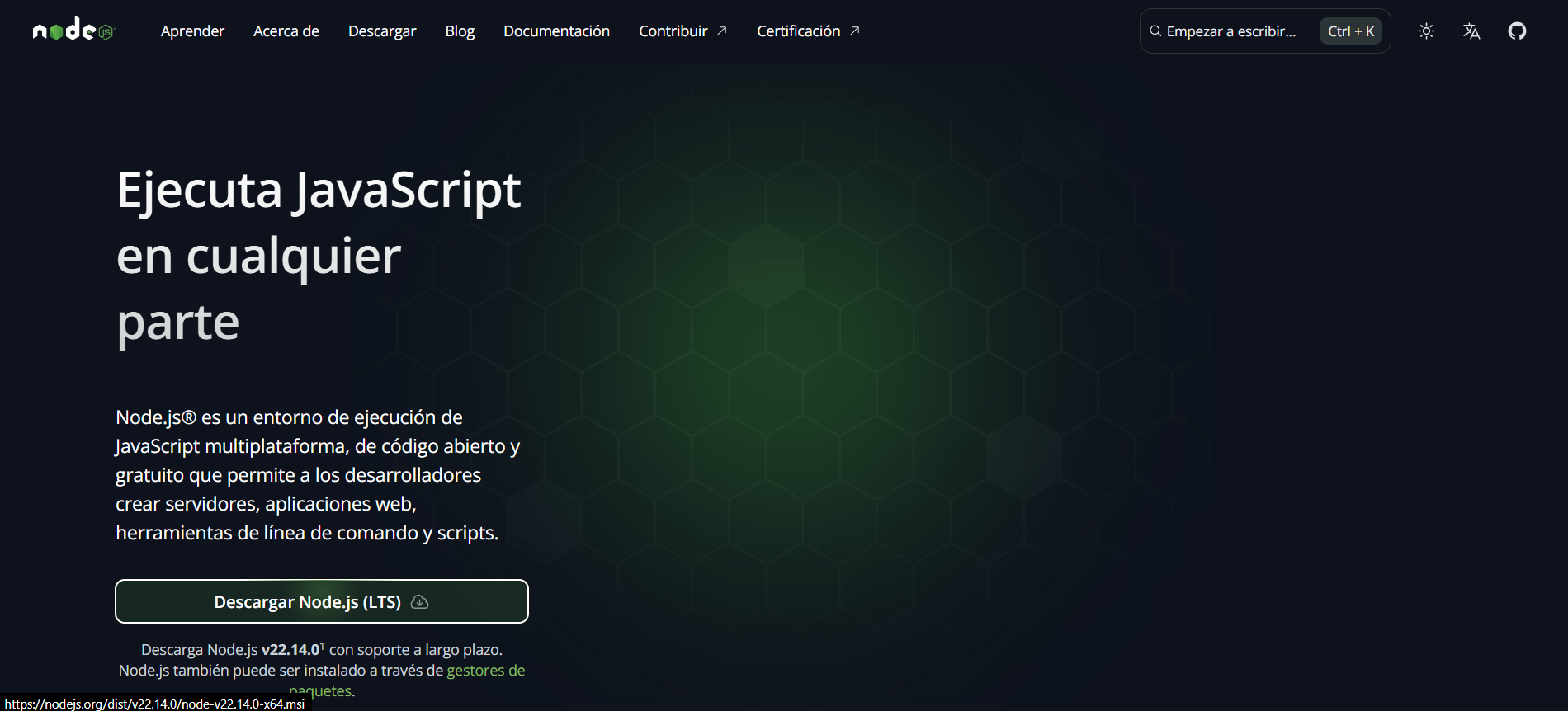
# Trabajo Práctico: Instalación de herramientas

### **Descargar Node.js**

1. Abre tu navegador web y dirígete al sitio oficial de Node.js:<https://nodejs.org>
2. En la página de inicio, verás un boton:
   * **Descargar Node.js LTS (Long Term Support):** Recomendado para la mayoría de los usuarios, ya que es más estable.
3. Descarga la versión que prefieras según tu sistema operativo (Windows, macOS o Linux).



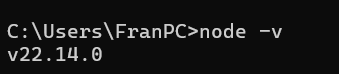
### **Instalar Node.js**

#### **En Windows:**

1. Ejecuta el archivo descargado (.msi).
2. Sigue el asistente de instalación:
   * Acepta los términos y condiciones.
   * Selecciona las opciones predeterminadas.
   * Espera a que finalice la instalación.
3. Una vez instalado, abre la terminal (cmd o PowerShell) y ejecuta:

node -v

Si muestra un número de versión, significa que Node.js se ha instalado correctamente.



npm -v

Si muestra un número de versión, significa que npm se ha instalado correctamente.



#### **En macOS:**

1. Abre el archivo .pkg descargado y sigue los pasos del asistente.
2. Una vez completada la instalación, abre la Terminal y ejecuta:

node -v

Si muestra una versión, la instalación fue exitosa.

#### **En Linux (Ubuntu/Debian):**

1. Abre la terminal y ejecuta los siguientes comandos para instalar Node.js desde el repositorio oficial

curl -fsSL https://deb.nodesource.com/setup\_lts.x | sudo -E bash -

sudo apt install -y nodejs

1. Verifica la instalación con:

node -v

1. Verificar la Instalación

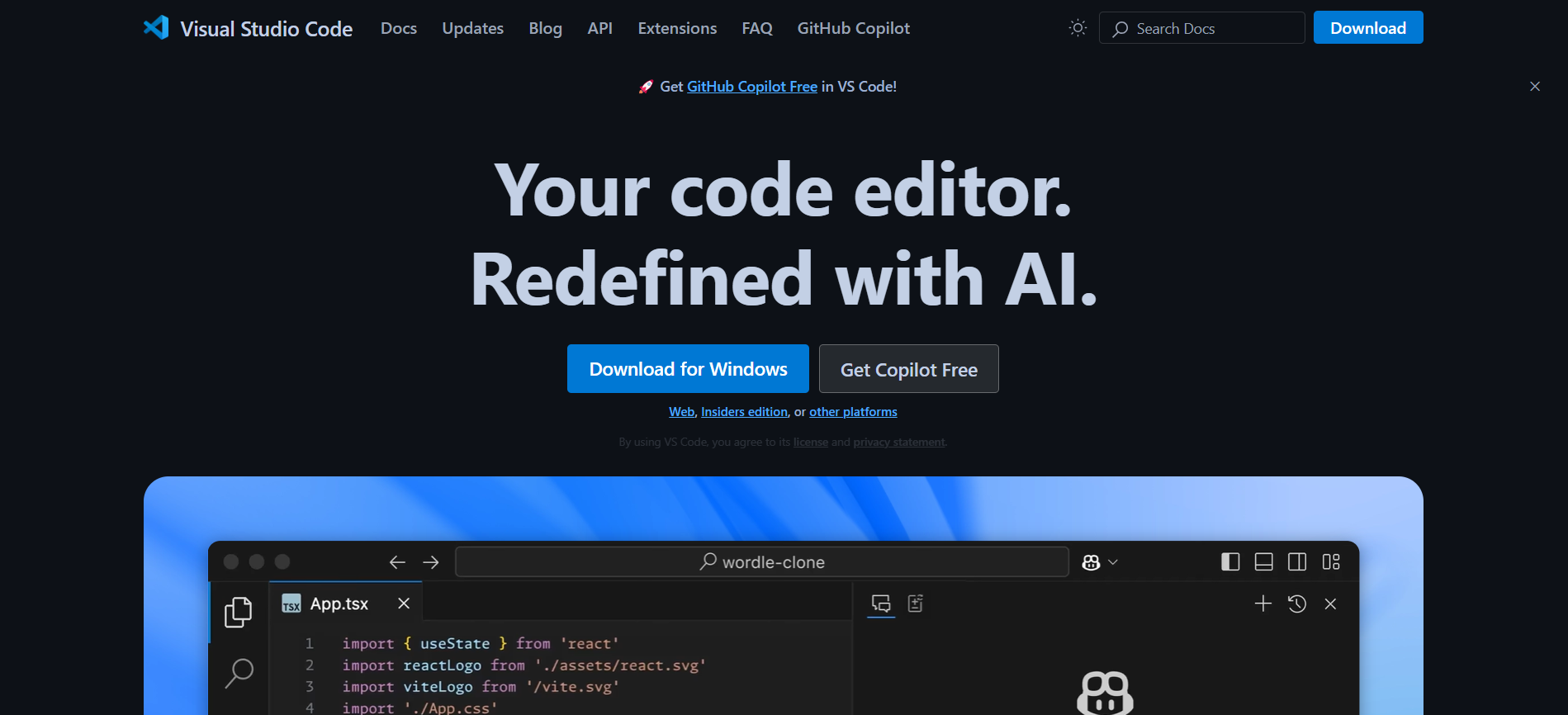
Después de la instalación, verifica también la versión de npm (Node Package Manager) con:

npm -v

Si ves un número de versión, significa que npm también se instaló correctamente.

**Descargar e Instalar Visual Studio Code**

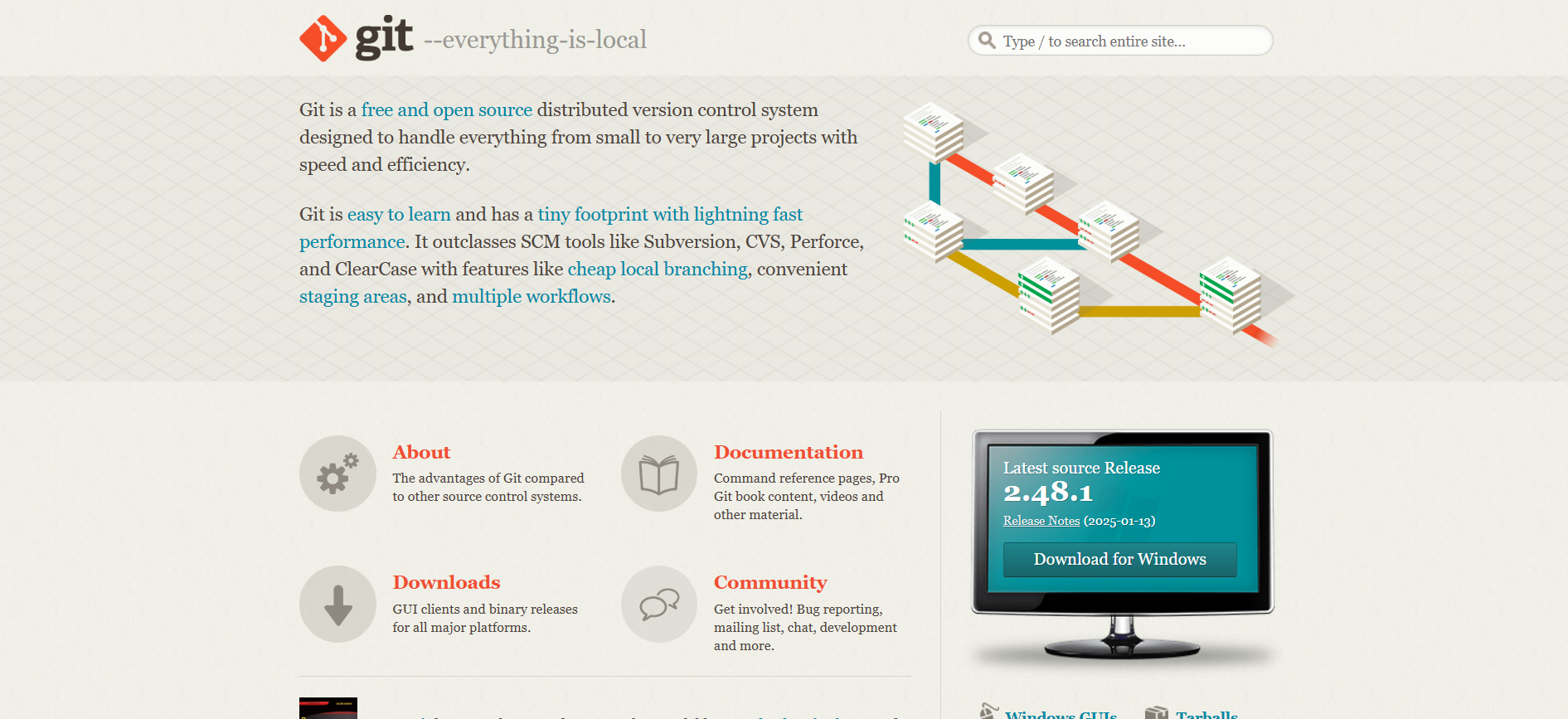
1. Abre tu navegador web y dirígete al sitio oficial de Visual Studio Code:<https://code.visualstudio.com>



1. Haz clic en el botón de descarga y selecciona la versión correspondiente a tu sistema operativo (Windows, macOS o Linux).
2. Una vez descargado el archivo de instalación:
   * **En Windows:** Ejecuta el archivo .exe, acepta los términos y sigue el asistente de instalación.
   * **En macOS:** Abre el archivo .zip descargado y mueve Visual Studio Code a la carpeta Aplicaciones.
   * **En Linux:** Dependiendo de la distribución, instala el archivo .deb o .rpm usando la terminal.
3. Abre Visual Studio Code y verifica que funciona correctamente.

### **Descargar e Instalar Git**

1. Abre tu navegador web y dirígete al sitio oficial de Git:<https://git-scm.com>



1. Descarga la versión adecuada para tu sistema operativo (Windows, macOS o Linux).
2. Instala Git siguiendo los pasos correspondientes:
   * **En Windows:** Ejecuta el instalador .exe, acepta los términos y sigue el asistente dejando las opciones predeterminadas.
   * **En macOS:** Instala Git usando Homebrew ejecutando en la terminal:

brew install git

* + **En Linux (Ubuntu/Debian):** Ejecuta en la terminal:  
    sudo apt install git

1. Verifica la instalación ejecutando en la terminal:  
   git --version

Si ves un número de versión, significa que Git se ha instalado correctamente.

### **Descargar e Instalar MongoDB Compass**

1. Abre tu navegador web y dirígete al sitio oficial de MongoDB Compass:<https://www.mongodb.com/try/download/compass>
2. Descarga la versión adecuada para tu sistema operativo (Windows, macOS o Linux).
3. Instala MongoDB Compass siguiendo los pasos correspondientes:
   * **En Windows:** Ejecuta el instalador .exe, acepta los términos y sigue el asistente de instalación.
   * **En macOS:** Abre el archivo .dmg descargado y mueve MongoDB Compass a la carpeta Aplicaciones.
   * **En Linux:** Dependiendo de la distribución, instala el archivo .deb o .rpm usando la terminal.
4. Abre MongoDB Compass y verifica que funciona correctamente.

### **Descargar e Instalar Docker Desktop**

1. Abre tu navegador web y dirígete al sitio oficial de Docker Desktop: https://www.docker.com/products/docker-desktop
2. Descarga la versión adecuada para tu sistema operativo (Windows o macOS).
3. Instala Docker Desktop siguiendo los pasos correspondientes:
   * **En Windows:** Ejecuta el instalador .exe, acepta los términos y sigue el asistente de instalación. Asegúrate de habilitar la opción de integración con WSL 2 si usas Windows.
   * **En macOS:** Abre el archivo .dmg descargado y arrastra Docker a la carpeta Aplicaciones.
4. Inicia Docker Desktop y espera a que se configure correctamente.
5. Verifica la instalación abriendo una terminal y ejecutando:  
   docker --version  
   Si muestra un número de versión, significa que Docker se ha instalado correctamente.

**Actividad**

Responder las siguientes preguntas sobre node

1. ¿Qué es Node.js y en qué se diferencia de JavaScript en el navegador?

Node.js es un entorno de ejecución de JavaScript del lado del servidor basado en el motor V8 de Chrome, mientras que javascript es un lenguaje de programación de alto nivel. La principal diferencia con JavaScript es que Node.js permite ejecutar JavaScript fuera del navegador, lo que lo hace ideal para crear aplicaciones del lado del servidor, como APIs, servidores web, etc. Además, Node.js tiene acceso a módulos y APIs de bajo nivel del sistema operativo, como el sistema de archivos, lo que no es posible en el navegador.

1. ¿Qué son las versiones LTS?

son versiones de Node.js que reciben actualizaciones de seguridad y mantenimiento durante un periodo extendido (generalmente 18 meses). Estas versiones son recomendadas para producción, ya que garantizan estabilidad y soporte durante un tiempo considerable.

1. Explica la diferencia entre un proceso síncrono y uno asíncrono

En un proceso síncrono la ejecución del código ocurre de manera secuencial. Cada operación debe esperar a que la anterior termine antes de continuar. Esto puede causar bloqueos si alguna operación es lenta, ya que el resto del código no se ejecutará hasta que termine la operación anterior. En cambio, en un proceso asíncrono, el código se ejecuta sin esperar que otras operaciones terminen. Esto es útil para tareas que toman tiempo, como leer archivos o hacer solicitudes HTTP, ya que el código puede continuar ejecutándose mientras espera la respuesta de esas tareas.

1. ¿Cómo se maneja la asincronía en Node.js? Explica las diferencias entre callbacks, Promises y async/await.

Las callbacks son funciones que se pasan como argumentos a otras funciones y se ejecutan una vez que se completa una tarea. Sin embargo, el uso excesivo de callbacks puede llevar al "callback hell" o "pirámide de la muerte", dificultando la lectura del código. Las promises son objetos que representan el resultado de una operación asíncrona que puede ser completada con éxito (resolve) o fallar (reject). Las Promises permiten encadenar operaciones de manera más limpia que los callbacks. Por último, las async/await son una forma más moderna y legible de trabajar con Promises. async marca una función como asíncrona y await permite esperar de manera sincrónica a que una Promise se resuelva antes de continuar con la ejecución del código.

1. ¿Qué es el objeto global en Node.js y en qué se diferencia de window en el navegador?

El objeto global en Node.js es un objeto que contiene todas las variables y funciones globales accesibles desde cualquier parte del código. Se puede acceder a él a través de global. En el navegador, el objeto equivalente es window, que representa el contexto global del navegador. La diferencia principal es que global en Node.js es un entorno de servidor y no tiene acceso a las APIs del navegador como el DOM.

1. ¿Node.js es multithreading?

No, Node.js no es multithreading en el sentido tradicional. Aunque utiliza múltiples hilos internamente para operaciones como el manejo de archivos y otras tareas de I/O, su modelo de ejecución es de un solo hilo (event loop). Sin embargo, se pueden usar módulos como worker\_threads para crear hilos en Node.js si se necesita ejecutar tareas paralelas.

1. ¿Qué es el Event Loop en Node.js y cuál es su papel en la ejecución de código asíncrono?

El Event Loop es el mecanismo que permite a Node.js manejar operaciones asíncronas sin bloquear el hilo principal. Node.js ejecuta las tareas asíncronas, como leer archivos o hacer solicitudes HTTP, y coloca las operaciones pendientes en una cola de eventos. El Event Loop procesa esas operaciones cuando el hilo principal está libre, asegurando que el código siga ejecutándose mientras se esperan respuestas de tareas asíncronas.

1. Mencionar tres ventajas de Node.js

 **Rendimiento elevado**: Debido a su arquitectura de un solo hilo y su modelo de I/O no bloqueante, Node.js es muy eficiente para aplicaciones que requieren un alto rendimiento y manejo de muchas conexiones simultáneas.

 **Escalabilidad**: Su capacidad para manejar múltiples conexiones simultáneas lo hace ideal para aplicaciones escalables, como aplicaciones web en tiempo real.

 **Gran ecosistema**: Node.js tiene un ecosistema muy grande gracias a npm, lo que permite a los desarrolladores acceder a una gran cantidad de bibliotecas y módulos.

1. menciona tres desventajas de Node.js.

 **No es adecuado para tareas de CPU intensivas**: Dado que Node.js es de un solo hilo, las operaciones que requieren mucho poder de CPU (como procesamiento de imágenes o algoritmos complejos) pueden bloquear el event loop y afectar el rendimiento.

 **Callback hell**: Aunque las Promises y async/await han mejorado esto, el abuso de callbacks puede llevar a un código difícil de mantener y leer.

 **Inmadurez para aplicaciones de alto rendimiento**: A pesar de su excelente rendimiento en operaciones de I/O, Node.js no es la mejor opción para aplicaciones que requieren un rendimiento extremo en operaciones de CPU intensivas.

1. ¿Podrías nombrar algunas bibliotecas que los desarrolladores utilizan frecuentemente con Node.js?

 **Express**: Framework minimalista para crear servidores web y APIs.

 **Socket.io**: Para la creación de aplicaciones en tiempo real que requieren comunicación bidireccional entre el servidor y el cliente (como chats en vivo).

 **Mongoose**: Biblioteca para interactuar con bases de datos MongoDB de manera fácil y estructurada.

 **Lodash**: Utilitaria que ofrece funciones útiles para trabajar con arreglos, objetos y funciones en JavaScript.

 **Async**: Proporciona control sobre funciones asíncronas y facilita la gestión de flujos asíncronos complejos.