UNIVERSIDAD TECNOLÓGICA DE AGUASCALIENTES

TEMA:

“Memorama (Primera Entrega)”

NOMBRE ALUMNO:

Talamantes Castañeda Ángela María.

MATRÍCULA:

191243.

CARRERA/AULA/GRADO/GRUPO:

IDGS ~ ICarniage, 10°A.

PROFESOR:

Luis Fernando Perea Gallosso.

MATERIA:

Optativa.

LINK:

Repository:

https://github.com/AngelaMTC/MemoryGame_OI

GitHub Pages:

https://angelamtc.github.io/MemoryGame_OI/

Análisis de todas las funcionalidades que se utilizarán:

Se utilizará Bootstrap para la ayuda del diseño:

```
 <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.
min.css" rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
```

SweetAlert:

```
<script src="./js/sweetalert2.all.min.js"></script>
```

Se usó un archive .js para su utilización (no funcionó el script).

Vue:

```
<script src="https://cdn.jsdelivr.net/npm/vue@2.7.13/dist/vue.js"></script>
```

Para el diseño de la página de juego.

Functions / Methods:

flip()

Intercambia la clave con su valor correspondiente. En JavaScript, la matriz primero se convierte en una colección y luego la función se aplica a la colección.

random()

Devuelve un número pseudoaleatorio de punto flotante que es mayor o igual a 0 y menor que 1, con una distribución aproximadamente uniforme en ese rango, que luego puede escalar a su rango deseado.

slice()

Devuelve una copia de una parte del array dentro de un nuevo array empezando por inicio hasta fin (fin no incluido). El array original no se modificará.

forEach()

Ejecuta la función indicada una vez por cada elemento del array.

addEventListener()

Configura una función que se llamará cada vez que el evento especificado se entregue al objetivo.

appendChild()

Agrega un nodo (elemento) como el último hijo de un elemento.

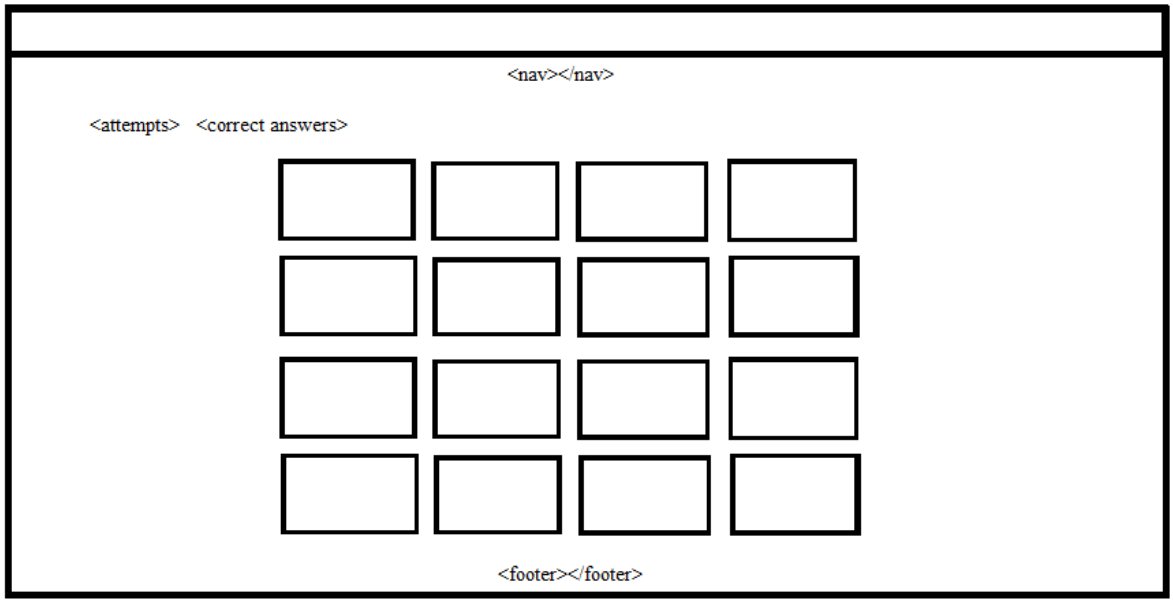
removeChild()

La interfaz elimina un nodo secundario del DOM y devuelve el nodo eliminado.

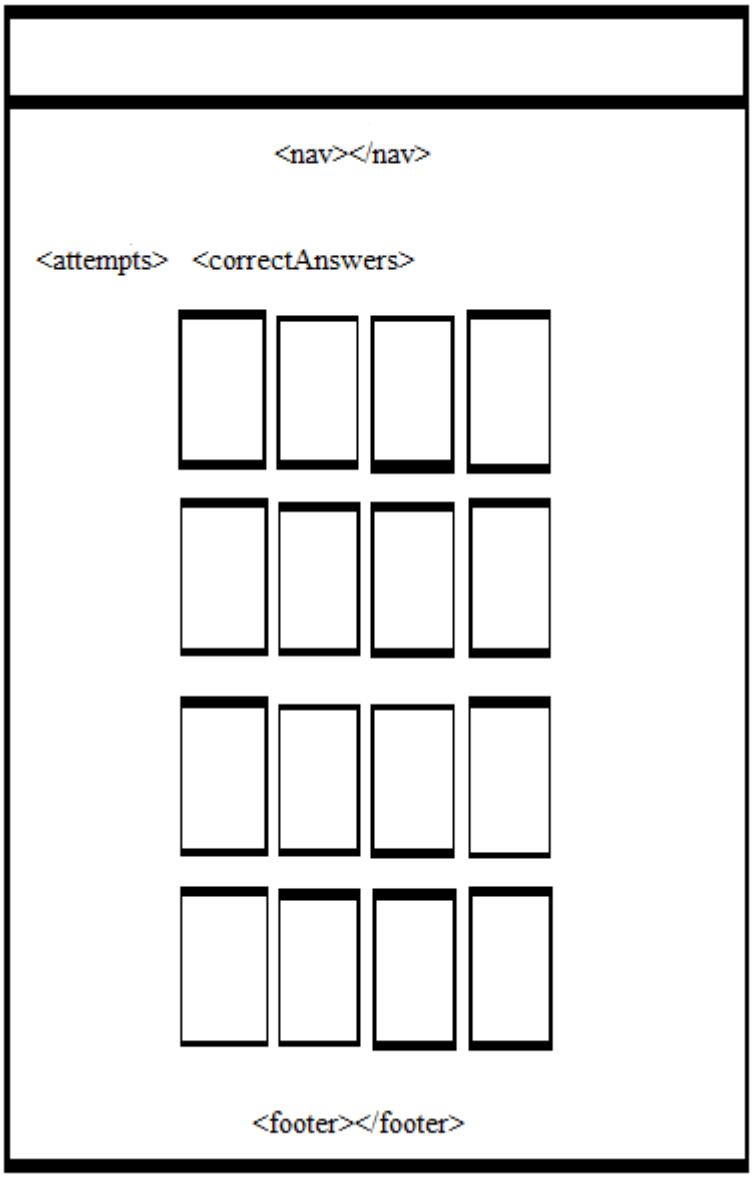
Diseño de la aplicación:

Será responsiva:

WEB:



MÓVIL:



Relación de los objetos que se usarán:

```
const maxAttempts = 8, // attempts máximos que tiene el jugador
  columns = 4, // columns del memorama
  timeFlip = 1, // Por cuántos segundos mostrar ambas imágenes
  imgHidden = "../styles/images/block.jpg"; // La imagen que se muestra cuando la
real está oculta
new Vue({
  el: "#app",
  data: () => ({
    // La ruta de las imágenes. Puede ser relativa o absoluta
    images: [
```

```
      "../styles/images/1up.ico",
      "../styles/images/feather.ico",
      "../styles/images/flower_fire.ico",
      "../styles/images/mushroom.png",
      "../styles/images/mario.jpg",
      "../styles/images/boo.png",
      "../styles/images/goomba.jfif",
      "../styles/images/luma.png",
    ],
    memorama: [],
    // Para saber cuál fue la carta anteriormente seleccionada
    lastClick: {
      iFile: null,
      iImg: null,
    },
    imgHidden: imgHidden,
    maxAttempts: maxAttempts,
    attempts: 0,
    correctAnswer: 0,
    waitTime: false,
  }),
```

```
// Para mezclar un arreglo:
    random(random) {
      var j, x, i;
      for (i = random.length - 1; i > 0; i--) {
        j = Math.floor(Math.random() * (i + 1));
        x = random[i];
        random[i] = random[j];
        random[j] = x;
      }
      return random;
    },
```

```
flip(iFile, iImg) {
      // Si se está regresando una imagen a su estado original, detener flujo
      if (this.waitTime) {
        return;
      }
      // Si es una imagen se acierta:
      if (this.memorama[iFile][iImg].sucess) {
        return;
      }
```

```
      // Selección de imagen por primera vez:
      if (this.lastClick.iFile === null && this.lastClick.iImg === null) {
        this.memorama[iFile][iImg].show = true;
        this.lastClick.iFile = iFile;
        this.lastClick.iImg = iImg;
        return;
      }
      // Ocultar la imagen que no es par:
      let imgSelected = this.memorama[iFile][iImg];
      let ultimaimgSelected =
        this.memorama[this.lastClick.iFile][this.lastClick.iImg];
      if (iFile === this.lastClick.iFile && iImg === this.lastClick.iImg) {
        this.memorama[iFile][iImg].show = false;
        this.lastClick.iFile = null;
        this.lastClick.iImg = null;
        this.attemptIncrease();
        return;
      }
```

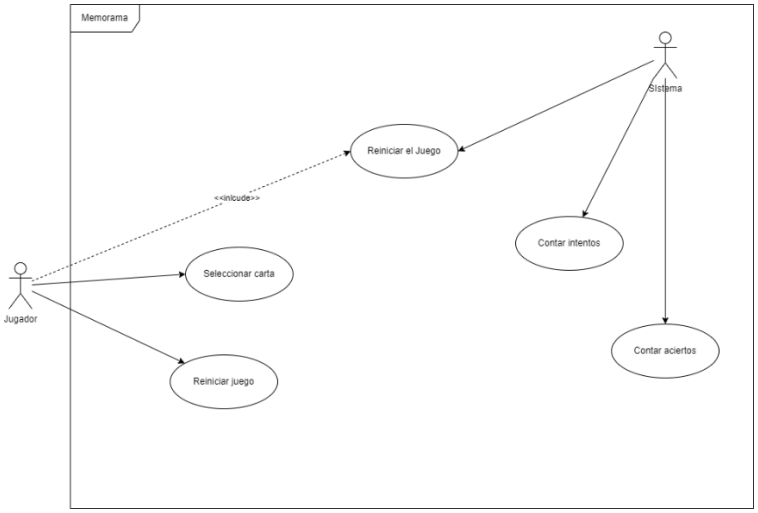
```
// Al seleccionar imagen par (se guarda la última seleccionada):
      this.memorama[iFile][iImg].show = true;
      if (imgSelected.ruta === ultimaimgSelected.ruta) {
        this.aciertos++;
        this.memorama[iFile][iImg].sucess = true;
        this.memorama[this.lastClick.iFile][this.lastClick.iImg].sucess = true;
        this.lastClick.iFile = null;
        this.lastClick.iImg = null;
        // Se comprueba cada vez que se gana:
        if (this.winner()) {
          this.victory();
        }
      } else {
        // Si no  se acierta, entonces gira ambas imágenes:
        this.waitTime = true;
        setTimeout(() => {
          this.memorama[iFile][iImg].show = false;
          this.memorama[iFile][iImg].animacion = false;
          this.memorama[this.lastClick.iFile][this.lastClick.iImg].show = false;
          this.lastClick.iFile = null;
          this.lastClick.iImg = null;
          this.waitTime = false;
        }, timeFlip * 1000);
        this.attemptIncrease();
      }
```

```
    },
    restartGame() {
      let memorama = [];
      this.images.forEach((imagen, indice) => {
        let imgMemorama = {
          ruta: imagen,
          show: false, // No se muestra la original
          sucess: false, // No acertada
        };
        // Poner dos veces la misma imagen:
        memorama.push(imgMemorama, Object.assign({}, imgMemorama));
      });
```

```
// Mandar aleatorio las imágenes:
      this.random(memorama);

      // Dividirlo en subarreglos o columns
      let memoramaDividido = [];
      for (let i = 0; i < memorama.length; i += columns) {
        memoramaDividido.push(memorama.slice(i, i + columns));
      }
      // Reiniciar attempts
      this.attempts = 0;
      this.aciertos = 0;
      // Asignar a instancia de Vue para que lo dibuje
      this.memorama = memoramaDividido;
    },
```

Casos de uso: describiendo los procesos del juego:



Vista final de la entrega uno: