

Trabajo voluntario

José López Arribas

Grado Superior en Desarrollo de Aplicaciones Web

Desarrollo Web en Entorno Cliente



Índice

1. Contenido	3
2. Ejecución	15
3. Problemas	18
4. Bibliografía	19

1. Contenido

El ejercicio ha sido resuelto en dos documentos HTML. De modo externo, han sido incluidos la funcionalidad (archivo JavaScript) y el estilo (dos archivos CSS, uno para cada HTML). Decidí mostrar una primera página de inicio que enlazara a la página de formulario. En esta última se carga la tabla de las citas.

El **HTML** de la página de inicio es simple. Cuenta con una cabecera con el nombre de la empresa a modo de título, una sección que contiene un enlace a la página de formulario y un pie de página con los datos de contacto.

```
17 <body>
18
19 <header>
20   <h1>MedacDent</h1>
21 </header>
22
23 <section class="container">
24   <div class="row">
25     <a id="nueva" href="/TrabajoClienteFormulario.html">Nueva cita</a>
26   </div>
27 </section>
28
29 <footer class="container">
30   <div class="row">
31     <div class="col-xs-5 col-sm-6 col-md-7 col-lg-8">
32       <p id="pieNombre">MedacDent</p>
33     </div>
34
35     <div class="col-xs-4 col-sm-3 col-md-3 col-lg-2">
36       <p id="pieEmail">Email: medacdent@gmail.com</p>
37     </div>
38
39     <div class="col-xs-3 col-sm-3 col-md-2 col-lg-2">
40       <p id="pieTelefono">Teléfono: 640319934</p>
41     </div>
42   </div>
43 </footer>
44
45 </body>
```

La página del formulario tiene una pequeña cabecera que nos sitúa. La parte principal es una gran sección que embute al formulario. Con dos partes: 'Día de la cita' e 'Información del paciente'. En total, siete campos a rellenar por el usuario. Un pie de página con el contacto termina por definir la estructura.



```
16 <body onload="cargarDatos()">
17
18 <header class="container">
19   <div class="row">
20     <div class="col-xs-6 col-sm-6 col-md-6 col-lg-6">
21       <h1>MedacDent</h1>
22     </div>
23
24     <div class="col-xs-6 col-sm-6 col-md-6 col-lg-6">
25       <h4>Nueva cita</h4>
26     </div>
27   </div>
28 </header>
```

```
30 <section class="container">
31   <div class="row">
32     <form onsubmit="return crearTabla()">
33       <fieldset id="diaCita">
34         <legend><h3>Día de la cita</h3></legend>
35         <input type="date" name="" id="dia" onchange="validarFecha()" required>
36       </fieldset>
37
38       <fieldset id="formularioParte2">
39         <legend><h3>Información del paciente</h3></legend>
40
41         <div class="row">
42           <div class="col-xs-6 col-sm-6 col-md-6 col-lg-6">
43             <label for="nombre" id="nombreLabel">Nombre</label>
44             <input type="text" name="nombre" id="nombre" pattern="[A-Za-z]+" maxlength="15" required>
45           </div>
46
47           <div class="col-xs-6 col-sm-6 col-md-6 col-lg-6">
48             <label for="apellidos" id="apellidosLabel">Apellidos</label>
49             <input type="text" name="apellidos" id="apellidos" pattern="[A-Za-z ]+" maxlength="30" required>
50           </div>
51         </div>
52
53         <div class="row">
54           <div class="col-xs-6 col-sm-6 col-md-6 col-lg-6">
55             <label for="dni" id="dniLabel">DNI</label>
56             <input type="text" name="dni" id="dni" pattern="[0-9]{8}[A-Z]{1}" maxlength="9" minlength="9" required>
57           </div>
58
59           <div class="col-xs-6 col-sm-6 col-md-6 col-lg-6">
60             <label for="telefono" id="telefonoLabel">Telefono</label>
61             <input type="text" name="telefono" id="telefono" pattern="[6]{1}[0-9]{8}" maxlength="9" minlength="9" required>
62           </div>
63         </div>
```

```
100
101 <footer class="container">
102   <div class="row">
103     <div class="col-xs-5 col-sm-6 col-md-7 col-lg-8">
104       <p id="pieNombre">MedacDent</p>
105     </div>
106
107     <div class="col-xs-4 col-sm-3 col-md-3 col-lg-2">
108       <p id="pieEmail">Email: medacdent@gmail.com</p>
109     </div>
110
111     <div class="col-xs-3 col-sm-3 col-md-2 col-lg-2">
112       <p id="pieTelefono">Teléfono: 640319934</p>
113     </div>
114   </div>
115 </footer>
116
117 <script src="/JS/TrabajoCliente.js"></script>
118
119 </body>
120 </html>
```

Se puede observar en las imágenes anteriores el uso de las clases **Bootstrap** para estilizar el proyecto y el responsive. He usado este framework en su versión última, la 5.3, junto a **CSS**. Los dos archivos HTML comparten el siguiente 'head' con los enlaces al framework, a las fuentes y al fichero de estilo, respectivamente.

```
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>Trabajo de Entorno Cliente. Formulario</title>
7
8   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-T3c6CoIi6uLn
9
10  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
11  <link href="https://fonts.googleapis.com/css2?family=Brygada+1918:wght@500&family=Gruppo&display=swap" rel="stylesheet">
12
13  <link rel="stylesheet" href="/CSS/TrabajoClienteFormulario.css">
```

El **JS** es largo. Contabilizo 16 funciones, un número considerable de líneas de código y un espacio que penaliza y excede el recomendado en las instrucciones de entrega.

La función 'inicio' nos da acceso desde la página de inicio al formulario.

```
24 function inicio() {
25   window.location.href = "/TrabajoClienteInicio.html";
26 }
```

'CargarDatos' es llamada desde el 'body' de la página formulario. Muestra la tabla con las citas o una tabla vacía, en función de si tenemos o no datos almacenados.

```
28 // Llamada a la función cargarDatos()
29 window.onload = cargarDatos;
```

```
38 function cargarDatos() {
39   let citas = JSON.parse(localStorage.getItem("citas")) || [];
40
41   if(citas.length == 0) {
42     mostrarTablaVacia();
43   } else {
44     mostrarTabla(citas);
45   }
46 }
```



```
47 function mostrarTablaVacia() {
48     let tabla = document.getElementById("tabla");
49
50     //thead
51     crearThead();
52
53     //tbody
54     let tbody = document.createElement("tbody");
55
56     let tr = document.createElement("tr");
57     let td = document.createElement("td");
58     let tdTextoVacio = document.createTextNode("No hay citas.");
59     td.appendChild(tdTextoVacio);
60     tr.appendChild(td);
61     tbody.appendChild(tr);
62
63     tabla.appendChild(tbody);
64 }
```

```
65 function crearThead() {
66     let thead = document.createElement("thead");
67
68     let thOrden = document.createElement("th");
69     let thTextoOrden = document.createTextNode("Orden de la cita");
70     thOrden.appendChild(thTextoOrden);
71     thead.appendChild(thOrden);
72
73     let thDia = document.createElement("th");
74     let thTextoDia = document.createTextNode("Día de la cita");
75     thDia.appendChild(thTextoDia);
76     thead.appendChild(thDia);
77
78     let thNombre = document.createElement("th");
79     let thTextoNombre = document.createTextNode("Nombre");
80     thNombre.appendChild(thTextoNombre);
81     thead.appendChild(thNombre);
82
83     let thApellido = document.createElement("th");
84     let thTextoApellido = document.createTextNode("Apellidos");
85     thApellido.appendChild(thTextoApellido);
86     thead.appendChild(thApellido);
87
88     let thDNI = document.createElement("th");
89     let thTextoDNI = document.createTextNode("DNI");
90     thDNI.appendChild(thTextoDNI);
91     thead.appendChild(thDNI);
92
93     let thTelefono = document.createElement("th");
94     let thTextoTelefono = document.createTextNode("Teléfono");
95     thTelefono.appendChild(thTextoTelefono);
96     thead.appendChild(thTelefono);
97
98     let thNacimiento = document.createElement("th");
99     let thTextoNacimiento = document.createTextNode("Fecha de nacimiento");
100    thNacimiento.appendChild(thTextoNacimiento);
101    thead.appendChild(thNacimiento);
}
```

```
103     let thObservaciones = document.createElement("th");
104     let thTextoObservaciones = document.createTextNode("Observaciones");
105     thObservaciones.appendChild(thTextoObservaciones);
106     thead.appendChild(thObservaciones);
107
108     tabla.appendChild(thead);
109 }
```

‘CrearTabla’ es una de las funciones principales. Es llamada desde un evento ‘onsubmit’ en el formulario. Crea y renderiza la tabla con las citas almacenadas. Incorpora otras funciones para almacenar la cita (‘[pedirCita](#)¹’) y mostrarlas en una tabla (‘[mostrarTabla](#)’).

```
121 function crearTabla() {
122     //1.
123     let diaCita = document.getElementById("dia").value;
124     let nombre = document.getElementById("nombre").value;
125     let apellidos = document.getElementById("apellidos").value;
126     let dni = document.getElementById("dni").value;
127     let telefono = document.getElementById("telefono").value;
128     let nacimiento = document.getElementById("nacimiento").value;
129     let observaciones = document.getElementById("observaciones").value;
130
131     //2.
132     let id = diaCita.concat(nombre);
133
134     //3.
135     let citas = JSON.parse(localStorage.getItem("citas")) || [];
136
137     //4.
138     pedirCita(citas, id, diaCita, nombre, apellidos, dni, telefono, nacimiento, observaciones);
139
140     //5.
141     localStorage.setItem("citas", JSON.stringify(citas));
142
143     //6.
144     mostrarTabla(citas);
145     return false;
146 }
```

¹ Ver apartado **Problemas**, página . La función `pedirCita` interfiere al modificar una cita.

```
167 function pedirCita(citas, id, diaCita, nombre, apellidos, dni, telefono, nacimiento, observaciones) {  
168     let citaPedida = false;  
169  
170     for(let i = 0; i < citas.length; i++) {  
171         for(let atributo in citas[i]) {  
172             if(atributo == "id") {  
173                 if(id == citas[i][atributo]) {  
174                     //alert("No puede pedir más de una cita.");  
175                     citaPedida = true;  
176                     return false;  
177                 }  
178             }  
179             if(atributo == "diaCita") {  
180                 if(!validarFecha(diaCita)) {  
181                     return false;  
182                 }  
183             }  
184             if(atributo == "nacimiento") {  
185                 if(!validarNacimiento(nacimiento)) {  
186                     return false;  
187                 }  
188             }  
189         }  
190     }  
191  
192     if(!citaPedida) {  
193         let nuevaCita = {  
194             id: id,  
195             diaCita: diaCita,  
196             nombre: nombre,  
197             apellidos: apellidos,  
198             dni: dni,  
199             telefono: telefono,  
200             nacimiento: nacimiento,  
201             observaciones: observaciones  
202         };  
203     }
```

```
204     citas.push(nuevaCita);  
205 }  
206  
207 limpiarCampos();  
208 limpiarBotonG();  
209 }
```

‘LimpiarCampos’ elimina de la pantalla las letras de los campos de formulario. ‘LimpiarBotonG’ hace lo mismo con el botón ‘Guardar los datos’ que aparece el modificar una cita. En las siguientes tomas, muestro las únicas validaciones que hago en JS, la del día de la fecha (‘validarFecha()’) y la del día de nacimiento (‘validarNacimiento()’).


```
224 function validarFecha(diaCita) {
225     let fechaHoy = new Date();
226
227     let dia = fechaHoy.getDate();
228     let mes = fechaHoy.getMonth() +1;
229     let año = fechaHoy.getFullYear();
230
231     dia = dia<10 ? `0${dia}` : dia;
232     mes = mes<10 ? `0${mes}` : mes;
233
234     let fechaFormato = `${año}-${mes}-${dia}`;
235     //-----
236     if(new Date(diaCita) < fechaHoy) {
237         let info = document.getElementById("info");
238         info.innerHTML = "Selecciona fecha correcta";
239         info.classList.add("final");
240         return false;
241     }
242     return true;
243 }
244 function validarNacimiento(nacimiento) {
245     let fechaHoy = new Date();
246
247     let dia = fechaHoy.getDate();
248     let mes = fechaHoy.getMonth() +1;
249     let año = fechaHoy.getFullYear();
250
251     dia = dia<10 ? `0${dia}` : dia;
252     mes = mes<10 ? `0${mes}` : mes;
253
254     let fechaFormato = `${año}-${mes}-${dia}`;
255     //-----
256     if(new Date(nacimiento) > fechaHoy) {
257         let info = document.getElementById("info");
258         info.innerHTML = "Selecciona fecha correcta";
259         info.classList.add("final");
260         return false;
261     }
262     return true;
```

```
268 function mostrarTabla(citas) {
269     limpiarTabla();
270     let idCita;
271
272     //thead
273     crearThead();
274
275     //tbody
276     let tbody = document.createElement("tbody");
277
278     for(let i = 0; i < citas.length; i++) {
279         let tr = document.createElement("tr");
280
281         //Columna de Orden de la cita
282         let tdOrden = document.createElement("td");
283         let tdOrdenTexto = document.createTextNode(i+1);
284         tdOrden.appendChild(tdOrdenTexto);
285         tr.appendChild(tdOrden)
286
287         //Las demás columnas
288         for(let atributo in citas[i]) {
289             if(atributo != "id") {
290                 let td = document.createElement("td");
291                 let tdTexto = document.createTextNode(citas[i][atributo]);
292                 td.appendChild(tdTexto);
293                 tr.appendChild(td);
294             } else {
295                 idCita = citas[i][atributo];
296             }
297         }
298
299         //Botones modificar y eliminar
300         let botonModificar = document.createElement("button");
301         botonModificar.innerHTML = "Modificar";
302         botonModificar.setAttribute("id", "botonM")
303         botonModificar.setAttribute("onclick", `modificar('${idCita}')`);
304         tr.appendChild(botonModificar);
305
306         let botonEliminar = document.createElement("button");
307         botonEliminar.innerHTML = "Eliminar";
308         botonEliminar.setAttribute("id", "botonE");
309         tr.appendChild(botonEliminar);
310
311         tbody.appendChild(tr);
312         tabla.appendChild(tbody);
313     }
314 }
315
316
317 }
```

Con 'modificar' el usuario puede alterar sus datos almacenados de cita en base a su 'id' único. Va unida a 'guardarCambios', que es la función que guarda las modificaciones.

```
330 function modificar(id) {  
331  
332     limpiarBotonG();  
333  
334     let citas = JSON.parse(localStorage.getItem("citas")) || [];  
335  
336     for(let i = 0; i < citas.length; i++) {  
337         if(id == citas[i]["id"]) {  
338  
339             idCita = citas[i]["id"];  
340  
341             let d = document.getElementById("dia");  
342             d.value = citas[i]["diaCita"];  
343  
344             let nom = document.getElementById("nombre");  
345             nom.value = citas[i]["nombre"];  
346  
347             let ape = document.getElementById("apellidos");  
348             ape.value = citas[i]["apellidos"];  
349  
350             let dn = document.getElementById("dni");  
351             dn.value = citas[i]["dni"];  
352  
353             let tel = document.getElementById("telefono");  
354             tel.value = citas[i]["telefono"];  
355  
356             let nac = document.getElementById("nacimiento");  
357             nac.value = citas[i]["nacimiento"];  
358  
359             let ob = document.getElementById("observaciones");  
360             ob.value = citas[i]["observaciones"];  
361  
362             break;  
363         }  
364     }  
365 }
```

```
367     let botonG = document.getElementById("botonG");  
368     let botonGuardar = document.createElement("button");  
369     botonGuardar.innerHTML = "Guardar los datos";  
370     botonGuardar.setAttribute("id", "botonGuardar");  
371     botonGuardar.addEventListener("click", function () {  
372         guardarCambios(idCita);  
373     })  
374     botonG.appendChild(botonGuardar);  
375  
376 }
```



```
396 function guardarCambios(id) {
397
398     limpiarTabla();
399
400     let citaInfo = false;
401
402     let diaCitaModificada = document.getElementById("dia").value;
403     let nombreModificado = document.getElementById("nombre").value;
404     let apellidosModificado = document.getElementById("apellidos").value;
405     let dniModificado = document.getElementById("dni").value;
406     let telefonoModificado = document.getElementById("telefono").value;
407     let nacimientoModificado = document.getElementById("nacimiento").value;
408     let observacionesModificado = document.getElementById("observaciones").value;
409     let idModificado = diaCitaModificada.concat(nombreModificado);
410
411     let citas = JSON.parse(localStorage.getItem("citas")) || [];
412
413     for(let i = 0; i < citas.length; i++) {
414         if(id == citas[i]["id"]) {
415             citas[i]["id"] = idModificado;
416             citas[i]["nombre"] = nombreModificado;
417             citas[i]["apellidos"] = apellidosModificado;
418             citas[i]["dni"] = dniModificado;
419             citas[i]["telefono"] = telefonoModificado;
420             citas[i]["observaciones"] = observacionesModificado;
421
422             if(diaCitaModificada) {
423                 if(!validarFecha(diaCitaModificada)) {
424                     return false;
425                 } else {
426                     citas[i]["diaCita"] = diaCitaModificada;
427                 }
428             }

```

```
430         if(nacimientoModificado) {
431             if(!validarNacimiento(nacimientoModificado)) {
432                 return false;
433             } else {
434                 citas[i]["nacimiento"] = nacimientoModificado;
435             }
436         }
437
438         localStorage.setItem("citas", JSON.stringify(citas));
439         citaInfo = true;
440         break;
441     }
442 }
443
444
445 limpiarBotonG();
446
447 limpiarCampos();
448
449 mostrarInfo(citaInfo);
450
451 mostrarTabla(citas);
452
453 return true;
454
455 }
```

'Eliminar' borra la cita almacenada.

```
471 function eliminar(id) {
472
473     let citaInfo = true;
474
475     let citas = JSON.parse(localStorage.getItem("citas")) || [];
476
477     for(let i = 0; i < citas.length; i++) {
478         if(id == citas[i]["id"]) {
479             citas.splice(i, 1);
480             localStorage.setItem("citas", JSON.stringify(citas));
481             citaInfo = false;
482             limpiarBotonG();
483             limpiarTabla();
484             limpiarCampos();
485             cargarDatos();
486             mostrarInfo(citaInfo);
487         }
488     }
489 }
490 }
```

'MostrarInfo()' ² añade una información en pantalla para alertar al usuario que sus datos de cita han sido modificados o eliminados.

```
501 function mostrarInfo(citaInfo) {
502
503     let info = document.getElementById("info");
504
505     if(info) {
506         if(citaInfo == true) {
507             info.innerHTML = "Cita modificada";
508             info.classList.add("final");
509         } else {
510             info.innerHTML = "Cita eliminada";
511             info.classList.add("final");
512         }
513     } else {
514         console.log("elemento no encontrado")
515     }
516 }
```

² Ver apartado **Problemas**, página .. . La función 'mostrarInfo' no actúa todas las veces que es llamada.



Por último, las tres funciones de limpieza que eliminan de la pantalla algunos elementos.

```
523 function limpiarCampos() {  
524     let dialLimpiar = document.getElementById("dia");  
525     dialLimpiar.value = "";  
526  
527     let nombreLimpiar = document.getElementById("nombre");  
528     nombreLimpiar.value = "";  
529  
530     let apellidosLimpiar = document.getElementById("apellidos");  
531     apellidosLimpiar.value = "";  
532  
533     let dniLimpiar = document.getElementById("dni");  
534     dniLimpiar.value = "";  
535  
536     let telefonoLimpiar = document.getElementById("telefono");  
537     telefonoLimpiar.value = "";  
538  
539     let nacimientoLimpiar = document.getElementById("nacimiento");  
540     nacimientoLimpiar.value = "";  
541  
542     let obLimpiar = document.getElementById("observaciones");  
543     obLimpiar.value = "";  
544 }
```

```
552 function limpiarTabla() {  
553     let tabla = document.getElementById("tabla");  
554     tabla.innerHTML = "";  
555 }
```

```
562 function limpiarBotonG() {  
563     let botonG = document.getElementById("botonG");  
564     botonG.innerHTML = "";  
565 }
```



2. Ejecución

El usuario entra en la web y ve la siguiente pantalla:



Entra en 'Nueva cita'.

MedacDent

Nueva cita

Día de la cita

dd/mm/aaaa

Información del paciente

Nombre

DNI

Fecha de nacimiento dd/mm/aaaa

Apellidos

Teléfono

Observaciones

Pedir cita Borrar los datos Atrás

Orden de la cita	Día de la cita	Nombre	Apellidos	DNI	Teléfono	Fecha de nacimiento	Observaciones
No hay citas.							



Observa el **formulario** con los datos para pedir la cita, **tres botones** (uno para confirmar, otro para borrar los datos y otro para volver a inicio), y la **tabla** de citas, donde aún no hay citas almacenadas. El usuario introduce sus datos.

Parece que está teniendo problemas. Debe introducir un DNI con el formato correcto.



Lo hace y, automáticamente, su cita aparece en la tabla como almacenada. Resulta que quiere modificar el día de la cita. Para ello clica en '**Modificar**'. Los datos de la cita vuelven a los campos del formulario listos para ser cambiados. Tan solo debe seleccionar el día y **guardar los datos** en el nuevo botón que aparece.

The screenshot shows the MedacDent web application interface. At the top, there's a header with the logo and navigation links. The main content area is titled 'Día de la cita' and features a date picker set to '21/01/2024'. Below this, the 'Información del paciente' section contains form fields for 'Nombre' (Jose), 'Apellidos' (Lopez Arribas), 'DNI' (45737919A), 'Telefono' (694549445), and 'Fecha de nacimiento' (01/01/2024). A 'Guardar los datos' button is prominently displayed. Below the form, there are three buttons: 'Pedir cita', 'Borrar los datos', and 'Atrás'. At the bottom, a table lists appointments. The first row shows an appointment for 'Jose Lopez Arribas' on '2024-01-19' with a 'Modificar' button.

Orden de la cita	Día de la cita	Nombre	Apellidos	DNI	Teléfono	Fecha de nacimiento	Observaciones
1	2024-01-19	Jose	Lopez Arribas	45737919A	694549445	2024-01-01	

La cita del paciente ha sido modificada.

The screenshot shows the MedacDent web application interface after the appointment has been modified. The 'Día de la cita' date picker now shows '21/01/2024'. The 'Información del paciente' section remains the same. The 'Cita modificada' button is now visible. The table at the bottom shows the appointment for 'Jose Lopez Arribas' on '2024-01-21' with a 'Modificar' button.

Orden de la cita	Día de la cita	Nombre	Apellidos	DNI	Teléfono	Fecha de nacimiento	Observaciones
1	2024-01-21	Jose	Lopez Arribas	45737919A	694549445	2024-01-01	



¿Y si se arrepiente y la quiere **eliminar**?

The screenshot shows a web browser window with the URL `127.0.0.1:5500/TrabajoClienteFormulario.html`. The page is titled 'MedacDent' and features a form for managing appointments. The form includes fields for 'Día de la cita' (Appointment day), 'Nombre' (Name), 'Apellidos' (Surnames), 'DNI', 'Teléfono' (Phone), 'Fecha de nacimiento' (Date of birth), and 'Observaciones' (Observations). Below the form, there are three buttons: 'Pedir cita' (Request appointment), 'Borrar los datos' (Delete data), and 'Atrás' (Back). At the bottom, a table displays the appointment details, showing 'No hay citas.' (No appointments).

Cita eliminada. No hay citas.

3. Problemas

A pesar de que el programa funciona: almacena, valida, modifica y elimina; hay un tipo de interferencia en la función `'pedirCita'` a la hora de modificar los datos. Creo que no debiera porqué haberlo. No obstante, no he acertado a resolver. Mi solución ha sido de pega, pues el problema sigue estando; solo eliminé un `'alert'` que informaba de que un mismo id no podía pedir más de una cita para el mismo día. Hasta esa línea profunda de función distinta penetra la aplicación a la hora de modificar.

¿Por qué la información que quiero mostrar al modificar y al eliminar una cita solo aparece la primera vez que se hacen tales acciones? Tampoco lo he podido detectar.

Ambos defectos permanecen.



4. Bibliografía

- Temario asignatura 'Desarrollo Web en Entorno Cliente', temas 1-9.
- W3School. <https://www.w3schools.com/>
- MDN web docs. <https://developer.mozilla.org/es/>
- OpenAI. (2023). ChatGPT.