# Google - Frontend Developer

## Interview Process

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

## Interview Questions

1. Differentiate between MySQL and MongoDB.
2. How can we remove duplicate entries from a JavaScript Array?
3. What are some ways to decrease page load time?
4. Talk about the Grid system in CSS.
5. What is stringify?
6. Explain what internal and im Frontend happens when a user clicks on a link in a web browser.
7. Explain the JavaScript module model. When would you use it?
8. If you have text on a web page, how many shapes do you know? Make the text disappear?
9. How can I get a response regarding updating the webpage?
10. Define and describe the need and use of the cap.
11. Define and describe when you will use the JavaScript module template.
12. Name and talk about some version control systems you know.
13. Explain the improvement.
14. Explains some techniques for improving performance when creating or maintaining a new website.
15. How to design a task scheduling system?
16. How will you style Google Docs?
17. Given Find with a string A and a string B the minimal window of A containing all the characters of B in time complexity O(n).
18. Given a row x column binary matrix filled with 0s and 1s, would you find the largest rectangle with only 1 and return its area?
19. How would you reverse a linked list?
20. Yes, I will give you an encoded string. Write the code to return your decoded string.
21. How can I implement a SnapshotArray that also supports predefined interfaces?
22. How to find the width of a reverse binary tree?
23. How do I know if two binary trees are the same?
24. Find the maximum sum of paths in a given binary tree, where the path can start and end at any node.
25. How do you make a function that takes a callback function fn and returns a function that calls fn on a timeout?

26. Implement the outline view for a Google doc.
27. DFS on HTML nodes.
28. Implement throttle.
29. How do you make a function that only calls input function f every 50 milliseconds?
30. Given a timeline write the JavaScript to select all nodes within selection of timeline
31. Design a slider component.
32. Design a Tic-Tac-Toe game/design an algorithm for Tic-Tac-Toe game.
33. Implement nested checkboxes (when the parent is checked, children are checked and vice versa. Use <input type="checkbox">). Similar to Indeterminate checkboxes.
34. Design a webpage which can auto load new posts when you reach the bottom of the page by using JavaScript. You may use AJAX and JavaScript event listeners.
35. Write a UI using HTML, CSS, JavaScript that allows users to enter the number of rows and columns in text input fields within a form and renders a table.
36. Explain the CSS Box Model.
37. What happens when you type a URL into the browser and hit enter?
38. Given some text on a web page, how many ways can you make the text disappear?
39. How do you send data from a web page to a server without a page refresh?
40. Design emoji autocomplete.
41. Design JS Bin.
42. How would you create a Google Analytics SDK used by webpages?

**Solutions**

**Q1: Differentiate between MySQL and MongoDB.**
**A1:** MySQL is a relational database management system (RDBMS) that uses structured query language (SQL) for managing structured data with predefined schemas. MongoDB is a NoSQL database that stores data in flexible, JSON-like documents, making it more suited for unstructured or semi-structured data. MySQL enforces relationships between tables, while MongoDB is schema-less, offering greater flexibility in data storage and retrieval.

**Q2: How can we remove duplicate entries from a JavaScript Array?**
**A2:** Duplicates can be removed from a JavaScript array by converting it to a `Set`, which inherently ensures unique values. For example:

```
let uniqueArray = [...new Set(array)];
```

This method creates a new array with only unique elements.

**Q3: What are some ways to decrease page load time?**
**A3:** To decrease page load time, you can optimize images, minify CSS and JavaScript files, leverage browser caching, use Content Delivery Networks (CDNs), defer or asynchronously load JavaScript, and reduce the number of HTTP requests by combining files.

**Q4: Talk about the Grid system in CSS.**
**A4:** The CSS Grid system is a powerful layout system that allows you to design complex web layouts using a two-dimensional grid-based structure. It divides the page into rows and columns, making it easier to control the placement and alignment of items within a grid container. CSS Grid is highly flexible, enabling responsive design without relying heavily on floats or positioning.

**Q5: What is stringify?**
**A5:** `JSON.stringify()` is a method in JavaScript that converts a JavaScript object or value to a JSON string. It is commonly used to send data over the network or to store it in a string-based format, such as local storage.

**Q6: Explain what internal and in Frontend happens when a user clicks on a link in a web browser.**
**A6:** When a user clicks on a link, the browser sends an HTTP request to the server for the resource specified by the URL. Internally, the browser performs a DNS lookup, establishes a TCP connection, and sends the request. The server processes the request and returns the appropriate response, which the browser then renders, updating the frontend by either reloading the page or modifying the DOM based on the response.

**Q7: Explain the JavaScript module model. When would you use it?**
**A7:** The JavaScript module model allows developers to split code into reusable modules, each with its own scope. Modules can export functions, objects, or primitives and be imported into other files as needed. This model is useful for organizing code, avoiding global namespace pollution, and making maintenance easier in large projects.

**Q8: If you have text on a web page, how many shapes do you know? Make the text disappear.**
**A8:** Text on a webpage can be manipulated into various shapes using CSS properties like `transform`, `clip-path`, or using SVG. To make the text disappear, you can use

`visibility: hidden;`, `display: none;`, or `opacity: 0;`, depending on whether you want to hide it visually or remove it from the document flow.

**Q9: How can I get a response regarding updating the webpage?**
**A9:** You can use JavaScript to get a response and update the webpage using AJAX (Asynchronous JavaScript and XML). This allows you to send HTTP requests to the server and update parts of a webpage without reloading the entire page.

**Q10: Define and describe the need and use of the cap.**
**A10:** In programming and computing, a "cap" generally refers to a limit or maximum value that can be assigned to a variable or resource. It is used to prevent resource exhaustion, maintain system stability, and ensure fair distribution of resources.

**Q11: Define and describe when you will use the JavaScript module template.**
**A11:** The JavaScript module template is used when you want to create a blueprint for organizing your code into self-contained, reusable units. You would use it when building large-scale applications to keep the code modular, maintainable, and easy to test.

**Q12: Name and talk about some version control systems you know.**
**A12:** Common version control systems include Git, SVN (Subversion), and Mercurial. Git is the most widely used, known for its distributed model that allows multiple developers to work on a project simultaneously. SVN is a centralized version control system, suitable for smaller teams, while Mercurial is similar to Git but designed to be easier to use.

**Q13: Explain the improvement.**
**A13:** Improvement in the context of development generally refers to the process of optimizing code, enhancing performance, fixing bugs, or refactoring existing code to make it more efficient and maintainable. Continuous improvement is a key aspect of agile methodologies.

**Q14: Explain some techniques for improving performance when creating or maintaining a new website.**
**A14:** Techniques for improving website performance include optimizing images, minifying CSS and JavaScript, using lazy loading for images and scripts, leveraging browser caching, reducing

server response time, and using a Content Delivery Network (CDN) to distribute content more efficiently.

**Q15: How to design a task scheduling system?**
**A15:** Designing a task scheduling system involves defining tasks, assigning priorities, and setting execution times. The system should manage dependencies between tasks, handle concurrency, and ensure efficient resource allocation. A common approach is using a priority queue or time-based scheduling algorithm like round-robin or priority scheduling.

**Q16: How will you style Google Docs?**
**A16:** Styling Google Docs involves creating a user-friendly and responsive interface, utilizing CSS for layout, typography, and visual elements. You would apply consistent themes, use CSS Grid or Flexbox for layout, and ensure accessibility by following best practices for contrast, font size, and navigation.

**Q17: Given Find with a string A and a string B the minimal window of A containing all the characters of B in time complexity O(n).**
**A17:** To find the minimal window in string A containing all characters of B in O(n) time, you can use a sliding window technique combined with a frequency map to track the characters in B. Expand the window by including characters from A until all characters of B are covered, then shrink the window from the left to find the smallest valid window.

**Q18: Given a row x column binary matrix filled with 0s and 1s, would you find the largest rectangle with only 1 and return its area?**
**A18:** To find the largest rectangle of 1s in a binary matrix, you can treat each row as the base of a histogram and use a stack to find the maximum area in each histogram. Iterate through each row, updating the histogram heights, and calculate the largest rectangle for each histogram.

**Q19: How would you reverse a linked list?**
**A19:** To reverse a linked list, you can iterate through the list and reverse the direction of each pointer. Initialize three pointers: `prev`, `current`, and `next`. For each node, update its `next` pointer to `prev`, then move `prev` and `current` one step forward until the entire list is reversed.

**Q20: Yes, I will give you an encoded string. Write the code to return your decoded string.**
**A20:** The code to decode a string depends on the encoding algorithm used. For a simple example, if the string is Base64 encoded, you can decode it using:

```
let decodedString = atob(encodedString);
```

Replace `encodedString` with the actual string you want to decode.

**Q21: How can I implement a SnapshotArray that also supports predefined interfaces?**
**A21:** A SnapshotArray can be implemented by maintaining a list of snapshots, where each snapshot stores the state of the array at a specific time. You would define methods for setting values, taking snapshots, and retrieving values based on a snapshot ID. Use a dictionary or map to store only the changes between snapshots to optimize memory usage.

**Q22: How to find the width of a reverse binary tree?**
**A22:** The width of a binary tree is defined as the maximum width among all levels. For a reversed binary tree, the process remains the same. Perform a level-order traversal using a queue and calculate the number of nodes at each level to determine the maximum width.

**Q23: How do I know if two binary trees are the same?**
**A23:** To check if two binary trees are the same, you can perform a recursive traversal of both trees simultaneously. At each step, check if the current nodes have the same value and if their left and right subtrees are also identical. If all corresponding nodes match, the trees are the same.

**Q24: Find the maximum sum of paths in a given binary tree, where the path can start and end at any node.**
**A24:** To find the maximum sum of paths in a binary tree, you can use a recursive approach where for each node, you calculate the maximum path sum including that node, and keep track of the global maximum. The path sum can either continue through the parent or terminate at the current node.

**Q25: How do you make a function that takes a callback function `fn` and returns a function that calls `fn` on a timeout?**
**A25:** You can create a function that returns a new function with a set timeout like this:

```
function withTimeout(fn, delay) {
    return function(...args) {
        setTimeout(() => fn(...args), delay);
    };
}
```

This function will execute `fn` after the specified delay when invoked.

**Q26: Implement the outline view for a Google doc.**
**A26:** The outline view in a Google Doc can be implemented using a sidebar that lists headings and subheadings in a nested manner. JavaScript can be used to extract headings from the document, dynamically generate the outline structure, and allow users to click on an entry to jump to the corresponding section in the document.

**Q27: DFS on HTML nodes.**
**A27:** To perform a Depth-First Search (DFS) on HTML nodes, you can write a recursive function that traverses each node starting from a root element, visiting its child nodes before moving to its sibling nodes. Use the DOM methods like `firstElementChild` and `nextElementSibling` to navigate through the nodes.

**Q28: Implement throttle.**
**A28:** Throttling a function ensures it is only executed once within a specified time interval. Here's an implementation:

```
function throttle(fn, limit) {
    let lastFunc;
    let lastRan;
    return function(...args) {
        const context = this;
        if (!lastRan) {
            fn.apply(context, args);
            lastRan = Date.now();
        } else {
            clearTimeout(lastFunc);
            lastFunc = setTimeout(function() {
                if ((Date.now() - lastRan) >= limit) {
```

```
                    fn.apply(context, args);
                    lastRan = Date.now();
                }
            }, limit - (Date.now() - lastRan));
        }
    };
}
```

**Q29: How do you make a function that only calls input function `f` every 50 milliseconds?**
**A29:** You can create a throttled version of `f` to ensure it only executes every 50 milliseconds:

```
function throttle(f, interval = 50) {
    let lastCall = 0;
    return function(...args) {
        const now = Date.now();
        if (now - lastCall >= interval) {
            lastCall = now;
            return f(...args);
        }
    };
}
```

**Q30: Given a timeline, write the JavaScript to select all nodes within the selection of timeline.**
**A30:** Assuming a timeline with selectable nodes, you can use a range selection technique. Listen for the selection event, get the selected range, and iterate through the nodes to determine which fall within the range using DOM methods like `getBoundingClientRect()`.

**Q31: Design a slider component.**
**A31:** A slider component can be designed using HTML, CSS, and JavaScript. The component should consist of a range input element (`<input type="range">`) styled with CSS for aesthetics. JavaScript can be used to update the displayed value dynamically and handle custom events like sliding animations or snapping to specific values.

**Q32: Design a Tic-Tac-Toe game/design an algorithm for a Tic-Tac-Toe game.**
**A32:** A Tic-Tac-Toe game can be implemented using a 2D array to represent the board. The game alternates between two players, updating the board with each move. The algorithm checks for win conditions (three in a row horizontally, vertically, or diagonally) after each move and declares a winner or a draw when the board is full.

**Q33: Implement nested checkboxes (when the parent is checked, children are checked and vice versa. Use `<input type="checkbox">`). Similar to Indeterminate checkboxes.**
**A33:** Implementing nested checkboxes involves adding event listeners to each parent checkbox. When a parent checkbox is checked or unchecked, all its child checkboxes are automatically toggled to match. If any child is checked/unchecked, the parent checkbox's state is updated accordingly, possibly showing an indeterminate state if not all children are uniformly checked.

**Q34: Design a webpage which can auto-load new posts when you reach the bottom of the page by using JavaScript. You may use AJAX and JavaScript event listeners.**
**A34:** To auto-load new posts, use the `scroll` event listener to detect when the user reaches the bottom of the page. When the bottom is reached, an AJAX request is triggered to fetch additional posts from the server, which are then dynamically appended to the DOM without refreshing the page.

**Q35: Write a UI using HTML, CSS, JavaScript that allows users to enter the number of rows and columns in text input fields within a form and renders a table.**
**A35:** The UI includes a form with two input fields for rows and columns. Upon submission, JavaScript retrieves the values and generates an HTML table with the specified number of rows and columns, appending it to the DOM. Styling can be applied with CSS to enhance the table's appearance.

**Q36: Explain the CSS Box Model.**
**A36:** The CSS Box Model describes the rectangular boxes generated for elements in a web page, including content, padding, border, and margin. The content area is the actual content, padding surrounds the content, the border encloses the padding, and the margin is the space outside the border that separates the element from others.

**Q37: What happens when you type a URL into the browser and hit enter?**
**A37:** When you type a URL and hit enter, the browser performs a DNS lookup to resolve the

domain name to an IP address. It then sends an HTTP/HTTPS request to the server at that IP address. The server processes the request, sending back an HTML response, which the browser parses and renders to display the web page.

**Q38: Given some text on a web page, how many ways can you make the text disappear?**
**A38:** Text can be made to disappear using `display: none;`, `visibility: hidden;`, `opacity: 0;`, `text-indent: -9999px;`, or positioning it off-screen using `position: absolute; left: -9999px;`. Additionally, JavaScript can be used to remove the text node from the DOM entirely.

**Q39: How do you send data from a web page to a server without a page refresh?**
**A39:** Data can be sent to a server without refreshing the page using AJAX or the Fetch API in JavaScript. This involves creating an asynchronous HTTP request, sending the data to the server, and processing the response, all while keeping the user on the current page.

**Q40: Design emoji autocomplete.**
**A40:** Emoji autocomplete can be designed using an input field that listens for key events. When the user types a colon (`:`), a dropdown appears with emoji suggestions. As the user continues typing, the suggestions filter down to match the input. Upon selecting an emoji, the text is replaced with the selected emoji in the input field.

**Q41: Design JS Bin.**
**A41:** JS Bin is an online code editor that allows users to write HTML, CSS, and JavaScript, and see the results in real-time. To design something similar, create a split-pane interface with editors for HTML, CSS, and JS. Use an iframe to render the output, updating it in real-time as the user types. Implement features like code syntax highlighting, sharing bins, and embedding them into web pages.

**Q42: How would you create a Google Analytics SDK used by webpages?**
**A42:** To create a Google Analytics-like SDK, develop a JavaScript library that can be embedded into web pages. This SDK would track user interactions like page views, clicks, and events. It would send the data asynchronously to a backend server for aggregation and analysis. Implement features for customizing the data tracked and ensuring compliance with privacy laws.