



Diseño Web
Carlos Tarriño Espinosa

Desarrollo web en entorno cliente





1. Contenido

1.1 HTML

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8">
5   <title>MedacDent - Gestión de Citas</title>
6   <link rel="stylesheet" href="estilos.css">
7   <link rel="icon" href="img/icono.png" type="image/x-icon">
8 </head>
9 <body>
10  <header>
11    
12    <h1>MedacDent</h1>
13  </header>
14
15  <section>
16    <form id="formCita">
17      <h2>Programar Cita</h2>
18
19      <!-- Campo oculto para almacenar id de la cita -->
20      <input type="hidden" id="idCita">
21
22      <!-- Datos de la cita -->
23      <label for="fecha">Fecha:</label>
24      <input type="date" id="fecha" required>
25
26      <label for="hora">Hora:</label>
27      <input type="time" id="hora" required>
28
29      <!-- Datos del paciente -->
30      <label for="nombre">Nombre:</label>
31      <input type="text" id="nombre" pattern="[A-Za-z ]+" title="Solo letras y espacios permitidos" required>
32
33      <label for="apellidos">Apellidos:</label>
34      <input type="text" id="apellidos" pattern="[A-Za-z ]+" title="Solo letras y espacios permitidos" required>
35
36      <label for="dni">DNI:</label>
37      <input type="text" id="dni" pattern="[0-9]{8}[A-Za-z]" title="8 números seguidos de una letra" required>
38
39      <label for="telefono">Teléfono:</label>
40      <input type="tel" id="telefono" pattern="[0-9]{9}" title="Solo 9 números permitidos" required>
41
42      <label for="fechaNacimiento">Fecha de Nacimiento:</label>
43      <input type="date" id="fechaNacimiento" required>
44
45      <!-- Observaciones -->
46      <label for="observaciones">Observaciones:</label>
47      <textarea id="observaciones"></textarea>
48
49      <!-- Guardar cita -->
50      <button type="submit">Guardar Cita</button>
51    </form>
52  </section>
53
54  <h2 id="Citasprogramadas"> <b>Citas programadas</b></h2>
55
56  <section>
57    <table id="tablaCitas">
58      <thead>
59        <tr>
60          <th>Orden</th>
61          <th>Fecha</th>
62          <th>Hora</th>
63          <th>Nombre</th>
64          <th>Apellidos</th>
65          <th>DNI</th>
66          <th>Teléfono</th>
67          <th>Fecha de Nacimiento</th>
68          <th>Observaciones</th>
69          <th>Acciones</th>
70        </tr>
71      </thead>
72      <tbody>
73        <!-- Las citas se añadirán aquí -->
74      </tbody>
75    </table>
76  </section>
77
78  <footer>
79    <p>Contacto: +34 123 456 789</p>
80    <p>Dirección: C. de Albalá, 5, San Blas-Canillejas, 28037 Madrid</p>
81    <p><a href="https://www.google.com/maps?q=C. de Albalá, 5, San Blas-Canillejas, 28037 Madrid"
82      target="_blank">Ver en Google Maps</a></p>
83  </footer>
84  <script src="scripts.js"></script>
85 </body>
86 </html>
87
88
```

1.2 JavaScript

```
1 // Esta sección de código se ejecuta cuando se carga completamente el documento HTML.
2 document.addEventListener('DOMContentLoaded', (event) => {
3     establecerFechaMinima(); //Establece condiciones en las fechas y el horario
4     cargarCitas(); // Carga y muestra las citas existentes.
5
6     // Agrega un evento al enviar el formulario.
7     const form = document.getElementById('formCita');
8     form.addEventListener('submit', function (e) {
9         e.preventDefault(); // Evita el comportamiento por defecto de envío del formulario.
10        if (validarFormulario()) { // Verifica si el formulario es válido.
11            guardarCita(); // Guarda la cita si el formulario es válido.
12        }
13    });
14 });
15
16 //Establece condiciones en las fechas y el horario
17 function establecerFechaMinima() {
18     var today = new Date(); // Obtiene la fecha y hora actual.
19     var dd = String(today.getDate()).padStart(2, '0'); // Obtiene el día actual y lo formatea con dos dígitos.
20     var mm = String(today.getMonth() + 1).padStart(2, '0'); // Obtiene el mes actual (0-11) y lo ajusta a formato (1-12) con dos dígitos.
21     var yyyy = today.getFullYear(); // Obtiene el año actual.
22
23     var fechaActual = yyyy + '-' + mm + '-' + dd;
24
25     document.getElementById('fecha').setAttribute('min', fechaActual); // Establece la fecha mínima para la fecha
26     document.getElementById('fechaNacimiento').setAttribute('max', fechaActual); // Establece la fecha máxima para la fecha de nacimiento
27
28     // Restringe el rango de horas seleccionables
29     document.getElementById('hora').setAttribute('min', '08:00');
30     document.getElementById('hora').setAttribute('max', '20:00');
31 }
32
33 // Función para cargar y mostrar todas las citas guardadas.
34 function cargarCitas() {
35     let citas = JSON.parse(localStorage.getItem('citas')) || [];
36     let cuerpoTabla = document.querySelector('#tablaCitas tbody');
37     cuerpoTabla.innerHTML = '';
38
39     if (citas.length === 0) {
40         // Si no hay citas, muestra una fila con el mensaje "Dato vacío"
41         let fila = cuerpoTabla.insertRow();
42         let celda = fila.insertCell();
43         celda.colSpan = 10; //Asume un total de 10 columnas
44         celda.textContent = "Dato vacío";
45         celda.className = 'dato-vacio'; // Se agrega una clase para poder identificar esta fila
46     } else {
```

```
46     } else {
47         // Si hay citas, muestra cada una en la tabla
48         citas.forEach((cita, index) => {
49             let fila = cuerpoTabla.insertRow();
50
51             // Añade la celda para el número de orden
52             fila.insertCell().textContent = index + 1;
53
54             // Añade las celdas para los demás datos de la cita
55             fila.insertCell().textContent = cita.fecha;
56             fila.insertCell().textContent = cita.hora;
57             fila.insertCell().textContent = cita.nombre;
58             fila.insertCell().textContent = cita.apellidos;
59             fila.insertCell().textContent = cita.dni;
60             fila.insertCell().textContent = cita.telefono;
61             fila.insertCell().textContent = cita.fechaNacimiento;
62             fila.insertCell().textContent = cita.observaciones;
63
64             // Añade las celdas para las acciones (editar/eliminar)
65             let tdAcciones = fila.insertCell();
66
67             //Botón editar
68             let btnEditar = document.createElement('button');
69             btnEditar.textContent = 'Editar';
70             btnEditar.className = 'editar';
71             btnEditar.onclick = function () { editarCita(cita.id); };
72             tdAcciones.appendChild(btnEditar);
73
74             //Botón eliminar
75             let btnEliminar = document.createElement('button');
76             btnEliminar.textContent = 'Eliminar';
77             btnEliminar.className = 'eliminar';
78             btnEliminar.onclick = function () { eliminarCita(cita.id); };
79             tdAcciones.appendChild(btnEliminar);
80         });
81     }
82 }
83
84
85 // Esta función valida los campos del formulario antes de guardar una cita.
86 function validarFormulario() {
87     let valido = true;
88 }
```

```
86 function validarFormulario() {
87     let valido = true;
88
89     // Aquí se realizan múltiples validaciones para cada campo:
90     // Validación para el nombre
91     const nombre = document.getElementById('nombre').value;
92     if (nombre.trim() === '') {
93         mostrarError('nombre', 'El nombre es obligatorio');
94         valido = false;
95     } else {
96         limpiarError('nombre');
97     }
98
99     // Validación para el DNI
100    const dni = document.getElementById('dni').value;
101    const regexDNI = /^[0-9]{8}[A-Z]$/;
102    if (!regexDNI.test(dni)) {
103        mostrarError('dni', 'El DNI no es válido (debe tener 8 números seguidos de una letra mayúscula)');
104        valido = false;
105    } else {
106        limpiarError('dni');
107    }
108
109    // Validación para los apellidos
110    const apellidos = document.getElementById('apellidos').value;
111    if (apellidos.trim() === '') {
112        mostrarError('apellidos', 'Los apellidos son obligatorios');
113        valido = false;
114    } else {
115        limpiarError('apellidos');
116    }
117
118    // Validación para el teléfono
119    const telefono = document.getElementById('telefono').value;
120    const regexTelefono = /^[0-9]{9}$/;
121    if (!regexTelefono.test(telefono)) {
122        mostrarError('telefono', 'El teléfono no es válido (debe tener 9 dígitos)');
123        valido = false;
124    } else {
125        limpiarError('telefono');
126    }
127
```

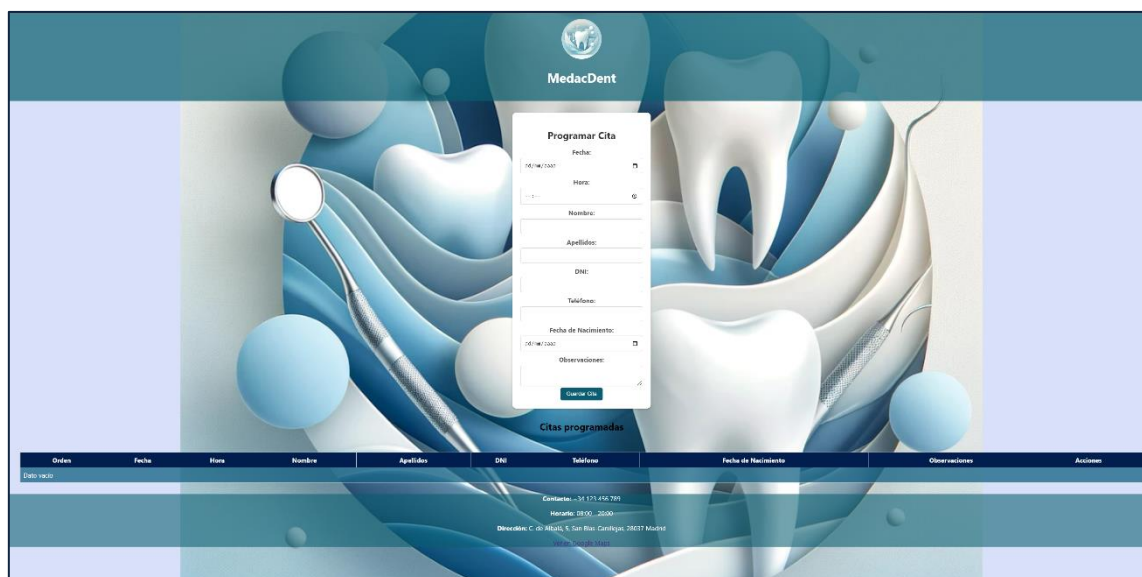
```
128
129    // Validación para la fecha: asegurándose de que no sea en el pasado
130    const fechaCita = document.getElementById('fecha').value;
131    const fechaActual = new Date();
132    // Establecer tiempo a 00:00:00 para comparar solo la fecha
133    fechaActual.setHours(0, 0, 0, 0);
134    // Convertir la fecha del input a un objeto Date para comparación
135    const fechaSeleccionada = new Date(fechaCita);
136
137    if (!fechaCita) {
138        mostrarError('fecha', 'La fecha es obligatoria');
139        valido = false;
140    } else if (fechaSeleccionada < fechaActual) {
141        mostrarError('fecha', 'La fecha de la cita no puede ser en el pasado');
142        valido = false;
143    } else {
144        limpiarError('fecha');
145    }
146
147    // Validación para la hora
148    const hora = document.getElementById('hora').value;
149    if (!hora) {
150        mostrarError('hora', 'La hora es obligatoria');
151        valido = false;
152    } else {
153        limpiarError('hora');
154    }
155
156    // Validación para la fecha de nacimiento
157    const fechaNacimiento = document.getElementById('fechaNacimiento').value;
158    if (!fechaNacimiento) {
159        mostrarError('fechaNacimiento', 'La fecha de nacimiento es obligatoria');
160        valido = false;
161    } else {
162        limpiarError('fechaNacimiento');
163    }
164
165    return valido;
166
167    // Función para mostrar un mensaje de error cerca de un campo del formulario.
168    function mostrarError(campo, mensaje) {
169        const elemento = document.getElementById(campo);
170        const divError = document.createElement('div');
171        divError.textContent = mensaje;
172        divError.className = 'error';
173        if (!elemento.nextSibling) {
174            elemento.parentNode.insertBefore(divError, elemento.nextSibling);
175        }
176    }
177
```

```
175     }
176   }
177
178   // Función para eliminar un mensaje de error existente.
179   function limpiarError(campo) {
180     const elemento = document.getElementById(campo);
181     if (elemento.nextSibling && elemento.nextSibling.className === 'error') {
182       elemento.parentNode.removeChild(elemento.nextSibling);
183     }
184   }
185
186   // función que limpie el formulario al enviar la cita
187   function limpiarFormulario() {
188     document.getElementById('idCita').value = '';
189     document.getElementById('fecha').value = '';
190     document.getElementById('hora').value = '';
191     document.getElementById('nombre').value = '';
192     document.getElementById('apellidos').value = '';
193     document.getElementById('dni').value = '';
194     document.getElementById('telefono').value = '';
195     document.getElementById('fechaNacimiento').value = '';
196     document.getElementById('observaciones').value = '';
197   }
198
199   // Función para guardar una cita en el almacenamiento local del navegador.
200   function guardarCita() {
201     const idCita = document.getElementById('idCita').value;
202     const cita = {
203       id: idCita || new Date().getTime(),
204       fecha: document.getElementById('fecha').value,
205       hora: document.getElementById('hora').value,
206       nombre: document.getElementById('nombre').value,
207       apellidos: document.getElementById('apellidos').value,
208       dni: document.getElementById('dni').value,
209       telefono: document.getElementById('telefono').value,
210       fechaNacimiento: document.getElementById('fechaNacimiento').value,
211       observaciones: document.getElementById('observaciones').value
212     };
213
214     let citas = JSON.parse(localStorage.getItem('citas')) || [];
215     const indice = idCita ? citas.findIndex(c => c.id === idCita) : -1;
216
217     if (indice > -1) {
218       citas[indice] = cita; // Actualiza la cita existente
219       alert("Cita modificada correctamente"); // Mensaje de confirmación
220     } else {
221       citas.push(cita); // Agrega una nueva cita
222       alert("Cita guardada correctamente"); // Mensaje de confirmación
223     }
224
225     localStorage.setItem('citas', JSON.stringify(citas));
226     cargarCitas();
227     limpiarFormulario(); // Limpia el formulario después de guardar la cita
228   }
229
230   // Función para cargar los datos de una cita en el formulario para su edición.
231   function editarCita(idCita) {
232     const citas = JSON.parse(localStorage.getItem('citas')) || [];
233     const citaAEditar = citas.find(cita => cita.id === idCita);
234
235     if (citaAEditar) {
236       document.getElementById('fecha').value = citaAEditar.fecha;
237       document.getElementById('hora').value = citaAEditar.hora;
238       document.getElementById('nombre').value = citaAEditar.nombre;
239       document.getElementById('apellidos').value = citaAEditar.apellidos;
240       document.getElementById('dni').value = citaAEditar.dni;
241       document.getElementById('telefono').value = citaAEditar.telefono;
242       document.getElementById('fechaNacimiento').value = citaAEditar.fechaNacimiento;
243       document.getElementById('observaciones').value = citaAEditar.observaciones;
244
245       // Guardar el ID de la cita en un campo oculto
246       document.getElementById('idCita').value = idCita;
247     }
248   }
249
250
251   // Función para eliminar una cita del almacenamiento local.
252   function eliminarCita(idCita) {
253     let citas = JSON.parse(localStorage.getItem('citas')) || [];
254     citas = citas.filter(cita => cita.id !== idCita);
255     localStorage.setItem('citas', JSON.stringify(citas));
256     cargarCitas();
257     alert("Cita eliminada correctamente"); // Mensaje de confirmación
258   }
```

2. Ejecución

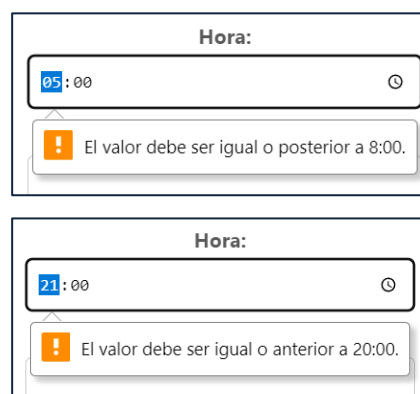
A continuación, explicaré el proceso de ejecución de la aplicación web detallando las funciones que la definen:

- 1) Interfaz web:** Aquí se muestra el diseño de la interfaz web aproximadamente, ya que para realizar la captura he tenido que ampliar el zoom del navegador, dañando así el diseño.

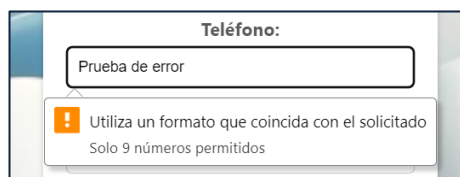


- 2) Restricciones en el formulario:** El formulario presenta diferentes restricciones:

- ❖ **Restricción fecha:** No se puede elegir una fecha anterior a la actual:
- ❖ **Restricción hora:** Ya que he establecido un horario de 08:00-20:00, no se puede elegir una hora de cita fuera de ese margen:



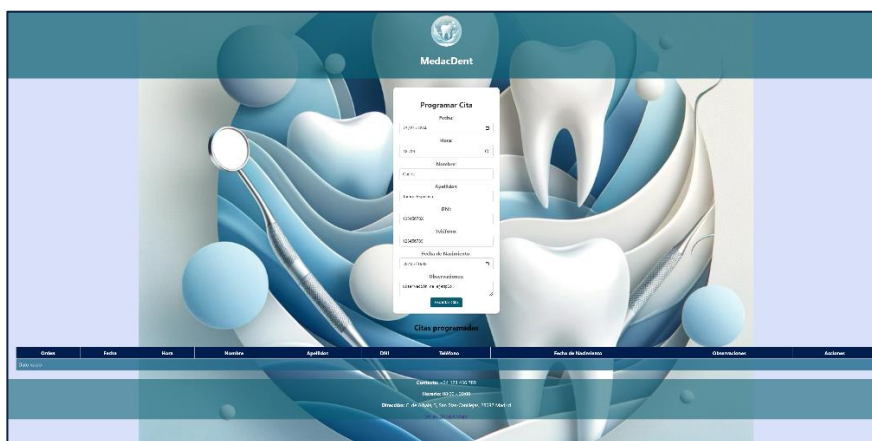
- ❖ **Restricción de campos:** Realicé una prueba aleatoria de error de los campos alfanuméricos en el campo "Teléfono".



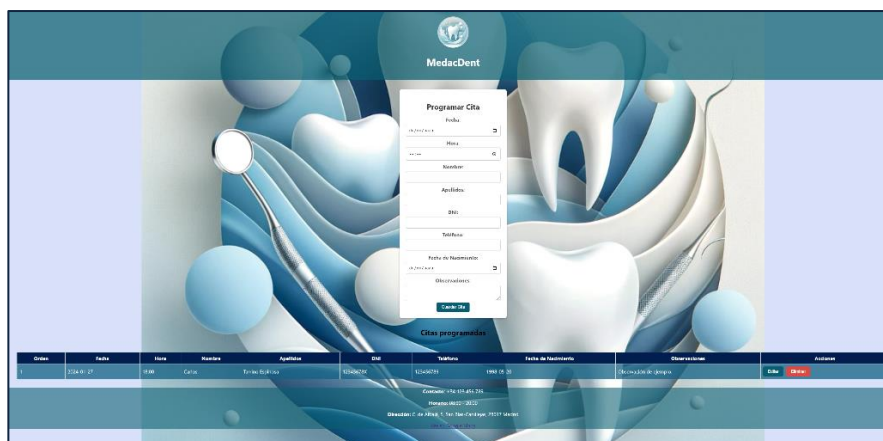
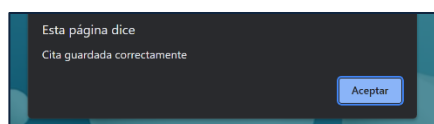
- ❖ **Restricción fecha de nacimiento:** No se puede seleccionar una posterior a la actual



- 3) Cita antes de guardar:** Se muestra el formulario relleno de la cita antes de guardar:



- 4) Cita guardada:** Al pulsar en el botón "Guardar cita" se lanza un aviso de confirmación tras validar los campos, se guarda la cita en la tabla y se limpia el formulario para poder añadir más citas:





5) Creación de segunda cita: Creo una segunda cita para lucir el diseño de la tabla y mostrar el correcto funcionamiento del campo de orden:

Orden	Fecha	Hora	Nombre	Apellidos	DNI	Teléfono	Fecha de Nacimiento	Observaciones	Acciones
1	2024-01-27	18:00	Carlos	Tarrio Espinosa	123456789	123456789	1998-05-20	Observación de ejemplo	Editar Eliminar
2	2024-01-25	17:30	Nombre	Apellidos	000000000	111122233	2004-01-25	Segunda cita de prueba	Editar Eliminar

6) Modificación de la segunda cita: Al pulsar en el botón editar de la columna acciones de la segunda cita, se muestran de nuevo los datos en el formulario para poder ser modificados:

Programar Cita

Fecha: 2024-01-27

Hora: 18:00

Nombre: Carlos

Apellidos: Tarrio Espinosa

DNI: 123456789

Teléfono: 123456789

Fecha de Nacimiento: 1998-05-20

Observaciones: Observación de ejemplo

Guardar Cita

Orden	Fecha	Hora	Nombre	Apellidos	DNI	Teléfono	Fecha de Nacimiento	Observaciones	Acciones
1	2024-01-27	18:00	Carlos	Tarrio Espinosa	123456789	123456789	1998-05-20	Observación de ejemplo	Editar Eliminar
2	2024-01-25	17:30	Nombre	Apellidos	000000000	111122233	2004-01-25	Segunda cita de prueba	Editar Eliminar

7) Cambio de valores de la segunda cita: Cambio los valores de la segunda cita:

Programar Cita

Fecha: 31/01/2024

Hora: 18:30

Nombre: Nombre Nuevo

Apellidos: Apellidos Nuevo

DNI: 000000008

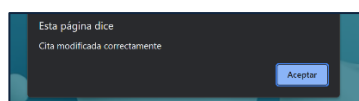
Teléfono: 444555666

Fecha de Nacimiento: 24/01/2024

Observaciones: Segunda cita de prueba editada

Guardar Cita

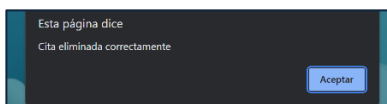
8) Edición exitosa de la segunda cita: Al guardar con los nuevos valores, aparece un aviso de confirmación, se cambian los valores de la cita en la tabla y se limpia el formulario:



Orden	Fecha	Hora	Nombre	Apellidos	DNI	Teléfono	Fecha de Nacimiento	Observaciones	Acciones
1	2024-01-27	18:00	Carlos	Tarrio Espinosa	123456789	123456789	1998-05-20	Observación de ejemplo	Editar Eliminar
2	2024-01-31	18:30	Nombre Nuevo	Apellidos Nuevo	000000008	444555666	2024-01-24	Segunda cita de prueba editada	Editar Eliminar



9) Eliminación de citas: Si pulsamos en el botón de eliminar de ambas citas, aparece un aviso y estas se eliminan, volviendo al estado inicial de la tabla:



Orden	Fecha	Hora	Nombre	Apellidos	DNI	Teléfono	Fecha de Nacimiento	Observaciones	Acciones
Datos vacíos									

3. Problemas

Aunque no haya sido un desarrollo sencillo y he tenido que dedicar mucho tiempo y esfuerzo para solventar algunos errores en la funcionalidad de la aplicación, no ha habido ninguna funcionalidad que no haya podido realizar.

4. Anexos

A continuación, agrego también la hoja de estilos externa:

```
1 body {
2   font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
3   background-color: #f4d47af;
4   background-image: url('img/imagen de fondo.webp');
5   background-size: 70%;
6   background-position: center;
7   background-repeat: no-repeat;
8   background-attachment: fixed;
9   color: #5a5a5a;
10  margin: 0;
11  padding: 0;
12 }
13
14 header {
15   background-color: #095d70b5;
16   color: white;
17   text-align: center;
18   padding: 1rem 0;
19 }
20
21 header img {
22   max-height: 100px;
23   border-radius: 50%;
24   object-fit: cover;
25 }
26
27 section {
28   display: flex;
29   justify-content: center;
30   text-align: center;
31   margin: 1%;
32 }
33
34 #Citasprogramadas {
35   color: #000000;
36   text-align: center;
37   margin-left: auto;
38   margin-right: auto;
39 }
40
41 form {
42   background: white;
43   padding: 20px;
44   border-radius: 10px;
45   box-shadow: 0 4px 5px #000;
46   max-width: 600px;
47 }
48
49 form h2 {
50   color: #333;
51   margin-bottom: 1rem;
52 }
53
54 label {
55   display: block;
56   margin-top: 10px;
57   font-weight: bold;
58 }
59
60 input[type="text"],
61 input[type="tel"],
62 input[type="date"],
63 input[type="time"],
64 textarea {
65   width: 100%;
66   padding: 10px;
67   margin-top: 5px;
68   border-radius: 5px;
69   border: 1px solid #ccc;
70   box-sizing: border-box;
71   transition: border-color 0.3s;
72 }
73
74 input[type="text"]:focus,
75 input[type="tel"]:focus,
76 input[type="date"]:focus,
77 input[type="time"]:focus,
78 textarea:focus {
79   border-color: #4a69bd;
80 }
81
82 button {
83   background-color: #095d70fd;
84   color: white;
85   padding: 8px 15px;
86   border: none;
87   border-radius: 5px;
88   cursor: pointer;
89   transition: background-color 0.3s ease;
90 }
91
92 button:hover {
93   background-color: #012f3afd;
94 }
95
96 button.editar {
97   margin-right: 10px;
98 }
99
100 button.eliminar {
101   background-color: #f44336da;
102   color: white;
103   border-color: #f44336da;
104 }
105
106 button.eliminar:hover {
107   background-color: #da190bda;
108   border-color: #dc3545da;
109 }
110
111 table {
112   width: 100%;
113   border-collapse: collapse;
114   margin-top: 20px;
115 }
116
117 table, th, td {
118   border: 1px solid #ddd;
119 }
120
121 th, td {
122   text-align: left;
123   padding: 8px;
124 }
125
126 th {
127   text-align: center;
128   background-color: #011e4f;
129   color: white;
130 }
131
132 tr {
133   background-color: #136183be;
134   color: white;
135 }
136
137 tr:nth-child(even) {
138   background-color: #f2f2f2;
139   color: #136183be;
140 }
141
142 footer {
143   background-color: #095d7093;
144   color: white;
145   text-align: center;
146   margin-top: 20px;
147   margin-bottom: 0;
148   padding: 0;
149   bottom: 0;
150   width: 100%;
151 }
152
153 .error {
154   color: red;
155   font-size: 0.8em;
156 }
157
```