Zomato- Frontend Developer

Interview Process

Round 1: Telephonic Round
Round 2: Assignment
Round 3: Coding Round
Round 4: HR Round

Interview Questions

1. Tell me about yourself.
2. Tell me about your projects.
3. What is REST?
4. HTTP, its Versions, and Methods, (OPTIONS method as separate question)
5. Can we post or update data using GET, if yes then how?
6. Bundler, what it does with some example?
7. Server Side Rendering & Hydration
8. Static Site Generation
9. Incremental Site Generation
10. HTTP vs HTTPS, why HTTPS is better
11. Load Balancer
12. What's NextJS, and its benefits?
13. Vite & why it is better than Webpack?
14. Framework vs Library, with Examples
15. Can (a==1 && a==2 && a==3) ever evaluate to true
16. If c!=c what could be the type of c.
17. 3 way handshaking, how to handle overhead if more requests
18. Tailwind CSS, why better than CSS
19. What is CORS, without cors package
20. Virtualization in React
21. Redux and how it's different with Context and Props
22. === vs ==
23. return from forEach
24. JS is single threaded or multi, and how to create multi thread on JS
25. Web Workers
26. Service Worker
27. !! operator
28. Temporal Dead Zone
29. What tags do you have in React?
30. null vs undefined
31. Maximum Subarray Sum
32. Optimizing performance of React Component (Avoid re-renders)

Solutions

Q1: What is REST?
A1: REST (Representational State Transfer) is an architectural style for designing networked applications. It relies on stateless, client-server communication, typically over HTTP, and uses standard HTTP methods like GET, POST, PUT, DELETE to perform CRUD operations on resources, which are represented by URLs.

Q2: HTTP, its Versions, and Methods.
A2: HTTP (HyperText Transfer Protocol) is the foundation of data communication on the web. The most common versions are HTTP/1.1, HTTP/2, and HTTP/3, each introducing improvements in performance and security. HTTP methods include GET (retrieve data), POST (send data), PUT (update data), DELETE (remove data), PATCH (partially update data), HEAD (retrieve headers), and OPTIONS (describe communication options).

Q3: What is the OPTIONS method in HTTP?
A3: The OPTIONS method in HTTP is used to describe the communication options for the target resource. It is often used in CORS (Cross-Origin Resource Sharing) to determine which HTTP methods and headers are allowed when accessing a resource from a different origin.

Q4: Can we post or update data using GET, if yes then how?
A4: Although the HTTP GET method is designed for data retrieval, it is technically possible to send data in the query string of a GET request, thereby "updating" data on the server if the server is designed to handle such requests. However, this is not recommended because GET requests should be idempotent and not alter server state.

Q5: Bundler, what it does with some example?
A5: A bundler is a tool that compiles multiple JavaScript files, along with their dependencies (like CSS, images, etc.), into a single or few optimized files (bundles) for use in production. For example, Webpack is a popular bundler that takes your JS, CSS, and other files and creates a bundle that can be served to users efficiently.

Q6: Server Side Rendering & Hydration
A6: Server-Side Rendering (SSR) involves rendering a web page on the server and sending the fully rendered HTML to the client. Hydration is the process where JavaScript on the client side

takes over this server-rendered HTML and makes it interactive by attaching event listeners and converting it into a dynamic React component.

Q7: Static Site Generation
A7: Static Site Generation (SSG) pre-renders pages at build time, creating static HTML files for each page. This results in faster load times and improved performance since the content is served as static files, without requiring server-side rendering on each request.

Q8: Incremental Site Generation
A8: Incremental Site Generation (ISG) extends Static Site Generation by allowing you to build or rebuild specific pages incrementally as needed, instead of generating all pages at once during the build process. This is useful for large sites where content updates frequently.

Q9: HTTP vs HTTPS, why HTTPS is better
A9: HTTP is the basic protocol for web communication, but it transmits data in plain text, making it vulnerable to interception. HTTPS (HTTP Secure) encrypts the data using SSL/TLS, protecting it from eavesdropping and ensuring data integrity, which makes HTTPS a more secure choice.

Q10: Load Balancer
A10: A load balancer is a device or software that distributes incoming network traffic across multiple servers to ensure no single server becomes overwhelmed, thereby improving application availability and reliability. It can also improve scalability and fault tolerance.

Q11: What's NextJS, and its benefits?
A11: Next.js is a React framework that provides features like server-side rendering, static site generation, and automatic code splitting. It simplifies React application development by offering built-in tools for routing, data fetching, and performance optimization, making it a popular choice for building modern web applications.

Q12: Vite & why it is better than Webpack?
A12: Vite is a build tool that offers fast development by leveraging native ES modules in the browser and using pre-bundling to optimize dependencies. It is considered better than Webpack

for development because of its significantly faster hot module replacement (HMR) and build times, thanks to its lightweight and modern architecture.

Q13: Framework vs Library, with Examples
A13: A library is a collection of pre-written code that developers can call upon when needed, like React for building UI components. A framework provides a complete structure or skeleton for building applications, enforcing a certain way of doing things, like Angular or Django. Libraries offer more flexibility, while frameworks provide a more opinionated, comprehensive approach.

Q14: Can (a==1 && a==2 && a==3) ever evaluate to true?
A14: Yes, this expression can evaluate to true if the value of `a` is an object that defines custom logic in its `toString` or `valueOf` method to return different values each time it's compared. For example, using a getter that increments on each access.

Q15: If c!=c what could be the type of c.
A15: If `c != c`, then `c` is `NaN` (Not a Number), because `NaN` is the only value in JavaScript that is not equal to itself.

Q16: 3 way handshaking, how to handle overhead if more requests
A16: The three-way handshake is a process used in TCP/IP networks to establish a connection between a client and server. It involves SYN, SYN-ACK, and ACK packets. To handle overhead with more requests, techniques like connection pooling, keep-alive, or using UDP for non-critical data can reduce the number of handshakes.

Q17: Tailwind CSS, why better than CSS
A17: Tailwind CSS is a utility-first CSS framework that allows developers to apply styles directly in HTML using predefined classes, resulting in faster styling and more maintainable code. It eliminates the need for custom CSS, reduces context switching, and makes it easier to manage complex designs compared to traditional CSS.

Q18: What is CORS, without cors package
A18: CORS (Cross-Origin Resource Sharing) is a security feature implemented by browsers to restrict web pages from making requests to a different domain than the one that served the web

page. It is handled by the server through specific headers like `Access-Control-Allow-Origin` to grant permission for cross-origin requests without needing additional packages.

Q19: Virtualization in React
A19: Virtualization in React refers to rendering only the visible portion of a large list or grid, rather than the entire set of items. Libraries like `react-window` or `react-virtualized` implement this technique, significantly improving performance by reducing the number of DOM elements rendered at once.

Q20: Redux and how it's different with Context and Props
A20: Redux is a state management library that centralizes the application's state in a single store, enabling predictable state changes and easier debugging. In contrast, React's Context API and Props allow passing data through the component tree, but they lack Redux's centralized control and can become unwieldy for large-scale applications.

Q21: === vs ==
A21: The `===` operator in JavaScript checks for strict equality, meaning both value and type must be the same, whereas `==` checks for loose equality, performing type coercion if the types are different. It is generally safer to use `===` to avoid unexpected type conversions.

Q22: return from forEach
A22: The `forEach` method in JavaScript does not return a value and cannot be exited early using `return`. If you need to break out of a loop or return a value, use a `for` loop or the `Array.prototype.every` or `Array.prototype.some` methods instead.

Q23: JS is single threaded or multi, and how to create multi thread on JS
A23: JavaScript is single-threaded, meaning it executes one task at a time in the event loop. However, you can create multithreading-like behavior using Web Workers, which run scripts in the background on separate threads, allowing for parallel execution without blocking the main thread.

Q24: Web Workers
A24: Web Workers are a feature in JavaScript that allow you to run scripts in the background, independent of the main thread. This enables you to perform heavy computations or handle long-running tasks without freezing the user interface, improving application performance and responsiveness.

Q25: Service Worker
A25: A Service Worker is a script that runs in the background of your browser, separate from the web page, enabling features like offline caching, background sync, and push notifications. It acts as a network proxy, intercepting network requests and serving cached content if the network is unavailable.

Q26: !! operator
A26: The `!!` operator in JavaScript is a double negation that converts a value to its boolean equivalent. It ensures that a truthy value becomes `true` and a falsy value becomes `false`. It is often used to explicitly cast a value to a boolean.

Q27: Temporal Dead Zone
A27: The Temporal Dead Zone (TDZ) in JavaScript refers to the period between the entering of a block and the initialization of a `let` or `const` variable within that block. Accessing the variable before initialization results in a `ReferenceError`.

Q28: What tags do you have in React?
A28: In React, the most common tags are JSX tags which are similar to HTML tags, like `<div>`, `<span>`, and `<input>`. Additionally, React components can be used as custom tags (e.g., `<MyComponent />`). Special React tags include fragments (`<>` or `<React.Fragment>`), and portals (`ReactDOM.createPortal`).

Q29: null vs undefined
A29: `null` is an assignment value representing the intentional absence of any object value, while `undefined` means a variable has been declared but not yet assigned a value. `undefined` is typically used by JavaScript itself, whereas `null` is used by programmers to indicate "no value."

Q30: Maximum Subarray Sum
A30: The maximum subarray sum problem can be solved using Kadane's algorithm, which runs in O(n) time. The algorithm iteratively calculates the maximum sum subarray ending at each position, keeping track of the global maximum sum found so far.

Q31: Optimizing performance of React Component (Avoid re-renders)
A31: To optimize React component performance and avoid unnecessary re-renders, use `React.memo` to memoize functional components, `useCallback` to memoize functions, and `useMemo` to memoize computed values. Also, ensure that components only re-render when necessary by implementing `shouldComponentUpdate` in class components or using the `useEffect` hook wisely in functional components.