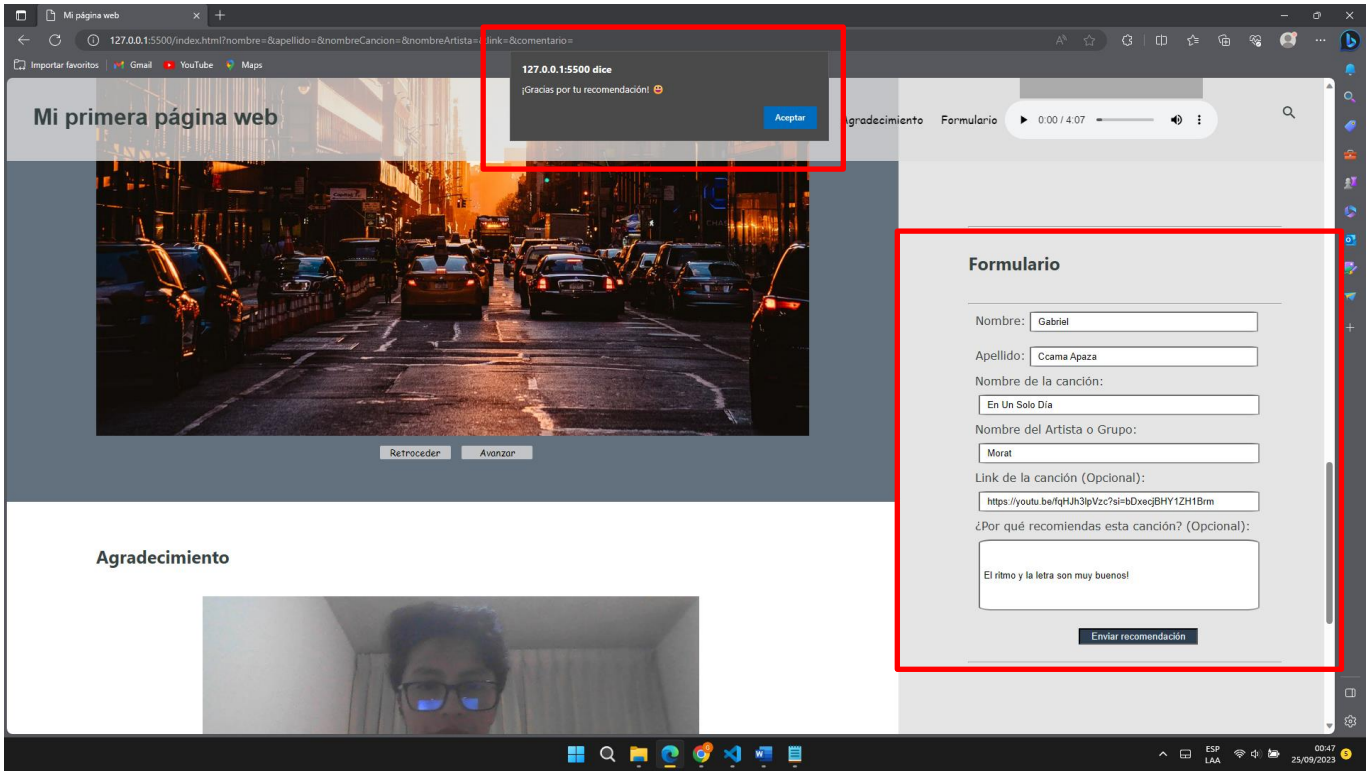
Se agregó código en JavaScript para realizar un formulario de recomendaciones de canciones validando campos y mostrando mensajes

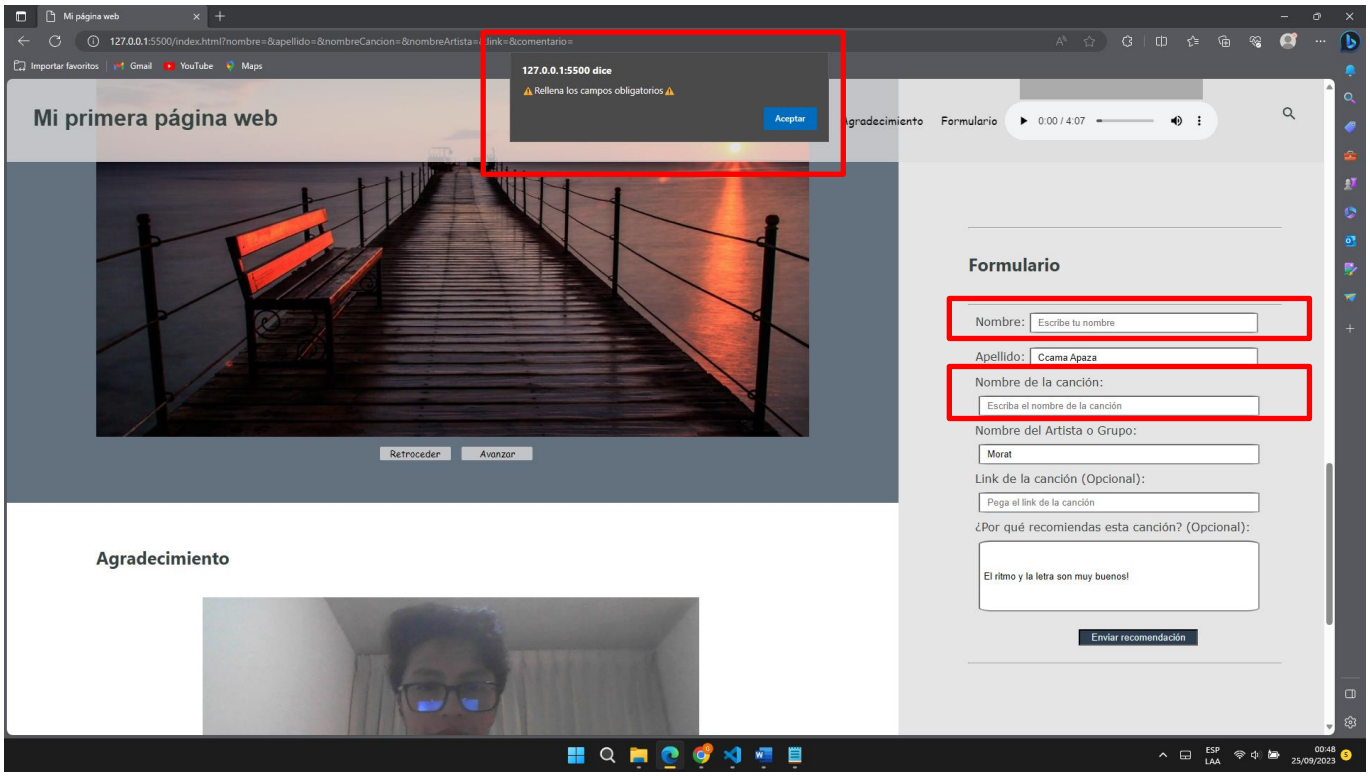
index.html | main.js | style.css

main.js > ...

```
// FORMULARIO - RECOMENDACION

function enviar(){
    var nombre = document.querySelector('#nombre').value;
    var apellido = document.querySelector('#apellido').value;
    var nombreCancion = document.querySelector('#nombreCancion').value;
    var nombreArtista = document.querySelector('#nombreArtista').value;

    if (nombre !== '' && apellido !== '' && nombreCancion !== '' && nombreArtista !== ''){
        alert('¡Gracias por tu recomendación! 😃');
    }else{
        alert('⚠️Rellena los campos obligatorios⚠️');
    }
}
```




Mi página web
127.0.0.1:5500 dice
¡Gracias por tu recomendación!
Aceptar
Mi primera página web
Agradecimiento
Formulario
Retroceder
Avanzar
Formulario
Nombre: Gabriel
Apellido: Ccama Apaza
Nombre de la canción:
En Un Solo Día
Nombre del Artista o Grupo:
Morat
Link de la canción (Opcional):
https://youtu.be/fqHJh3lpVzc?si=bDxecjBHY1ZH1Brm
¿Por qué recomiendas esta canción? (Opcional):
El ritmo y la letra son muy buenos!
Enviar recomendación
Agradecimiento


Mi página web
127.0.0.1:5500 dice
Rellena los campos obligatorios
Aceptar
Mi primera página web
Agradecimiento
Formulario
Retroceder
Avanzar
Formulario
Nombre: Escribe tu nombre
Apellido: Ccama Apaza
Nombre de la canción:
Escriba el nombre de la canción
Nombre del Artista o Grupo:
Morat
Link de la canción (Opcional):
Pega el link de la canción
¿Por qué recomiendas esta canción? (Opcional):
El ritmo y la letra son muy buenos!
Enviar recomendación
Agradecimiento

OBSERVACIONES:

- Aprender los aspectos básicos de JavaScript sienta las bases para futuros desarrollos más avanzados.
- Comprender la estructura lógica de JavaScript permite crear lógica personalizada y respuesta a diferentes situaciones en el desarrollo de una página web.
- Los aspectos básicos, como la manipulación de arreglos y bucles, permiten crear funcionalidades en una página web.

CONCLUSIONES:

- Se concluye que conocer los aspectos básicos de JavaScript es esencial para ingresar al mundo de la programación de páginas web dinámicas. Estos aspectos incluyen funciones, arreglos, bucles, operadores y estructuras condicionales, y permiten el desarrollo de una amplia gama de funcionalidades en nuestras páginas web.
- Se concluye que la incorporación de JavaScript en una página HTML es fundamental para transformar páginas estáticas en interactivas y dinámicas. JavaScript proporciona las herramientas necesarias para crear una interacción fluida entre el usuario y la página.
- Se reconoce que comprender la estructura de JavaScript es crucial para abordar y resolver problemas de manera efectiva utilizando la lógica de programación. Esta comprensión proporciona una base sólida para el desarrollo de aplicaciones web más complejas en el futuro.
- Se destaca que el trabajo propuesto al final del laboratorio nos permitió poner en práctica los conocimientos adquiridos. Esta aplicación práctica contribuye a consolidar la comprensión de JavaScript, HTML y CSS.
- Como evidencia de la aplicación de los conceptos básicos aprendidos en el laboratorio, se logró implementar con éxito un Slider de imágenes, tanto automático como manual, y un formulario de recomendaciones de música.