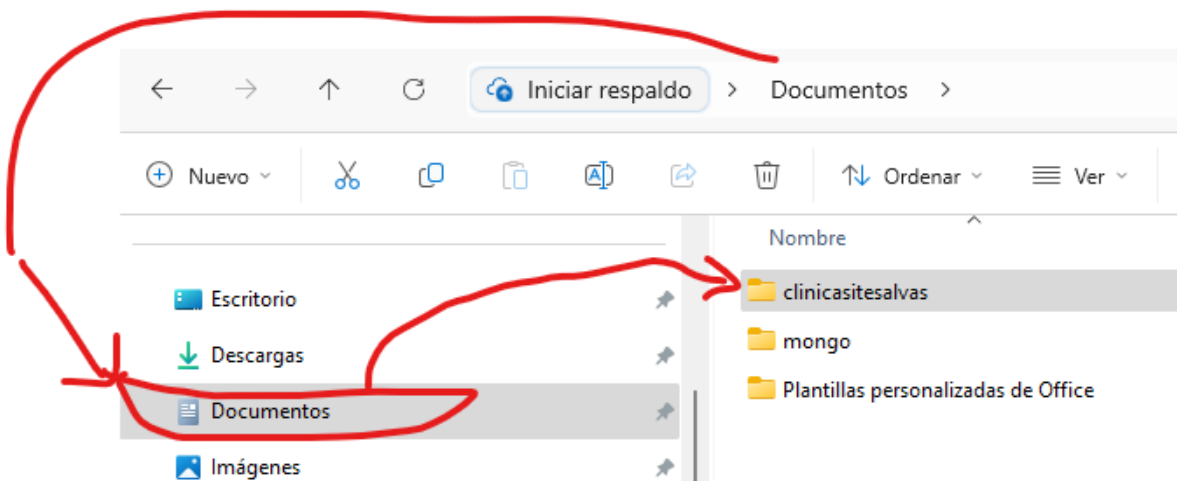
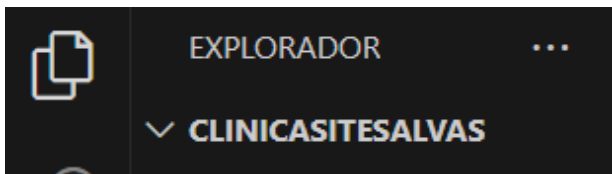


Ejercicio crear crud sin bases de datos

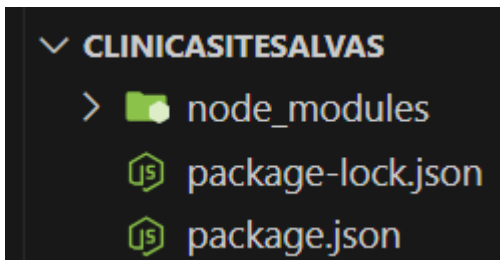
1 Crear carpeta raíz.(clnicasitesalvas)



2 conectar la carpeta raíz con visual code



3 instalar node al proyecto



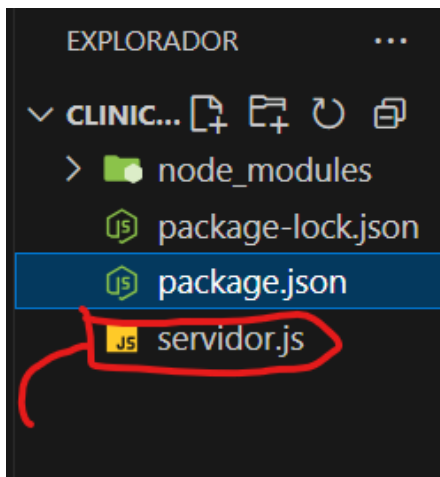
4 configurar package.json

```
package.json > ...  
1  {  
2    "name": "clnicasitesalvas",  
3    "version": "1.0.0",  
4    "main": "servidor.js",  
    Depurar  
5    "scripts": {  
6      "dev": "node --watch servidor.js"  
7    },
```

5 instalar librerías express y cors

```
package.json > ...  
1  {  
2    "name": "clnicasitesalvas",  
3    "version": "1.0.0",  
4    "main": "servidor.js",  
    Depurar  
5    "scripts": {  
6      "dev": "node --watch servidor.js"  
7    },  
8    "keywords": [],  
9    "author": "",  
10   "license": "ISC",  
11   "description": "",  
12   "dependencies": {  
13     "cors": "^2.8.5",  
14     "express": "^5.1.0",  
15     "save": "^2.9.0"  
16   }  
17 }
```

6 crear el archivo principal el mismo nombre configurado en la línea 4 del package.json



Codificar el archivo principal

1 llamar guardando en constate las librerías.

```
1  const express = require('express');
2  const cors = require('cors');      4.5k (gzipped: 1.9k)
3  const app = express();
```

2 crear los midelware

```
app.use(cors());
app.use(express.json());
app.use(express.urlencoded({ extended: true }));
```

3 crear las rutas y el microservicio de get

```
9  let medicos = [];
10
11  app.get('/medicos', (req, res) => {
12    res.json(medicos);
13  });
```

Ahora creamos el microservicio post

```

15 app.post('/medicos', (req, res) => {
16   const { idMedico, identificacion, nombres, telefono, correo } = req.body;
17
18   const existe = medicos.find(m => m.idMedico === idMedico);
19   if (existe) {
20     return res.status(400).json({ error: 'El médico ya existe' });
21   }
22
23   const nuevoMedico = { idMedico, identificacion, nombres, telefono, correo };
24   medicos.push(nuevoMedico);
25   res.status(201).json(nuevoMedico);
26 });

```

Ahora creamos el microservicio put

```

28 app.put('/medicos/:id', (req, res) => {
29   const id = req.params.id;
30   const { identificacion, nombres, telefono, correo } = req.body;
31
32   const medicoIndex = medicos.findIndex(m => m.idMedico === id);
33   if (medicoIndex === -1) {
34     return res.status(404).json({ error: 'Médico no encontrado' });
35   }
36
37   if (identificacion) medicos[medicoIndex].identificacion = identificacion;
38   if (nombres) medicos[medicoIndex].nombres = nombres;
39   if (telefono) medicos[medicoIndex].telefono = telefono;
40   if (correo) medicos[medicoIndex].correo = correo;
41
42   res.json(medicos[medicoIndex]);
43 });

```

Ahora creamos el microservicio delete

```

45 app.delete('/medicos/:idMedico', (req, res) => {
46   const id = req.params.idMedico;
47
48   const medicoIndex = medicos.findIndex(m => m.idMedico === id);
49   if (medicoIndex === -1) {
50     return res.status(404).json({ error: 'Médico no encontrado' });
51   }
52
53   const eliminado = medicos.splice(medicoIndex, 1);
54
55   res.json({ mensaje: 'Médico eliminado', medico: eliminado[0] });
56 });

```

Ahora crear el codigo para servidor

```
58 app.listen(3000, () => {
59   console.log('Servidor escuchando en http://localhost:3000');
60 });
```

Se prueba en el programa postman

POST http://localhost:3000/medicos

Body

Key	Value	Description
t1	102	
t2	fulanito Gonzalez	
t3	3001111112	
t4	fg@gmail.com	
t5	3	

Body

```
1 {
2   "idMedico": "3",
3   "identificacion": "102",
4   "nombres": "fulanito Gonzalez",
5   "telefono": "3001111112",
6   "correo": "fg@gmail.com"
7 }
```

201 Created · 4 ms · 390 B

Hay guardo un medico

Se prueba la ruta get

GET http://localhost:3000/medicos

Query Params

Key	Value	Description
Key	Value	Description

Body

```
1 [
2   {
3     "idMedico": "1",
4     "identificacion": "100",
5     "nombres": "pepito perez",
6     "telefono": "3001111111",
7     "correo": "pepito@gmail.com"
8   }
9 ]
```

200 OK · 4 ms · 386 B

Se prueba en el navegador



A screenshot of a web browser window. The address bar shows 'localhost:3000/medicos'. The page content displays a JSON array of three doctor objects. The first object has 'idMedico': '1', 'identificacion': '100', 'nombres': 'pepito perez', 'telefono': '3001111111', and 'correo': 'pepito@gmail.com'. The second object has 'idMedico': '2', 'identificacion': '101', 'nombres': 'juanito perez', 'telefono': '3001111112', and 'correo': 'juanito@gmail.com'. The third object has 'idMedico': '3', 'identificacion': '102', 'nombres': 'fulanito Gonzalez', 'telefono': '3001111112', and 'correo': 'fg@gmail.com'.

```
{
  {
    "idMedico": "1",
    "identificacion": "100",
    "nombres": "pepito perez",
    "telefono": "3001111111",
    "correo": "pepito@gmail.com"
  },
  {
    "idMedico": "2",
    "identificacion": "101",
    "nombres": "juanito perez",
    "telefono": "3001111112",
    "correo": "juanito@gmail.com"
  },
  {
    "idMedico": "3",
    "identificacion": "102",
    "nombres": "fulanito Gonzalez",
    "telefono": "3001111112",
    "correo": "fg@gmail.com"
  }
}
```

Probando el put

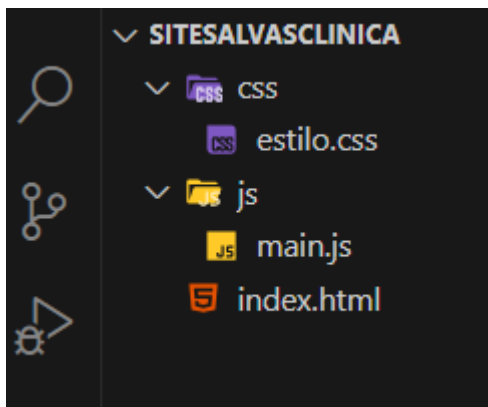
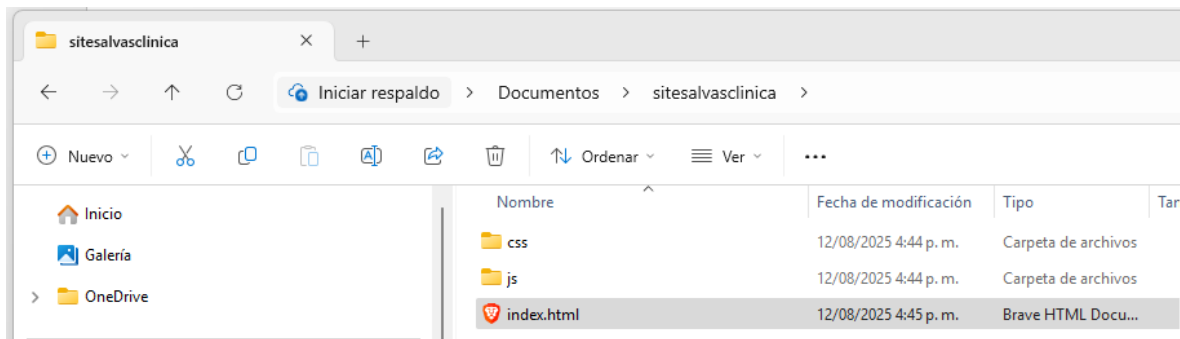


A screenshot of a JSON editor interface. At the top, there are tabs for 'JSON', 'Preview', and 'Visualize'. The 'JSON' tab is selected. Below the tabs, a single JSON object is displayed with the following fields: 'idMedico': '3', 'identificacion': '102', 'nombres': 'fulanito Gonzalez', 'telefono': '3001111112', and 'correo': 'fg@gmail.com'.

```
1 {
2   "idMedico": "3",
3   "identificacion": "102",
4   "nombres": "fulanito Gonzalez",
5   "telefono": "3001111112",
6   "correo": "fg@gmail.com"
7 }
```

Este se modificara

Carpeta frontend



Archivo index.html

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8" />
5   <meta name="viewport" content="width=device-width, initial-scale=1" />
6   <title>Gestión Médicos</title>
7   <link rel="stylesheet" href="css/estilo.css">
8 </head>
9 <body>
```

```
11 <h1>Gestión de Médicos</h1>
12
13 <form id="medicoForm">
14   <label for="idMedico">ID Médico</label>
15   <input type="text" id="idMedico" required />
16
17   <label for="identificacion">Identificación</label>
18   <input type="text" id="identificacion" required />
19
20   <label for="nombres">Nombres</label>
21   <input type="text" id="nombres" required />
22
23   <label for="telefono">Teléfono</label>
24   <input type="text" id="telefono" required />
25
26   <label for="correo">Correo</label>
27   <input type="email" id="correo" required />
28
29   <button type="submit">Guardar Médico</button>
30 </form>
```



```
30 <table>
31   <thead>
32     <tr>
33       <th>ID Médico</th>
34       <th>Identificación</th>
35       <th>Nombres</th>
36       <th>Teléfono</th>
37       <th>Correo</th>
38       <th>Acciones</th>
39     </tr>
40   </thead>
41   <tbody id="tablaMedicosBody">
42     <!-- Aquí van los médicos -->
43   </tbody>
44 </table>
45 <script src="js/main.js"></script>
46 </body>
47 </html>
```

Archivo estilo.css

```
1  body {
2      font-family: Arial, sans-serif;
3      max-width: 800px;
4      margin: 20px auto;
5      padding: 0 20px;
6      background: #f5f5f5;
7  }
8  h1 {
9      text-align: center;
10     color: #333;
11 }
12 form {
13     background: white;
14     padding: 15px;
15     border-radius: 6px;
16     margin-bottom: 30px;
17     box-shadow: 0 2px 5px rgba(0,0,0,0.1);
18 }
19 form input {
20     width: 100%;
21     padding: 8px;
22     margin: 6px 0 12px;
23     border: 1px solid #ccc;
24     border-radius: 4px;
25     box-sizing: border-box;
26 }
```

```
27 form button {
28     background-color: #28a745;
29     color: white;
30     border: none;
31     padding: 10px 15px;
32     border-radius: 4px;
33     cursor: pointer;
34     font-weight: bold;
35 }
36 form button:hover {
37     background-color: #218838;
38 }
39 table {
40     width: 100%;
41     border-collapse: collapse;
42     background: white;
43     box-shadow: 0 2px 5px rgba(0,0,0,0.1);
44 }
45 th, td {
46     padding: 12px;
47     border-bottom: 1px solid #ddd;
48     text-align: left;
49 }
50 th {
51     background-color: #007bff;
52     color: white;
53 }
54 tr:hover {
55     background-color: #f1f1f1;
56 }
```

```
57     button.edit-btn, button.delete-btn {
58         padding: 6px 10px;
59         border: none;
60         border-radius: 4px;
61         cursor: pointer;
62         font-size: 14px;
63     }
64     button.edit-btn {
65         background-color: #ffc107;
66         color: #212529;
67         margin-right: 6px;
68     }
69     button.edit-btn:hover {
70         background-color: #e0a800;
71     }
72     button.delete-btn {
73         background-color: #dc3545;
74         color: white;
75     }
76     button.delete-btn:hover {
77         background-color: #bd2130;
78     }
```

Crear el archivo js main.js

```
1  const apiUrl = 'http://localhost:3000/medicos';
2
3  const medicoForm = document.getElementById('medicoForm');
4  const idMedicoInput = document.getElementById('idMedico');
5  const identificacionInput = document.getElementById('identificacion');
6  const nombresInput = document.getElementById('nombres');
7  const telefonoInput = document.getElementById('telefono');
8  const correoInput = document.getElementById('correo');
9  const tablaMedicosBody = document.getElementById('tablaMedicosBody');
10
11  // Cargar médicos al iniciar la página
12  window.onload = () => {
13    cargarMedicos();
14  };
15
16  // Función para cargar los médicos y mostrarlos en la tabla
17  function cargarMedicos() {
18    fetch(apiUrl)
19      .then(res => res.json())
20      .then(data => {
21        tablaMedicosBody.innerHTML = ''; // limpiar tabla
22        data.forEach(medico => {
23          const tr = document.createElement('tr');
24          tr.innerHTML = `
25            <td>${medico.idMedico}</td>
26            <td>${medico.identificacion}</td>
27            <td>${medico.nombres}</td>
28            <td>${medico.telefono}</td>
29            <td>${medico.correo}</td>
```

```

30         <td>
31             <button class="edit-btn" onclick="editarMedico('${medico.idMedico}')">Editar</button>
32             <button class="delete-btn" onclick="eliminarMedico('${medico.idMedico}')">Eliminar</button>
33         </td>
34     `;
35     tablaMedicosBody.appendChild(tr);
36 });
37 })
38 .catch(err => alert('Error cargando médicos: ' + err));
39 }
40
41 // Guardar o actualizar médico
42 medicoForm.addEventListener('submit', e => {
43     e.preventDefault();
44
45     const idMedico = idMedicoInput.value.trim();
46     const identificacion = identificacionInput.value.trim();
47     const nombres = nombresInput.value.trim();
48     const telefono = telefonoInput.value.trim();
49     const correo = correoInput.value.trim();
50
51     if (!idMedico || !identificacion || !nombres || !telefono || !correo) {
52         alert('Por favor complete todos los campos.');
```

```

56     const medicoData = { idMedico, identificacion, nombres, telefono, correo };
57
58     // Si existe el ID, hacemos PUT (editar); si no, POST (crear)
59     const metodo = existeMedico(idMedico) ? 'PUT' : 'POST';
60     const url = metodo === 'POST' ? apiUrl : `${apiUrl}/${idMedico}`;
61
62     fetch(url, {
63         method: metodo,
64         headers: { 'Content-Type': 'application/json' },
65         body: JSON.stringify(medicoData),
66     })
67     .then(res => {
68         if (!res.ok) return res.json().then(e => Promise.reject(e.error));
69         return res.json();
70     })
71     .then(() => {
72         alert(metodo === 'POST' ? 'Médico agregado' : 'Médico actualizado');
73         limpiarFormulario();
74         cargarMedicos();
75     })
76     .catch(err => alert('Error: ' + err));
77 });
```

```

59   const metodo = existeMedico(idMedico) ? 'PUT' : 'POST';
60   const url = metodo === 'POST' ? apiUrl : `${apiUrl}/${idMedico}`;
61
62   fetch(url, {
63     method: metodo,
64     headers: { 'Content-Type': 'application/json' },
65     body: JSON.stringify(medicoData),
66   })
67     .then(res => {
68       if (!res.ok) return res.json().then(e => Promise.reject(e.error));
69       return res.json();
70     })
71     .then(() => {
72       alert(metodo === 'POST' ? 'Médico agregado' : 'Médico actualizado');
73       limpiarFormulario();
74       cargarMedicos();
75     })
76     .catch(err => alert('Error: ' + err));
77   });
78
79   // Verifica si ya existe un médico con ese ID (solo en frontend para saber si es PUT o POST)
80   function existeMedico(idMedico) {
81     return Array.from(tablaMedicosBody.children).some(
82       row => row.children[0].textContent === idMedico
83     );

```

```

84   }
85
86   // Llenar el formulario para editar
87   function editarMedico(idMedico) {
88     fetch(apiUrl)
89       .then(res => res.json())
90       .then(data => {
91         const medico = data.find(m => m.idMedico === idMedico);
92         if (!medico) {
93           alert('Médico no encontrado');
94           return;
95         }
96         idMedicoInput.value = medico.idMedico;
97         identificacionInput.value = medico.identificacion;
98         nombresInput.value = medico.nombres;
99         telefonoInput.value = medico.telefono;
100        correoInput.value = medico.correo;
101      });
102   }
103

```

