MERN STACK INTERVIEW QUESTION AND ANSWER:

1. **What is the MERN stack?**
   - MERN stack is a collection of technologies used for building web applications, consisting of MongoDB, Express.js, React.js, and Node.js.

2. **Explain each component of MERN.**
   - **MongoDB:** A NoSQL database.
   - **Express.js:** A web application framework for Node.js.
   - **React.js:** A JavaScript library for building user interfaces.
   - **Node.js:** A runtime environment for executing JavaScript on the server side.

3. **What is MongoDB?**
   - MongoDB is a NoSQL database that stores data in flexible, JSON-like documents.

4. **What is the difference between SQL-based and NoSQL-based databases?**
   - **SQL-based databases** are relational, structured, and use tables.
   - **NoSQL-based databases** are non-relational, flexible, and use documents, key-value pairs, graphs, or wide-columns.

5. **What are the advantages of MongoDB?**
   - Flexible schema, high performance, scalability, horizontal scaling, and JSON-like storage format.

6. **What is the difference between Node.js and React.js?**
   - **Node.js:** Server-side runtime environment.
   - **React.js:** Client-side library for building user interfaces.

7. **What is the difference between Express.js and Node.js?**
   - **Node.js:** Runtime environment for executing JavaScript on the server side.
   - **Express.js:** A framework built on top of Node.js to simplify server-side coding.

8. **Why is Node.js called a single-threaded environment?**
    - Node.js uses a single-threaded event loop to handle multiple concurrent requests.

9. **What is asynchronous programming in Node.js?**
    - A programming paradigm that allows non-blocking operations, enabling other processes to continue while waiting for an operation to complete.

10. **What is a promise in JavaScript?**
    - An object representing the eventual completion or failure of an asynchronous operation.

11. **What is async/await in JavaScript?**
    - Syntactic sugar built on promises to write asynchronous code in a synchronous manner.

12. **What is the React.js component lifecycle?**
    - The sequence of events (mounting, updating, unmounting) that a component goes through during its existence.

13. **What is the difference between functional components and class components in React.js?**
    - **Functional components:** Stateless, simpler, use hooks.
    - **Class components:** Stateful, more complex, use lifecycle methods.

14. **What are controlled components in React.js?**
    - Components where the form data is handled by the state within the component.

15. **What are uncontrolled components in React.js?**
    - Components where the form data is handled by the DOM itself.

16. **What are props in React.js?**
    - Short for properties, props are read-only inputs passed to components.

17. **What is state in React.js?**
    - An object that holds dynamic data and determines the component's behavior.

18. **What is the difference between props and state in React.js?**
    - **Props:** Read-only, passed from parent to child.
    - **State:** Mutable, managed within the component.

19. **What is the context API in React.js?**
    - A way to pass data through the component tree without manually passing props down at every level.

20. **What are higher-order components (HOCs) in React.js?**
    - Functions that take a component and return a new component, enhancing it with additional behavior or data.

21. **What are hooks in React.js?**
    - Functions that let you use state and other React features in functional components.

22. **What is the difference between useEffect and useLayoutEffect in React.js?**
    - **useEffect:** Runs asynchronously after render.
    - **useLayoutEffect:** Runs synchronously after render but before the DOM updates.

23. **What is an API?**
    - Application Programming Interface, a set of rules that allow different software entities to communicate.

24. **What is REST API?**

- Representational State Transfer API, an architectural style for designing networked applications.

25. **What is the full form of REST?**
    - Representational State Transfer.

26. **What is the difference between REST API and SOAP API?**
    - **REST API:** Uses HTTP, more flexible and simpler.
    - **SOAP API:** Protocol-based, more rigid, uses XML.

27. **How can you get data from an API?**
    - By making HTTP requests using fetch or Axios in JavaScript.

28. **How can you manage JSON data?**
    - Convert JSON to JavaScript objects using `JSON.parse()` and objects to JSON using `JSON.stringify()`.

29. **How do you convert JSON to an object and vice versa?**
    - **JSON to object:** `JSON.parse()`
    - **Object to JSON:** `JSON.stringify()`

30. **What is the fetch approach to call an API?**
    - A modern JavaScript method for making network requests and handling responses.

31. **What is Axios?**
    - A promise-based HTTP client for making API requests in JavaScript.

32. **What is the difference between fetch and Axios?**
    - **Fetch:** Built-in, less intuitive error handling.

  - **Axios:** Third-party library, easier to use, better error handling.

33. **What is the difference between PATCH and PUT methods in HTTP?**
    - **PATCH:** Partially updates a resource.
    - **PUT:** Fully updates or creates a resource.

34. **How many types of HTTP methods are there?**
    - Common types include GET, POST, PUT, PATCH, DELETE, OPTIONS, and HEAD.

35. **What is a status code in HTTP?**
    - A code sent by the server to indicate the result of the client's request.

36. **What is the meaning of 1xx, 2xx, 3xx, 4xx, and 5xx status codes?**
    - **1xx:** Informational responses.
    - **2xx:** Success.
    - **3xx:** Redirection.
    - **4xx:** Client errors.
    - **5xx:** Server errors.

37. **What is the difference between Node.js and React.js?**
    - **Node.js:** Server-side runtime.
    - **React.js:** Client-side library.

38. **What is the difference between Express.js and Node.js?**
    - **Node.js:** Runtime environment.
    - **Express.js:** Framework built on Node.js for server-side applications.

39. **What is the role of async/await in Node.js programming?**
    - Simplifies writing asynchronous code, making it look synchronous and easier to read.

40. **How to compile and execute Node.js program code?**
    - Using the `node` command in the terminal, e.g., `node app.js`.

41. **How does Node.js work internally?**
    - It uses the V8 JavaScript engine and an event-driven, non-blocking I/O model.

42. **Which compiler is built-in under the Node.js environment?**
    - The V8 JavaScript engine compiler.

43. **What is the React context API?**
    - A way to manage global state in React applications without prop drilling.

44. **What are hooks in React?**
    - Functions that let you use state and lifecycle features in functional components.

45. **Name five hooks methods you should know.**
    - `useState`, `useEffect`, `useContext`, `useReducer`, `useRef`.

46. **What is the difference between useEffect and useLayoutEffect?**
    - **useEffect:** Runs after rendering.
    - **useLayoutEffect:** Runs before the browser paints the screen.

47. **What is the difference between functional components and class components in React?**
    - **Functional components:** Stateless, use hooks.
    - **Class components:** Stateful, use lifecycle methods.

48. **What are the advantages of using Node.js?**

- Non-blocking I/O, single-threaded, event-driven, scalable, and uses JavaScript.

49. **Name five libraries of Node.js.**
    - Express, Mongoose, Lodash, Async, and Moment.

50. **What is the fetch approach to call an API?**
    - Using the fetch function to make HTTP requests and handle responses.

51. **What is Axios?**
    - A promise-based HTTP client for making API requests.

52. **What is the difference between fetch and Axios?**
    - **Fetch:** Built-in, less intuitive error handling.
    - **Axios:** Third-party library, more intuitive error handling.

53. **What is the difference between PATCH and PUT methods in HTTP?**
    - **PATCH:** Partially updates a resource.
    - **PUT:** Fully updates or creates a resource.

54. **How many types of HTTP methods are there?**
    - Common methods include GET, POST, PUT, PATCH, DELETE, OPTIONS, and HEAD.

55. **What is a status code in HTTP?**
    - A code indicating the result of the client's request.

56. **What is the meaning of 1xx, 2xx, 3xx, 4xx, and 5xx status codes?**
    - **1xx:** Informational responses.
    - **2xx:** Success.
    - **3xx:** Redirection.

- **4xx:** Client errors.
- **5xx:** Server errors.