MERN Stack Interview Preparation

Tech Stack: **(MERN STACK)**

JavaScript, TypeScript, React, Redux Toolkit, Git, Bitbucket, React-hook-form, Next.js, HTML5, CSS3, Axios, CDN, Caching, React-query, Storybook, Unit Testing, Test coverage, MSW, Node.js, Express.js, GraphQL, RESTful Services, JWT Tokens, HTTP Status codes, HTTP Caching, HTTP cookies, HTTP Authentication, CORS, Request & Response Headers, MongoDB, MySQL, Web Performance, Web Optimization, Web Security, Accessibility, Responsive/Adaptive/Fluid Design etc.

HTML:

1. What is the use of DOCTYPE. What if we don't use it in our app?
2. Use of Meta tags
3. HTML5 tags with examples
4. Inline vs block elements
5. Semantic elements
6. <meta> tag (it's very important. Learn everything about this tag)
7. Async, defer
8. Pre-connect, prefetch, dns-prefetch and preload
9. SOE in HTML5
10. Local Storage vs session storage
11. Media tags
12. HTML5 forms
13. Button vs link
14. How to optimize website assets loading?
15. Web workers with examples
16. Service workers
17. Web APIs
18. Lazy loading on images
19. HTML elements vs tags
20. XSS vs CSRF attack
21. HTML cookie and how we can read /write it

CSS3:

1. Box sizing
2. Position (relative vs absolute)
3. Box model
4. Specificity
5. Media and Container queries
6. Centre a div horizontally and vertically
7. Remember precedence order inline style > external style > id > classes
8. Units like %, VH, VW
9. Rem vs em unit
10. What are the different types of Selectors in CSS?
11. What is the difference between inline, inline-block, and block?
12. Flex vs Grid
13. What are Pseudo elements and Pseudo classes?
14. What are the different ways to hide the element using CSS?
15. display:none vs visibility:hidden
16. nth-child() selector, :root selector
17. Inline vs block elements
18. What do the following CSS selectors mean? (Must learn below selectors)
    - div, p
    - div p
    - div ~ p
    - div + p

JavaScript:

1. Definition
2. Learn about JavaScript Engine (Chrome V8)
3. https://www.freecodecamp.org/news/execution-context-how-javascript-works-behind-the-scenes/
4. Prototypal Inheritance
5. Closure
6. Hoisting
7. Functional declaration vs function expression
8. ES6 vs ES5, explain ES6 features
9. Array methods like filter, reduce, find, map, forEach, sort etc.

10. Learn sort method on numbers and string how it is used to sort numbers and strings
11. String methods like split, join, indexOf, includes etc.
12. Functions(very imp) methods: call, apply, bind
13. Currying
14. __proto vs prototype property
15. Object.create()
16. Default parameters in functions
17. Prototype chain
18. Reverse a string
19. Find the intersection/union of two arrays
20. Two sum problem
21. Array flattening
22. Objects, Object.freeze(), Object.seal(), Object.keys(), Object.values()
23. Temporal dead zone
24. Difference among let, var and const
25. Debounce and throttling
26. Memoization
27. Web performance
28. Web optimization
29. Web Accessibility
30. Promises
31. Promise, Promise.all, Promise.race, Promise.any and Promise.allSetlled polyfills
32. Call, Bind and Apply polyfills
33. Map, Filter, Reduce and forEach polyfills
34. Callbacks, Callback Hell
35. Event Loop, Micro Queue, Macro Queue (https://velog.io/@jiseong/Microtask-Macrotask)
36. OOP (Object Oriented Programming)
37. ES6 features (Rest Operator, spread operator)
38. Shallow Copy and Deep copy
39. Object de-structuring, array de-structuring
40. Aysnc Await
41. Learn about function constructor vs classes
42. This keyword and how it behaves differently with function and objects
43. This keyword with arrow functions
44. Arrow functions
45. Event bubbling and event capturing
46. Event delegation

47. Sets, Maps
48. Garbage collection (mark and sweep, reference counting algorithms)
49. Stack and Heap (use of these in script execution)
50. Scope Chaining
51. Scoping like global scope, block scope and function scope (learn the order of these)
52. Types of JS errors (Reference/syntax/type)
53. Different ways to long poll a server request? =>Long Polling, Web Sockets, Server Sent Events
54. Design patterns:
    - Prototype Pattern
    - Module Pattern
    - Factory Pattern
    - Singleton Pattern
    - Builder Pattern
    - Proxy Pattern etc.

React:

1. What is React and how it is different from Angular and Vue?
2. What is virtual DOM and what how it differs from the real DOM ?
3. What is JSX? Does the browser understand it? (Hint: The browser doesn't understand the JSX. The browser simply makes use of createElement method in React to convert the JSX into plain JavaScript object and that' how the JS Engine understand and compile it)
4. Explain the react lifecycle ad its order (https://reactjs.org/docs/react-component.html#the-component-lifecycle)
5. Re-rendering in React how we can stop it. Hint (learn React.PureComponent, ShouldComponentUpdate method and React.memo)
6. Higher Order Components in React with example
7. Render props
8. State and props with example
9. Explain setState method in class components and how it updates the state?
   (Hint: setState takes a function which accepts the previousState and update the current state)

10. Data binding in React
11. What is the use of keys in React?
12. Synthetic events
13. What are hooks in React?
14. Fetch data from an API (https://jsonplaceholder.typicode.com/)
15. React. Fragment and <> </> let you render multiple components/elements in React
16. How can you pass data from child to parent component and vice versa?
17. useMemo vs useCallback with example
18. useState vs useReducer
19. Context API vs Redux. When to use either one
20. Controlled vs uncontrolled forms in React
21. Explain different types of hooks eg) useState, useEffect, useMemo, useCallback, useReducer, useRef etc.
22. What is useEffect and how it works?
23. Context API
24. What is Redux & Redux Toolkit?
25. Explain the Redux flow
26. Explain the terms in Redux like action types, reducer, store and middlewares
27. What is Redux Thunk and what is its use?
28. Can we directly update the state in Redux? (Hint: we can' t, we always make a copy of the state and update it)
29. useSelector and useDispatch (Redux hooks)
30. What is tree Shaking?
31. What is React.lazy and React.suspense?
32. What can we do to optimize a React application? (Hint: we can use optimization techniques such as React.PureComponents, useMemo, useCallback, React.Memo, tree shaking, React.lazy and React.suspense)
33. Can we use setState inside of render() method? (Hint: we can't , it'll lead the application to infinite loop)
34. Accessibility in React (https://reactjs.org/docs/accessibility.html#why-accessibility)
35. Web performance in React (https://reactjs.org/docs/optimizing-performance.html#gatsby-focus-wrapper)
36. Web optimization in React
37. Dynamic Form
38. Multi-Step Form
39. Form Validation (YUP Validation etc.)
40. React Testing Library & Jest, Cypress(Unit/Integration/E2E Testing)

41. Storybook
42. Rendering Techniques: SSR/CSR/SSG/ISR

Deployment:

1. Cloud AWS/GCP/Azure
2. Bitbucket
3. Jenkins
4. Ansible
5. Nagios
6. Grafana
7. Terraform
8. Docker
9. Kubernetes
10. Linux RHEL, NGINX etc.

Node.js:

1. What is Node.js?
2. How does Node.js work?
2. What is the difference between Node.js and JavaScript?
3. How does Node.js handle asynchronous programming?
4. What is an event loop in Node.js?
5. What is the purpose of the `require` function in Node.js?
6. How do you handle errors in Node.js?
7. What is the difference between `process.nextTick()` and `setImmediate()` in Node.js?
8. How do you handle file operations in Node.js?
9. What is the purpose of the `fs` module in Node.js?
10. How do you create a server in Node.js?
11. What is the purpose of the `http` module in Node.js?
12. How do you handle routing in Node.js?
13. What is middleware in Node.js?
14. How do you handle form data in Node.js?
15. What is the purpose of the `npm` package manager in Node.js?
16. How do you handle authentication and authorization in Node.js?
17. What is the purpose of the `express` framework in Node.js?
18. How do you handle database operations in Node.js?
19. What is the purpose of the `mongoose` library in Node.js?
20. How do you handle sessions and cookies in Node.js?
21. What is the purpose of the `socket.io` library in Node.js?
22. How do you handle real-time communication in Node.js?

23. What is clustering in Node.js?
24. How do you handle child processes in Node.js?
25. What is the purpose of the `cluster` module in Node.js?
26. How do you handle caching in Node.js?
27. What is the purpose of the `redis` database in Node.js?
28. How do you handle testing in Node.js?
29. What is the purpose of the `mocha` framework in Node.js?
30. How do you handle logging in Node.js?
31. What is the purpose of the `winston` library in Node.js?
32. How do you handle security in Node.js?
33. What is the purpose of the `helmet` library in Node.js?
34. How do you handle performance optimization in Node.js?
35. What is the purpose of the `pm2` process manager in Node.js?
36. How do you handle deployment in Node.js?
37. What is the purpose of the `Docker` containerization platform in Node.js?
38. How do you handle scaling in Node.js?
39. What is the purpose of the `load balancing` in Node.js?
40. How do you handle logging in Node.js?
41. What is the purpose of the `winston` library in Node.js?
42. How do you handle security in Node.js?
43. What is the purpose of the `helmet` library in Node.js?
44. How do you handle performance optimization in Node.js?
45. What is the purpose of the `pm2` process manager in Node.js?
46. How do you handle deployment in Node.js?
47. What is the purpose of the `Docker` containerization platform in Node.js?
48. How do you handle scaling in Node.js?
49. What is the purpose of the `load balancing` in Node.js?
50. How do you handle logging in Node.js?

Express.js:

1. What is Express.js?
2. What is middleware in Express.js?
3. How do you create a server in Express.js?
4. What is routing in Express.js?
5. How do you handle HTTP requests in Express.js?
6. What is the difference between app.get() and app.post() in Express.js?
7. How do you handle form data in Express.js?
8. What is the purpose of the body-parser middleware in Express.js?

9. How do you handle static files in Express.js?
10. What is the purpose of the express.static() middleware in Express.js?
11. How do you handle cookies and sessions in Express.js?
12. What is the purpose of the cookie-parser middleware in Express.js?
13. What is the purpose of the express-session middleware in Express.js?
14. How do you handle authentication and authorization in Express.js?
15. What is the purpose of the passport middleware in Express.js?
16. How do you handle database operations in Express.js?
17. What is the purpose of the mongoose library in Express.js?
18. How do you handle error handling in Express.js?
19. What is the purpose of the error-handler middleware in Express.js?
20. How do you handle logging in Express.js?
21. What is the purpose of the morgan middleware in Express.js?
22. How do you handle testing in Express.js?
23. What is the purpose of the supertest library in Express.js?
24. How do you handle security in Express.js?
25. What is the purpose of the helmet middleware in Express.js?
26. How do you handle performance optimization in Express.js?
27. What is the purpose of the compression middleware in Express.js?
28. How do you handle deployment in Express.js?
29. What is the purpose of the pm2 process manager in Express.js?
30. How do you handle scaling in Express.js?
31. What is the purpose of the load balancing in Express.js?
32. How do you handle logging in Express.js?
33. What is the purpose of the morgan middleware in Express.js?
34. How do you handle security in Express.js?
35. What is the purpose of the helmet middleware in Express.js?
36. How do you handle performance optimization in Express.js?
37. What is the purpose of the compression middleware in Express.js?
38. How do you handle deployment in Express.js?
39. What is the purpose of the pm2 process manager in Express.js?
40. How do you handle scaling in Express.js?

MySQL:

1. What is MySQL?
2. What is the difference between MySQL and SQL?
3. How do you create a database in MySQL?
4. How do you create a table in MySQL?
5. What are the different data types in MySQL?
6. How do you insert data into a table in MySQL?

7. How do you update data in a table in MySQL?
8. How do you delete data from a table in MySQL?
9. What is the difference between DELETE and TRUNCATE in MySQL?
10. How do you retrieve data from a table in MySQL?
11. What is the difference between SELECT and SELECT DISTINCT in MySQL?
12. How do you join tables in MySQL?
13. What are the different types of joins in MySQL?
14. How do you use GROUP BY and HAVING clauses in MySQL?
15. How do you use ORDER BY clause in MySQL?
16. What is the difference between INNER JOIN and OUTER JOIN in MySQL?
17. How do you create indexes in MySQL?
18. What is the purpose of indexes in MySQL?
19. How do you handle NULL values in MySQL?
20. What is the difference between CHAR and VARCHAR data types in MySQL?
21. How do you create a stored procedure in MySQL?
22. How do you call a stored procedure in MySQL?
23. What is the difference between a primary key and a unique key in MySQL?
24. How do you create a foreign key in MySQL?
25. What is the purpose of foreign keys in MySQL?
26. How do you perform transactions in MySQL?
27. What is the purpose of transactions in MySQL?
28. How do you backup and restore a MySQL database?
29. What is the difference between MyISAM and InnoDB storage engines in MySQL?
30. How do you optimize MySQL queries?
31. What is the purpose of the EXPLAIN statement in MySQL?
32. How do you handle database security in MySQL?
33. What is the purpose of user privileges in MySQL?
34. How do you handle database replication in MySQL?
35. What is the purpose of database replication in MySQL?
36. How do you handle database performance tuning in MySQL?
37. What is the purpose of database performance tuning in MySQL?
38. How do you handle database backup and recovery in MySQL?
39. What is the purpose of database backup and recovery in MySQL?
40. How do you handle database indexing in MySQL?

MongoDB:

1. What is MongoDB?
2. What is the difference between MongoDB and SQL databases?
3. What is a document in MongoDB?
4. What is a collection in MongoDB?
5. What is a database in MongoDB?
6. How do you create a database in MongoDB?
7. How do you create a collection in MongoDB?
8. How do you insert a document in MongoDB?
9. How do you update a document in MongoDB?
10. How do you delete a document in MongoDB?
11. What is the difference between update and updateMany in MongoDB?
12. How do you retrieve data from a collection in MongoDB?
13. What is the difference between find and findOne in MongoDB?
14. How do you use the $in operator in MongoDB?
15. How do you use the $or operator in MongoDB?
16. How do you use the $and operator in MongoDB?
17. How do you use the $regex operator in MongoDB?
18. How do you use the $sort operator in MongoDB?
19. How do you use the $limit operator in MongoDB?
20. How do you use the $skip operator in MongoDB?
21. How do you use the $aggregate operator in MongoDB?
22. What is the purpose of indexes in MongoDB?
23. How do you create an index in MongoDB?
24. What is the difference between a single field index and a compound index in MongoDB?
25. How do you handle NULL values in MongoDB?
26. What is the purpose of the $exists operator in MongoDB?
27. How do you handle database security in MongoDB?
28. What is the purpose of user roles in MongoDB?
29. How do you handle database replication in MongoDB?
30. What is the purpose of database replication in MongoDB?
31. How do you handle database sharding in MongoDB?
32. What is the purpose of database sharding in MongoDB?
33. How do you handle database backup and recovery in MongoDB?
34. What is the purpose of database backup and recovery in MongoDB?
35. How do you handle database performance tuning in MongoDB?
36. What is the purpose of database performance tuning in MongoDB?

37. How do you handle database indexing in MongoDB?
38. What is the purpose of database indexing in MongoDB?
39. How do you handle database aggregation in MongoDB?
40. What is the purpose of database aggregation in MongoDB?