

UNIVERSIDAD DON BOSCO



Docente:

José Darwin Bermúdez Portillo.

Asignatura:

Desarrollo de Aplic. Web con Soft. Interpret. en el Cliente.

Integrantes:

José Guillermo Argueta Méndez (AM231054)

Jorge Enriquez Panameño Pubill (PP232243)

Nancy Mariela Cabrera Rosales (CR232269)

Marcos Ceseus Medina Velásquez (MV232358)

Dania Gorety Tejada Campos (TC232020)

Grupo:

G03T

Fecha de entrega:

29 de octubre de 2023.

Creación de una API

Una API (Interfaz de Programación de Aplicaciones) es una interfaz que actúa como un puente de comunicación entre diferentes aplicaciones de software o servicios. Está diseñada para permitir que estas aplicaciones se comuniquen y compartan datos de manera estructurada y segura.

- Lo primero que vamos a hacer es configurar nuestro entorno, por lo tanto, es necesario tener node.js y npm instalados.
- Crearemos un directorio y ejecutaremos el comando "npm init" para generar un archivo package.json.
- Instalaremos express usando "npm install express --save" y después de completar la instalación, iniciaremos la aplicación con "npm start".
- Crearemos un archivo index.js y este será su contenido.
- Ejecutaremos "node index.js" y luego podremos comprobar escribiendo <http://localhost:3000> en el navegador para ver el mensaje "¡Hola mundo! La solicitud GET se ha ejecutado con éxito".
- Realizamos la petición de manera específica, ya que dependemos de la ruta principal, pues de ahí obtendremos nuestro recurso, con el uso de los distintos verbos, entre los cuales tenemos GET, POST, DELETE, PUT, dependiendo de lo que queramos realizar. Express js te acepta una función con dos parámetros, el primero es el request (petición) y el response (lo que manda de vuelta).

```

// importamos express
let express = require('express');

// Inicializamos las variables
let app = express();

let puerto = 3000;
// sera una llamada GET a la ruta principal "/"
app.get('/', (req, res, next) => {

    res.status(200).json({
        ok: true,
        message: 'Hola Mundo, GET ejecutado con éxito'
    });

});

// cual es el puerto que escuchara las peticiones http://localhost:3000
app.listen(puerto, () => {
    // `` con estas comillas podemos agregar variables usando ${}
    console.log(`Express listo y escuchando en el puerto: ${puerto}`);
});

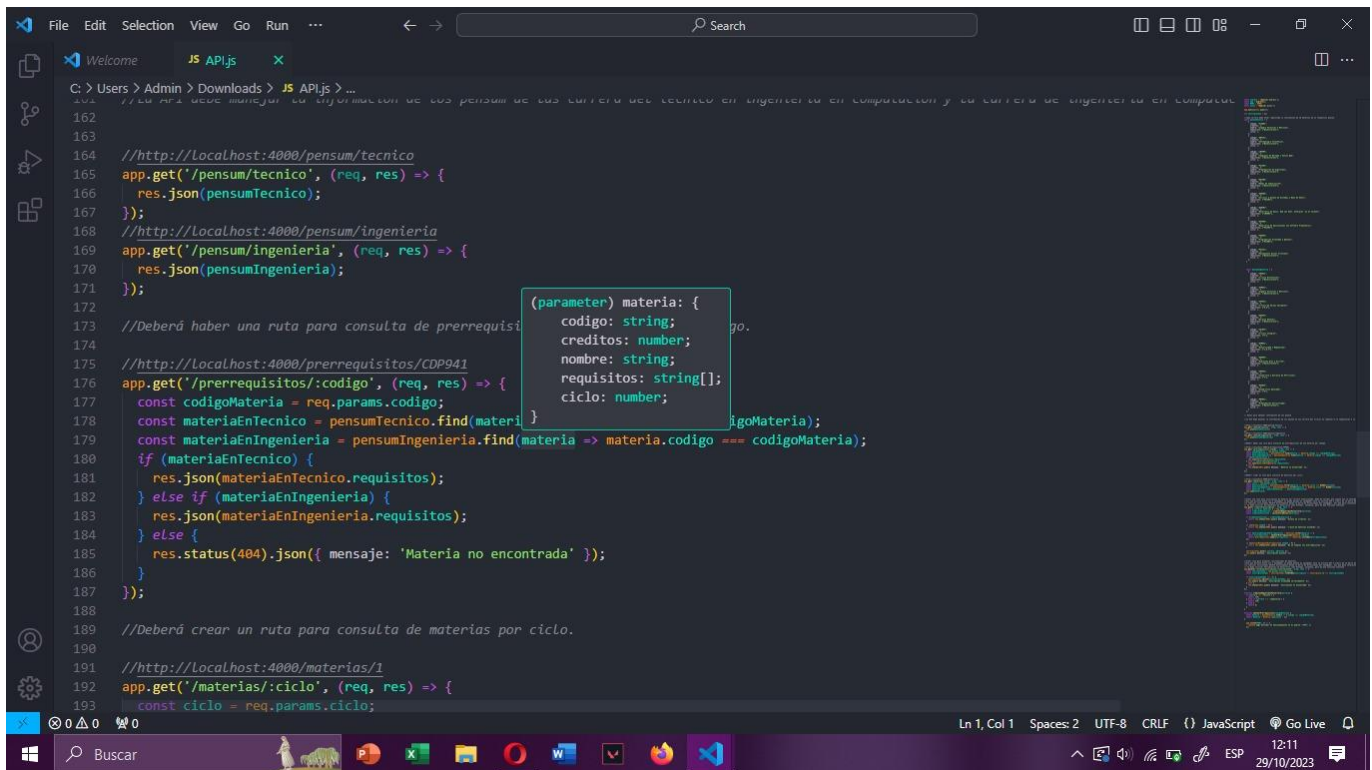
```

Ya con lo instalado se tiene que ir a la terminal del visual code y ejecutar los comando que se muestran en la imagen de referencia

```

1 const express = require('express');
2 const app = express();
3 const PORT = 4000;
4 const axios = require('axios');
5
6 app.use(express.json());
7
8 let inscripciones = [];
9
10 //cada carrera debe tener registrada la información de 10 materias de su respectivo pensum.
11 const pensumTecnico = [
12   {
13     codigo: "ALG501",
14     credits: 4,
15     nombre: "Álgebra Vectorial y Matrices",
16     requisitos: ["Bachillerato"],
17     ciclo: 1,
18   },
19   {
20     codigo: "ANF231",
21     credits: 3,
22     nombre: "Antropología Filosófica",
23     requisitos: ["Bachillerato"],
24     ciclo: 1,
25   },
26   {
27     codigo: "LME404",
28     credits: 4,
29     nombre: "Lenguajes de Marcado y Estilo Web",
30     requisitos: ["Bachillerato"],
31     ciclo: 1,
32   },
33 ];

```



```
162 //Deberá haber una ruta para consulta de materias por ciclo.
163
164 //http://localhost:4000/pensum/tecnico
165 app.get('/pensum/tecnico', (req, res) => {
166   res.json(pensumTecnico);
167 });
168 //http://localhost:4000/pensum/ingenieria
169 app.get('/pensum/ingenieria', (req, res) => {
170   res.json(pensumIngenieria);
171 });
172
173 //Deberá haber una ruta para consulta de prerequisitos de una materia por su código.
174
175 //http://localhost:4000/prerequisitos/codigo
176 app.get('/prerequisitos/:codigo', (req, res) => {
177   const codigoMateria = req.params.codigo;
178   const materiaEnTecnico = pensumTecnico.find(materia => materia.codigo === codigoMateria);
179   const materiaEnIngenieria = pensumIngenieria.find(materia => materia.codigo === codigoMateria);
180   if (materiaEnTecnico) {
181     res.json(materiaEnTecnico.requisitos);
182   } else if (materiaEnIngenieria) {
183     res.json(materiaEnIngenieria.requisitos);
184   } else {
185     res.status(404).json({ mensaje: 'Materia no encontrada' });
186   }
187 });
188
189 //Deberá crear una ruta para consulta de materias por ciclo.
190
191 //http://localhost:4000/materias/1
192 app.get('/materias/:ciclo', (req, res) => {
193   const ciclo = req.params.ciclo;
```

```
(parameter) materia: {
  codigo: string;
  creditos: number;
  nombre: string;
  requisitos: string[];
  ciclo: number;
```

¿Como funciona nuestro código?

Este código es un ejemplo de un servidor Node.js que utiliza el framework Express para crear una API web que gestiona información relacionada con los pensum de dos carreras: "Técnico en Ingeniería en Computación" y "Ingeniería en Computación" de una institución educativa (UDB).

¿Qué permite?

- Obtener información de materias en ambos programas.
- Consultar prerequisitos de una materia por su código.
- Buscar materias por ciclo.
- Inscribirse en materias, verificando requisitos y límites de créditos.
- Eliminar inscripciones previas.

Usamos un array llamado inscripciones que se utilizará para almacenar las inscripciones de materias por parte de los estudiantes.

Se definen dos arreglos de objetos pensumTecnico y pensumIngenieria, que representan el plan de estudios de las dos carreras. Cada objeto en estos arreglos contiene información sobre una materia, como su código, créditos, nombre, requisitos y ciclo al que pertenece.

Funciones auxiliares:

CreditosRequeridosPorCarrera(carrera): Esta función devuelve la cantidad de créditos requeridos para completar una carrera, ya sea "tecnico" o "ingenieria".

ObtenerPrerrequisitos(codigoMateria): Esta función devuelve los requisitos de una materia específica en el plan de estudios de "Técnico en Ingeniería en Computación".

Rutas que se usaron en API

/pensum/tecnico y /pensum/ingenieria: Estas rutas responden a las solicitudes GET y devuelven la información del plan de estudios para las dos carreras, respectivamente.

/prerrequisitos/:codigo: Esta ruta responde a las solicitudes GET con un parámetro codigo y devuelve los requisitos de una materia específica en ambas carreras. Se busca la materia en los arreglos pensumTecnico y pensumIngenieria y se devuelve su lista de requisitos.

/materias/:ciclo: Esta ruta responde a las solicitudes GET con un parámetro ciclo y devuelve las materias que pertenecen a un ciclo específico en ambas carreras. Se filtran las materias en función del ciclo y se devuelven como respuesta.

/realizarInscripcion: Esta ruta permite a un estudiante inscribir materias en su carrera. Se espera que el cliente envíe una solicitud POST con datos que incluyen la carrera y una lista de materias a inscribir. El servidor verifica si se cumplen los requisitos para la inscripción, como la cantidad de créditos y prerrequisitos, y luego almacena la inscripción en el arreglo inscripciones si es válida.

/eliminarInscripcion/:inscripcionId: Esta ruta permite eliminar una inscripción previamente realizada. El cliente debe enviar una solicitud DELETE con el ID de la inscripción que se desea eliminar.

Finalmente, el servidor se inicia en el puerto especificado (en este caso, el puerto 4000), y se muestra un mensaje en la consola indicando que el servidor está en funcionamiento. Además El código establece una API para administrar datos relacionados con cursos, prerrequisitos, inscripciones y cancelaciones de cursos en dos programas educativos. Los estudiantes pueden emplear esta API para inscribirse en cursos de acuerdo con los requisitos de sus programas académicos.