1. What is Express.js, and how does it differ from Node.js?
2. Explain the role of middleware in Express.js.
3. How do you handle errors in Express.js?
4. What is the difference between `app.use()` and `app.get()` in Express?
5. How can you structure an Express application for scalability?
6. Explain how routing works in Express.js.
7. How do you handle static files in Express.js?
8. What are the benefits of using Express.js with MongoDB?
9. How can you implement authentication in an Express app?
10. Describe how to create a RESTful API using Express.js.
11. How do you handle CORS in Express.js?
12. What are route parameters, and how do you use them in Express?
13. How can you implement pagination in Express?
14. How do you secure an Express application?
15. Explain the process of deploying an Express.js application.
16. How do you manage sessions in Express?
17. What is the purpose of `next()` in Express middleware?
18. How do you implement file uploads in an Express application?
19. Explain how to handle form data in Express.
20. What are the differences between `app.route()` and `app.all()`?
21. How can you optimize the performance of an Express.js application?
22. Describe the process of using a template engine with Express.
23. How do you manage environment variables in an Express.js application?
24. Explain how to implement logging in an Express app.
25. What are some common security vulnerabilities in Express apps?
26. How do you use Express.js with WebSockets?
27. Describe how to implement rate limiting in Express.
28. What is a reverse proxy, and how can you set it up with Express?
29. How do you handle errors in asynchronous code in Express.js?
30. Explain the concept of a middleware stack in Express.
31. How do you handle file downloads in Express?
32. What are the differences between `req.query`, `req.params`, and `req.body`?
33. How do you implement role-based access control in Express?
34. Explain the difference between synchronous and asynchronous middleware in Express.
35. How do you test an Express.js application?
36. Describe how to use Promises with Express.js.
37. How do you manage database connections in an Express app?

38. What is the role of `express.Router` in an application?
39. How can you implement a multi-language application using Express?
40. Explain how to handle file caching in Express.
41. What is the significance of HTTP status codes in an Express application?
42. How do you handle JSON Web Tokens (JWT) in Express?
43. What is the difference between a GET and POST request in Express?
44. How do you prevent SQL injection in an Express.js application?
45. Explain how to handle multipart/form-data in Express.
46. How do you configure HTTPS in an Express.js application?
47. Describe how to implement OAuth2 in an Express app.
48. How do you handle large file uploads in Express.js?
49. Explain the concept of middleware chaining in Express.
50. How do you handle nested routes in an Express application?
51. How do you implement social login in Express.js?
52. What are some best practices for organizing routes in an Express app?
53. How do you implement data validation in Express.js?
54. Explain the purpose of `res.locals` in Express.
55. How can you use Express.js with a front-end framework like React?
56. What is a middleware function, and how do you create one?
57. How do you implement CSRF protection in Express?
58. Explain how to set custom headers in an Express response.
59. How do you handle redirects in Express.js?
60. How do you use the `express-validator` library?
61. What is the purpose of `express.json()` and `express.urlencoded()`?
62. How do you handle database transactions in Express?
63. Explain the process of setting up a development environment for Express.
64. How do you implement email sending functionality in Express?
65. How can you integrate third-party APIs with Express.js?
66. What is the role of `process.env` in Express?
67. How do you manage large data sets in Express.js?
68. How do you implement a custom error handler in Express?
69. What is the role of `app.listen()` in an Express application?
70. How do you implement request throttling in Express.js?
71. Explain how to use middleware for authentication in Express.
72. How do you configure a 404 error page in Express?
73. How do you use the `morgan` logging library with Express?
74. How do you handle timeouts in Express.js?
75. Explain how to integrate a payment gateway in an Express application.

76. How do you handle WebSocket connections in an Express app?
77. What is the purpose of the `req.app` property in Express?
78. How do you implement input sanitization in Express?
79. Explain how to use the `cookie-parser` middleware in Express.
80. How do you implement internationalization (i18n) in Express?
81. How do you manage versioning in an Express API?
82. What are the benefits of using `express-async-handler`?
83. How do you implement JWT-based authentication in Express?
84. How do you set up a development and production environment for Express.js?
85. How do you optimize an Express application for SEO?
86. How do you handle file encryption in Express.js?
87. What are some common performance bottlenecks in Express applications?
88. How do you implement a REST API versioning strategy in Express?
89. What is the difference between PUT and PATCH requests in Express?
90. How do you handle user roles and permissions in Express.js?
91. How do you implement a caching mechanism in Express?
92. How do you manage API keys in an Express application?
93. How do you implement a custom response format in Express.js?
94. How do you handle cross-origin resource sharing (CORS) in Express?
95. What is the difference between a stream and a buffer in Node.js/Express?
96. How do you use the `compression` middleware in Express.js?
97. How do you handle nested JSON data in Express?
98. How do you implement a real-time chat application using Express?
99. How do you integrate Redis with Express.js for session management?
100. How do you handle file system operations in an Express.js application?