



UNIVERSIDAD TÉCNICA DE MANABÍ

FACULTAD DE CIENCIAS INFOMÁTICAS

01/07/2024

SOFTWARE 6

DESARROLLO DE APLICACIONES WEB

ING. EDISON SOLÓRZANO

ALCIVAR FUENTES HANSEL ADRIÁN

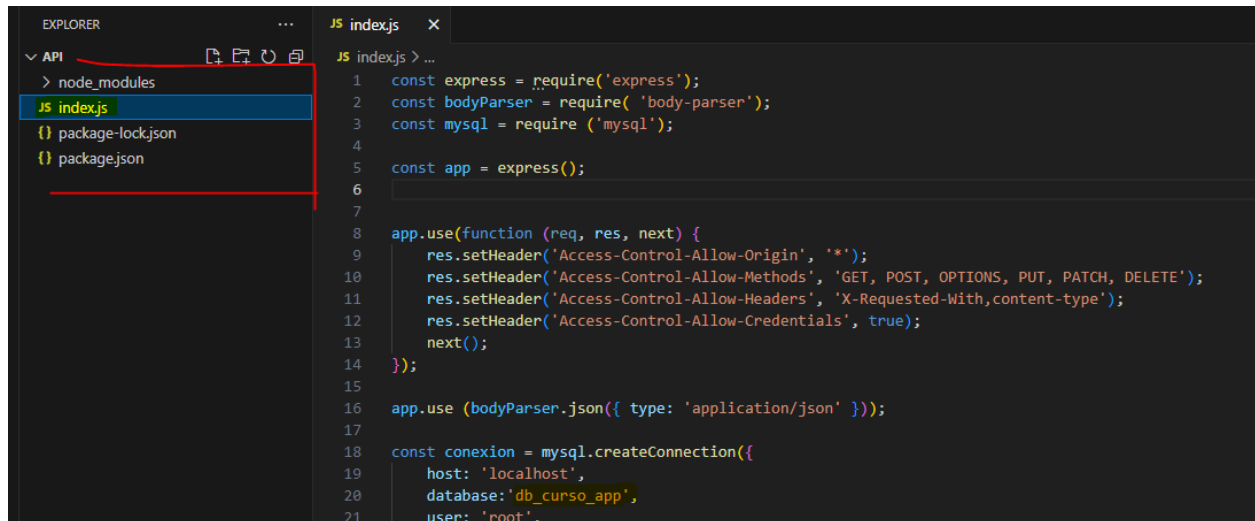
API REST con NODE JS

Desarrollar una Api Rest con Node JS.

Crear un api con node JS para realizar la conexión con la base de datos y realizar envío de datos con la herramienta Postman.

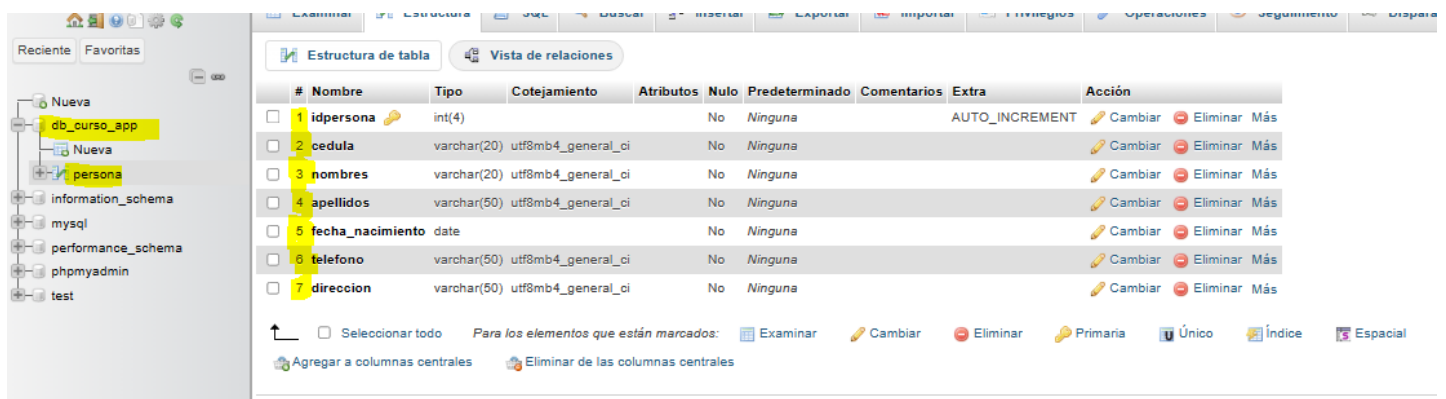
Desarrollo

1. Desarrollamos el archivo [api] “index.js” el cual contendrán las apis para realizar las funciones necesarias-



```
1  const express = require('express');
2  const bodyParser = require('body-parser');
3  const mysql = require('mysql');
4
5  const app = express();
6
7
8  app.use(function (req, res, next) {
9    res.setHeader('Access-Control-Allow-Origin', '*');
10   res.setHeader('Access-Control-Allow-Methods', 'GET, POST, OPTIONS, PUT, PATCH, DELETE');
11   res.setHeader('Access-Control-Allow-Headers', 'X-Requested-With,content-type');
12   res.setHeader('Access-Control-Allow-Credentials', true);
13   next();
14 });
15
16 app.use(bodyParser.json({ type: 'application/json' }));
17
18 const conexion = mysql.createConnection({
19   host: 'localhost',
20   database: 'db_curso_app',
21   user: 'root',
```

2. Creación de la base de datos llamada “db_curso_app”. Creación de la tabla llamada “persona” con 7 campos.



#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	idpersona	int(4)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Más
2	cedula	varchar(20)	utf8mb4_general_ci		No	Ninguna			Cambiar Eliminar Más
3	nombres	varchar(20)	utf8mb4_general_ci		No	Ninguna			Cambiar Eliminar Más
4	apellidos	varchar(50)	utf8mb4_general_ci		No	Ninguna			Cambiar Eliminar Más
5	fecha_nacimiento	date			No	Ninguna			Cambiar Eliminar Más
6	telefono	varchar(50)	utf8mb4_general_ci		No	Ninguna			Cambiar Eliminar Más
7	direccion	varchar(50)	utf8mb4_general_ci		No	Ninguna			Cambiar Eliminar Más

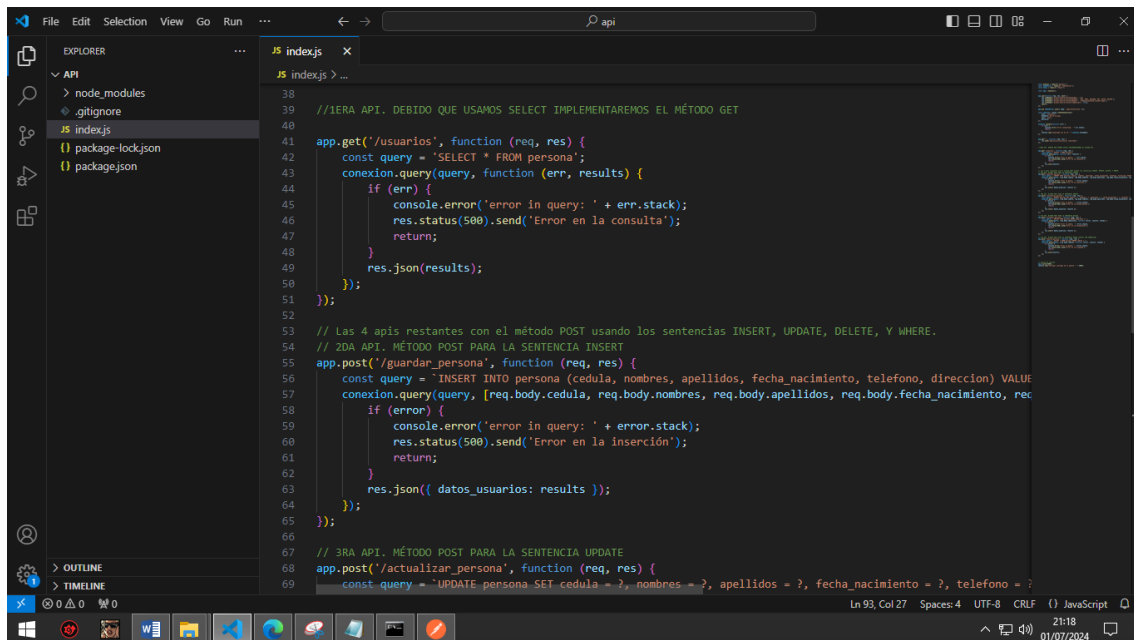
Posteriormente, hemos agregado algunos registros.



	idpersona	cedula	nombres	apellidos	fecha_nacimiento	telefono	direccion
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	1	1316151918	Hansel	Alcivar Fuentes	2014-07-01	0967053608	Portoviejo, Av manabí
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	2	1316151928	Fernando	Mendoza Álava	2014-08-02	0967053678	Portoviejo, Av manabí. Depto utm
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	3	1316151100	Anderson	Velásquez Zambrano	2014-10-02	0967054123	Portoviejo, Av Ejército
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	4	1316154159	Jostyn	Alcivar Montesdeoca	2014-02-02	0967841359	Chone pero no se donde
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	5	1314152698	Blood Street	Fuentes Destalles	0000-00-00	0967053608	Los Santos Internacional

3. Creamos las API'S con los métodos GET y POST.

Una breve ilustración de las 5 apis elaboradas

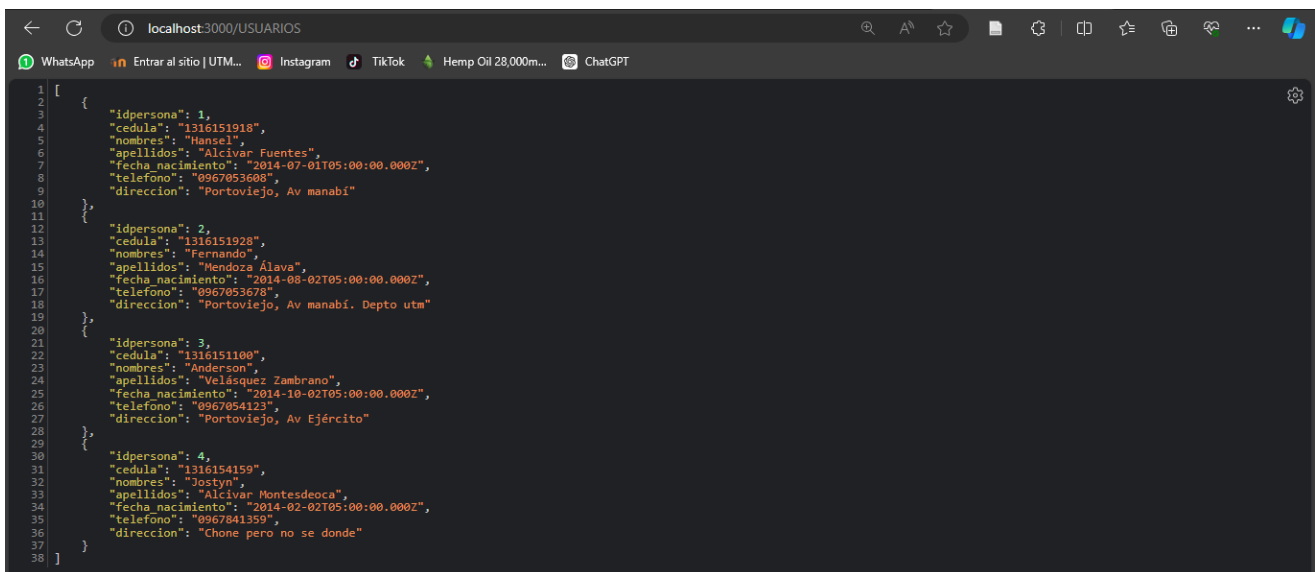


```
38
39 //1ERA API, DEBIDO QUE USAMOS SELECT IMPLEMENTAREMOS EL MÉTODO GET
40
41 app.get('/usuarios', function (req, res) {
42   const query = 'SELECT * FROM persona';
43   conexion.query(query, function (err, results) {
44     if (err) {
45       console.error('error in query: ' + err.stack);
46       res.status(500).send('Error en la consulta');
47       return;
48     }
49     res.json(results);
50   });
51 });
52
53 // Las 4 apis restantes con el método POST usando los sentencias INSERT, UPDATE, DELETE, Y WHERE.
54 // 2DA API. MÉTODO POST PARA LA SENTENCIA INSERT
55 app.post('/guardar_persona', function (req, res) {
56   const query = 'INSERT INTO persona (cedula, nombres, apellidos, fecha_nacimiento, telefono, direccion) VALUES (' + req.body.cedula + ', ' + req.body.nombres + ', ' + req.body.apellidos + ', ' + req.body.fecha_nacimiento + ', ' + req.body.telefono + ', ' + req.body.direccion + ')';
57   conexion.query(query, function (err, results) {
58     if (err) {
59       console.error('error in query: ' + err.stack);
60       res.status(500).send('Error en la inserción');
61       return;
62     }
63     res.json({ datos_usuarios: results });
64   });
65 });
66
67 // 3RA API. MÉTODO POST PARA LA SENTENCIA UPDATE
68 app.post('/actualizar_persona', function (req, res) {
69   const query = 'UPDATE persona SET cedula = ?, nombres = ?, apellidos = ?, fecha_nacimiento = ?, telefono = ?';
```

4. Validación de las apis

Usamos la herramienta “Postman” para testear las funciones y validar si los envíos de los datos son almacenados exitosamente en la base de datos.

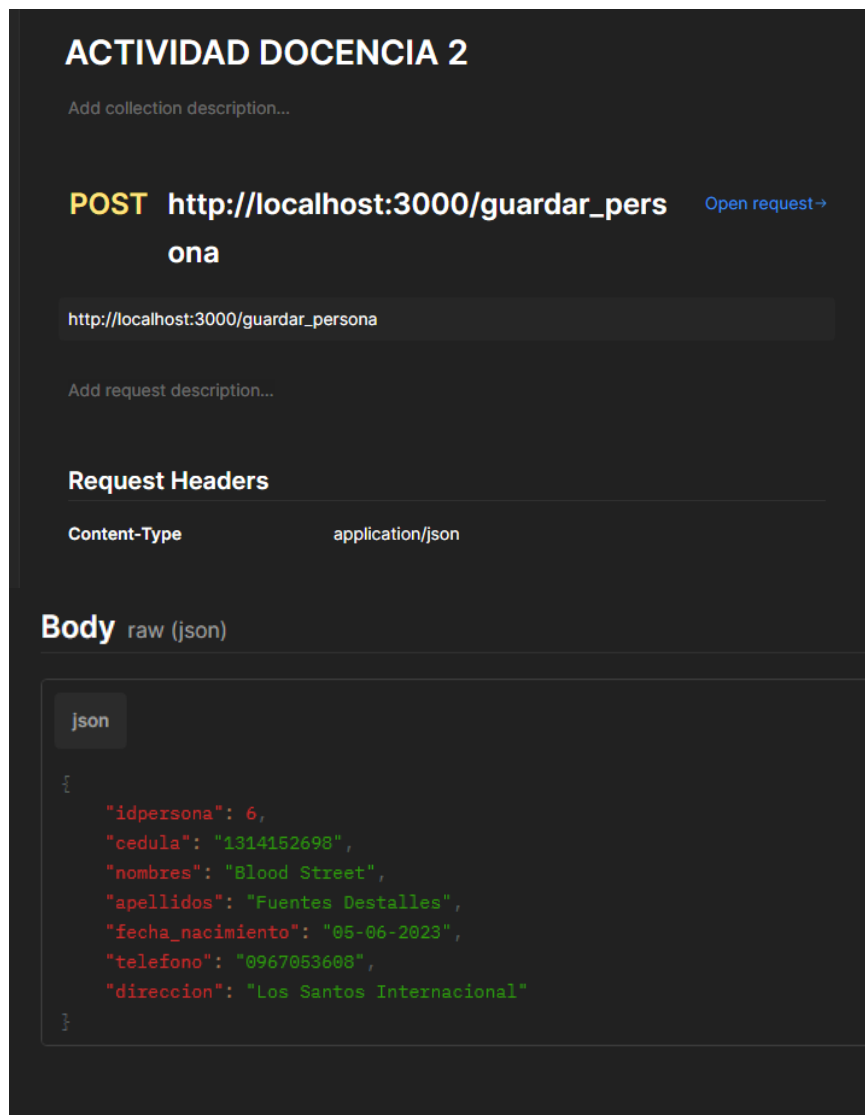
Método Get a través del navegador.



```
1 [
2   {
3     "idpersona": 1,
4     "cedula": "1316151918",
5     "nombres": "Hansel",
6     "apellidos": "Alcivar Fuentes",
7     "fecha_nacimiento": "2014-07-01T05:00:00.000Z",
8     "telefono": "0967053608",
9     "direccion": "Portoviejo, Av manabí"
10  },
11  {
12    "idpersona": 2,
13    "cedula": "1316151928",
14    "nombres": "Fernando",
15    "apellidos": "Mendoza Alava",
16    "fecha_nacimiento": "2014-08-02T05:00:00.000Z",
17    "telefono": "0967053678",
18    "direccion": "Portoviejo, Av manabí. Depto utm"
19  },
20  {
21    "idpersona": 3,
22    "cedula": "1316151100",
23    "nombres": "Anderson",
24    "apellidos": "Velásquez Zambrano",
25    "fecha_nacimiento": "2014-10-02T05:00:00.000Z",
26    "telefono": "0967054123",
27    "direccion": "Portoviejo, Av Ejército"
28  },
29  {
30    "idpersona": 4,
31    "cedula": "1316154159",
32    "nombres": "Jostyn",
33    "apellidos": "Alcivar Montesdeoca",
34    "fecha_nacimiento": "2014-02-02T05:00:00.000Z",
35    "telefono": "0967841359",
36    "direccion": "Chone pero no se donde"
37  }
38 ]
```

Método POST a través de la herramienta Postman:

- /guardar_persona: Este api usada la sentencia INSERT



- /actualizar persona: Usando la sentencia UPDATE

Actualizamos el primer registro.

POST

http://localhost:3000/actualizar_persona

[Open request→](#)

http://localhost:3000/actualizar_persona

Actualizamos el ["idpersona": 1] cambiándole los datos de todos los campos.

Request Headers

Content-Type

application/json










Body raw (json)

json

```
{
  "idpersona": 1,
  "cedula": "1314152698",
  "nombres": "Nombre",
  "apellidos": "Actualizado",
  "fecha_nacimiento": "05-06-2024",
  "telefono": "0985997468",
  "direccion": "Direccion actualizada"
}
```

Se actualizó correctamente en la base de datos.

Opciones extra

		idpersona	cedula	nombres	apellidos	fecha_nacimiento	telefono	direccion
<input type="checkbox"/>	 Editar  Copiar  Borrar	1	1314152698	Nombre	Actualizado	0000-00-00	0985997468	Direccion actualizada
<input type="checkbox"/>	 Editar  Copiar  Borrar	2	1316151928	Fernando	Mendoza Álava	2014-08-02	0967053678	Portoviejo, Av manabí. Depto utm
<input type="checkbox"/>	 Editar  Copiar  Borrar	3	1316151100	Anderson	Velásquez Zambrano	2014-10-02	0967054123	Portoviejo, Av Ejército

- /eliminar_persona: Usando la sentencia DELETE

Eliminamos el registro que anteriormente actualizamos.

POST

http://localhost:3000/eliminar_persona

[Open request→](#)

http://localhost:3000/eliminar_persona

Eliminamos el primer registro de la base de datos.

Request Headers

Content-Type

application/json

Body

raw (json)

json

```
{
  "idpersona": 1,
  "cedula": "1314152698",
  "nombres": "Nombre",
  "apellidos": "Actualizado",
  "fecha_nacimiento": "05-06-2024",
  "telefono": "0985997468",
  "direccion": "Direccion actualizada"
}
```

El primer registro [idpersona: 1] no se encuentra en la tabla persona, significa que se eliminó exitosamente de la base de datos.

Opciones extra

		▼	idpersona	cedula	nombres	apellidos	fecha_nacimiento	telefono	direccion	
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	2	1316151928	Fernando	Mendoza Álava	2014-08-02	0967053678	Portoviejo, Av manabí. Depto utm
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	3	1316151100	Anderson	Velásquez Zambrano	2014-10-02	0967054123	Portoviejo, Av Ejército
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	4	1316154159	Jostyn	Alcivar Montesdeoca	2014-02-02	0967841359	Chone pero no se donde
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	5	1314152698	Blood Street	Fuentes Destalles	0000-00-00	0967053608	Los Santos Internacional

- /buscar_persona: Usando la sentencia WHERE (un SELECT con la condición WHERE).

Hacemos la petición de búsqueda del segundo registro [idpersona: 2].

POST **http://localhost:3000/buscar_persona** [Open request →](#)

`http://localhost:3000/buscar_persona`

Add request description...

Request Headers

Content-Type	application/json
--------------	------------------

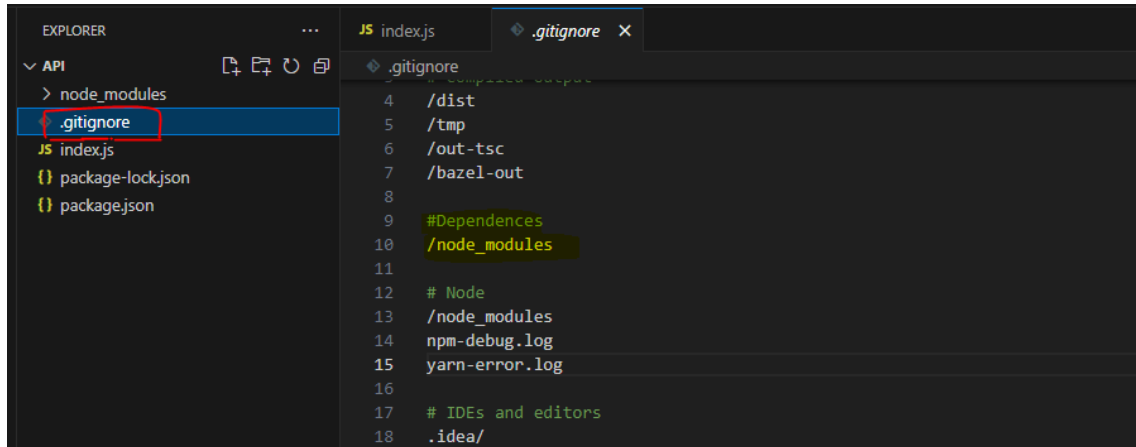
Body raw (json)

json

```
{
  "idpersona": 2,
  "cedula": "1316151928",
  "nombres": "Fernando",
  "apellidos": "Mendoza Álava",
  "fecha_nacimiento": "02-08-2014",
  "telefono": "0967053678",
  "direccion": "Portoviejo, Av manabí. Depto utm"
}
```

5. Archivos

Creamos el archivo “.gitignore” y dentro de este añadimos la dependencia “node_modules”.



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays the project structure with files like index.js, package-lock.json, and package.json. The .gitignore file is highlighted. The main editor area shows the content of .gitignore, which includes patterns for ignoring various files and directories.

```
1 # Compiled output
2
3
4 /dist
5 /tmp
6 /out-tsc
7 /bazel-out
8
9 #Dependencies
10 /node_modules
11
12 # Node
13 /node_modules
14 npm-debug.log
15 yarn-error.log
16
17 # IDEs and editors
18 .idea/
```

6. Git Hub

Enlace del proyecto en reposito de Git Hub: