

**Facultad:** Ciencias de la Ingeniería  
**Asignatura:** Desarrollo de Software II  
**Docente:** Ms. Carlos Salazar  
**Estudiante:** Jeison Medranda  
**Carrera:** Sistemas de la información 7mo A  
**Descripción:** Tarea Semana 1

### Laboratorio Semana 1: Introducción a la instalación y uso de herramientas para el desarrollo de software con Node.js

En este laboratorio teórico práctico, el estudiante logrará instalar y configurar las herramientas esenciales para el desarrollo de aplicaciones modernas con Node.js. En este sentido, se abordará el proceso de instalación de MongoDB, Node.js, dependencias básicas y postman. Para finalizar se realizará un pequeño ejercicio que permite explorar la creación de un proyecto Node.js y un servidor indagando la creación de rutas y endpoints probándolos con Postman.

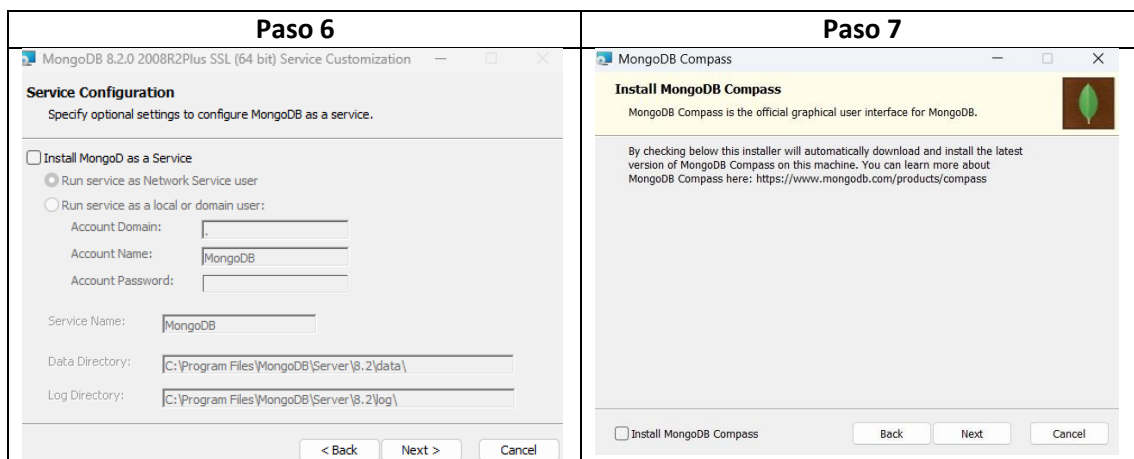
## Instalación de Mongo DB

### Edición comunitaria

La edición comunitaria de MongoDB se trata de la versión gratuita y de código abierto de este sistema de base de datos NoSQL. Se encuentra diseñado para que desarrolladores, estudiantes e investigadores puedan aprender, experimentar y construir aplicaciones sin necesidad de pagar licencias, en otras palabras, es la puerta de entrada gratuita al ecosistema MongoDB.

### Pasos a seguir para la instalación en plataforma Windows

1. Ingresar al sitio oficial de mongoDB
2. Ingresar a la sección Edición Comunitaria
3. Buscar y descargar MongoDB Community Edition
4. Descargar la versión 8.2.0, package msi
5. Instalar con la configuración Completa
6. Deshabilitar MongoD como servicio
7. Deshabilitar la instalación de MongoDB Compass



### Actividad 1:

Explica de manera técnica:

1. ¿Cuál sería la razón para deshabilitar mongod como servicio durante la instalación?

Se deshabilita el servicio mongod para evitar que MongoDB se ejecute automáticamente.

Control Manual y Específico.

2. ¿Qué es MongoDB Compass y por qué se suele deshabilitar su instalación automática?  
MongoDB Compass es una potente GUI para consultar, agregar y analizar sus datos de MongoDB en un entorno visual.

¿Por qué se suele deshabilitar su instalación?

Para mantener una instalación más limpia, ligera y controlada, especialmente en entornos de desarrollo o producción.

### Dbpath

En MongoDB, el parámetro `--dbpath` representa la ruta (carpeta) donde se almacenarán los datos de la base de datos. Cuando se arranca el servidor MongoDB necesita un directorio configurado para guardar:

- Archivos de las colecciones (documentos)
- Índices
- Información de configuración de la base de datos

La importancia de personalizar esta configuración radica en:

- Es posible tener diferentes dbpath para distintos proyectos o entornos (desarrollo, prueba, producción).
- Permite mover la base de datos a otra ubicación o disco con mayor espacio.
- Es útil cuando no se requiere utilizar la ruta por defecto cuando se trabaja en entornos compartidos.

Una vez se ha instalado Mongo, es importante configurar la ruta (carpeta) donde se almacenarán los datos de la base de datos

### Actividad 2:

Configura el parámetro `--dbpath` considerando:

1. En la raíz del disco C, crea un directorio denominado data8 que contenga otro directorio nombrado db.

2. Desde el símbolo del sistema, ingresa al bin de la instalación MongoDB, la ruta debe ser similar `C:\Program Files\MongoDB\Server\8.2\bin`

3. En esta ruta desde la terminal ejecuta:

```
C:\Program Files\MongoDB\Server\8.2\bin>mongod.exe --dbpath C:\data8\db
C:\Program Files\MongoDB\Server\8.2\bin>mongof.exe --dbpath C:\data8 \db
```

Para ejecutar el servidor en sistemas Windows es importante ejecutar mongod.exe ya que es el motor que pone en marcha MongoDB en el puerto por defecto 27017 y se encarga de manejar:

- Lectura y escritura de documentos.
- Conexión con clientes (como mongo, Compass, aplicaciones en Node.js, etc.)
- El almacenamiento de datos en el directorio definido por –dbpath

## Mongo Shell

Mongo Shell es la interfaz de línea de comandos (CLI) que permite a los desarrolladores interactuar directamente con el servidor de MongoDB.

- Funciona como un cliente que se conecta al servidor mongod
- Usa el mismo lenguaje de MongoDB: JavaScript
- Permite ejecutar comandos y consultas para crear bases de datos, colecciones, documentos y administrar la base de datos.
- Es la forma más directa y ligera de trabajar con MongoDB
- Ideal para aprender, probar consultas, administrar y depurar

Para su instalación, se debe seguir los siguientes pasos:

1. En el sitio oficial de Mongo en la sección comunitaria, se debe buscar en Tools – MongoDB Shell: versión 2.5.8 y package msi
2. Instalar en la ruta donde se instaló MongoDB en una nueva carpeta llamada mongosh  
`C:\Program Files\MongoDB\mongosh`
3. Desde la terminal ingresar al directorio anterior y ejecutar el comando mongosh.exe  
`C:\Program Files\MongoDB\mongosh>mongosh.exe`. Recuerde que debe tener ejecutando el servicio mongod.exe desde otra terminal.

## Primeros pasos en MongoDB

Una vez se encuentre ejecutando el servicio mongosh.exe, es posible crear bases de datos e interactuar con la información.

### Actividad 3:

En esta sección debe proporcionar una imagen evidenciando el cumplimiento de las siguientes actividades:

1. Crear una base de datos para el registro de vehículos, se debe contemplar: marca, modelo, año, chasis, placa, propietario
2. Ingresar dos vehículos del año 2023 y uno del año 2026
3. Consultar todos los vehículos.
4. Consultar los vehículos de marca KIA
5. Eliminar el vehículo del año 2026
6. Consultar todos los vehículos.

```
registro_vehiculos> show dbs
admin 40.00 KiB
config 88.00 KiB
local 40.00 KiB
registro_vehiculos> use registro_vehiculos
switched to db registro_vehiculos
registro_vehiculos> db.createCollection('vehiculos')
{ ok: 1 }
registro_vehiculos> db.vehiculos.insertMany([ {marca: "TOYOTA", modelo: "YARIS", anio: 2023, chasis: "ABC123", placa: "PBJ7859", propietario: "Pedro Perez"}, {marca: "KIA", modelo: "SOLUTO", anio: 2026, chasis: "XYZ789", placa: "STU478", propietario: "Hector Lopez"}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('68eb2610e27c80f9dccebea8'),
    '1': ObjectId('68eb2610e27c80f9dccebea9')
  }
}
```

```
registro_vehiculos> db.vehiculos.find().pretty()
[
  {
    _id: ObjectId('68eb2610e27c80f9dccebea8'),
    marca: 'TOYOTA',
    modelo: 'YARIS',
    anio: 2023,
    chasis: 'ABC123',
    placa: 'PBJ7859',
    propietario: 'Pedro Perez'
  },
  {
    _id: ObjectId('68eb2610e27c80f9dccebea9'),
    marca: 'KIA',
    modelo: 'SOLUTO',
    anio: 2026,
    chasis: 'XYZ789',
    placa: 'STU478',
    propietario: 'Hector Lopez'
  }
]
```

```
registro_vehiculos> db.vehiculos.insertMany([ { marca: "TOYOTA", modelo: "Yaris", anio: 2023, chasis: "ABC123", placa: "PBJ", propietario: "Pedro Lopez" }, { marca: "KIA", modelo: "Soluto", anio: 2026, chasis: "XYZ123", placa: "STU478", propietario: "Hector Lopez" }])
registro_vehiculos> db.vehiculos.find({ marca: "KIA" }).pretty()
[
  {
    _id: ObjectId('68eb2610e27c80f9dccebea9'),
    marca: 'KIA',
    modelo: 'SOLUTO',
    anio: 2026,
    chasis: 'XYZ789',
    placa: 'STU478',
    propietario: 'Hector Lopez'
  }
]
registro_vehiculos> db.vehiculos.deleteOne({ anio: 2026 })
{ acknowledged: true, deletedCount: 1 }
registro_vehiculos> db.vehiculos.find().pretty()
[
  {
    _id: ObjectId('68eb2610e27c80f9dccebea8'),
    marca: 'TOYOTA',
    modelo: 'YARIS',
    anio: 2023,
    chasis: 'ABC123',
    placa: 'PBJ7859',
    propietario: 'Pedro Perez'
  }
]
```

## MongoDB Compass

MongoDB Compass es la interfaz gráfica oficial de MongoDB que permite administrar y visualizar datos de manera intuitiva, sin necesidad de escribir todos los comandos en la consola (mongosh). Es importante en el desarrollo de software porque hace más fácil explorar, cons

y administrar datos, especialmente en entornos de aprendizaje, permitiendo comprender mejor cómo funciona una base de datos NoSQL.

#### Actividad 4:

- Elaborar un pequeño video de máximo un minuto explicando como ingresar, editar y eliminar un vehículo. Se debe proporcionar un enlace en Google Drive para visualizar el recurso.

[https://drive.google.com/file/d/1gGcnmRHCCd10-6tY7O79QOpsR44QbJ6-/view?usp=drive\\_link](https://drive.google.com/file/d/1gGcnmRHCCd10-6tY7O79QOpsR44QbJ6-/view?usp=drive_link)

## NodeJS

Es un entorno de ejecución de JavaScript que permite ejecutar este lenguaje del lado del servidor (no solo en el navegador).

- Está construido sobre el motor V8 de Google Chrome, lo que lo hace muy rápido al interpretar el código.
- Permite usar JavaScript en backend, de manera que un mismo lenguaje puede utilizarse tanto en frontend como en backend.
- Se basa en un modelo asíncrono y orientado a eventos, lo que significa que puede manejar muchas operaciones simultáneas sin bloquearse.
- Es una tecnología central en el desarrollo de aplicaciones modernas, microservicios y sistemas en tiempo real.

### Instalación

Para la instalación de Node.js considere las siguientes versiones:

- Node 22.20.0
- npm 10.9.3

### Postman

Es una herramienta de colaboración y pruebas de APIs (Application Programming Interfaces).

- Permite enviar solicitudes HTTP (GET, POST, PUT, DELETE, etc.) a un servidor y analizar las respuestas.
- Ofrece una interfaz gráfica amigable para trabajar con APIs sin necesidad de programar desde cero un cliente.
- Es ampliamente usado por desarrolladores, testers y equipos de QA.
- Es importante porque simplifica el desarrollo, mejora la colaboración en equipo y asegura que las APIs funcionen correctamente antes de ser usadas en producción.

#### Actividad 5:

- Elaborar un pequeño video de máximo un minuto visualizando las versiones instaladas de node y npm desde la terminal. Además, es necesario visualizar Postman funcionando en su computador. Se debe proporcionar un enlace en Google Drive para visualizar el

recurso.

[https://drive.google.com/file/d/1yObYn4LnHb1YsqXFQ\\_arbcnKp-EUgdqp/view?usp=drive\\_link](https://drive.google.com/file/d/1yObYn4LnHb1YsqXFQ_arbcnKp-EUgdqp/view?usp=drive_link)

## Configurar un nuevo proyecto Node.js

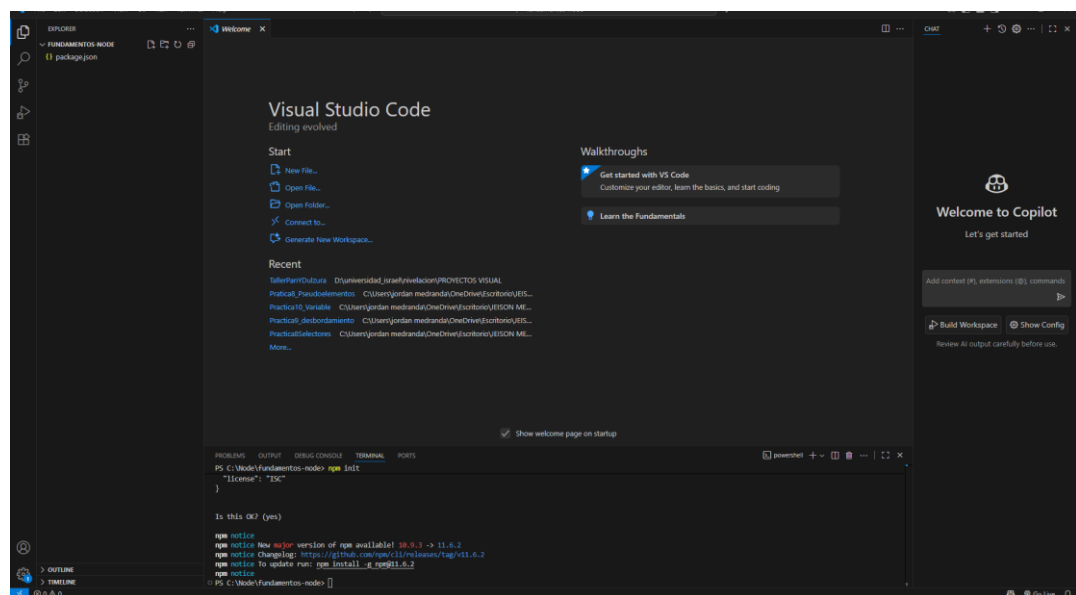
Para crear un nuevo proyecto Node.js, se utiliza el comando `npm init` desde la terminal para crear el archivo `package.json` en el proyecto.

El archivo `package.json` es el núcleo del proyecto, porque guarda toda la información necesaria sobre:

- El nombre y la versión del proyecto
- Una breve descripción.
- El punto de entrada (archivo principal, como `index.js` o `app.js`)
- Dependencias (librerías que necesita el proyecto)
- Scripts personalizados (por ejemplo, `npm start` para arrancar el servidor)
- Autor y licencia

Para organizar el directorio donde se almacenarán los proyectos se requiere:

1. Crear un nuevo directorio en la raíz C con el nombre Node
2. Crear un directorio con el nombre fundamentos-node
3. Cargar la carpeta anterior en Visual Studio Code
4. Desde la terminal ubicarse en el directorio fundamentos-node
5. Ejecutar el comando `npm init`
6. Verificar que se ha generado el archivo `package.json` en el proyecto desde Visual Studio Code.



## Express

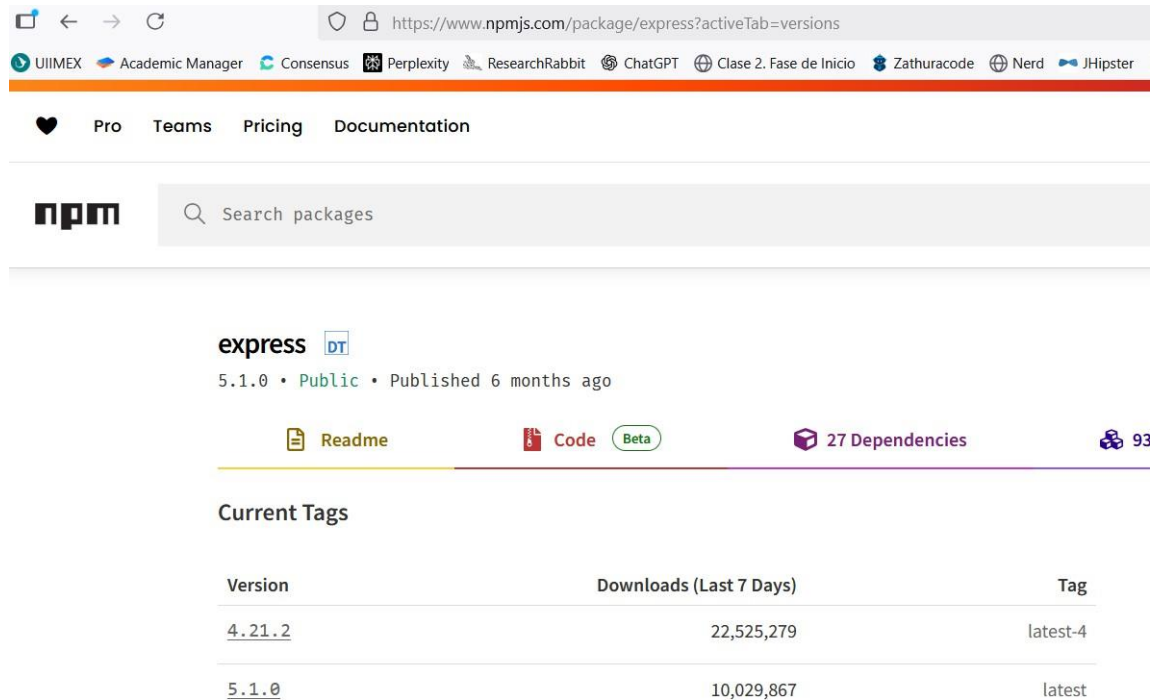
Express.js es un framework minimalista para Node.js que facilita la creación de aplicaciones web y APIs.

- Está construido sobre Node.js y simplifica mucho el manejo de rutas, peticiones HTTP,

respuestas y middlewares (funciones que se ejecutan en el ciclo de una petición como: validación, seguridad, autenticación o manejo de errores).

- Es rápido, flexible y ligero, lo que lo ha convertido en el framework más popular del ecosistema de Node.
- Es importante porque reduce la complejidad de trabajar con Node.js puro, fomenta la modularidad y es la base del desarrollo de backends modernos en JavaScript.

Para su instalación, observe las siguientes ilustraciones:



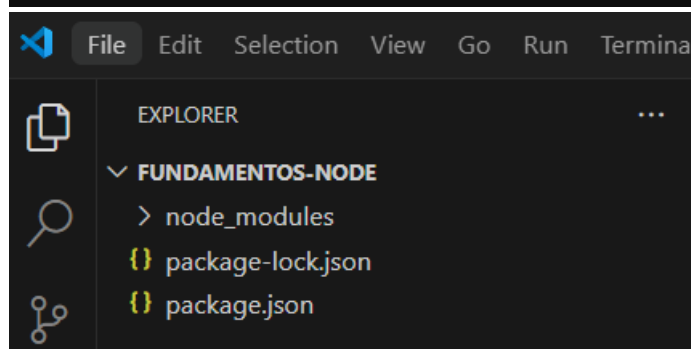
Version	Downloads (Last 7 Days)	Tag
<a href="#">4.21.2</a>	22,525,279	latest-4
<a href="#">5.1.0</a>	10,029,867	latest

```
C:\Node\fundamentos-node>npm install express@4.21.2 --save
added 69 packages, and audited 70 packages in 3s

14 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

C:\Node\fundamentos-node>
```

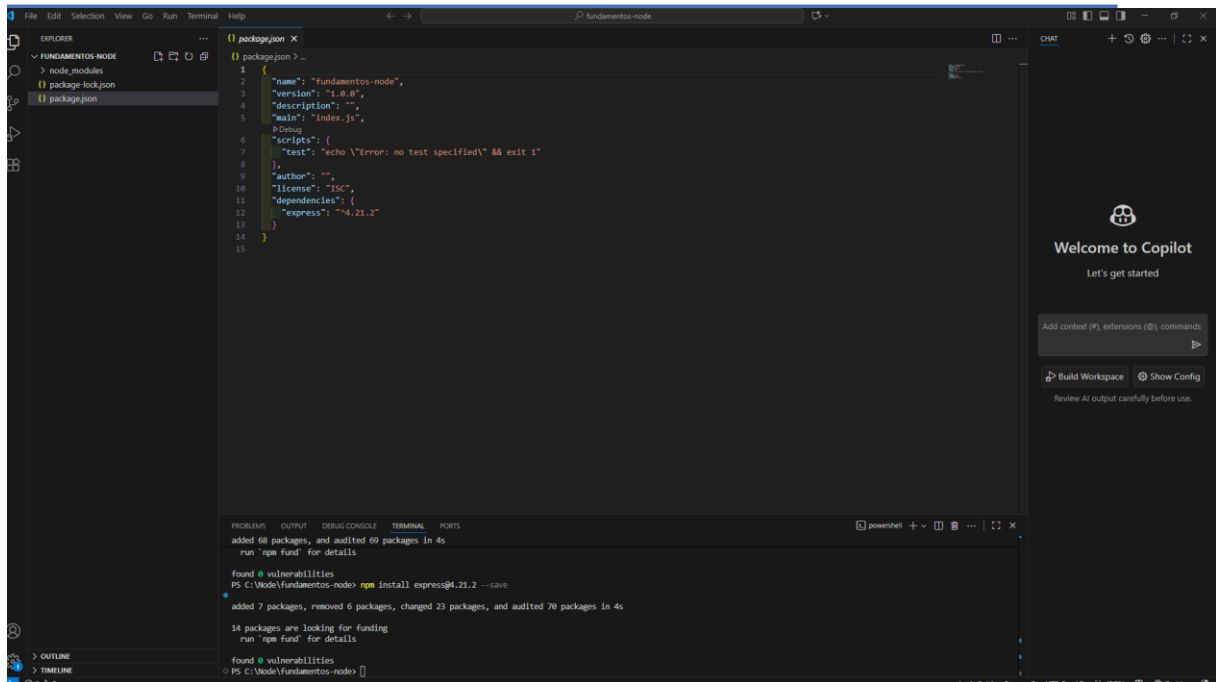


File Edit Selection View Go Run Termina

EXPLORER

✓ FUNDAMENTOS-NODE

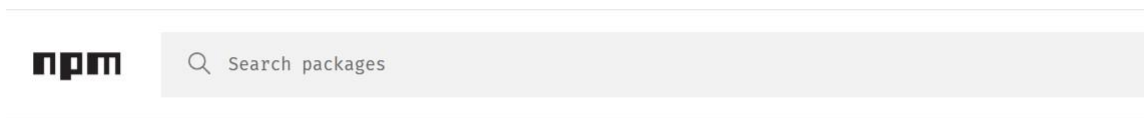
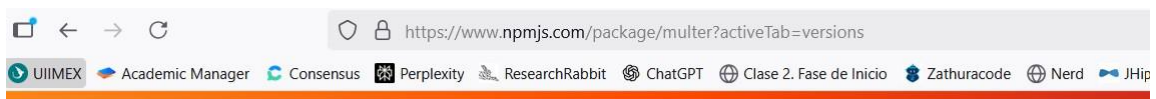
- > node\_modules
- { package-lock.json
- { package.json



## Multer

Es un middleware de Node.js para Express que permite manejar la carga de archivos (file uploads) desde el cliente al servidor.

- Se utiliza especialmente para manejar solicitudes HTTP tipo multipart/form-data, que es el formato en el que normalmente viajan los archivos en formularios web.
- Permite subir imágenes, documentos, audios, videos, etc., y guardarlos en una carpeta del servidor o procesarlos en memoria.
- Facilita a Express manejar formularios con archivos adjuntos.



multer DT

2.0.2 • Public • Published 3 months ago

[Readme](#)

[Code](#) Beta

7 Dependencies

```
C:\Node\fundamentos-node>npm install multer@1.4.5-lts.1 --save
npm warn deprecated multer@1.4.5-lts.1: Multer 1.x is impacted by a number of vulnerabilities,
n 2.x. You should upgrade to the latest 2.x version.

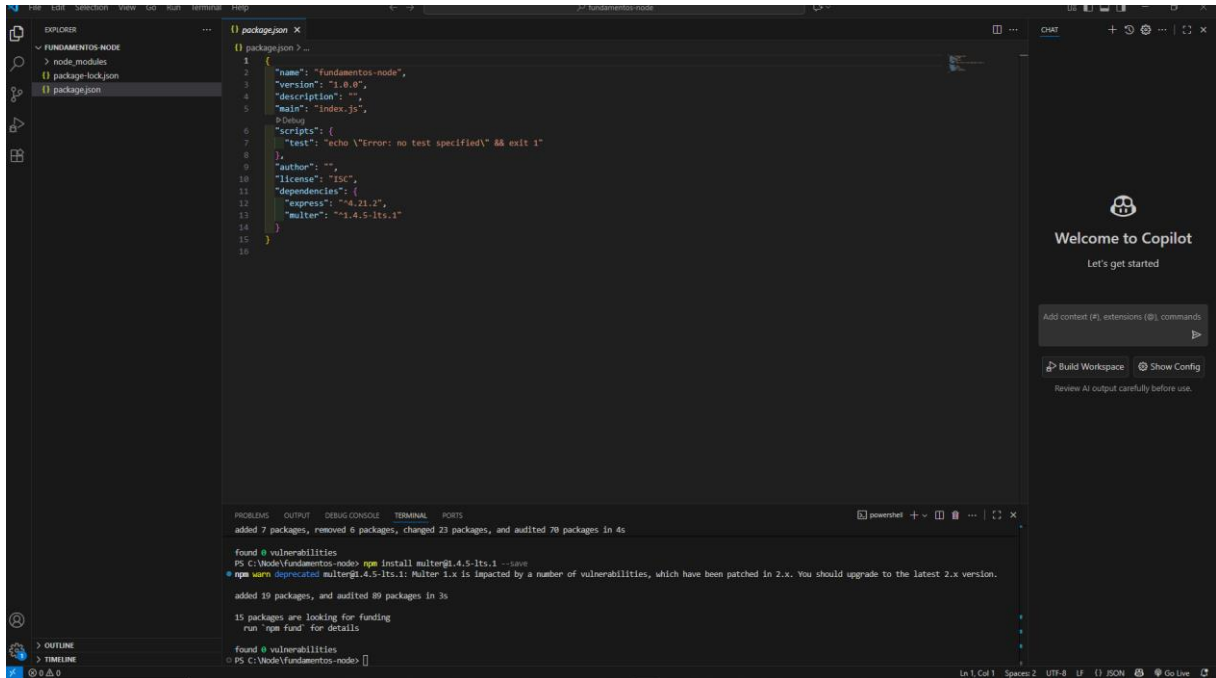
added 19 packages, and audited 89 packages in 2s

15 packages are looking for funding
  run 'npm fund' for details

found 0 vulnerabilities
```



Para su instalación, observe las siguientes ilustraciones:



## Mongoose

Mongoose sirve como puente entre el código de Node.js y la base de datos MongoDB, permitiendo manejar documentos como si fueran objetos de JavaScript y aplicar reglas de validación y estructuras de datos más organizadas. Es decir, es una librería ODM para Node.js que facilita el trabajo con MongoDB.

Es importante porque agrega estructuras, validaciones y modelos, lo que ayuda a mantener ordenados y consistentes los datos en aplicaciones modernas.

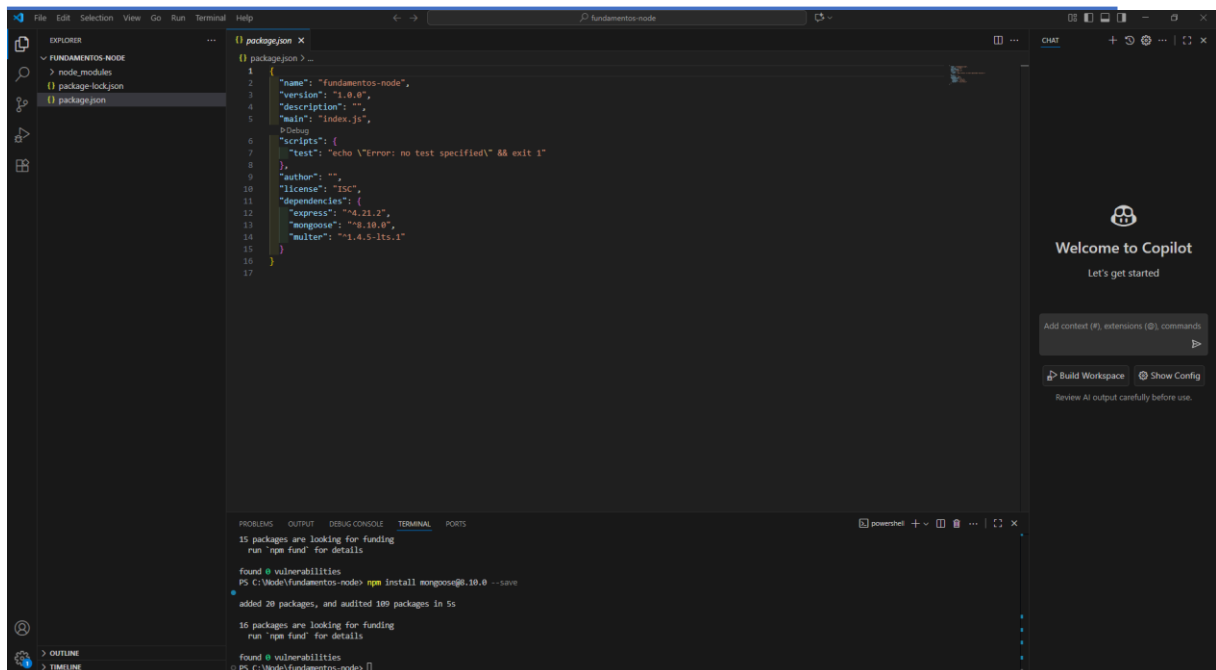
Para su instalación, observe la siguiente ilustración:

```
C:\Node\fundamentos-node>npm install mongoose@8.10.0 --save

added 20 packages, and audited 109 packages in 4s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```



The screenshot shows a VS Code editor with a project named 'fundamentos-node'. The Explorer sidebar on the left shows the file structure with 'package.json' selected. The main editor displays the content of 'package.json':

```
1 {
2   "name": "fundamentos-node",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "",
10  "license": "ISC",
11  "dependencies": {
12    "express": "^4.21.2",
13    "mongoose": "^8.10.0",
14    "multer": "^1.4.5-lts.1"
15  }
16 }
17
```

The terminal at the bottom shows the output of running 'npm install':

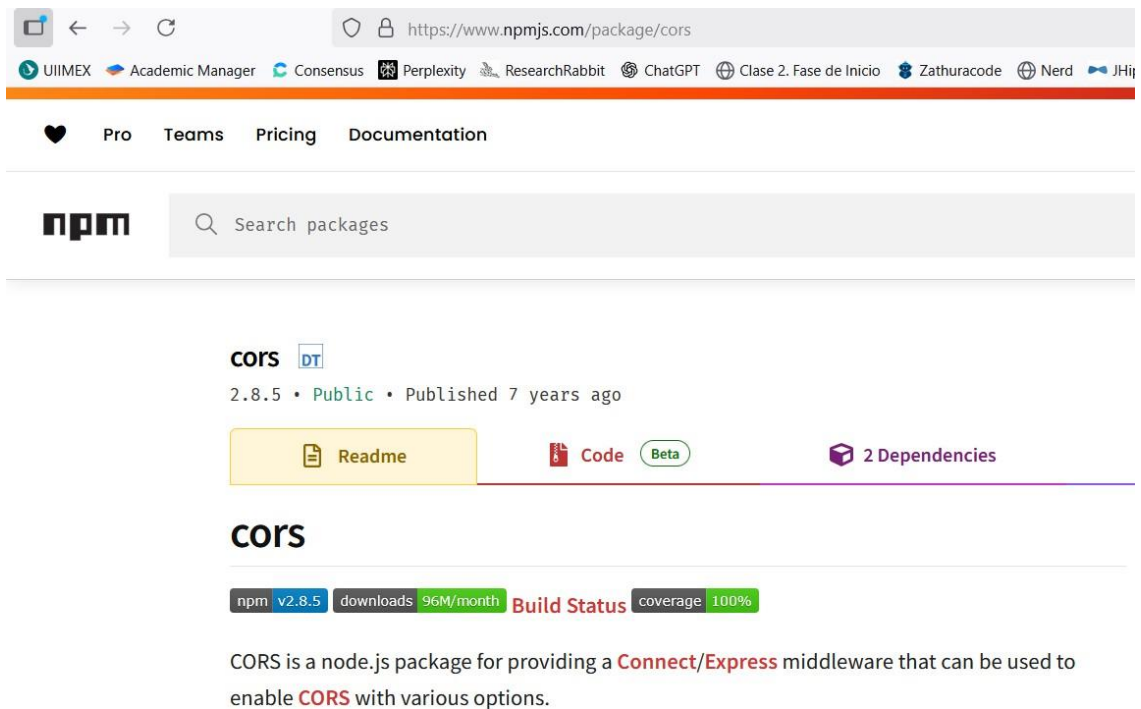
```
15 packages are looking for funding
run 'npm fund' for details

found 0 vulnerabilities

PS C:\code\fundamentos-node> npm install mongoose@8.10.0 --save
added 20 packages, and audited 100 packages in 5s
16 packages are looking for funding
run 'npm fund' for details
found 0 vulnerabilities
PS C:\code\fundamentos-node>
```

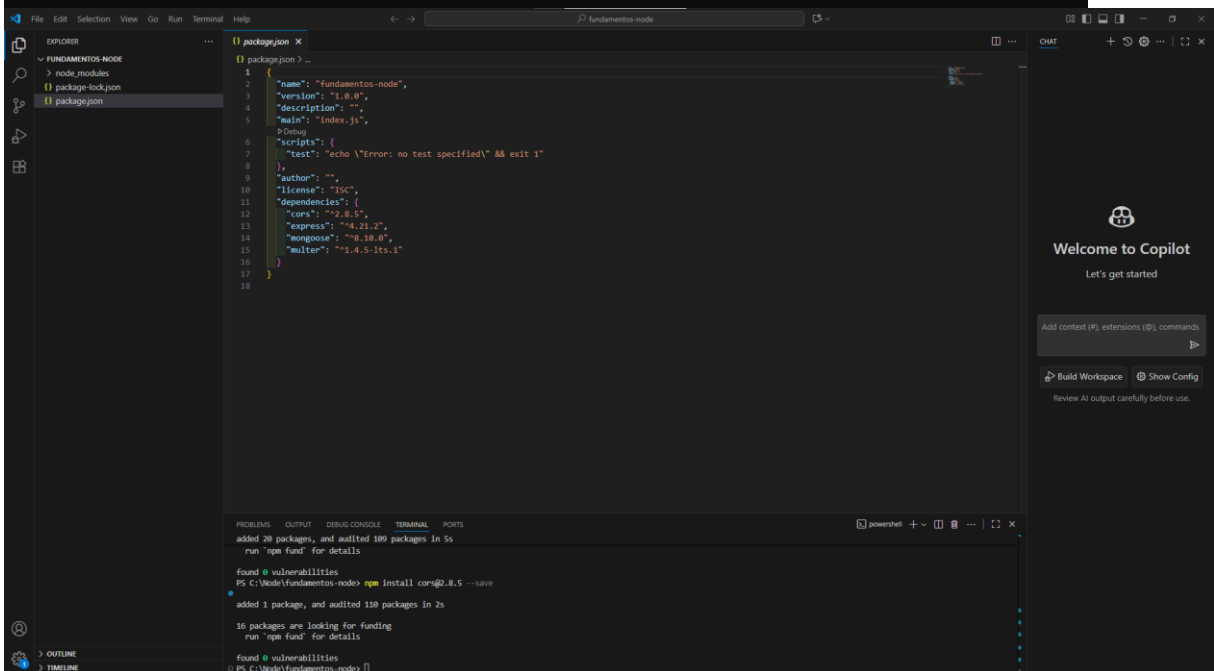
## Cors

Es un mecanismo de seguridad que controla qué orígenes externos pueden acceder a los recursos de un servidor. Su importancia radica en permitir la comunicación segura entre frontend y backend en aplicaciones modernas, evitando accesos no autorizados y habilitando solo los que el desarrollador defina.



The screenshot shows the npm website for the 'cors' package. The browser address bar displays 'https://www.npmjs.com/package/cors'. The page header includes navigation links: 'Pro', 'Teams', 'Pricing', and 'Documentation'. Below the header is a search bar with the text 'Search packages'. The main content area displays the package details for 'cors' (version 2.8.5, published 7 years ago). It includes buttons for 'Readme', 'Code', and 'Beta', along with a '2 Dependencies' badge. The package statistics show 'npm v2.8.5', 'downloads 96M/month', 'Build Status', and 'coverage 100%'. The description states: 'CORS is a node.js package for providing a **Connect/Express** middleware that can be used to enable **CORS** with various options.'

```
C:\Node\fundamentos-node>npm install cors@2.8.5 --save  
added 1 package, and audited 110 packages in 1s  
  
16 packages are looking for funding  
  run 'npm fund' for details  
  
found 0 vulnerabilities
```



## Validator

Validator.js es una librería de Node.js para validar y sanitizar datos de entrada, su importancia radica en la mejora de la seguridad, evita errores y garantiza la calidad de los datos en aplicaciones modernas, especialmente en APIs y formularios web.

Para su instalación observe las siguientes ilustraciones:

UIIMEX Academic Manager Consensus Perplexity ResearchRabbit ChatGPT Clase 2. Fase de Inicio Zathuracode Nerd JHip

Pro Teams Pricing Documentation

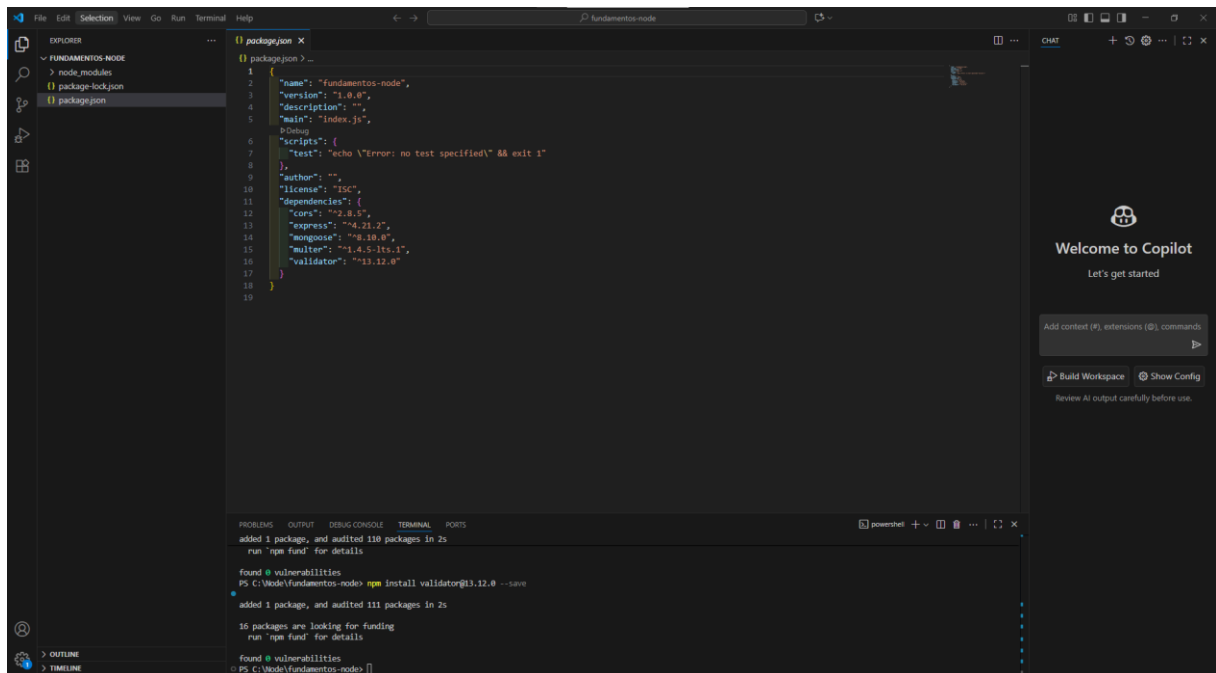
npm

Search packages

validator [DT](#)

13.15.15 • Public • Published 4 months ago

```
C:\Node\fundamentos-node>npm install validator@13.12.0 --save  
added 1 package, and audited 111 packages in 1s  
  
16 packages are looking for funding  
  run 'npm fund' for details  
  
found 0 vulnerabilities
```



## Nodemon

Nodemon es una herramienta de desarrollo para aplicaciones Node.js que reinicia automáticamente el servidor cada vez que detecta cambios en los archivos del proyecto.

- Se instala como una dependencia en el proyecto
- En lugar de ejecutar `node app.js`, se ejecuta `nodemon app.js`
- Así cada vez que se guarda cambios en el código, Nodemon reinicia el servidor sin necesidad de hacerlo manualmente.
- Nodemon debe ser visto como un asistente que observa el proyecto refresca el servidor en automático mientras se desarrolla y se realiza cambios.

Para su instalación, observe las siguientes ilustraciones:

## nodemon TS

3.1.10 • Public • Published 5 months ago

Readme

Code Beta

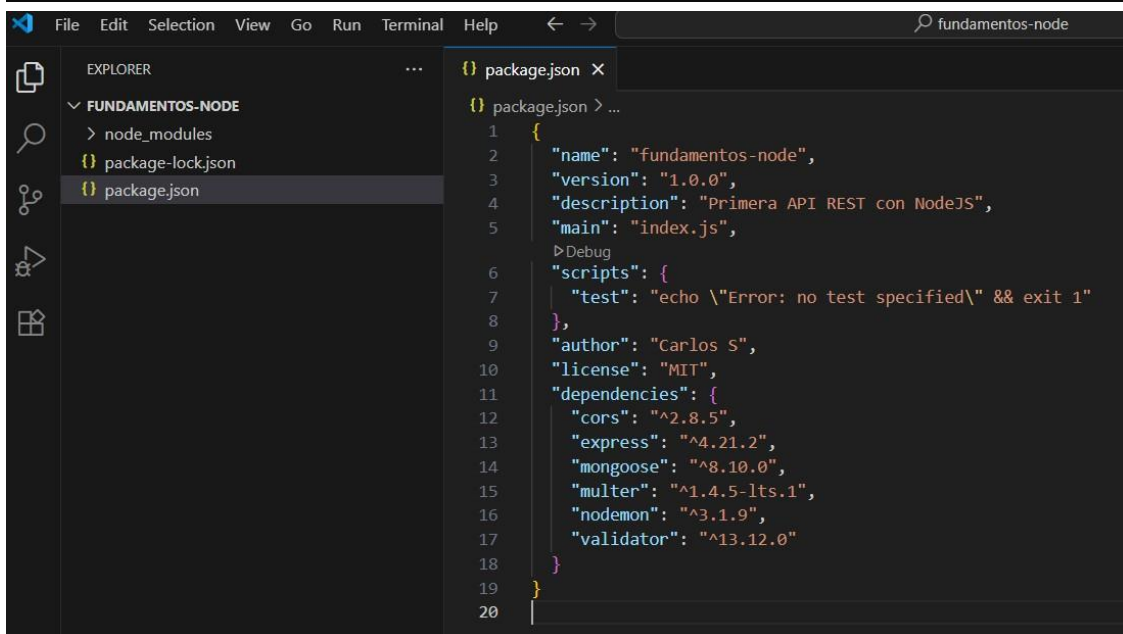
10 Dependencies



```
C:\Node\fundamentos-node>npm install nodemon@3.1.9 --save
added 29 packages, and audited 140 packages in 2s

20 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

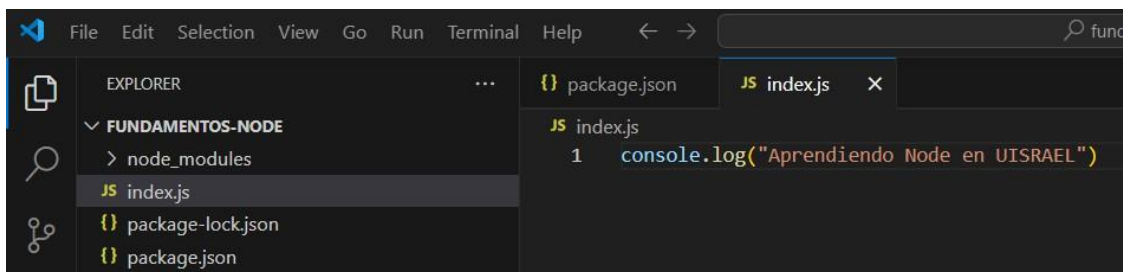


The screenshot shows the Visual Studio Code interface with the 'package.json' file open. The Explorer sidebar on the left shows the project structure: 'FUNDAMENTOS-NODE' containing 'node\_modules', 'package-lock.json', and 'package.json'. The main editor displays the content of 'package.json' with the following JSON structure:

```
{
  "name": "fundamentos-node",
  "version": "1.0.0",
  "description": "Primera API REST con NodeJS",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "author": "Carlos S",
  "license": "MIT",
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^4.21.2",
    "mongoose": "^8.10.0",
    "multer": "^1.4.5-lts.1",
    "nodemon": "^3.1.9",
    "validator": "^13.12.0"
  }
}
```

Al finalizar, el fichero package.json debe ser similar a la siguiente imagen:

Para verificar la correcta instalación, se creará el fichero index.js con un mensaje por consola para visualizar su impresión desde la terminal y cuando el mensaje sea modificado, automáticamente se debe actualizar en la consola. Observe las ilustraciones:



The screenshot shows the Visual Studio Code interface with the 'index.js' file open. The Explorer sidebar on the left shows the project structure: 'FUNDAMENTOS-NODE' containing 'node\_modules', 'package-lock.json', 'package.json', and 'index.js'. The main editor displays the content of 'index.js' with the following JavaScript code:

```
1 console.log("Aprendiendo Node en UISRAEL")
```

```
C:\Node\fundamentos-node>npm start

> fundamentos-node@1.0.0 start
> nodemon index.js

[nodemon] 3.1.9
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
Aprendiendo Node en UISRAEL
[nodemon] clean exit - waiting for changes before restart
```

```
{ } package.json  JS index.js  X
JS index.js
1 console.log("Aprendiendo Node en UISRAEL - ECUADOR")
```

```
C:\Node\fundamentos-node>npm start

> fundamentos-node@1.0.0 start
> nodemon index.js

[nodemon] 3.1.9
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
Aprendiendo Node en UISRAEL
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
Aprendiendo Node en UISRAEL - ECUADOR
[nodemon] clean exit - waiting for changes before restart
```

#### Actividad 6:

- Preparar el ejercicio anterior para la revisión en vivo en el salón de clase (Presencial o Virtual)

The image shows a Visual Studio Code interface with a Node.js project named 'FUNDAMENTOS-NODE'. The file explorer on the left shows the project structure, including 'index.js', 'package-lock.json', and 'package.json'. The main editor window displays the content of 'index.js', which contains a single line of JavaScript code: `console.log('Aprendiendo Node en UISRAEL - ECUADOR');`. The bottom panel features a terminal window where the command `npm start` has been executed. The terminal output shows the following sequence of events: `[nodemon] 3.1.9`, `[nodemon] to restart at any time, enter 'rs'`, `[nodemon] watching path(s): *.*`, `[nodemon] watching extensions: *.js,*.json`, `[nodemon] starting node index.js`, `Aprendiendo Node en UISRAEL`, `[nodemon] clean exit - waiting for changes before restart`, `[nodemon] restarting due to changes...`, `[nodemon] starting node index.js`, `Aprendiendo Node en UISRAEL - ECUADOR`, and `[nodemon] clean exit - waiting for changes before restart`. On the right side of the editor, a Copilot chat window is open, displaying the 'Welcome to Copilot' message and options to 'Build Workspace' or 'Show Context'.