



Hola Bienvenido!

Registra tu asistencia

<https://forms.gle/6hfdT5U5v5fF34RQ8>



Node.js: llevando JavaScript al lado del servidor

Objetivo general

Comprender qué es Node.js, cómo funciona y por qué es una herramienta clave en el desarrollo web moderno, especialmente cuando se combina con frameworks de frontend como **Vue.js**.



Node.js es un **entorno de ejecución de JavaScript**
del lado del servidor





Node.js es un entorno de ejecución usado para ejecutar código JavaScript en los servidores. Permite ejecutar JavaScript sin necesidad de un navegador web. Node.js es compatible con sistemas operativos como Windows, macOS y Linux. Es una plataforma de código abierto, por lo que todos los usuarios pueden acceder al código fuente de forma gratuita.

Características clave de Node:



- Permite usar **JavaScript** fuera del navegador.
- Ideal para crear **aplicaciones web rápidas y escalables**.
- Usa un modelo **asíncrono y orientado a eventos**, lo que mejora el rendimiento

Ejemplos de uso



Servidores web

APIs REST

Aplicaciones en tiempo real (chats, juegos)

Arquitectura y funcionamiento



- Event Loop: sistema que gestiona múltiples tareas sin bloquear el hilo principal.
- I/O no bloqueante: Node puede atender varias peticiones al mismo tiempo sin esperar que una finalice.
- Módulos: piezas de código reutilizables (propios o instalados desde NPM).
- NPM (Node Package Manager): repositorio de millones de librerías para ampliar las capacidades de Node.



Práctica de node + Express para correr un servidor backend





Primeros pasos con Node

<https://nodejs.org/en/download>

- Instalación y verificación (`node -v`, `npm -v`).
- Crear un archivo `app.js` con un simple:
- `console.log('Hola desde Node.js!');`
- Ejecutar el archivo en consola con:
- `node app.js`

Inicializar proyecto con: `npm init -y`

Instalar paquetes con:

`npm install express`



Creación de un servidor básico

- Módulo `http` o uso de **Express.js** para simplificar:

```
import express from 'express';  
const app = express();
```

```
app.get('/', (req, res) =>  
  res.send('Hola desde mi servidor  
  Node.js'));  
app.listen(3000, () =>  
  console.log('Servidor en  
  http://localhost:3000'));
```

`node express.js`

Conceptos:

- `req` → petición del cliente
- `res` → respuesta del servidor
- `listen()` → abre un puerto para recibir solicitudes




Proyecto: “Mi primer servidor web con Node.js y Express”

Objetivo:

Aplicar los conceptos vistos para crear un servidor que atienda diferentes rutas.

```
mkdir mi-servidor //crear carpeta  
cd mi-servidor //entramos a la ruta  
npm init -y //inicializamos proyecto  
npm install express //instalamos express
```

Crear un archivo `server.js`



```
import express from 'express';
const app = express();
const PORT = 3000;

app.get('/', (req, res) => res.send('Bienvenido a mi servidor Node.js'));
app.get('/contacto', (req, res) => res.send('Página de contacto'));
app.get('/productos', (req, res) => res.json([{ nombre: 'Laptop', precio: 12000 }]]);

app.listen(PORT, () => console.log(`Servidor activo en http://localhost:${PORT}`));
```

Ejecutar:

```
node server.js
```

Abrir navegador en <http://localhost:3000>.



Resultado esperado

- Página principal muestra un mensaje de bienvenida.
- Rutas `/contacto` y `/productos` responden con texto y JSON.
- De esta manera se lograría comprender a estructura básica de un backend.



Node.js en el ecosistema moderno

- Node sirve como **base del desarrollo web actual**:
 - Ejecuta herramientas de compilación como **Vite**, **Webpack** o **Parcel**.
 - Permite crear **APIs REST** para comunicar frontend (Vue.js) y backend.
 - Es la plataforma detrás del **Full Stack JavaScript**.



Vite




Vuetify3



Vue.js

Relación directa entre Node y Vue

 Herramienta	Función	Soporte de Node
Vite	Servidor de desarrollo rápido	Corre sobre Node
NPM	Instalación de dependencias (Vue, Vite)	Administrado por Node
APIs backend	Fuente de datos (Express, MongoDB)	Desarrolladas en Node
Frontend Vue	Interfaz dinámica con fetch/axios	Se comunica con el backend Node



Hemos visto en otra perspectiva el backend con node js

Fullstack= Frontend + Backend.... vamos genial :)

Las demás clases complementaremos el frontend profundizando estructura en vue js , utilizando estilizado css tailwind hasta complementar con backend y uso de API's para lograr una app estética y funcional