

INTRODUCCIÓN AL DESARROLLO BACKEND CON NODEJS 2023



SECRETARÍA DE
EXTENSIÓN
UNIVERSITARIA
UTN - FRC



*UTN
Facultad Regional Córdoba

Agencia
CÓRDOBA
JOVEN



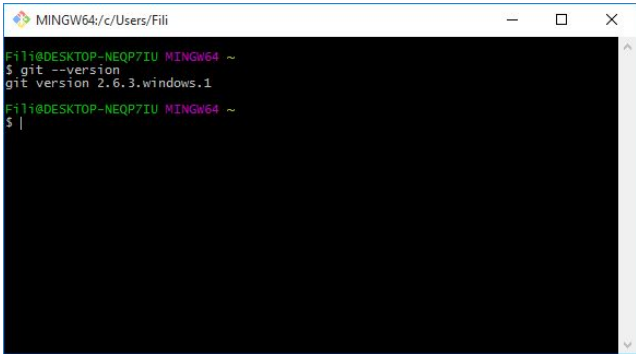


Node.js y NPM

- Herramientas de Desarrollo
 - VSCode
 - Línea de Comandos.
 - Control de versiones. Git.
- ¿Qué es Node.js?
- Instalación de Node.js y NPM
- Creación de un proyecto con Node.js y NPM
- Módulos y dependencias



Herramientas de Desarrollo

Editor de Código Fuente	Control de Versiones	Linea de Comando
 Visual Studio Code		

Visual Studio Code (VSCode)

VSCode es un editor de código altamente popular y potente, además de uso libre.

Características destacadas de VSCode:

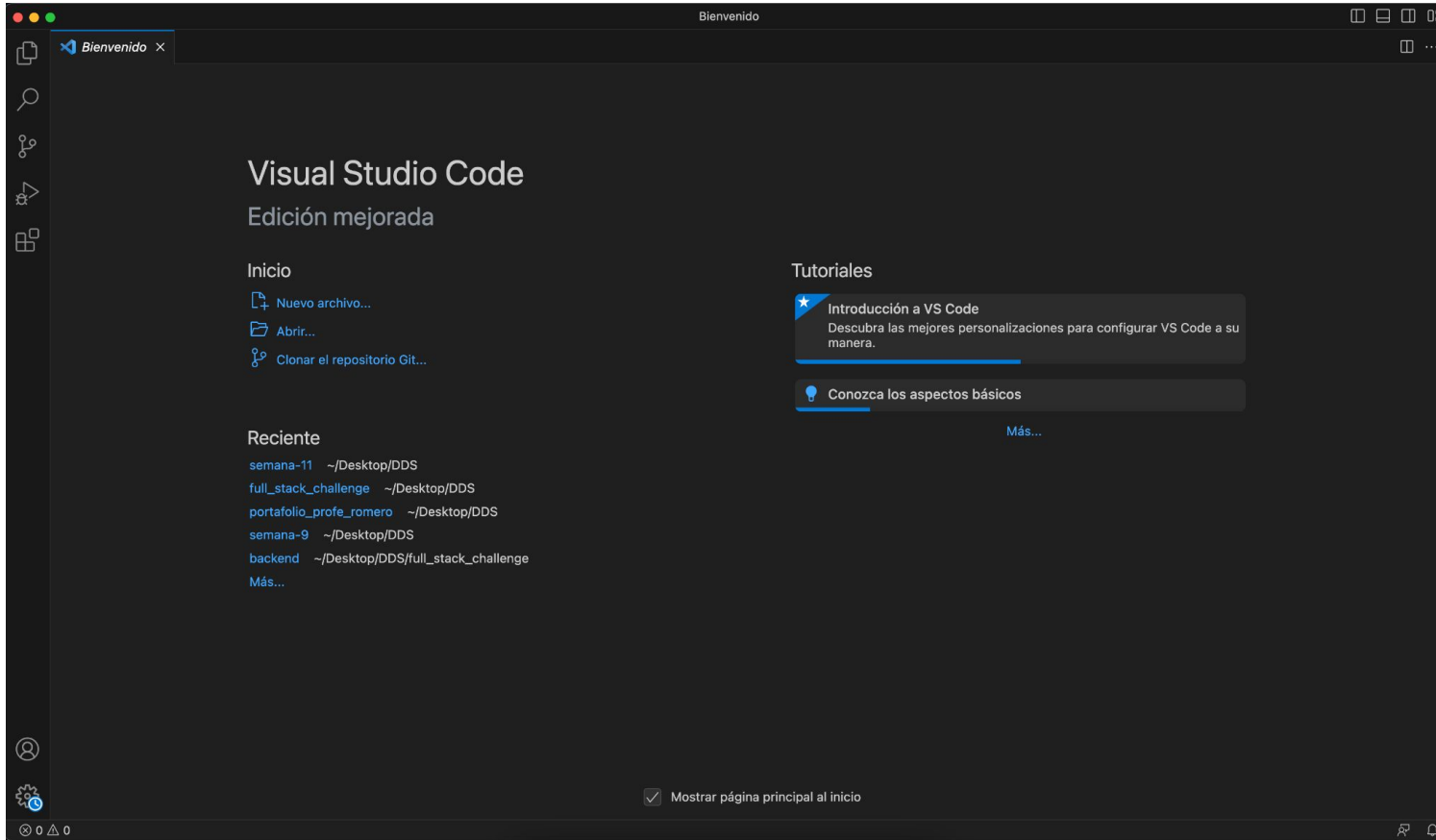
- **Resaltado de sintaxis:** facilita la lectura y comprensión del código.
- **Autocompletado inteligente:** ahorra tiempo al sugerir automáticamente código mientras escribes.
- **Depuración integrada:** permite identificar y solucionar errores de manera más eficiente.
- **Integración con control de versiones:** facilita la colaboración y el seguimiento de cambios.
- **Extensiones personalizables:** amplía la funcionalidad de VSCode según tus necesidades.



Instalar VSCode

- Descargar archivo de instalación usando el siguiente link:
 - <https://code.visualstudio.com/>
- Seguir las instrucciones de instalación.

Un paseo por Visual Studio Code (VSCode)



Instalar Extensiones Javascript

- **ESLint:** es un linter que analiza estáticamente su código para encontrar problemas en función de un conjunto de reglas preconfiguradas.
- **JavaScript Booster:** El asistente de "solución rápida" predeterminado en VS Code.
- **Prettier:** es un formateador de código y aplica un estilo coherente que ha preconfigurado en su proyecto.
- **GitHub Copilot:** Este es esencialmente un programador de pares de IA que brinda sugerencias sobre lo que cree que va a escribir en su código. Incluso puede escribir funciones automáticamente.



Visual Studio Code

Control de versiones con Git



Instalar Git

- Descargar archivo de instalación según el sistema operativo:
 - <https://git-scm.com/downloads>
- Seguir las instrucciones de instalación.

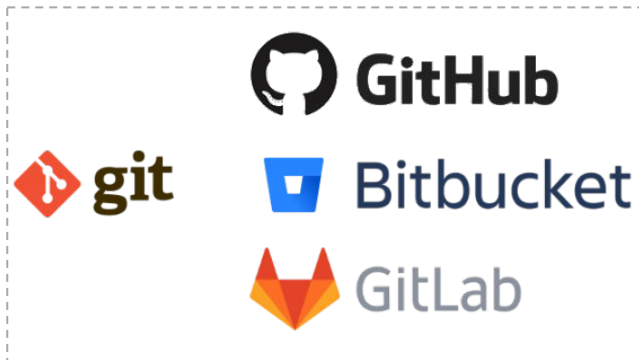
El control de versiones es esencial para el desarrollo colaborativo y el seguimiento de cambios en el código.

Permite a los desarrolladores **rastrear y gestionar los cambios en el código fuente** de manera eficiente.

Conceptos básicos de Git:

- **Repositorio:** almacena los archivos y su historial de cambios.
- **Commit:** registra los cambios realizados en el repositorio en un momento específico.
- **Branch:** crea ramas para trabajar en diferentes líneas de desarrollo.
- **Merge:** combina los cambios de diferentes ramas en una sola.

Git y Plataformas de Alojamiento



Resumiendo

- Git es la herramienta de control de versiones que permite el seguimiento de cambios en el código fuente.
- Las plataformas de alojamiento (GitLab, GitHub y Bitbucket) son lugares donde se pueden almacenar, colaborar y administrar los repositorios Git de manera más conveniente y efectiva.

GitLab, GitHub y Bitbucket son plataformas de alojamiento de repositorios basadas en Git.

Alojamiento de Repositorios:

- Proporcionan un lugar centralizado donde los desarrolladores pueden almacenar y gestionar sus repositorios Git.
- Los repositorios alojados contienen el historial completo de cambios, ramas y etiquetas del proyecto.

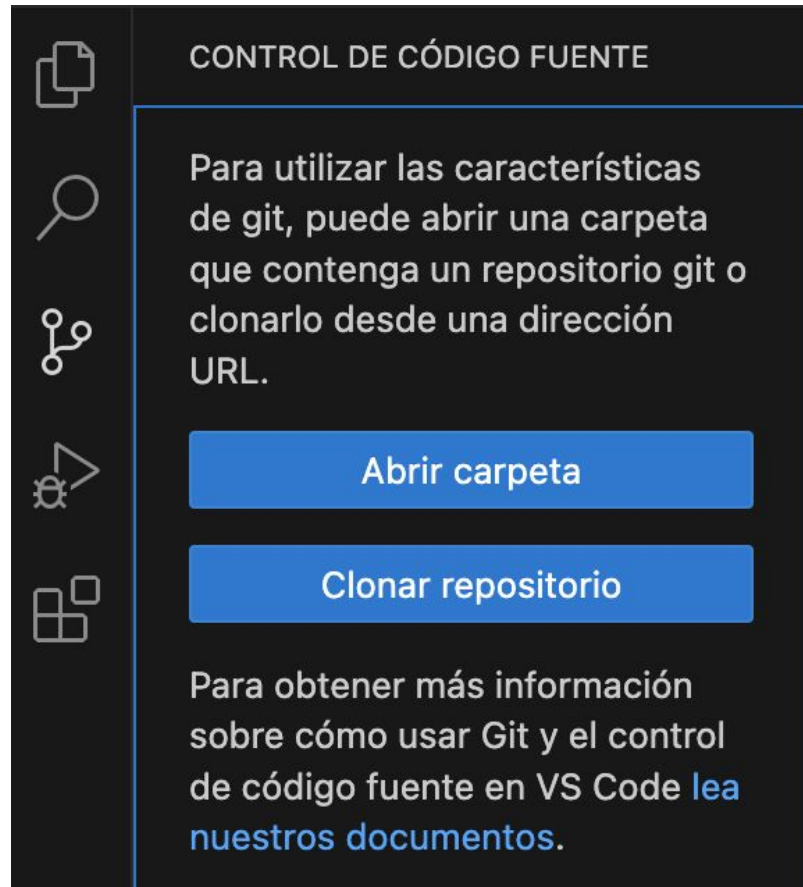
Colaboración y Gestión de Proyectos:

- Permiten a los equipos de desarrollo colaborar en los proyectos de forma más efectiva.
- Ofrecen características como gestión de problemas, seguimiento de tareas, asignación de tareas y comentarios en línea.

Integración Continua y Entrega Continua (CI/CD):

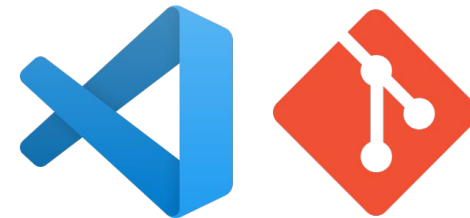
- Proporcionan herramientas para automatizar el proceso de construcción, pruebas y entrega de software.
- Facilitan la implementación rápida y confiable de nuevas versiones del software.

Git en VSCode



- **Extensiones VSCode**

- **GitLens:** Proporciona una visualización avanzada de información de Git, incluyendo quién modificó cada línea de código.
- **Git History:** Permite ver el historial de cambios de Git dentro de VSCode y realizar un seguimiento detallado de los commits.
- **Git Graph:** Muestra el historial de commits y las ramas de Git en un gráfico interactivo.



Línea de Comandos

- Las herramientas de línea de comandos son interfaces de texto que permiten **interactuar con el sistema operativo** y ejecutar comandos mediante instrucciones escritas en forma de texto.
- Estas herramientas ofrecen una forma poderosa y eficiente de interactuar con el sistema operativo, proporcionando una amplia variedad de funcionalidades y capacidades.
- Algunos ejemplos de herramientas de línea de comandos populares incluyen el :
 - **Terminal** en **macOS** y **Linux**.
 - **PowerShell** en **Windows**.

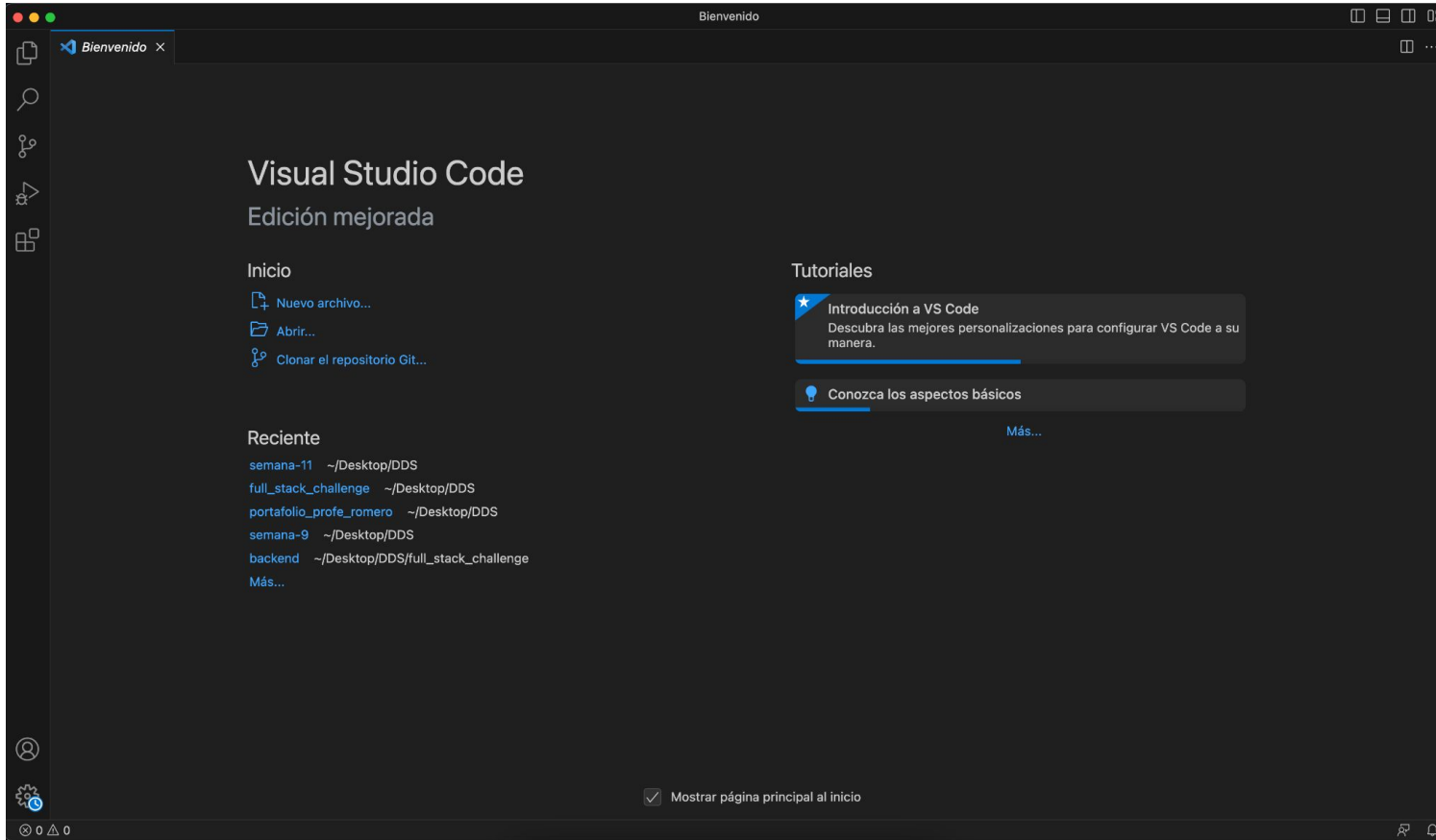


Comandos más usados

- **cd**: Cambia el directorio actual.
 - Ejemplo 1: **cd documentos** cambia al directorio "documentos".
 - Ejemplo 2: **cd ..** cambia al directorio padre.
- **ls (en Linux/Mac) o dir (en Windows)**: Lista los archivos y directorios en el directorio actual.
 - Ejemplo: **ls** muestra los archivos y directorios en el directorio actual.
- **mkdir**: Crea un nuevo directorio.
 - Ejemplo: **mkdir nueva_capeta** crea un directorio llamado "nueva_capeta".
- **pwd**: Muestra la ruta completa del directorio actual (present working directory).
 - Ejemplo: **pwd** muestra la ruta completa del directorio actual en la línea de comandos.



Línea de Comandos en VSCode



¿Qué es Node.js?

- Node.js es un entorno de tiempo de ejecución de JavaScript de código abierto y multiplataforma.
- Permite ejecutar código JavaScript fuera del navegador, en el servidor o en cualquier dispositivo que tenga instalado Node.js.
- Características principales de Node.js:
 - Basado en el motor V8 de Google Chrome, lo que proporciona un rendimiento rápido y eficiente.
 - Utiliza un modelo de operaciones de entrada/salida sin bloqueo y orientado a eventos, lo que lo hace altamente escalable y adecuado para aplicaciones en tiempo real y de alto rendimiento.
 - Posee un ecosistema rico y activo de paquetes y módulos npm (Node Package Manager) que permiten a los desarrolladores reutilizar y compartir código fácilmente.



NodeJs: Instalación

Node.js® es un entorno de ejecución para JavaScript construido con V8, motor de JavaScript de Chrome.

Descargar para macOS

18.16.0 LTS

Recomendado para la mayoría

20.2.0 Actual

Últimas características

[Otras Descargas](#) | [Cambios](#) | [Documentación de la API](#) [Otras Descargas](#) | [Cambios](#) | [Documentación de la API](#)

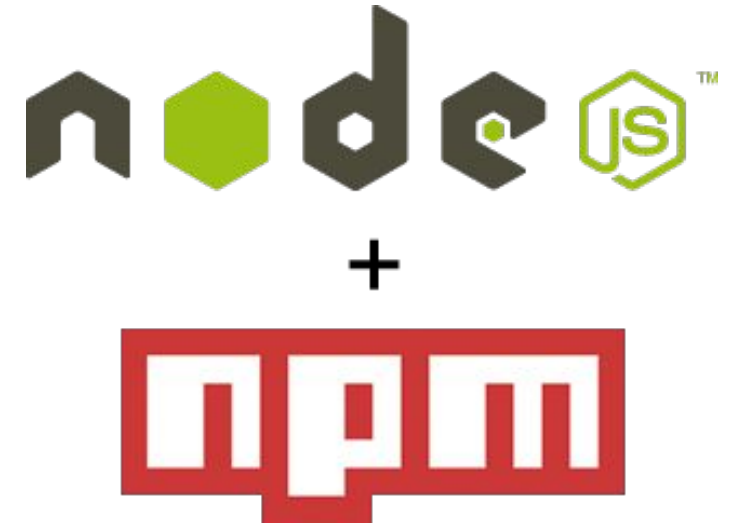
O eche un vistazo al [Programa de soporte a largo plazo \(LTS\)](#).

Instalar Nodejs

- Descargar archivo de instalación:
 - <https://nodejs.org/es>
- Seguir las instrucciones de instalación.

Nodejs - NPM

- **npm (Node Package Manager)** es el gestor de paquetes predeterminado para Node.js.
- Es una herramienta que nos **permite descargar, instalar y administrar fácilmente paquetes de código reutilizable** para nuestros proyectos de Node.js.
- Características principales:
 - **Repositorio de paquetes:** npm cuenta con un repositorio en línea que alberga miles de paquetes de código abierto. <https://www.npmjs.com/>
 - **Instalación sencilla:** Permite instalar paquetes específicos o dependencias de proyectos mediante comandos simples.
 - **Manejo de dependencias:** Administra automáticamente las dependencias de un proyecto y asegura que las versiones correctas estén instaladas.
 - **Scripts personalizados:** Permite ejecutar scripts personalizados definidos en el archivo "package.json" de un proyecto.
 - **Actualizaciones regulares:** npm se actualiza periódicamente para agregar nuevas funciones y mejoras.



¿Para qué usamos npm?

- Esta herramienta se utiliza con la línea de comandos.
- Para iniciar el uso de npm en un proyecto, debemos ejecutar el comando

> *npm init -y*

para generar un archivo "package.json" que almacena información sobre el proyecto y sus dependencias.

- Luego, puedes utilizar comandos como:

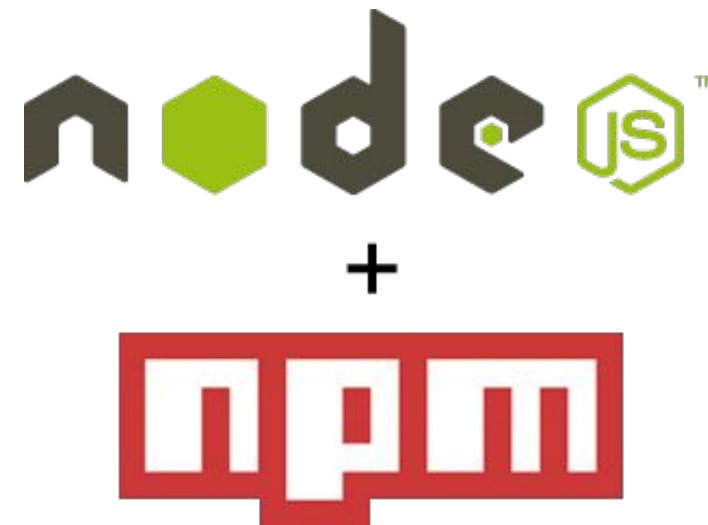
> *npm install*

para instalar las dependencias necesarias.

- Además, puedes buscar paquetes en el repositorio de npm y utilizar

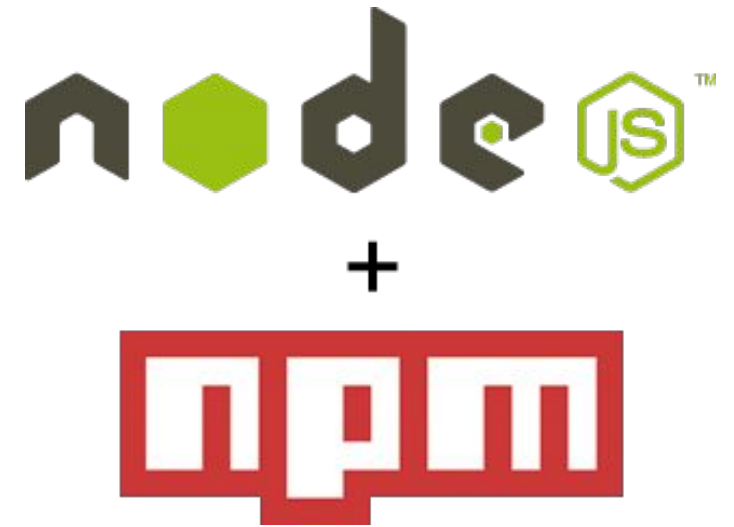
> *npm install <nombre_del_paquete>*

para descargar e instalarlos en tu proyecto.



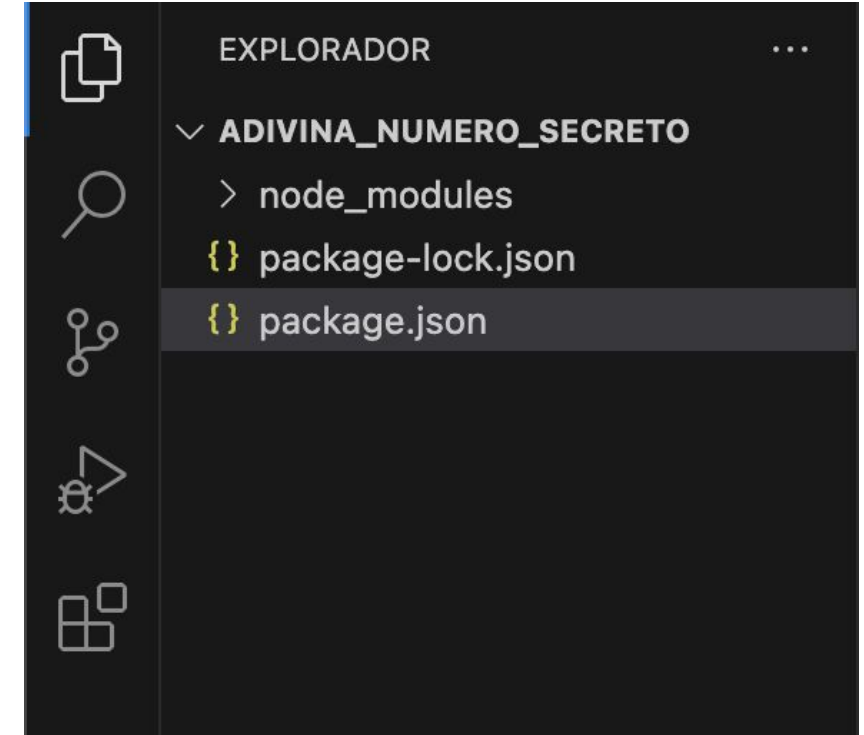
npm - package.json

```
{ } package.json > ...
1  {
2    "dependencies": {
3      "chance": "^1.1.11"
4    },
5    "name": "nodejs_npm",
6    "version": "1.0.0",
7    "main": "app.js",
8    "scripts": {
9      "test": "echo \"Error: no test specified\" && exit 1"
10   },
11   "author": "",
12   "license": "ISC",
13   "description": ""
14 }
```



Instalación paquetes npm

- Como resultado de instalar un paquete npm sucede lo siguiente:
 - Genera una carpeta dentro de node_modules con el nombre del paquete. (La carpeta node_modules se genera la primera vez que instalamos un paquete o si ejecutar el comando npm install).
 - Genera un archivo package-lock.json, donde indica los paquetes que están en la carpeta node_modules.json.
 - Modifica el archivo package.json para agregar una dependencia.
- Tanto el archivo package-lock.json como la carpeta node_modules se generan automáticamente por npm, por lo tanto podemos borrarlas en cualquier momento y que se generen nuevamente con el comando npm install.



Bueno, Vamo a Codea!!!

Vamos a utilizar **Chance** es una biblioteca que genera datos aleatorios de manera realista y controlada. Puede ser útil para simular datos en tus aplicaciones o para demostrar cómo trabajar con datos generados aleatoriamente.

1. Instalar librería **chance** utilizando el siguiente comando en tu terminal.

```
> npm install chance
```

2. Una vez que **chance** esté instalada, puedes utilizarlo en tu aplicación. Creamos un nuevo archivo JavaScript, por ejemplo, app.js, y agrega el siguiente código:
3. Guarda el archivo app.js.
4. Ejecuta tu aplicación en la terminal utilizando el siguiente comando:

```
> node app.js
```

5. Resultado esperado: Veremos que se imprime un nombre, una edad y un email aleatorios en la consola.



```
// Importa la biblioteca chance
const Chance = require('chance');

// Crea una instancia de Chance
const chance = new Chance();

// Genera datos aleatorios utilizando Chance
const randomName = chance.name();
const randomAge = chance.age();
const randomEmail = chance.email();

// Imprime los datos aleatorios en la consola
console.log('Nombre aleatorio:', randomName);
console.log('Edad aleatoria:', randomAge);
console.log('Email aleatorio:', randomEmail);
```

Introducción a los Módulos de Node.js

- Los módulos de Node.js son una forma de **organizar** y **reutilizar el código** en aplicaciones de Node.js.
- Los módulos de Node.js se pueden cargar mediante la función `require()` y se exportan utilizando `module.exports` o `exports`.



```
const modulo = require('../../../modulo');
```

Ejemplo de Uso de Módulos en Node.js

- Supongamos que tenemos dos archivos: calculadora.js y app.js.

En **calculadora.js**, definimos una función para sumar dos números:

```
JS calculadora.js > ...  
1  const sumar = (a, b) => {  
2    |    return a + b;  
3  }  
4  
5  module.exports = { sumar };
```

En **app.js**, cargamos el módulo calculadora.js utilizando require() y utilizamos la función de suma:

```
JS app.js > ...  
1  const { sumar } = require('./calculadora');  
2  
3  console.log(sumar(5,3));
```

Seguimos Codeando!!!

- En este caso NO vamos a usar la herramienta javascript-tester, sino que ahora en VSCode.
- Crear un programa en JavaScript que permita realizar un registro de estudiantes. El programa debe tener las siguientes funcionalidades:
 1. Permitir al usuario ingresar la cantidad de estudiantes que desea registrar.
 2. Solicitar al usuario que ingrese los nombres y edades de los estudiantes.
 3. Almacenar la información de cada estudiante en un objeto con las propiedades nombre y edad.
 4. Guardar cada objeto del estudiante en un array.
 5. Mostrar en pantalla la lista de estudiantes registrados, mostrando el nombre y la edad de cada uno.



Llevemos nuestro código a GitHub



Aplicación Web Nodejs

```
const http = require('http');

const server = http.createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('¡Hola, mundo desde un servidor Node.js!');
});

server.listen(3000, 'localhost', () => {
  console.log('Servidor Node.js en ejecución en http://localhost:3000/');
});
```

¿Cómo iniciamos la aplicación?

- En VSCode buscamos una carpeta vacía para trabajar
- Creamos un archivo app.js
- Iniciamos un terminal de línea de comandos.
- Ejecutamos el siguiente comando:

node app.js

- Abrimos un navegador e ingresamos la siguiente url:

<http://localhost:3000>

- Resultado esperado: en el navegador debería ver la frase: **“¡Hola, mundo desde un servidor Node.js!”**

Actividad 2: Paso a Paso Aplicación Nodejs

- Seguir las instrucciones de la actividad publicada en la UVE.



MUCHAS GRACIAS

ANDÉN
Centro de Innovación
y Emprendimientos Tecnológicos

SECRETARÍA DE
EXTENSIÓN
UNIVERSITARIA
UTN - FRC

SEU

UTN
Facultad Regional Córdoba

Agencia
**CÓRDOBA
JOVEN**

 **CÓRDOBA**
entre todos