Pregunta 1: ¿Cómo mejora el rendimiento de una aplicación Node.js el módulo Cluster?

Respuesta correcta: Creando múltiples instancias de la aplicación Node.js

Teoría: El módulo Cluster en Node.js permite la creación de múltiples instancias del proceso Node.js que se ejecutan en paralelo en diferentes núcleos de CPU. Esto mejora el rendimiento de una aplicación al permitir que múltiples solicitudes sean atendidas simultáneamente, aprovechando mejor los recursos de hardware disponibles. Cada instancia (o "worker") puede manejar su propio conjunto de conexiones, distribuyendo la carga de trabajo.

Pregunta 2: ¿Qué método del módulo HTTP se usa para crear un servidor HTTP en Node.js?

Respuesta correcta: http.createServer()

Teoría: El método http.createServer() es utilizado para crear un servidor HTTP en Node.js. Este método acepta una función de callback que es ejecutada cada vez que el servidor recibe una solicitud (request). Dentro de esta función, puedes manejar las solicitudes y enviar respuestas.

Pregunta 3: ¿Para qué tipos de aplicaciones se usa comúnmente Node.js?

Respuesta correcta: Node.js se emplea comúnmente para aplicaciones de una sola página (SPAs), aplicaciones en tiempo real, aplicaciones para dispositivos de Internet de las Cosas (IoT) y aplicaciones de transmisión de datos.

Teoría: Node.js es muy utilizado en aplicaciones que requieren alta concurrencia y baja latencia, como aplicaciones en tiempo real (ej. chats en línea), SPAs (Single Page Applications), aplicaciones IoT, y aplicaciones de streaming de datos. Su modelo de I/O no bloqueante lo hace ideal para estas situaciones.

Pregunta 4: ¿Cuándo es adecuada la arquitectura serverless para el desarrollo de aplicaciones?

Respuesta correcta: Cuando deseas enfocarte en escribir código y no preocuparte por el aprovisionamiento de servidores

Teoría: La arquitectura serverless permite a los desarrolladores enfocarse en la escritura de código sin preocuparse por la gestión de la infraestructura subyacente. Las plataformas serverless manejan automáticamente el aprovisionamiento, escalado y mantenimiento de servidores, lo que es ideal para aplicaciones donde se busca simplicidad y costos reducidos, aunque a costa de tener menos control sobre la infraestructura.

Pregunta 5: ¿Cómo se diferencia la arquitectura de microservicios de la arquitectura monolítica?

Respuesta correcta: Los microservicios están acoplados de manera flexible y consisten en servicios que se pueden desplegar de forma independiente

Teoría: La arquitectura de microservicios se caracteriza por descomponer una aplicación en servicios pequeños, independientes y desacoplados que se pueden desplegar y escalar de manera independiente. A diferencia de la arquitectura monolítica, donde todos los componentes están estrechamente integrados en una única base de código, los microservicios permiten una mayor flexibilidad y mantenimiento a largo plazo.

Pregunta 6: ¿Qué método se usa para crear un clúster de procesos de Node.js usando el módulo Cluster?

Respuesta correcta: cluster.fork()

Teoría: El método cluster.fork() se utiliza en Node.js para crear un nuevo proceso worker que ejecuta el código de la aplicación. Este método permite aprovechar todos los núcleos de la CPU creando procesos que comparten el mismo puerto del servidor, manejando de forma eficiente múltiples solicitudes simultáneamente.

Pregunta 7: ¿Qué es Node.js y cuál de las siguientes afirmaciones sobre él es verdadera?

Respuesta correcta: Node.js es un entorno de ejecución de JavaScript de código abierto que se puede usar en varios sistemas operativos

Teoría: Node.js es un entorno de ejecución de JavaScript de código abierto basado en el motor V8 de Google Chrome. Se utiliza para desarrollar aplicaciones del lado del servidor y es compatible con varios sistemas operativos, incluidos Windows, macOS y Linux. Node.js es conocido por su modelo de I/O no bloqueante, lo que lo hace ideal para aplicaciones escalables.

Pregunta 9: ¿Cuál es una de las principales ventajas de usar Node.js para aplicaciones en tiempo real?

Respuesta correcta: Node.js ofrece una conexión continua a través de WebSockets, lo que permite tiempos de respuesta más rápidos

Teoría: Una de las grandes ventajas de Node.js es su capacidad para manejar conexiones continuas mediante WebSockets, lo que permite una comunicación bidireccional en tiempo

real entre el cliente y el servidor. Esto es especialmente útil para aplicaciones como chats en línea, juegos multijugador y otras aplicaciones en tiempo real.

Pregunta 10: ¿Cuál es la característica clave del event loop en Node.js?

Respuesta correcta: Gestiona operaciones asincrónicas, asegurando la ejecución no bloqueante

Teoría: El event loop es una característica fundamental de Node.js que permite manejar operaciones asincrónicas sin bloquear la ejecución del código. Esto significa que Node.js puede manejar múltiples operaciones I/O (como la lectura de archivos o solicitudes HTTP) de manera eficiente, sin esperar a que se completen antes de pasar a la siguiente tarea, lo que mejora significativamente el rendimiento.