# Interview Questions for Various Profile For EY Talent Initiative

## Front-End Technologies

### General

1. What are the core languages used in front-end development?
2. Can you explain the difference between == and === in JavaScript?
3. Describe the box model in CSS.
4. What is the Document Object Model (DOM)?
5. How do you optimize a website's performance?
6. Explain how CSS preprocessors work.
7. What are the benefits of using a front-end framework like React or Vue.js?
8. How do you ensure your website is accessible?
9. What is the purpose of a service worker?
10. How do you handle state management in single-page applications?

### HTML & CSS

1. What does DOCTYPE mean in HTML?
2. Explain the difference between block-level and inline elements.
3. What are semantic HTML elements? Give examples.
4. How do you make a website responsive?
5. What is the box model in CSS, and how does it affect layout?
6. Explain CSS specificity and how it works.
7. How do you create a grid layout in CSS?
8. What are CSS preprocessors, and how do they enhance CSS development?
9. How do you implement custom fonts on a web page?
10. Describe the process of making a web page accessible.
11. What is the purpose of the **alt** attribute on images?
12. How would you implement a multi-language website?
13. Explain the concept of CSS Flexbox.
14. What are media queries, and how are they used?
15. What is the difference between **id** and **class** selectors in CSS?
16. How can you optimize the loading of CSS files?
17. Describe how you would create a theme switcher (dark mode/light mode).
18. How do you handle browser-specific styling issues?
19. Explain the importance of mobile-first design.
20. What are pseudo-elements and pseudo-classes in CSS?

### JavaScript

1. What are the differences between **var**, **let**, and **const**?
2. Explain event bubbling and event capturing.
3. What are closures in JavaScript, and how are they used?
4. How do you handle exceptions in JavaScript?
5. Explain the concept of hoisting in JavaScript.
6. What are JavaScript Promises?
7. How does the **this** keyword work in JavaScript?
8. Explain the concept of prototypal inheritance.
9. What are arrow functions, and how do they differ from regular functions?
10. How do you copy an object in JavaScript?
11. What is event delegation, and why is it useful?
12. How do you ensure your JavaScript code runs after the HTML document is loaded?
13. What is JSON, and how do you work with it in JavaScript?
14. Explain the concept of modules in JavaScript.
15. What are Web Workers, and when would you use them?
16. How do you handle asynchronous operations in JavaScript?
17. What are some ES6 features that you find most beneficial?
18. How do you prevent a form from being submitted?
19. What is the Document Object Model (DOM)?
20. How do you find and fix memory leaks in JavaScript?

### Frameworks & Libraries

1. What are the differences between Angular and React?
2. How do you choose between using a framework or a library?
3. Explain the virtual DOM concept in React.
4. How do you manage state in React applications?
5. What are components in Angular and React?
6. How do you handle forms in React?
7. What is JSX, and why is it used in React?
8. Explain the concept of directives in Angular.
9. How do you create a single-page application (SPA) with Angular or React?
10. What are hooks in React, and how do you use them?
11. Describe the Redux flow.
12. How do you optimize React application performance?
13. What is server-side rendering, and when would you use it?
14. How do you handle API calls in React and Angular?
15. What are the benefits of TypeScript in front-end development?
16. Explain Angular's two-way data binding.
17. How do you ensure your Angular application is secure?
18. What is the context API in React?
19. How do you use Webpack for application bundling?
20. What are Progressive Web Apps (PWAs), and how do you create one?

### Performance & Optimization

1. How do you optimize website loading times?
2. Explain the importance of minimizing HTTP requests.
3. What are some tools you use for performance testing?
4. How do you ensure your images are optimized for the web?
5. What is lazy loading, and how do you implement it?
6. How does minifying CSS and JavaScript improve performance?
7. What are service workers, and how do they contribute to web performance?
8. How do you use browser caching to improve website performance?
9. Explain the critical rendering path.
10. What strategies do you use to reduce the time to first byte (TTFB)?

### Accessibility & Security

1. What are ARIA roles, and how do you use them?
2. How do you ensure your web applications are accessible to people with disabilities?
3. What are common web security threats, and how do you mitigate them?
4. Explain the importance of HTTPS.
5. How do you prevent cross-site scripting (XSS) attacks?
6. What are content security policies (CSP), and how do you implement them?
7. How do you secure data transmitted between client and server?
8. What steps do you take to ensure form data is validated?
9. How do you prevent clickjacking attacks?
10. What is cross-site request forgery (CSRF), and how do you prevent it?

### Miscellaneous

1. How do you stay current with front-end technologies?
2. Explain the concept of progressive enhancement.
3. How do you handle SEO in single-page applications?
4. What is the importance of code reviews, and how do you approach them?
5. How do you test your front-end code?
6. Describe your workflow when you start a new front-end project.
7. How do you handle feature toggles in your applications?
8. What is containerization, and how does it affect front-end development?
9. How do you handle user authentication in web applications?
10. Explain the importance of build tools in front-end development.

### Trends & Future

1. What are the upcoming trends in front-end development you're excited about?
2. How do you see WebAssembly impacting front-end development?
3. What is your opinion on the future of web design patterns like Material Design?
4. How do you think front-end development will evolve with the rise of AI and machine learning?
5. What role do you think front-end developers will play in the development of VR and AR web experiences?
6. How do you approach designing for voice interfaces and conversational UIs?
7. What do you think about the future of CSS, such as Houdini and container queries?
8. How do you see GraphQL changing the way front-end developers work with APIs?
9. What are the challenges and opportunities of developing for the Internet of Things (IoT)?
10. How do you prepare for the increasing importance of privacy and data protection in web development?

## Back-End Technologies

1. What are RESTful APIs, and how do you create one?
2. Explain the MVC architecture.
3. How do session and cookie work in web applications?
4. What is the difference between SQL and NoSQL databases?
5. Can you describe the process of authentication and authorization?
6. How do you handle file uploads in a web application?
7. What is a microservices architecture?
8. Explain how you would prevent SQL injection attacks.
9. What are some key differences between synchronous and asynchronous programming in Node.js?
10. How do you manage dependencies in a project?

## Databases

1. How do you design a relational database schema?
2. What are primary keys and foreign keys?
3. How do you perform a database normalization?
4. Explain the ACID properties of a transaction.
5. What are indexes, and why are they important?
6. How do you optimize a slow query?
7. What is a NoSQL database, and when would you use one?
8. How do you ensure data integrity in a database?
9. What are stored procedures?
10. How do you manage database migrations?

## Web Fundamentals

1. What is the difference between HTTP and HTTPS?
2. Explain how DNS works.
3. What are websockets, and when would you use them?
4. Describe the TCP/IP model.
5. How does a web browser render a webpage?
6. What are the main components of an HTTP request and response?
7. Explain the concept of CORS.
8. What is a CDN, and why would you use one?
9. How do you secure a web application?
10. What is the significance of load balancing?

## Development Practices

1. What is version control, and how do you use it?
2. Explain the concept of continuous integration and continuous deployment (CI/CD).
3. What is unit testing, and why is it important?
4. How do you debug a problem in your code?
5. What is the Agile development methodology?
6. How do you manage project dependencies?
7. Describe the process of code review.
8. How do you document your code and APIs?
9. What is containerization, and how does it help in development?
10. How do you ensure code quality?

## Problem Solving & Algorithms

1. Describe how you would solve a given coding problem.
2. Explain the difference between a linked list and an array.
3. How do you reverse a string in JavaScript?
4. What is a binary search algorithm?
5. Explain the concept of recursion with an example.
6. What are hash tables, and how are they implemented?
7. How do you detect a cycle in a linked list?
8. What is a sorting algorithm you are familiar with, and how does it work?
9. Explain dynamic programming and its benefits.
10. What is Big O notation, and why is it important?

## Soft Skills & Miscellaneous

1. How do you manage time when working on multiple projects?
2. Describe a challenging project you worked on and how you overcame the challenges.
3. How do you stay updated with the latest technology trends?
4. Explain the importance of team collaboration in software development.
5. Describe a situation where you had to learn a new technology quickly.
6. How do you handle feedback on your code?
7. What is your approach to testing and quality assurance?
8. How do you prioritize tasks in a project?
9. Explain the concept of scalability in web applications.
10. What do you enjoy most about being a full-stack developer?

## Advanced Topics

1. How do you implement security measures in a web application?
2. Explain the concept of serverless architecture.
3. What are containers, and how do they differ from virtual machines?
4. How do you implement a search feature in a web application?
5. What is GraphQL, and how does it compare to REST?
6. Explain the concept of state management in complex applications.
7. How do you handle large-scale data in web applications?
8. What are progressive web apps (PWAs)?
9. How do you optimize the performance of a React/Vue.js application?
10. What is the significance of micro-frontends?

## Specific Technologies & Frameworks

1. Can you explain the virtual DOM and its benefits?
2. How do you manage global state in a React application?
3. What are decorators in Angular, and how do you use them?
4. Explain middleware in the context of Express.js.
5. What are slots in Vue.js?
6. How do you handle authentication in a Single Page Application?
7. What are the benefits of using TypeScript?
8. How do you perform form validation in a front-end framework?
9. What is the Context API in React?
10. How do you use hooks in React?

## Deployment & Operations

1. Explain the process of deploying a web application.
2. How do you monitor and improve the performance of a live application?
3. What is Docker, and how do you use it?
4. Explain how to use environment variables in a project.
5. How do you automate tasks in your development workflow?
6. What are some common security vulnerabilities in web applications?
7. How do you ensure your application is scalable?
8. What is continuous integration/continuous deployment, and how do you implement it?
9. How do you manage application logging?
10. What tools do you use for performance profiling?

## Additional List of Questions to Sort Through

1. To develop a project from scratch, what technologies and languages would you need or what skills a full stack developer should have?
2. Which language is the most preferred by full-stack developers?
3. Explain Pair Programming.
4. What do you mean by CORS (Cross-Origin Resource Sharing)?
5. What is Callback Hell?
6. Explain Long Polling.
7. Can you tell me what are the latest trends in Full Stack Development? Also, how do you keep yourself updated about the new trends in the industry?
8. State difference between GraphQL and REST (Representational State Transfer).
9. What is CI (Continuous Integration)?
10. Explain the meaning of multithreading.
11. Explain the benefits and drawbacks of using "use strict".
12. What are some of the uses of Docker?
13. Explain event loop in Node.js.
14. Is there a way to decrease the load time of a web application?
15. Explain dependency injection.
16. What do you mean by observer pattern?
17. State difference between blue/green deployment and rolling deployment.
18. Explain inversion of control.
19. What do you mean by referential transparency in functional programming?
20. State difference between normalization and denormalization.
21. In Java, what is a connection leak? How can you fix this?
22. What is Promise and explain its states?
23. State the difference between GET and POST.
24. Explain the Restful API and write its usage.
25. What do you mean by MEAN Stack?
26. Do you know how to prevent a bot from scraping your publicly accessible API?
27. What makes MVC (Model View Controller) different from MVP (Model View Presenter)?
28. What do you mean by Temporal Dead Zone in ES6?
29. Why should arrow functions not be used in ES6?
30. What is event bubbling and capturing in JavaScript?
31. Tell me about a project that you worked on and the technologies you used. Why did you choose them?
32. In the past, what was the best implementation or debugging you did?
33. What is Full Stack development?
34. What do Full Stack Web Developers do?
35. Name a few Full Stack developer tools.
36. What skills do you need to be a full-stack developer?
37. What is CORS?
38. What is Inversion of Control (IoC)?
39. What is Dependency Injection?
40. What is Continuous Integration?
41. What is multithreading and how it is used?
42. How is GraphQL different from REST?
43. List the ways to improve your website load time and performance.
44. What is the Observer pattern?
45. What’s the difference between a Full Stack Engineer and a Full Stack Developer?
46. What is polling?
47. What’s the difference between GET and POST?
48. What’s the difference between abstract and interface?
49. How can you prevent a bot from scraping a publicly accessible API?
50. What is RESTful API?
51. What is a callback in JavaScript?
52. What do you mean by data attributes?
53. What's the difference between "resetting" and "normalizing" CSS?
54. What does ACID mean in Database systems?
55. How is rolling deployment different from blue-green deployment?
56. What is an Application server?
57. What is referential transparency?
58. What are the differences between Server-side Scripting and Client-side Scripting?
59. What are the types of design patterns?
60. What’s the difference between normalization and denormalization?
61. Name a few ways to optimize a website to be as efficient and scalable as possible?

# Sample Answers

## Front-End Technologies

1. **Core languages for front-end development:** HTML for structure, CSS for styling, and JavaScript for functionality.
2. **Difference between == and === in JavaScript:** **==** compares values after type coercion, while **===** compares both values and types, without conversion.
3. **Box model in CSS:** Consists of margins, borders, padding, and the actual content area, from outside to inside.

## Back-End Technologies

1. **What are RESTful APIs:** APIs that follow REST principles, allowing interaction with RESTful web services using standard HTTP methods.
2. **MVC architecture:** A design pattern that separates an application into three main components: Model, View, and Controller, to separate internal representations of information from the ways that information is presented and accepted by the user.

## Databases

1. **Designing a relational database schema:** Involves defining tables, keys, and relationships between tables that represent how data is stored and accessed.
2. **Primary keys and foreign keys:** A primary key uniquely identifies each record in a table, while a foreign key is a field (or collection of fields) in one table that uniquely identifies a row of another table.

## Web Fundamentals

1. **Difference between HTTP and HTTPS:** HTTPS is the secure version of HTTP, where communications are encrypted via TLS or SSL.
2. **How DNS works:** Translates domain names to IP addresses so browsers can load Internet resources.

## Development Practices

1. **What is version control:** A system that records changes to a file or set of files over time so that you can recall specific versions later. Git is a widely used system for version control.
2. **Continuous integration and continuous deployment (CI/CD):** Practices in software development where code changes are automatically built, tested, and deployed to production.

## Problem Solving & Algorithms

1. **Solving a coding problem:** Break down the problem into smaller parts, understand the input and output requirements, choose the appropriate data structures and algorithms, write the code, and then test it with different scenarios.
2. **Difference between a linked list and an array:** An array is a collection of items stored at contiguous memory locations, whereas a linked list is a sequence of elements where each element links to the next, allowing for efficient insertion and deletion.

## Soft Skills & Miscellaneous

1. **Managing time on multiple projects:** Prioritize tasks based on urgency and importance, use tools like calendars and to-do lists for organization, and don’t hesitate to communicate with team members about workload and deadlines.
2. **Challenging project and overcoming challenges:** Describe a specific project, the challenges faced (technical, team dynamics, etc.), the strategies employed to overcome them (collaboration, learning new skills, etc.), and the outcome.

## Advanced Topics

1. **Implementing security measures in a web application:** Include using HTTPS, sanitizing user input to prevent SQL injection and XSS attacks, implementing secure authentication and authorization, and keeping software up to date.
2. **Serverless architecture:** Allows developers to build and run applications and services without managing infrastructure, as the cloud provider automatically manages the allocation and provisioning of servers.