

Informe de Laboratorio 03

Tema: NodeJS con express

Nota

Estudiante	Escuela	Asignatura
Paulo Andre Hidalgo Chinchay phidalgo@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación web 2 Semestre: III Código: 20223011

Laboratorio	Tema	Duración
03	NodeJS con express	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - A	Del 2 Junio 2023	Al 5 Junio 2023

1. Tarea

- Cree una aplicación NodeJS con express, para administrar una agenda personal.
Crear evento: fecha y hora. (Si ya existe el archivo no debería ingresar el evento)(La primera línea es el título del evento, las demás líneas son la descripción del evento.)
Editar evento. (Se muestran el archivo donde esta el detalle del evento)
Eliminar evento.
Ver eventos. Utilizar el formato árbol especificado anteriormente, donde debería incluirse sólo el título del evento.
- Interfaz que trabaja todo un CRUD en una sola vista. (Se pueden usar ventanas emergentes)

2. Equipos, materiales y temas utilizados

- Sistema Operativo Ubuntu GNU Linux 23 lunar 64 bits Kernell 6.2.
- Sistema Operativo Windows 11 pro versión 22H2 de 64 bits.
- VIM 9.0.
- Git 2.39.2.
- Visual Studio Code 1.78.2.
- Cuenta en GitHub con el correo institucional.

- NodeJS 18.1.0
- Ejercicios resueltos.
- <https://github.com/rescobedoq/pw2/tree/main/labs/lab03>
- Saber usar File System de NodeJS.
- https://www.w3schools.com/nodejs/nodejs_filesystem.asp

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- <https://github.com/PauloUNSA/pw2-lab-c-23a.git>
- URL para el laboratorio 03 en el Repositorio GitHub.
- <https://github.com/PauloUNSA/pw2-lab-c-23a/tree/main/lab3>

4. App Agenda en NodeJS con express

4.1. Configurar espacio de trabajo

- Con el comando npm se instalaron los siguientes frameworks:
- express, body-parser, markdown-it y cors.
- Además se crearon los archivos index.js e index.html. de prueba para ver si funcionaba correctamente el entorno.

Listing 1: Crear subcarpeta para instalar frameworks

```
$ mkdir -p /express/
```

Listing 2: Instalar frameworks dentro de express

```
$ cd /express  
$ npm install express body-parser markdown-it cors
```

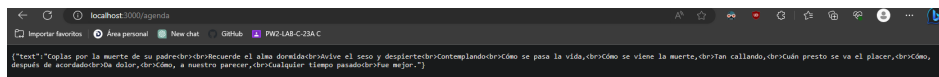
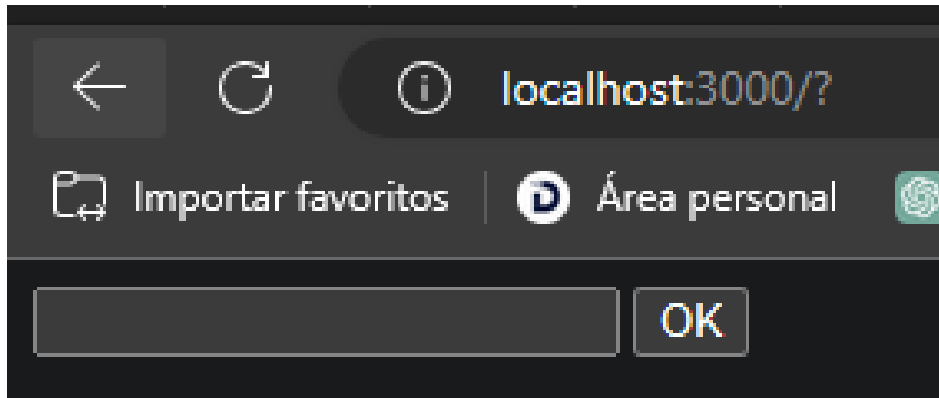
Listing 3: Index.js de prueba

```
1 const fs = require('fs')  
2 const path = require('path')  
3 const express = require('express')  
4 const app = express()  
5 app.use(express.static('pub'))  
6  
7 app.listen(3000, () => {  
8   console.log("Escuchando en: http://localhost:3000")  
9 });  
10  
11 app.get('/', (request, response) => {  
12   response.sendFile(path.resolve(__dirname, 'index.html'))  
13 })
```

```
14
15 app.get('/agenda', (request, response) => {
16     fs.readFile(path.resolve(__dirname, 'agenda/poema.txt'), 'utf8',
17         (err, data) => {
18             if (err) {
19                 console.error(err)
20                 response.status(500).json({
21                     error: 'message'
22                 })
23                 return
24             }
25             response.json({
26                 text: data.replace(/\n/g, '<br>')
27             })
28         })
29     //
30 })
31 /*
32 fs.appendFile('mynewfile1.txt', "global", function (err) {
33     if (err) throw err;
34     console.log('Saved!');
35 });
36 */
```

Listing 4: Index.html de prueba

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>Agenda</title>
8 </head>
9 <body>
10     <form id="form">
11         <input type="text" id="title">
12         <input type="submit" value="OK" onclick="enviarDatos()">
13     </form>
14     <!--<input type="text">--
15     <input type="submit" value="MOSTAR" onclick="mostrar()"-->
16
17     <div id="prueba"></div>
18
19     <script src="./script.js"></script>
20
21 </body>
22 </html>
```



- Como se observa funciona correctamente

4.2. Terminado con redireccion

- La página web estaba completa, sin embargo redireccionaba a otras paginas como eventos o borrar al completar un form o pedir datos como se muestra a continuación.

Listing 5: Index.js

```

1 const fs = require('fs')
2 const path = require('path')
3 const express = require('express')
4 const cors = require('cors');
5 const app = express()
6 const bp = require('body-parser')
7 app.use(cors());
8 app.use(express.static('./'))
9
10 app.use(bp.urlencoded({
11   extended: true
12 }))
13 app.use(express.urlencoded({
14   extended : false
15 }))
16 app.listen(3000, () => {
17   console.log("Escuchando en: http://localhost:3000")
18 });
19
20 app.get('/', (request, response) => {
21   response.sendFile(path.resolve(__dirname, 'index.html'))
22 })
23 app.get('/borrar', (request, response) => {
24   response.sendFile(path.resolve(__dirname, 'index.html'))
25 })
26 const agenda = path.join(__dirname, 'agenda');
27 var diaDir

```

```
28 app.post('/borrar', (request, response) => {
29   console.log(request.body)
30   const { date,time } = request.body;
31   diaDir = path.join(agenda, date);
32   const filePath = path.join(diaDir,`${time}.txt`);
33   fs.unlink(filePath, (err) => {
34     if (err) {
35       response.sendFile(path.resolve(__dirname, 'index.html'))
36     } else {
37       response.sendFile(path.resolve(__dirname, 'index.html'))
38     }
39   });
40 });
41 function eliminar(dir){
42   fs.readdir(dir, (err, files) => {
43     if (err) {
44       console.log(err);
45     } else {
46       console.log(files)
47       if (files===''||files.length == 0) {//si cumple elimina
48         console.log('La carpeta esta vacia'+files.length);
49         fs.rmdir(dir,{ recursive: true }, (err) => {
50           if (err) {
51             console.error(err);
52           } else {
53             console.log('La carpeta ha sido eliminada');
54           }
55         });
56       } else {
57         console.log('La carpeta no esta vacia');
58       }
59     }
60   });
61 }
62 app.post('/editar', (request, response) => {
63   console.log(request.body)
64   const { date,time,desc } = request.body;
65   const filePath2 = path.join(agenda, date,`${time}.txt`);
66
67   fs.writeFile(filePath2, desc, function (err) {
68     if (err) throw err;
69     console.log('Reemplazo correcto!');
70     response.sendFile(path.resolve(__dirname, 'index.html'))
71   });
72 });
73 app.post('/eventos/', (request, response) => {
74   console.log(request.body)
75   const {desc,fecha,hora} = request.body
76   console.log(hora)
77   //9:9 -9-9
78   const hora2 = hora.replace(':', '-')
79   console.log(hora2)
80   const filePath = path.join(agenda, fecha,`${hora2}.txt`);
81   if(desc != ''){
82     fs.access(filePath, fs.constants.F_OK, (err) => {
83       if (err) {
```

```

84     const content = desc;
85     fs.mkdir(path.join(agenda, fecha), { recursive: true }, (err) => {
86         if (err) {
87             response.sendFile(path.resolve(__dirname, 'index.html'))
88             //res.status(500).json({ error: 'Falla al crear evento' });
89         } else {
90             fs.writeFile(filePath, content, (err) => {
91                 if (err) {
92                     response.sendFile(path.resolve(__dirname, 'index.html'))
93                     //res.status(500).json({ error: 'Falla al crear evento' });
94                 } else {
95                     response.sendFile(path.resolve(__dirname, 'index.html'))
96                     //res.sendStatus(200);
97                 }
98             });
99         }
100     });
101     } else {
102         response.sendFile(path.resolve(__dirname, 'index.html'))
103         //res.status(409).json({ error: 'Evento ya existe' });
104     }
105     });
106     }else response.sendFile(path.resolve(__dirname, 'index.html'))
107 })
108 app.get('/eventos', (req, res) => {
109     if(diaDir){
110         eliminar(diaDir)
111     }
112     const agenda = readAgenda();
113     res.json(agenda);
114     });
115 function readAgenda() {
116     const agendaArr = [];
117     fs.readdirSync(agenda).forEach((date) => {
118         const datePath = path.join(agenda, date);
119         const events = fs.readdirSync(datePath).map((file) => {
120             const time = path.basename(file, '.txt');
121             const filePath = path.join(datePath, file);
122             const text = fs.readFileSync(filePath, 'utf8');
123             return { time, text };
124         });
125         agendaArr.push({ date, events });
126     });
127     return agendaArr;
128 }

```

Listing 6: Index.html terminado con redireccionamiento y js incrustado

```

1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>Agenda</title>
8 </head>

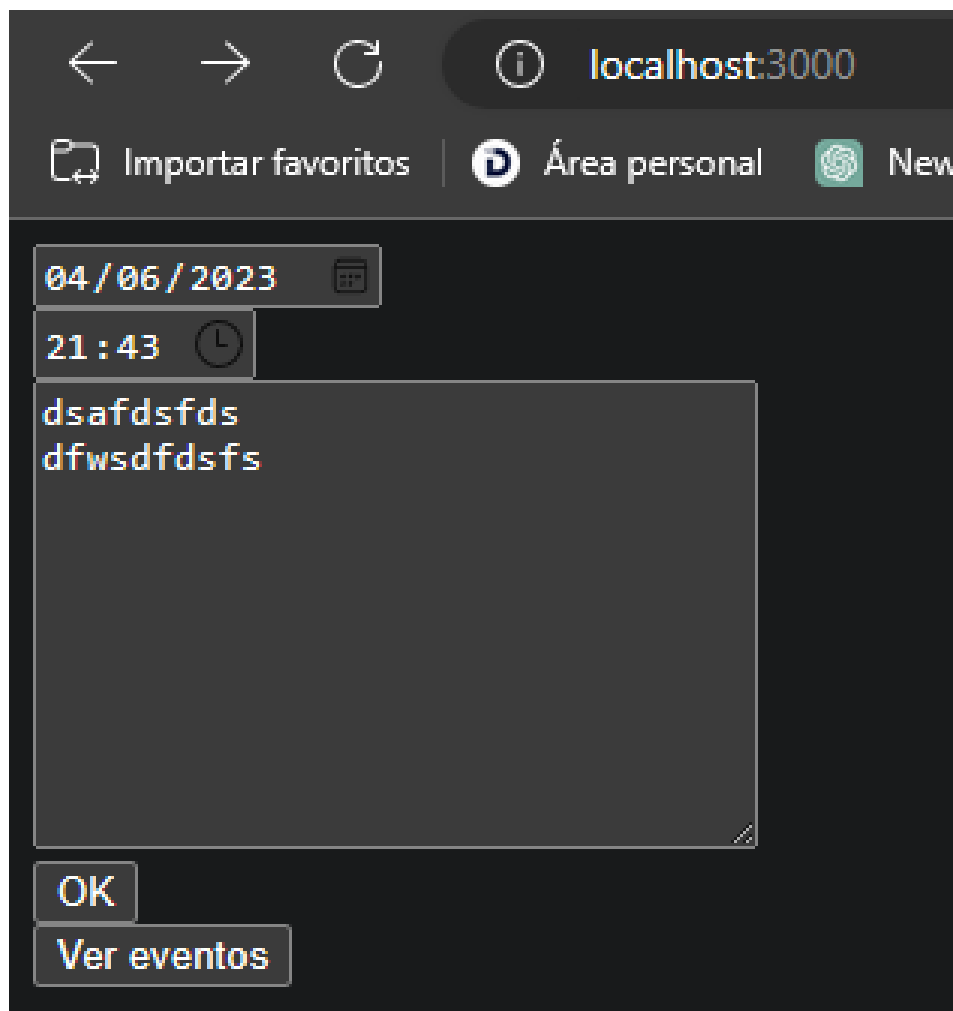
```

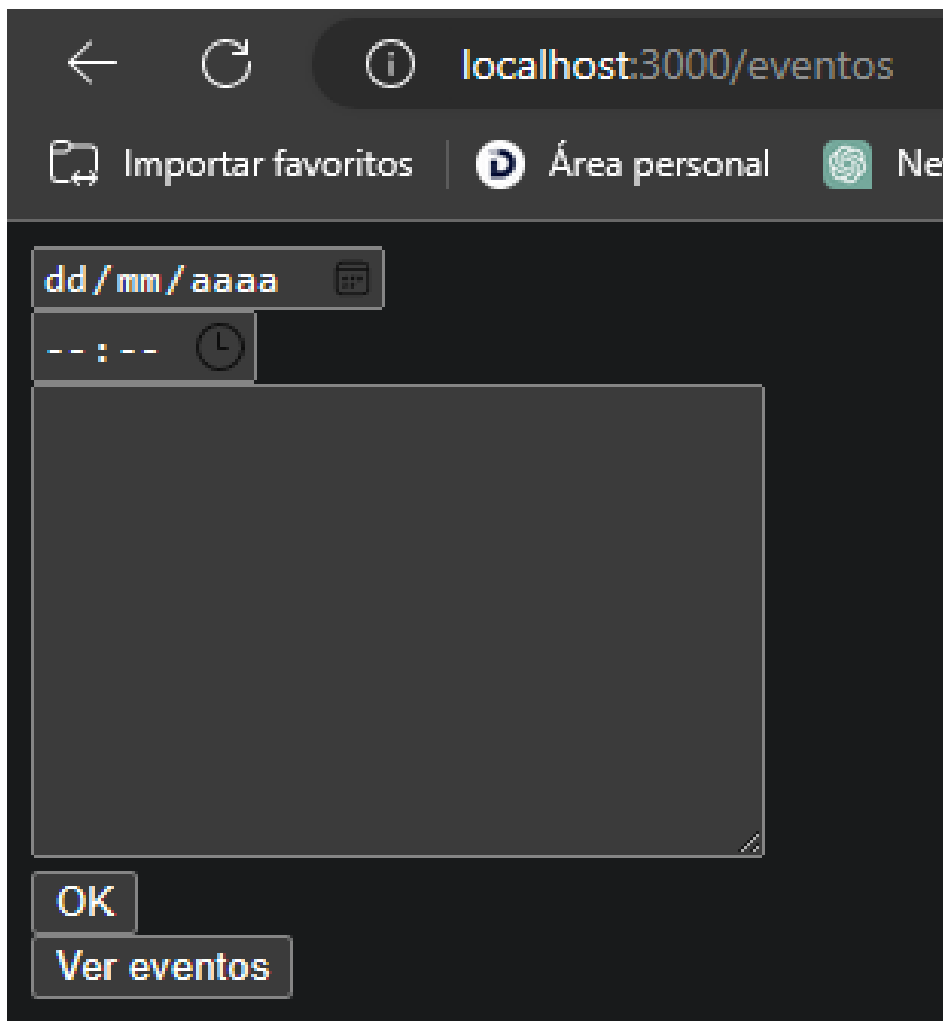
```

9 <body>
10 <form id="form" action="./eventos" method="post" >
11 <input type="date" name="fecha" required><br>
12 <input type="time" name="hora" required><br>
13 <textarea name="desc" cols="30" rows="10" required></textarea><br>
14 <input type="submit" value="OK">
15 </form>
16 <input type="button" value="Ver eventos" onclick="mostrar()" >
17 <div id="prueba"></div>
18 <div id="editar"></div>
19
20 <script>
21 function mostrar() {
22     const url = 'http://localhost:3000/eventos';
23     fetch(url, { mode: 'cors' })
24     .then(response => response.json())
25     .then(data => {
26         const agenda = data;
27         let html = '';
28         agenda.forEach(item => {
29             html += '<h3>${item.date}</h3>';
30             html += '<ul>';
31             item.events.forEach(event => {
32                 const salto = event.text.indexOf('\n')
33                 //console.log(salto)
34                 var titulo
35                 if(salto != -1){ titulo = event.text.substring(0,salto)}
36                 else{ titulo = event.text}
37                 let descrip = event.text.substring(salto+1, event.text.length)
38                 descrip = descrip.replace(/\r\n/g, "<br>")
39                 const texto = event.text.replace(/\r\n/g, "<br>")
40                 console.log(texto)
41
42                 html += '<form id="form" action="./borrar" method="post">
43                     <input type="button" value="${titulo}" onclick="print('${descrip}')">
44                     <input type="hidden" name="date" value="${item.date}">
45                     <input type="hidden" name="time" value="${event.time}">
46                     <input type="button" value="editar"
47                         onclick="editarFun('${item.date}','${event.time}','${texto}')">
48                     <input type="submit" value="eliminar">
49                 </form>';
50                 html += '</ul>';
51             });
52             document.querySelector("#prueba").innerHTML = html;
53         });
54     }
55     function print(texto){
56         document.querySelector("#editar").innerHTML = texto;
57     }
58     function editarFun(date, time, texto){
59         //console.log("entra aqui")
60         texto = texto.replace(<br>/g, "\r\n")
61         html = '<form id="form" action="./editar" method="post">
62             <input type="hidden" name="date" value="${date}">
63             <input type="hidden" name="time" value="${time}">

```

```
64         <textarea name="desc" cols="30" rows="10" required>${texto}</textarea><br>
65         <input type= "submit" value="confirmar" >
66     </form></li>‘
67     document.querySelector("#editar").innerHTML = html;
68 }
69 </script>
70 </body>
71 </html>
```





The image shows a web browser interface with a dark theme. The address bar displays "localhost:3000/eventos". Below the address bar, there are navigation links: "Importar favoritos", "Área personal", and "New". The main content area features a date and time picker. The date field is labeled "dd/mm/aaaa" and the time field is labeled "--:--". Below these fields is a large, empty rectangular area. At the bottom of the form, there are two buttons: "OK" and "Ver eventos".

dd/mm/aaaa

--:--

OK

Ver eventos

2023-05-26

dsfddad

editar

eliminar

2023-06-04

dsafdsfds

editar

eliminar

4.3. Terminado completamente

- Se borraron los `console.log()` en `index.js`.

- Se externalizo el script del html.
- Index.html completamente terminado y con estilo a parte.

Listing 7: Index.js limpio

```
1 const fs = require('fs')
2 const path = require('path')
3 const express = require('express')
4 const cors = require('cors');
5 const app = express()
6 const bp = require('body-parser')
7 app.use(cors());
8 app.use(express.static('./'))
9 app.use(bp.json())
10 app.use(bp.urlencoded({
11   extended: true
12 }))
13 app.use(express.urlencoded({
14   extended : false
15 }))
16 app.listen(3000, () => {
17   console.log("Escuchando en: http://localhost:3000")
18 });
19 const index = 'prueba.html'
20
21 app.get('/', (request, response) => {
22   response.sendFile(path.resolve(__dirname, index))
23 })
24 app.get('/borrar', (request, response) => {
25   response.sendFile(path.resolve(__dirname, index))
26 })
27 const agenda = path.join(__dirname, 'agenda');
28 var diaDir
29 app.post('/borrar', (request, response) => {
30   console.log('borrar '+request.body)
31   const { date,time } = request.body;
32   diaDir = path.join(agenda, date);
33   const filePath = path.join(diaDir,`${time}.txt`);
34   fs.unlink(filePath, (err) => {
35     if (err) {
36       response.status(500).json({ text: 'Falla al eliminar evento' });
37     } else {
38       response.status(200).json({ text: 'Se elimino el evento' });
39     }
40   });
41 });
42 function eliminar(dir){
43   fs.readdir(dir, (err, files) => {
44     if (err) {
45       console.log(err);
46     } else {
47       console.log(files)
48       if (files===''||files.length == 0) { //si cumple elimina
49         console.log('La carpeta esta vacia'+files.length);
50         fs.rmdir(dir,{ recursive: true }, (err) => {
51           if (err) {
```

```
52     console.error(err);
53   } else {
54     console.log('La carpeta ha sido eliminada');
55   }
56   });
57   } else {
58     console.log('La carpeta no esta vacia');
59   }
60   }
61   });
62 }
63 app.post('/editar', (request, response) => {
64   console.log(request.body)
65   const { date,time,desc } = request.body;
66   const filePath2 = path.join(agenda, date,`${time}.txt`);
67   fs.writeFile(filePath2, desc, function (err) {
68     if (err) throw err;
69     response.status(200).json({ text: 'Reemplazo correcto!' });
70     response.sendFile(path.resolve(__dirname, index))
71   });
72 });
73 app.post('/eventos/', (request, res) => {
74   console.log(request.body)
75   const {desc,fecha,hora} = request.body
76   const hora2 = hora.replace(':', '-')
77   const filePath = path.join(agenda, fecha,`${hora2}.txt`);
78   if(desc !== ''){
79     fs.access(filePath, fs.constants.F_OK, (err) => {
80       if (err) {
81         const content = desc;
82         fs.mkdir(path.join(agenda, fecha), { recursive: true }, (err) => {
83           if (err) {
84             res.status(500).json({ text: 'Falla al crear evento' });
85           } else {
86             fs.writeFile(filePath, content, (err) => {
87               if (err) {
88                 res.status(500).json({ text: 'Falla al crear evento' });
89               } else {
90                 res.status(200).json({ text: 'Se creo el evento' });
91               }
92             });
93           }
94         });
95       } else {
96         res.status(409).json({ text: 'Evento ya existe' });
97       }
98     });
99   }
100 });
101 app.get('/eventos', (req, res) => {
102   if(diaDir){
103     eliminar(diaDir)
104     diaDir = undefined
105   }
106   const agenda = readAgenda();
107   res.json(agenda);
```

```

108     });
109     function readAgenda() {
110         const agendaArr = [];
111         fs.readdirSync(agenda).forEach((date) => {
112             const datePath = path.join(agenda, date);
113             const events = fs.readdirSync(datePath).map((file) => {
114                 const time = path.basename(file, '.txt');
115                 const filePath = path.join(datePath, file);
116                 const text = fs.readFileSync(filePath, 'utf8');
117                 return { time, text };
118             });
119             agendaArr.push({ date, events });
120         });
121         return agendaArr;
122     }

```

Listing 8: Index.html terminado sin redireccionamiento con CSS y script externos

```

1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Agenda</title>
8      <link rel="stylesheet" href="./estilo.css">
9  </head>
10 <body>
11     <form id="formCrear" >
12         <input type="date" name="fecha" id="fecha" required><br>
13         <input type="time" name="hora" id="hora" required><br>
14         <textarea name="desc" id="desc" cols="30" rows="10" required></textarea><br>
15         <input type="submit" value="OK">
16     </form>
17     <input type="button" value="Ver eventos" onclick="mostrar()" >
18     <div id="show"></div>
19     <div id="editar"></div>
20     <script src="./script.js"></script>
21 </body>
22 </html>

```

Listing 9: script.js script de funciones para realizar acciones con el servidor NodeJS

```

1  var seMuestraEventos = false
2  document.querySelector('#formCrear').onsubmit = () => {
3      const desc = document.getElementById('desc').value
4      const fecha = document.getElementById('fecha').value
5      const hora = document.getElementById('hora').value
6      agregar(desc, fecha, hora)
7      document.querySelector('#formCrear').reset();
8      if(seMuestraEventos){mostrar()}
9      return false;
10 }
11 function agregar(desc, fecha, hora){
12     const url = 'http://localhost:3000/eventos'

```

```
13 const data = {
14   desc: desc,
15   fecha: fecha,
16   hora: hora
17 }
18 const request = {
19   method: 'POST', // Podria ser GET
20   headers: {
21     'Content-Type': 'application/json',
22   },
23   body: JSON.stringify(data),
24 }
25 fetch(url, request)
26 .then(response => response.json())
27 .then(data2 => {
28   print(data2.text)
29 })
30 .catch(error => {
31   console.error('Error:', error);
32 });
33 }
34 var i =0
35 function mostrar() {
36   i=0
37   seMuestraEventos = true
38   resetear()
39   const url = 'http://localhost:3000/eventos';
40   fetch(url, { mode: 'cors' })
41   .then(response => response.json())
42   .then(data => {
43     const agenda = data;
44     let html = '';
45     agenda.forEach(item => {
46       html += '<h3>${item.date}</h3>';
47       html += '<ul>';
48       item.events.forEach(event => {
49         const salto = event.text.indexOf("\n")
50         var titulo
51         let descrip
52         if(salto != -1){
53           titulo = event.text.substring(0,salto)
54           descrip = event.text.substring(salto+1, event.text.length)
55         }else{
56           titulo = event.text
57           descrip = 'No hay descripcion'
58         }
59         descrip = descrip.replace(/\n/g,"<br>")
60         const texto = event.text.replace(/\n/g,"<br>")
61         i++
62         html += '<form id="formBorrar${i}">
63           <input type="button" value="${titulo}" onclick="print('${descrip}')">
64           <input type="hidden" id="date${i}" value="${item.date}">
65           <input type="hidden" id="time${i}" value="${event.time}">
66           <input type="button" value="editar"
67             onclick="cargarEditor('${item.date}','${event.time}','${texto}')">
68           <input type="submit" value="eliminar"
```

```
68         style="background-color: red;
69         color: white;
70         padding: 10px 20px;
71         border: none;
72         cursor: pointer;">
73     </form><br>'//editar
74     });
75     html += '</ul>';
76     });
77     document.querySelector("#show").innerHTML = html;
78     if(i!=0){sePuedeBorrar();}
79     else{print("No hay eventos")}
80     });
81 }
82 function sePuedeBorrar(){
83     for (let index = 1; index <= i; index++) {
84         let nombre = '#formBorrar'+index
85         document.querySelector(nombre).onsubmit = () => {
86             eliminar(i)
87             return false;
88         }
89     }
90 }
91 function print(texto){
92     document.querySelector("#editar").innerHTML = texto;
93 }
94 function resetear(){
95     document.querySelector("#editar").innerHTML = '';
96 }
97 function cargarEditor(date, time,texto){
98     texto = texto.replace(/<br>/g,"\\r\\n")
99     html = '<form id="formEditor">
100         <input type="hidden" id="dateE" value="${date}">
101         <input type="hidden" id="timeE" value="${time}">
102         <textarea id="descE" cols="30" rows="10" required>${texto}</textarea><br>
103         <input type= "submit" value="confirmar">
104     </form>'
105     document.querySelector("#editar").innerHTML = html;
106     cargoEditor();
107     return false;
108 }//editar
109 function cargoEditor(){
110     document.querySelector('#formEditor').onsubmit = () => {
111         const date = document.getElementById('dateE').value
112         const time = document.getElementById('timeE').value
113         const desc2 = document.getElementById('descE').value
114         editar(date,time,desc2)
115         mostrar()
116         document.querySelector("#editar").innerHTML = '';
117         return false;
118     }//borrar
119 function eliminar(i){
120     const date = document.getElementById('date'+i).value
121     const time = document.getElementById('time'+i).value
122     borrar(date,time)
123     mostrar()
```

```

124     return false;
125 }
126 function borrar(date,time){
127     const url = 'http://localhost:3000/borrar'
128     const data = {
129         date: date,
130         time:time
131     }
132     const request = {
133         method: 'POST', // Podria ser GET
134         headers: {
135             'Content-Type': 'application/json',
136         },
137         body: JSON.stringify(data),
138     }
139     fetch(url, request)
140     .then(response => response.json())
141     .then(data2 => {
142         document.querySelector("#editar").innerHTML = data2.text;
143     });
144 }
145 function editar(date,time,desc){
146     const url = 'http://localhost:3000/editar'
147     const data = {
148         date: date,
149         time:time,
150         desc: desc
151     }
152     const request = {
153         method: 'POST', // Podria ser GET
154         headers: {
155             'Content-Type': 'application/json',
156         },
157         body: JSON.stringify(data),
158     }
159     fetch(url, request)
160     .then(response => response.json())
161     .then(data2 => {
162         //document.querySelector("#show").innerHTML = data2.text;
163         print(data2.text)
164     });
165 }

```

Listing 10: Hoja de estilos

```

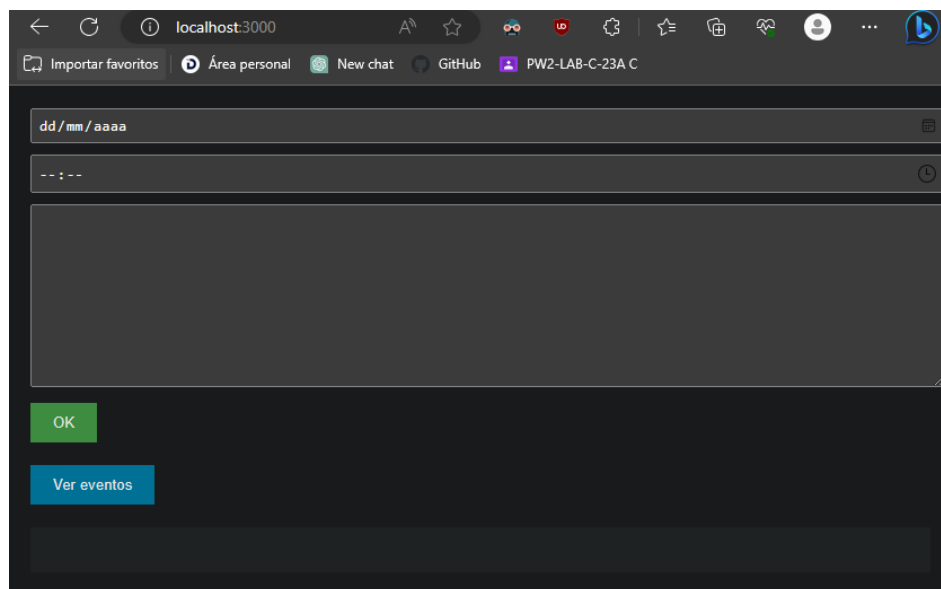
1 body {
2     font-family: Arial, sans-serif;
3     margin: 0;
4     padding: 20px;
5 }
6
7 /* Estilos del formulario */
8 #formCrear {
9     margin-bottom: 20px;
10 }
11

```



```
12 #formCrear input[type="date"],
13 #formCrear input[type="time"],
14 #formCrear textarea {
15     margin-bottom: 10px;
16     width: 100%;
17     padding: 5px;
18 }
19 #formEditar textarea {
20     margin-bottom: 10px;
21     width: 100%;
22     padding: 5px;
23 }
24 #formEditar input[type="submit"] {
25     background-color: #4CAF50;
26     color: white;
27     padding: 10px 20px;
28     border: none;
29     cursor: pointer;
30 }
31
32 #formCrear input[type="submit"] {
33     background-color: #4CAF50;
34     color: white;
35     padding: 10px 20px;
36     border: none;
37     cursor: pointer;
38 }
39
40 #formCrear input[type="submit"]:hover {
41     background-color: #45a049;
42 }
43
44 /* Estilos del boton "Ver eventos" */
45 input[type="button"] {
46     background-color: #008CBA;
47     color: white;
48     padding: 10px 20px;
49     border: none;
50     cursor: pointer;
51 }
52
53 input[type="button"]:hover {
54     background-color: #0077A7;
55 }
56
57 #show {
58     margin-bottom: 20px;
59 }
60
61 #editar {
62     background-color: #f2f2f2;
63     padding: 20px;
64     margin-bottom: 20px;
65 }
66
67 ul {
```

```
68 list-style-type: none;
69 padding: 0;
70 }
71
72 li {
73     background-color: #f9f9f9;
74     padding: 10px;
75     margin-bottom: 10px;
76 }
77
78 li:hover {
79     background-color: #f1f1f1;
80 }
81
82 li span {
83     font-weight: bold;
84     margin-right: 10px;
85 }
```



Se creó el evento

2023-06-04

Título	editar	eliminar

No hay eventos

4.4. Estructura de laboratorio 01

- El contenido que se entrega en este laboratorio es el siguiente:

```
//se omitieron las subcarpetas y archivos de node_modules al ocupar las de 500 lineas de
espacio
C:\USERS\PAULO\PW2-LAB-C-23A\LAB3
| |-----express
| |   estilo.css
| |   index.html
| |   index.js
| |   package-lock.json
| |   package.json
| |   script.js
| |
| |-----agenda
| |-----eventos
| |-----node_modules
| |   *
```

```
| |
| +-----priv
|     poema.txt
|
+-----latex
|     lab3_paulo-hidalgo.tex
|     lab3_paulo-hidalgo.pdf
|
|-----build
|     lab3_paulo-hidalgo.aux
|     lab3_paulo-hidalgo.fdb_latexmk
|     lab3_paulo-hidalgo.fls
|     lab3_paulo-hidalgo.log
|     lab3_paulo-hidalgo.out
|     lab3_paulo-hidalgo.pdf
|     lab3_paulo-hidalgo.synctex(busy)
|
|-----img
|     crea-evento.png
|     eliminar.png
|     l-eventos.png
|     localhost01.png
|     localhost02.png
|     localhost03.png
|     localhost_agenda.png
|     logo_abet.png
|     logo_episunsa.png
|     logo_unsa.jpg
|     Segundo-commit.png
|     ultimo-commit.png
|     ver-eventos01.png
|     ver-eventos02.png
|
+-----src
|     css01.css
|     index01.html
|     index02.html
|     index03.html
|     indexjs01.js
|     indexjs02.js
|     indexjs03.js
|     script.js
```

4.5. Pregunta: En el Ejemplo Hola Mundo con NodeJS. ¿Qué pasó con la línea: Content type ...?

- No esta debido a que no es necesario ya que devuelve una respuesta en JSON, que después es tomada por el cliente e insertada en un div de html ya existente.

5. Rúbricas

5.1. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumple con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos los ítems.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 1: Niveles de desempeño

Puntos	Nivel			
	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 2: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		20	

6. Referencias

- <https://github.com/rescobedoq/pw2/tree/main/labs/lab03>
- https://www.w3schools.com/nodejs/nodejs_filesystem.asp