**Node Js Interview Questions**

1. What is Node.js and what is it commonly used for?
2. What is the difference between Node.js and JavaScript?
3. How do you install Node.js on your machine?
4. What is npm, and what is its purpose in the Node.js ecosystem?
5. Explain the event-driven, non-blocking I/O model of Node.js.
6. What is a Node.js module?
7. How do you include external modules in a Node.js application?
8. Explain the difference between require and import in Node.js.
9. How can you create your own Node.js module?
10. What is a callback function in Node.js, and why is it used?
11. How can you avoid callback hell or the "Pyramid of Doom" in Node.js?
12. Explain the concept of Promises and how they help manage asynchronous operations.
13. What is the async/await feature in Node.js, and how does it work?
14. How do you read a file asynchronously in Node.js?
15. How can you write data to a file in Node.js?
16. What is the purpose of the fs module in Node.js?
17. What is an EventEmitter in Node.js, and how does it work?
18. How can you create and use custom events in Node.js using EventEmitters?
19. What are Streams in Node.js, and when are they useful?
20. Explain the difference between readable and writable streams.
21. How can you pipe data from one stream to another in Node.js?
22. What is Express.js, and why is it popular in Node.js development?
23. How do you create a simple web server using Express.js?
24. What is middleware in Express.js, and why is it important?
25. What is a RESTful API, and how does Express.js support REST?
26. How do you handle routing and parameters in an Express.js application?
27. What HTTP methods are commonly used in RESTful APIs?
28. How do you handle errors in a Node.js application?
29. What is the difference between synchronous and asynchronous error handling?
30. What are some common security concerns in Node.js applications, and how can you address them?
31. How can you prevent common vulnerabilities like SQL injection and cross-site scripting (XSS) in a Node.js application?
32. What is clustering in Node.js, and how does it improve performance?
33. How can you scale a Node.js application to handle a large number of concurrent connections?
34. What tools can you use for debugging Node.js applications?
35. How can you perform unit testing in Node.js?
36. Explain the concept of the Node.js Event Loop.
37. What are child processes in Node.js, and when would you use them?
38. What are Worker Threads, and how do they differ from child processes?
39. How can you handle memory leaks in a long-running Node.js application?
40. What are some best practices for deploying and monitoring Node.js applications in production?
41. What is the purpose of middleware in Express.js, and how does the middleware execution flow work?
42. Can you explain the difference between app.use() and router.use() in Express.js?
43. How can you implement authentication using JSON Web Tokens (JWT) in an Express.js application?
44. What is role-based access control, and how can it be implemented in an Express.js application for authorization?
45. What are some popular databases used with Node.js, and how can you connect to and query these databases in a Node.js application?
46. Explain the use of an Object-Relational Mapping (ORM) like Sequelize or Mongoose in Node.js applications.
47. What are WebSockets, and how can you implement real-time communication in a Node.js application using libraries like Socket.io?
48. What are some common use cases for implementing WebSocket communication in Node.js applications?
49. What is the purpose of the package.json file, and how can you manage package versions and dependencies in a Node.js project?
50. How do you update packages in a Node.js project to the latest versions while maintaining version stability?