Universidad Técnica de Manabí

Facultad Ciencias de la

InformaciónPortoviejo

Nombre:

José Samuel Itás

Moreira

Carrera:

Ingeniería de Software

Asignatura:

Desarrollo De

Aplicaciones

Web

Paralelo:

“A”

Actividad Docencia No. 2 - Api Rest

- Crear un Api Rest con Node JS.
- Conectar con la base de datos en postgres o mysql.
    - Importante: Nombre de la base de datos: db_curso_app
    - Esquema: esq_datos_personales
    - Tabla: persona
    - Columas:
        - idpersona serial4
        - cedula varchar(20)
        - nombres varchar(50)
        - apellidos varchar(50)
        - fecha_nacimiento date
        - telefono varchar(50)
        - direccion varchar(50)
- Crear 5 end points (insert, update, select, delete, where)

Creación de la base de datos:

Vamos a crear la base de datos con el nombre indicado en las instrucciones de la tarea:

```
CREATE DATABASE db_curso_app;
```

Con la base de datos ya creada vamos a crear la tabla persona:

```
USE db_curso_app;
CREATE TABLE persona (
    idpersona INT AUTO_INCREMENT PRIMARY KEY,
    cedula VARCHAR(20),
    nombres VARCHAR(50),
    apellidos VARCHAR(50),
    fecha_nacimiento DATE,
    telefono VARCHAR(50),
    direccion VARCHAR(50)
);
```

Creación del proyecto:

Primero vamos a crear una carpeta para el proyecto, para ello abrimos la terminal y ejectuamos el siguiente comando:

```
mkdir api-rest
```

Ingresamos a la carpeta creada con el comando:

```
cd api-rest
```

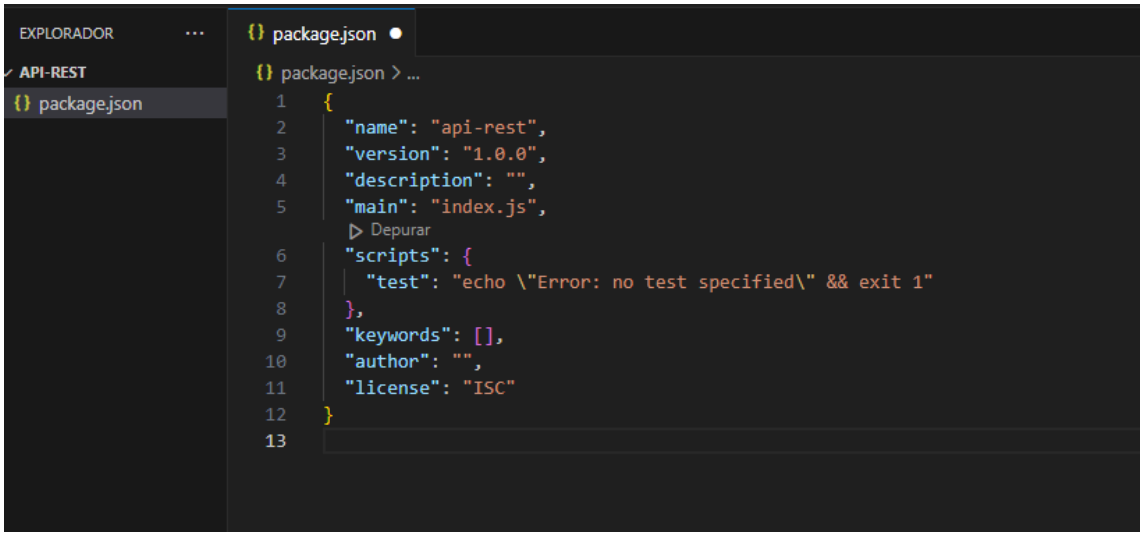
Una vez dentro de la carpeta creada procedemos a iniciar el proyecto:

```
npm init -y
```

```
{
  "name": "api-rest",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

Desarrollo de la api:

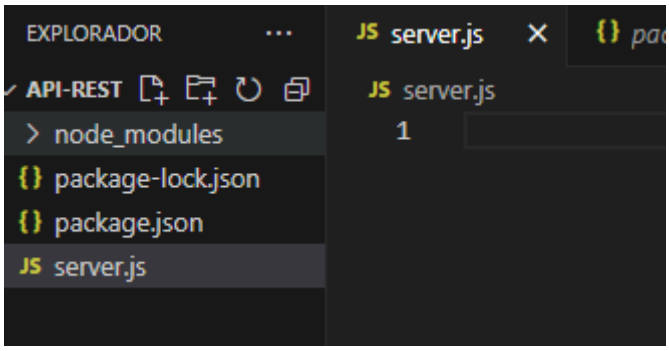
Ahora abrimos la carpeta donde esta el proyecto en Visual Studio Core:



Vamos a instalar los siguientes paquetes en nuestro proyecto:

```
>npm install express mysql
```

Creamos un nuevo archivo, server.js:



Configuramos el archivo:

```
const express = require('express');
const app = express();
const bodyParser = require('body-parser');
const mysql = require('mysql');

app.use(bodyParser.json({ type: 'application/json' }));

app.use(function (req, res, next) {
    res.setHeader('Access-Control-Allow-Origin', '*');
    res.setHeader('Access-Control-Allow-Methods', 'POST');
    res.setHeader('Access-Control-Allow-Headers', 'X-Requested-With,content-ty
    res.setHeader('Access-Control-Allow-Credentials', true);
    next();
});

const connection = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: '',
    database: 'db_curso_app'
});

connection.connect(function (err) {
    if (err) {
        console.error('error connecting: ' + err.stack);
        return;
    }
    console.log('connected as id ' + connection.threadId);
});
```

Ahora vamos a crear los endpoints:

Primero uno con insert que servirá para insertar una persona:

```
app.post("/insertar", function (req, res) {
    const { cedula, nombres, apellidos, fecha_nacimiento, telefono, direccion } = req.body;
    connection.query(
        'INSERT INTO persona (cedula, nombres, apellidos, fecha_nacimiento, telefono, direcci
        [cedula, nombres, apellidos, fecha_nacimiento, telefono, direccion],
        function (error, results) {
            if (error) throw error;
            res.json({ persona: results });
        }
    );
});
```

Ahora el select para seleccionar todas las personas:

```
app.post("/seleccionar", function (req, res) {
    connection.query('SELECT * FROM persona', function (error, results) {
        if (error) throw error;
        res.json({ personas: results });
    });
});
```

El WHERE para seleccionar una persona por id:

```
app.post("/seleccionarPorId", function (req, res) {
    const { idpersona } = req.body;
    connection.query('SELECT * FROM persona WHERE idpersona = ?', [idpersona], function (error, results) {
        if (error) throw error;
        if (results.length > 0) {
            res.json({ persona: results[0] });
        } else {
            res.status(404).json({ error: 'Persona no encontrada' });
        }
    });
```

Update para actualizar una persona:

```
app.post("/actualizar", function (req, res) {
    const { idpersona, cedula, nombres, apellidos, fecha_nacimiento, telefono, direccion } = req.body;
    connection.query(
        'UPDATE persona SET cedula = ?, nombres = ?, apellidos = ?, fecha_nacimiento = ?, telefono = ?, direccion = ? WHERE idpersona = ?'
        [cedula, nombres, apellidos, fecha_nacimiento, telefono, direccion, idpersona],
        function (error, results) {
            if (error) throw error;
            if (results.affectedRows > 0) {
                res.json({ persona: { idpersona, cedula, nombres, apellidos, fecha_nacimiento, telefono, direccion } });
            } else {
                res.status(404).json({ error: 'Persona no encontrada' });
            }
        }
    );
```

Delete para eliminar una persona:

```
app.post("/eliminar", function (req, res) {
    const { idpersona } = req.body;
    connection.query('DELETE FROM persona WHERE idpersona = ?', [idpersona], function (error, results) {
        if (error) throw error;
        if (results.affectedRows > 0) {
            res.json({ mensaje: 'Persona eliminada' });
        } else {
            res.status(404).json({ error: 'Persona no encontrada' });
        }
    });
});
```

Ejecución:

Ejecutamos el servidor en la consola:

```
api-rest> node server.js
```

```
Servidor iniciado en el puerto: 3001
connected as id 12
```

Link del proyecto:

https://github.com/Tuisnor/-ACTIVIDAD_AUTONOMA_2