Core Node.js Concepts

1. **What is Node.js, and how does it work?**

   Node.js is a runtime environment that allows JavaScript to run on the server side. It uses the V8 JavaScript engine and operates on a single-threaded, event-driven architecture, making it lightweight and efficient for I/O-heavy tasks.

2. **What is the Event Loop in Node.js?**

   The Event Loop is a mechanism that allows Node.js to perform non-blocking I/O operations. It continuously checks the call stack and processes events from the event queue, enabling asynchronous behavior.

3. **What is the role of libuv in Node.js?**

   libuv is a C library that provides the Event Loop and handles asynchronous I/O operations like file system tasks, networking, and timers.

4. **What is the difference between setImmediate() and process.nextTick()?**

   process.nextTick() executes immediately after the current operation, before the Event Loop continues.
   setImmediate() executes in the next iteration of the Event Loop.

5. **How does Node.js handle child threads?**

   Node.js uses the worker_threads module to create child threads for CPU-intensive tasks, keeping the main thread free for I/O operations.

6. **What is the purpose of the Buffer class in Node.js?**

   The Buffer class is used to handle binary data directly, which is essential for working with files, network protocols, and other I/O operations.

7. **What are streams in Node.js?**

   Streams are collections of data that allow processing data in chunks instead of loading it all into memory at once. They improve performance and efficiency.

8. **What are the different types of streams in Node.js?**

   Readable (e.g., reading a file)
   Writable (e.g., writing to a file)
   Duplex (e.g., TCP sockets)
   Transform (e.g., zlib compression)

9. **What is the difference between require and import?**

   require is CommonJS syntax, used in Node.js for module loading.
   import is ES6 syntax, used for static module loading in modern JavaScript.

10. **What is the global object in Node.js?**

    The global object is similar to the window object in browsers. It contains global variables and functions available throughout the application.

Advanced Node.js Concepts

1. **How does clustering improve Node.js performance?**

   Clustering allows you to create multiple instances of your Node.js application (workers) to leverage multi-core systems, improving performance by distributing the load across CPUs.

2. **What is the purpose of the cluster module in Node.js?**

   The cluster module enables the creation of child processes (workers) that share the same server port, allowing Node.js to handle more requests efficiently.

3. **How do you handle memory leaks in Node.js?**

   Use tools like node-inspect or Chrome DevTools to identify memory leaks. Common fixes include clearing timers, closing database connections, and avoiding global variables.

4. **What is the EventEmitter class in Node.js?**

   The EventEmitter class is used to create and handle custom events in Node.js. It is the foundation of the event-driven architecture.

5. **What is the difference between spawn(), exec(), and fork() in the child_process module?**

   spawn(): Launches a new process and returns a stream.
   exec(): Runs a command in a shell and buffers the output.
   fork(): A special case of spawn() used to create child processes for Node.js modules.

6. **What is the purpose of the util module in Node.js?**

   The util module provides utility functions like promisify (to convert callback-based functions to promises) and inherits (for inheritance).

7. **How do you debug a Node.js application?**

   Use console.log, node-inspect, or debugging tools in IDEs like VS Code. For advanced debugging, use Chrome DevTools or ndb.

8. **What is the purpose of the fs module in Node.js?**

   The fs module provides APIs to interact with the file system, such as reading, writing, and deleting files.

9. **What is the difference between readFile and createReadStream in the fs module?**

   readFile: Reads the entire file into memory
   createReadStream: Reads the file in chunks, making it more memory-efficient for large files.

10. **What is the purpose of the path module in Node.js?**

    The path module provides utilities to work with file and directory paths, such as joining paths, resolving relative paths, and extracting file extensions.

Express.js and Middleware

1. **What is Express.js?**

   Express.js is a web application framework for Node.js that simplifies building APIs and web applications by providing routing, middleware, and other utilities.

2. **What is middleware in Express.js?**

   Middleware are functions that have access to the request (req), response (res), and the next middleware in the cycle. They are used for tasks like logging, authentication, and error handling.

3. **How do you handle errors in Express.js?**

   Use error-handling middleware with four arguments: (err, req, res, next). This middleware catches errors and sends appropriate responses.

4. **What is the purpose of app.use() in Express.js?**

   app.use() is used to mount middleware functions that are executed for every request to the application.

5. **What is the difference between app.get() and app.use() in Express.js?**

   app.get(): Handles HTTP GET requests for a specific route.
   app.use(): Mounts middleware for all HTTP methods or specific routes.

Performance and Security

1. **How do you secure a Node.js application?**

   Use HTTPS, validate user input, sanitize data, implement authentication (e.g., JWT), and regularly update dependencies to patch vulnerabilities.

2. **What is the purpose of the helmet middleware in Express.js?**

   Helmet helps secure Express apps by setting HTTP headers like X-Content-Type-Options and X-Frame-Options.

3. **What is the purpose of the cors middleware in Express.js?**

   The cors middleware enables Cross-Origin Resource Sharing (CORS), allowing or restricting requests from different domains.

4. **How do you optimize the performance of a Node.js application?**

   Use clustering, caching (e.g., Redis), load balancing, and efficient algorithms. Also, minimize blocking operations and optimize database queries.

5. **What is the purpose of the PM2 process manager?**

   PM2 is used to manage and monitor Node.js applications. It provides features like auto-restart, load balancing, and logging.