

Universidad Tecnológica Metropolitana



Estructuras de Datos Aplicadas

ISC. Ruth Betsaida Martínez Domínguez, MGTI

Examen Parcial 1

Soberanis Acosta Jimena Monserrat

Desarrollo de Software Multiplataforma

Cuarto Cuatrimestre

4°B

Parcial I

Jueves, 5 de octubre de 2023

Creé el HTML que contiene principalmente el form de llenado para el paciente, y en botón llama a su respectivo método del javascript

```
<> index.html > html > body > div#paciente-form
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Hospital el Faro del Mayab</title>
7      <link rel="stylesheet" href="styles.css">
8  </head>
9  <body>
10     <h1>Hospital el Faro del Mayab</h1>
11     <h2>Registro de pacientes</h2>
12
13     <div id="paciente-form">
14         <h2>Registrar nuevo paciente</h2>
15         <input type="text" id="nombre" placeholder="Nombre">
16         <input type="text" id="edad" placeholder="Edad">
17         <input type="text" id="genero" placeholder="Género">
18         <input type="text" id="telefono" placeholder="Número telefónico">
19         <input type="text" id="diagnostico" placeholder="Diagnostico Médico"><br>
20         <button onclick="agregarPaciente()">Agregar paciente</button>
21     </div>
22
```

Luego esta el formulario de búsqueda por nombre o diagnóstico, que también tienen sus botones que llaman a sus métodos

```
23     <div id="busqueda-form">
24         <h2>Buscar Paciente</h2>
25         <input type="text" id="buscar-nombre" placeholder="Buscar por nombre">
26         <button onclick="buscarPaciente('nombre')">Buscar por Nombre</button>
27         <input type="text" id="buscar-diagnostico" placeholder="Buscar por diagnóstico">
28         <button onclick="buscarPaciente('diagnostico')">Buscar por Diagnostico</button>
29     </div>
30
```

Luego, están las dos listas, la de pacientes automática y la de búsqueda.

```
30
31     <div id="pacientes-lista">
32         <h2>Lista de pacientes</h2>
33         <ul id="lista">
34             <li></li>
35         </ul>
36     </div>
37
38     <div id="resultados-busqueda">
39         <h2>Resultados de búsqueda</h2>
40         <ul id="lista-pacientes-encontrados">
41             <li></li>
42         </ul>
43     </div>
44
```

Por último, está el apartado de las estadísticas:

```
42
43     <div id="estadisticas">
44         <h2>Estadísticas</h2>
45         <p id="info-total-pacientes">Total de pacientes: <span id="total-pacientes">0</span></p>
46         <p id="info-promedio">Edad promedio: <span id="edad-promedio">0</span></p>
47     </div>
48
49     <script src="script.js"></script>
50 </body>
51 </html>
```

Primero, creé la primera clase llamada Paciente, se utiliza para crear los objetos de un paciente, tiene 5 atributos: nombre, edad, genero teléfono y diagnostico, estos mismos se guardan en el constructor.

```
JS script.js > agregarPaciente
1  class Paciente {
2      constructor(nombre, edad, genero, telefono, diagnostico) {
3          this.nombre = nombre;
4          this.edad = edad;
5          this.genero = genero;
6          this.telefono = telefono;
7          this.diagnostico = diagnostico;
8      }
9  }
10
```

Después, está la clase RegistroPaciente que sirve para administrar un proceso de registro de pacientes y realizar ciertas operaciones con ellos.

Se inicia un constructor que crea un arreglo vacío que guardará los pacientes en objetos de tipo Paciente.

```
11 class RegistroPaciente {
12     constructor() {
13         this.pacientes = [];
14     }
15 }
```

Está el método de agregarPaciente, que agrega un nuevo objeto de tipo Paciente, al registro de pacientes almacenado en pacientes.

```
15
16 agregarPaciente(paciente) {
17     this.pacientes.push(paciente);
18 }
```

El método siguiente llamado `buscarPacienteRekursivamente`, como su nombre lo dice se utiliza para buscar pacientes en el registro de manera recursiva, pues al final vuelve a llamar al método para seguir con la búsqueda del paciente. Este método lo que hace es comparar un valor específico (en este caso es nombre o diagnóstico) con un campo particular (el nombre o diagnóstico de los pacientes) y devuelve una lista de pacientes que coinciden. Este método termina hasta que haya recorrido toda la lista, toda esta información se guarda en un arreglo vacío que se creó arriba.

```
20  buscarPacienteRekursivamente(valor, campo, indice = 0, pacientesEncontrados = []) {
21    if (indice >= this.pacientes.length) {
22      return pacientesEncontrados;
23    }
24
25    const paciente = this.pacientes[indice];
26    if (paciente[campo] === valor) {
27      pacientesEncontrados.push(paciente);
28    }
29
30    return this.buscarPacienteRekursivamente(valor, campo, indice + 1, pacientesEncontrados);
31  }
```

Luego, existen dos métodos que lo que hacen es iniciar la búsqueda de un paciente ya sea por el nombre o por el diagnóstico, está hecho de esta manera, pues sirven para los botones del HTML. En este método se llama al método que hace esa búsqueda, pero lo hace con la tabla que se le manda.

```
33  buscarPacientePorNombre(nombre) {
34    return this.buscarPacienteRekursivamente(nombre, "nombre");
35  }
36
37  buscarPacientePorDiagnostico(diagnostico) {
38    return this.buscarPacienteRekursivamente(diagnostico, "diagnostico");
39  }
```

Este método, se encarga de mostrar la lista completa de pacientes en la interfaz del usuario. Busca el id de lista en el HTML para crear por cada paciente un elemento de lista.

```
41  mostrarListaPacientes() {
42    const listaPacientes = document.getElementById('lista');
43    listaPacientes.innerHTML = '';
44
45    this.pacientes.forEach(paciente => {
46      const li = document.createElement('li');
47      li.textContent = `${paciente.nombre}, ${paciente.edad} años, ${paciente.diagnostico}`;
48      listaPacientes.appendChild(li);
49    });
50  }
```

El método `calcularEstadisticas` se trata de unas sencillas relacionadas con los pacientes, como el número total de pacientes y el promedio de edad. Busca los espacios específicos en el HTML para que se muestren ahí.

```

52     calcularEstadisticas() {
53         const totalPacientes = this.pacientes.length;
54         const sumaEdades = this.pacientes.reduce((suma, paciente) => suma + paciente.edad, 0);
55         const edadPromedio = totalPacientes > 0 ? sumaEdades / totalPacientes : 0;
56
57         document.getElementById('total-pacientes').textContent = totalPacientes;
58         document.getElementById('edad-promedio').textContent = edadPromedio.toFixed(2);
59     }
60

```

Por último en esta clase, está el método de eliminarPaciente, que hace lo que su nombre dice.

```

61     eliminarPaciente(nombre) {
62         this.pacientes = this.pacientes.filter(paciente => paciente.nombre !== nombre);
63     }
64 }
65

```

Esta parte del código, se crea un nuevo objeto llamado registro como instancia de la clase RegistroPaciente.

Después está la función agregarPaciente, que interactúa con el usuario, que se ejecuta cuando un usuario agrega un nuevo paciente al sistema. Captura los valores ingresados por el usuario en el formulario de la interfaz de usuario (nombre, edad, género, teléfono y diagnóstico) y utiliza estos valores para crear un nuevo objeto Paciente.

Al ingresarlos, se limpian los campos del formulario, y después se llama a dos métodos, uno para mostrarlo en la lista y el otro sobre las estadísticas.

```

66     const registro = new RegistroPaciente();
67
68     function agregarPaciente() {
69         const nombre = document.getElementById('nombre').value;
70         const edad = parseInt(document.getElementById('edad').value);
71         const genero = document.getElementById('genero').value;
72         const telefono = document.getElementById('telefono').value;
73         const diagnostico = document.getElementById('diagnostico').value;
74
75         const paciente = new Paciente(nombre, edad, genero, telefono, diagnostico);
76         registro.agregarPaciente(paciente);
77
78         document.getElementById('nombre').value = "";
79         document.getElementById('edad').value = "";
80         document.getElementById('genero').value = "";
81         document.getElementById('telefono').value = "";
82         document.getElementById('diagnostico').value = "";
83
84         registro.mostrarListaPacientes();
85         registro.calcularEstadisticas();
86     }
87

```

Esta función mostrarPacientesEncontrados se utiliza para mostrar en la interfaz de usuario la lista de pacientes que han sido encontrados durante una búsqueda. Primero, borra cualquier lista que

haya estado antes y crea una nueva con los pacientes encontrados, y por cada uno de ellos se crea el botón eliminar, que llama a su respectivo método eliminar.

```
87
88 function mostrarPacientesEncontrados(pacientes) {
89     const listaPacientesEncontrados = document.getElementById('lista-pacientes-encontrados');
90     listaPacientesEncontrados.innerHTML = '';
91
92     pacientes.forEach(paciente => {
93         const li = document.createElement('li');
94         li.textContent = `${paciente.nombre}, ${paciente.edad} años, ${paciente.diagnostico}`;
95
96         const eliminarBtn = document.createElement('button');
97         eliminarBtn.textContent = 'Eliminar';
98         eliminarBtn.addEventListener('click', () => eliminarPaciente(paciente.nombre));
99
100        li.appendChild(eliminarBtn);
101        listaPacientesEncontrados.appendChild(li);
102    });
103 }
104
```

Esta función se llama cuando un usuario desea buscar pacientes en el sistema, ya sea por nombre o diagnóstico, lo que hace es capturar el valor de búsqueda ingresado por el usuario en la interfaz y determinar si la búsqueda debe realizarse por nombre o diagnóstico. De ahí se llaman a los métodos respectivos, según el campo escogido.

```
105 function buscarPaciente(campo) {
106     const valorABuscar = document.getElementById(`buscar-${campo}`).value;
107     let pacientesEncontrados;
108
109     if (valorABuscar !== "") {
110         if (campo === 'nombre') {
111             pacientesEncontrados = registro.buscarPacientePorNombre(valorABuscar);
112         } else if (campo === 'diagnostico') {
113             pacientesEncontrados = registro.buscarPacientePorDiagnostico(valorABuscar);
114         }
115
116         mostrarPacientesEncontrados(pacientesEncontrados);
117     }
118 }
119
```

Por último, está la función eliminarPaciente, que se ejecuta con el botón creado arriba y que ve en el HTML, y elimina un paciente específico de la lista.

También lo elimina de las estadísticas y los pacientes encontrados, para limpiar el HTML

```
119 function eliminarPaciente(nombre) {
120     registro.eliminarPaciente(nombre);
121     registro.mostrarListaPacientes();
122     registro.calcularEstadisticas();
123     mostrarPacientesEncontrados([]);
124 }
```

Le di un estilo con CSS:

```
1  body {
2    font-family: Arial, sans-serif;
3    background-color: #f2f2f2;
4    margin: 20px;
5    padding: 0;
6  }
7
8  .container {
9    max-width: 800px;
10   margin: 0 auto;
11   padding: 20px;
12   background-color: #fff;
13   border-radius: 5px;
14   box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
15 }
16
17 h1 {
18   color: #333;
19   text-align: center;
20 }
21
22 h2 {
23   color: #555;
24   text-align: center;
25 }
26
27 #paciente-form {
28   margin-bottom: 20px;
29 }
30
31 input[type="text"] {
32   width: 90%;
33   padding: 10px;
34   margin-bottom: 10px;
35   margin-right: 10px;
36   border: 1px solid #ccc;
37   border-radius: 5px;
38 }
39
40 button {
41   background-color: #007bff;
42   color: #fff;
43   padding: 10px 20px;
44   border: none;
45   border-radius: 5px;
46   cursor: pointer;
47 }
48
49 #pacientes-lista ul {
50   list-style: none;
51   padding: 0;
52   text-align: center;
53 }
54
55 #pacientes-lista li {
56   margin-bottom: 10px;
57   padding: 10px;
58   background-color: #f9f9f9;
59   border: 1px solid #ddd;
60   border-radius: 5px;
61 }
62
63 #estadisticas p {
64   margin: 10px 0;
65 }
66
67 #busqueda-form button {
68   background-color: #28a745;
69   margin-bottom: 10px;
70 }
71
72 #info-total-pacientes, #info-promedio {
73   margin-bottom: 10px;
74   width: 15%;
75   padding: 10px;
76   background-color: #f9f9f9;
77   border: 1px solid #ddd;
78   border-radius: 5px;
79   text-align: center;
80 }
81
82 #resultados-busqueda ul {
83   list-style: none;
84   padding: 0;
85 }
86
87 #resultados-busqueda li {
88   margin-bottom: 10px;
89   padding: 10px;
90   background-color: #f9f9f9;
91   border: 1px solid #ddd;
92   border-radius: 5px;
93 }
94
```

Y con electron lancé la aplicación:

Hospital el Faro del Mayab

File Edit View Window Help

Hospital el Faro del Mayab

Registro de pacientes

Registrar nuevo paciente

Agregar paciente

Buscar Paciente

Buscar por Nombre

Lista de pacientes

Jimena Soberanis, 19 años, Sinusitis

Resultados de búsqueda

Estadísticas

Total de pacientes:
1

Edad promedio:
19.00

Lista de pacientes

Jimena Soberanis, 19 años, Sinusitis

Anna Soberanis, 16 años, Diarrea

Resultados de búsqueda

Estadísticas

Total de pacientes:
2

Edad promedio:
17.50

Jimena Soberanis

Buscar por Nombre

Buscar por diagnóstico

Buscar por Diagnostico

Lista de pacientes

Jimena Soberanis, 19 años, Sinusitis

Anna Soberanis, 16 años, Diarrea

Resultados de búsqueda

Jimena Soberanis, 19 años, Sinusitis

Eliminar

Buscar por Diagnostico

Lista de pacientes

Anna Soberanis, 16 años, Diarrea

Resultados de búsqueda

Estadísticas

Total de pacientes:
1

Edad promedio:
16.00