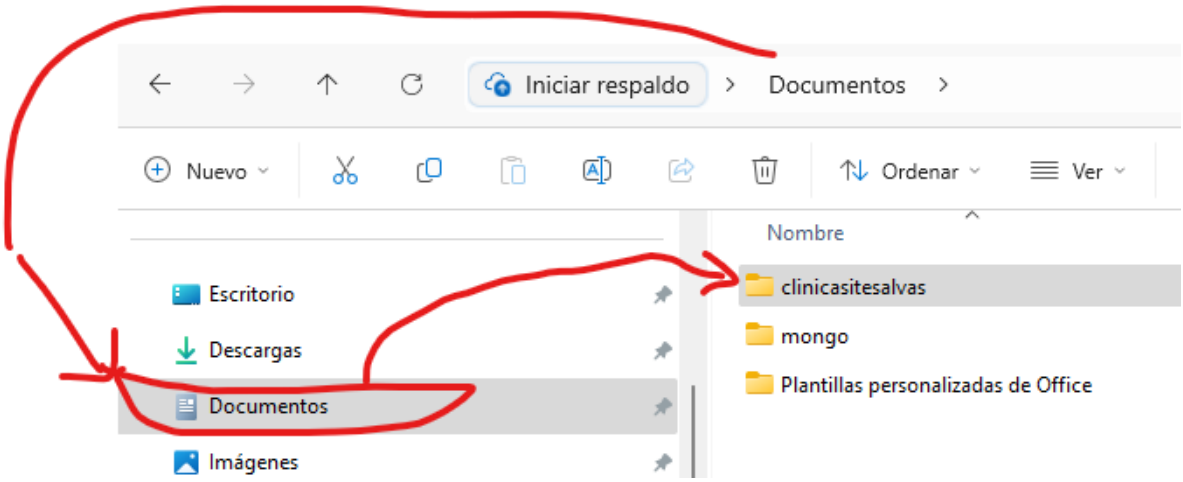
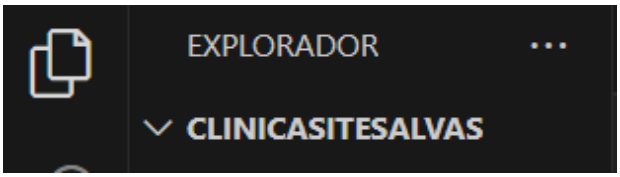
Ejercicio crear crud sin bases de datos

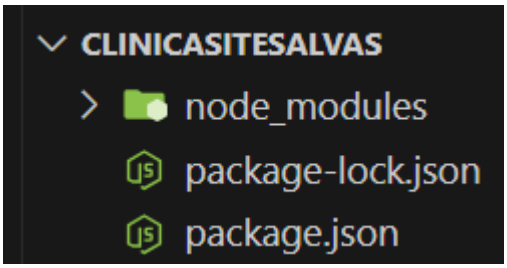
1 Crear carpeta raíz.(clinicasitesalvas)



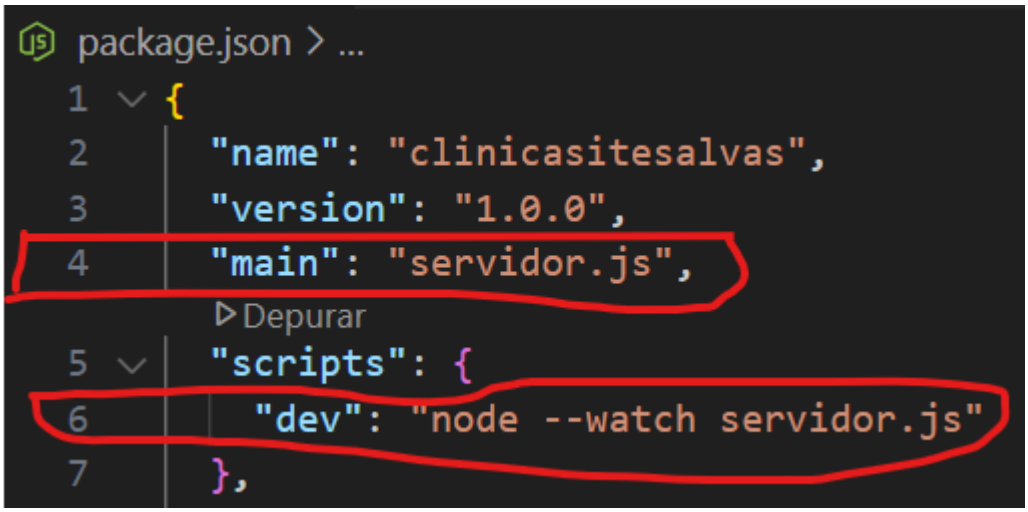
2 conectar la carpeta raíz con visual code



3 instalar node al proyecto

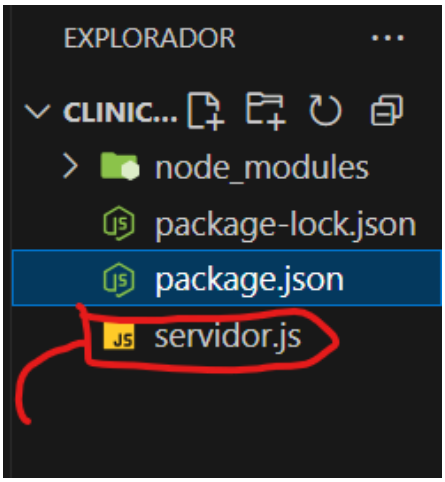


4 configurar package.json



5 instalar librerías express y cors

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

```
app.post('/medicos', (req, res) => {
  const { idMedico, identificacion, nombres, telefono, correo } = req.body;

  const existe = medicos.find(m => m.idMedico === idMedico);
  if (existe) {
    return res.status(400).json({ error: 'El médico ya existe' });
  }

  const nuevoMedico = { idMedico, identificacion, nombres, telefono, correo };
  medicos.push(nuevoMedico);
  res.status(201).json(nuevoMedico);
});
```

Ahora creamos el microservicio put

```
app.put('/medicos/:id', (req, res) => {
  const id = req.params.id;
  const { identificacion, nombres, telefono, correo } = req.body;

  const medicoIndex = medicos.findIndex(m => m.idMedico === id);
  if (medicoIndex === -1) {
    return res.status(404).json({ error: 'Médico no encontrado' });
  }

  if (identificacion) medicos[medicoIndex].identificacion = identificacion;
  if (nombres) medicos[medicoIndex].nombres = nombres;
  if (telefono) medicos[medicoIndex].telefono = telefono;
  if (correo) medicos[medicoIndex].correo = correo;

  res.json(medicos[medicoIndex]);
});
```

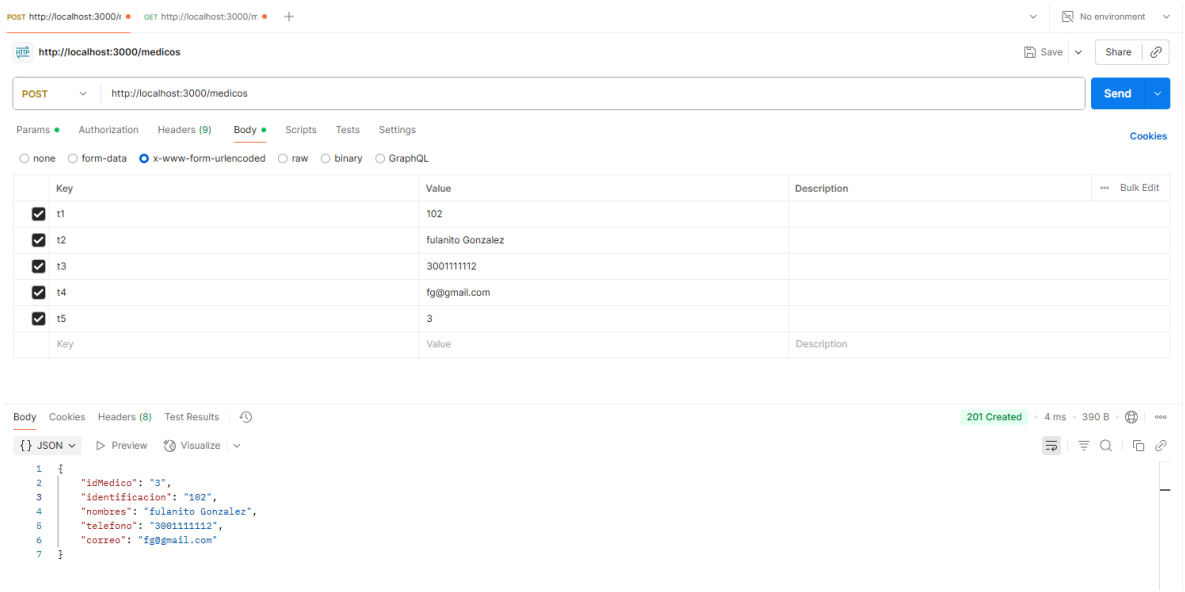
Ahora creamos el microservicio delete

```
app.delete('/medicos/:idMedico', (req, res) => {
  const id = req.params.idMedico;

  const medicoIndex = medicos.findIndex(m => m.idMedico === id);
  if (medicoIndex === -1) {
    return res.status(404).json({ error: 'Médico no encontrado' });
  }

  const eliminado = medicos.splice(medicoIndex, 1);

  res.json({ mensaje: 'Médico eliminado', medico: eliminado[0] });
});
```

Ahora crear el codigo para servidor

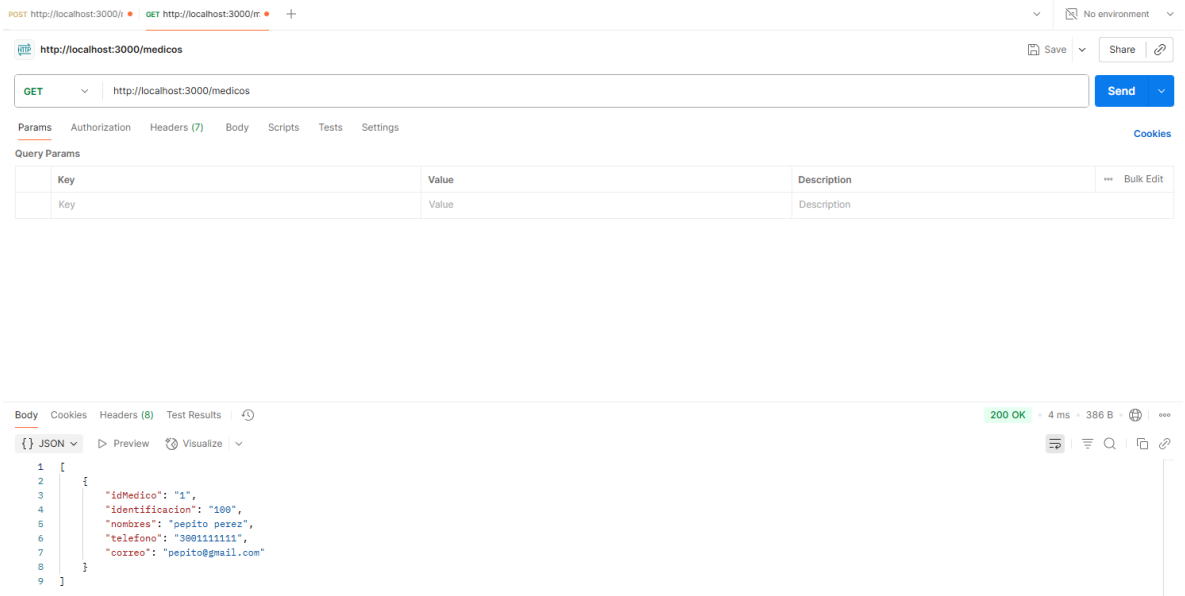
```
app.listen(3000, () => {
  console.log('Servidor escuchando en http://localhost:3000');
});
```

Se prueba en el programa postman



Hay guardo un medico

Se prueba la ruta get

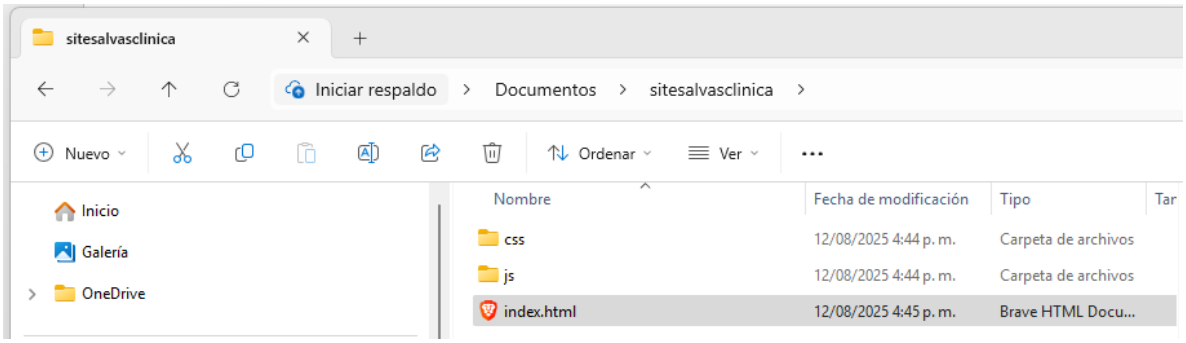


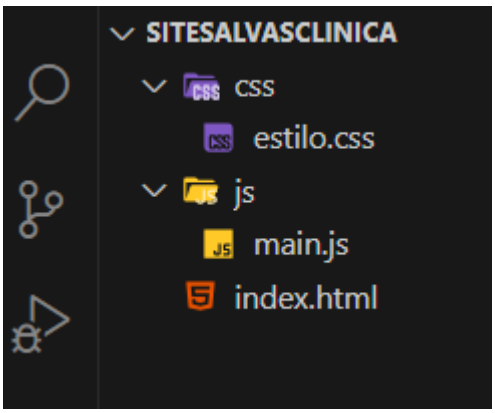
Se prueba en el navegador

Probando el put

Este se modificara

Carpeta frontend





Archivo index.html

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

```
  <h1>Gestión de Médicos</h1>

 <form id="medicoForm">
  <label for="idMedico">ID Médico</label>
  <input type="text" id="idMedico" required />

  <label for="identificacion">Identificación</label>
  <input type="text" id="identificacion" required />

  <label for="nombres">Nombres</label>
  <input type="text" id="nombres" required />

  <label for="telefono">Teléfono</label>
  <input type="text" id="telefono" required />

  <label for="correo">Correo</label>
  <input type="email" id="correo" required />

  <button type="submit">Guardar Médico</button>
</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

```css
body {
    font-family: Arial, sans-serif;
    max-width: 800px;
    margin: 20px auto;
    padding: 0 20px;
    background: #f5f5f5;
}
h1 {
    text-align: center;
    color: #333;
}
form {
    background: white;
    padding: 15px;
    border-radius: 6px;
    margin-bottom: 30px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
}
form input {
    width: 100%;
    padding: 8px;
    margin: 6px 0 12px;
    border: 1px solid #ccc;
    border-radius: 4px;
    box-sizing: border-box;
}
```

```
    form button {
      background-color: #28a745;
      color: white;
      border: none;
      padding: 10px 15px;
      border-radius: 4px;
      cursor: pointer;
      font-weight: bold;
    }
    form button:hover {
      background-color: #218838;
    }
    table {
      width: 100%;
      border-collapse: collapse;
      background: white;
      box-shadow: 0 2px 5px rgba(0,0,0,0.1);
    }
    th, td {
      padding: 12px;
      border-bottom: 1px solid #ddd;
      text-align: left;
    }
    th {
      background-color: #007bff;
      color: white;
    }
    tr:hover {
      background-color: #f1f1f1;
    }
```

```
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

```
const apiUrl = 'http://localhost:3000/medicos';

const medicoForm = document.getElementById('medicoForm');
const idMedicoInput = document.getElementById('idMedico');
const identificacionInput = document.getElementById('identificacion');
const nombresInput = document.getElementById('nombres');
const telefonoInput = document.getElementById('telefono');
const correoInput = document.getElementById('correo');
const tablaMedicosBody = document.getElementById('tablaMedicosBody');

// Cargar médicos al iniciar la página
window.onload = () => {
  cargarMedicos();
};

// Función para cargar los médicos y mostrarlos en la tabla
function cargarMedicos() {
  fetch(apiUrl)
    .then(res => res.json())
    .then(data => {
      tablaMedicosBody.innerHTML = ''; // limpiar tabla
      data.forEach(medico => {
        const tr = document.createElement('tr');
        tr.innerHTML = `
          <td>${medico.idMedico}</td>
          <td>${medico.identificacion}</td>
          <td>${medico.nombres}</td>
          <td>${medico.telefono}</td>
          <td>${medico.correo}</td>
```

```
            <td>
              <button class="edit-btn" onclick="editarMedico('${medico.idMedico}')">Editar</button>
              <button class="delete-btn" onclick="eliminarMedico('${medico.idMedico}')">Eliminar</button>
            </td>
          `;
          tablaMedicosBody.appendChild(tr);
        });
      })
      .catch(err => alert('Error cargando médicos: ' + err));
  }

  // Guardar o actualizar médico
  medicoForm.addEventListener('submit', e => {
    e.preventDefault();

    const idMedico = idMedicoInput.value.trim();
    const identificacion = identificacionInput.value.trim();
    const nombres = nombresInput.value.trim();
    const telefono = telefonoInput.value.trim();
    const correo = correoInput.value.trim();

    if (!idMedico || !identificacion || !nombres || !telefono || !correo) {
      alert('Por favor complete todos los campos.');
      return;
    }
```

```
    const medicoData = { idMedico, identificacion, nombres, telefono, correo };

    // Si existe el ID, hacemos PUT (editar); si no, POST (crear)
    const metodo = existeMedico(idMedico) ? 'PUT' : 'POST';
    const url = metodo === 'POST' ? apiUrl : `${apiUrl}/${idMedico}`;

    fetch(url, {
      method: metodo,
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(medicoData),
    })
      .then(res => {
        if (!res.ok) return res.json().then(e => Promise.reject(e.error));
        return res.json();
      })
      .then(() => {
        alert(metodo === 'POST' ? 'Médico agregado' : 'Médico actualizado');
        limpiarFormulario();
        cargarMedicos();
      })
      .catch(err => alert('Error: ' + err));
  });
```

```
  const metodo = existeMedico(idMedico) ? 'PUT' : 'POST';
  const url = metodo === 'POST' ? apiUrl : `${apiUrl}/${idMedico}`;

  fetch(url, {
    method: metodo,
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(medicoData),
  })
    .then(res => {
      if (!res.ok) return res.json().then(e => Promise.reject(e.error));
      return res.json();
    })
    .then(() => {
      alert(metodo === 'POST' ? 'Médico agregado' : 'Médico actualizado');
      limpiarFormulario();
      cargarMedicos();
    })
    .catch(err => alert('Error: ' + err));
});

// Verifica si ya existe un médico con ese ID (solo en frontend para saber si es PUT o POST)
function existeMedico(idMedico) {
  return Array.from(tablaMedicosBody.children).some(
    row => row.children[0].textContent === idMedico
  );
```

```
  }

  // Llenar el formulario para editar
  function editarMedico(idMedico) {
    fetch(apiUrl)
      .then(res => res.json())
      .then(data => {
        const medico = data.find(m => m.idMedico === idMedico);
        if (!medico) {
          alert('Médico no encontrado');
          return;
        }
        idMedicoInput.value = medico.idMedico;
        identificacionInput.value = medico.identificacion;
        nombresInput.value = medico.nombres;
        telefonoInput.value = medico.telefono;
        correoInput.value = medico.correo;
      });
  }
```

```
104  // Eliminar médico
105  function eliminarMedico(idMedico) {
106    if (!confirm('¿Seguro que quieres eliminar este médico?')) return;
107
108    fetch(`${apiUrl}/${idMedico}`, { method: 'DELETE' })
109      .then(res => {
110        if (!res.ok) throw new Error('Error al eliminar médico');
111        return res.json();
112      })
113      .then(() => {
114        alert('Médico eliminado');
115        cargarMedicos();
116      })
117      .catch(err => alert(err));
118  }
119
120  // Limpiar formulario
121  function limpiarFormulario() {
122    idMedicoInput.value = '';
123    identificacionInput.value = '';
124    nombresInput.value = '';
125    telefonoInput.value = '';
126    correoInput.value = '';
127  }
```