**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 que permite la ejecución de código JavaScript en el lado del servidor, utilizando el motor V8 de Google Chrome. La diferencia principal entre Node.js y JavaScript en el navegador es el entorno en el que se ejecuta el código. JavaScript en el navegador se ejecuta en el motor del navegador, Node.js se ejecuta en el servidor y usa V8.

1. ¿Qué son las versiones LTS?

son versiones con soporte a largo plazo, lo que significa que reciben actualizaciones de seguridad y correcciones críticas durante un período extendido.

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

Un proceso síncrono ejecuta las tareas una tras otra en orden, mientras que un proceso asíncrono permite ejecutar otras tareas sin esperar a que una termine.

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

La asincronía se maneja con callbacks, Promises y async/await

Una función de callback es una función que se pasa a otra función como un argumento, que luego se invoca dentro de la función externa para completar algún tipo de rutina o acción.

Promises, son objetos que representan un valor futuro; pueden estar en estado pendiente, resuelta o rechazada.

La palabra “async” antes de una función para decir que es asíncrona y la palabra “await”, que pausa la ejecución de la función a la espera de que se obtengan todos los datos y siempre debe estar dentro de la función async.

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

En Node.js, el objeto global es global, mientras que en los navegadores es window.

**Diferencias clave:**

* global en Node.js no tiene acceso al DOM ni a APIs del navegador.
* window en el navegador maneja elementos como document, localStorage, y eventos del usuario.
* En Node.js, se usa module.exports y require() para módulos, mientras que en el navegador se usan ES Modules (import/export).
* this en el ámbito global de Node.js no apunta a global, pero en el navegador sí apunta a window.

1. ¿Node.js es multithreading?

No, Node.js no es multithreading en su ejecución principal, ya que está basado en un modelo single-threaded y asíncrono impulsado por el event loop. Sin embargo, puede manejar múltiples operaciones concurrentes de manera eficiente gracias a su arquitectura basada en eventos y callbacks.

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

El Event Loop en Node.js es un mecanismo que permite la ejecución asíncrona y no bloqueante en un solo hilo. Su función principal es gestionar y coordinar la ejecución de callbacks, promesas, y otras operaciones asincrónicas, como lectura de archivos, peticiones HTTP y consultas a bases de datos.

1. Mencionar tres ventajas de Node.js

* Asincronía y no bloqueo: Gracias al Event Loop, Node.js maneja múltiples solicitudes de manera eficiente sin bloquear el hilo principal.
* Escalabilidad: Su arquitectura basada en eventos permite construir aplicaciones escalables con alto rendimiento, ideales para APIs y microservicios.
* Gran ecosistema: Cuenta con NPM (Node Package Manager), que ofrece miles de paquetes listos para usar, facilitando el desarrollo.

1. menciona tres desventajas de Node.js.

* No es ideal para tareas intensivas en CPU: Al ser single-threaded, puede tener problemas de rendimiento con procesos que requieren alto uso de CPU.
* Callback Hell: El uso excesivo de callbacks puede hacer que el código sea difícil de leer y mantener, aunque esto se ha mitigado con Promesas y async/await.
* Inestabilidad en paquetes: Algunos módulos de NPM pueden ser inseguros o carecer de mantenimiento adecuado, lo que puede generar problemas en proyectos a largo plazo.

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

Express.js (Framework web minimalista para crear APIs y aplicaciones web rápidamente.)

Socket.io (Biblioteca para implementar WebSockets y comunicación en tiempo real.)

Axios (Cliente HTTP basado en promesas para realizar peticiones a APIs externas.)

bcrypt.js (Biblioteca para hashing de contraseñas)