Front-End Engineer Interview Preparation Guide

This guide was created to prepare you for the challenges you will face in the Front-End Engineer interview, and the knowledge you must possess to be successful in each step of the interview process.

Process:

- 5 - 60-Minute Interviews
- 3 to 4 Coding Challenges
- 1 System design challenge
- Total of 10-12 behavioral questions that are looking for past examples where you exhibited one of Amazons 16 Leadership Principles.

Front-End Coding Interviews:

1. **Front-End (Data Structures and Algorithms)**
2. **Front-End (Logical and Maintainable)**
3. **Front-End (JavaScript, HTML, CSS, Frameworks)**
4. **Front-End (Problem Solving)**

Systems Design Challenge: (See below for details)

What can you expect?

- These interviews will require you to write code in an online code editor. These interviews are structured to help demonstrate coding skills, code quality and conceptual thinking.
- An important part of these interview is to assess the problem and discuss it with your interviewer. Demonstrating how you think and approach the problems is equally important as the end result.
- We understand that the problems are rarely linked to a single solution. Please discuss all the possible solutions and why the approach selected is the best fit. Highlighting other solutions helps the interviewer understand your thought process.
- As the emphasis is on the best practices we encourage all the candidates to test their code. It is best to avoid bugs, but the interviewer will not compile code so don't worry about the minor mistakes.

- Our preferred choice of language for Front End Engineering interview is Vanilla JavaScript. A LiveCode link for the code editor will be attached as part of an invitation email.

What do we look for?

During these interviews the interviewer will assess your performance on these areas:

- **Problem Solving** - We're evaluating how you comprehend and explain complex ideas. Are you providing the reasoning behind a particular solution?
- **Developing and comparing multiple solutions?** Using appropriate data structures? Speaking about space and time complexity?
- **Coding** - Can you convert solutions to executable code? Is the code organized and does it have the right logical structure? Is the HTML you are writing semantic and follow Web Standards?
- **Verification** - Are you considering a reasonable number of test cases or coming up with a good argument for why your code is correct?
- **Communication** - Are you asking for requirements and clarity when necessary, or are you just diving into the code? Don't forget to ask questions.
- **Think Big:** Is the code you are writing going to scale for one or any type of users? Have you considered accessibility implications? Can your user interact with the application you have written without the need for a mouse?

How to prepare?

- Get comfortable with JavaScript. You need to have a clear understanding of variables, functions, objects, arrays, asynchronous control flow, and closures. Check out https://javascript.info/ to refresh your knowledge.
- Understand the DOM. Front-End Engineers create complex applications with high interactivity for multiple types of users. Study the rendering capacities available to you and layout mechanisms available in CSS. Study common and advanced HTML tags as well as CSS selectors and rules.
- Think about different algorithms and algorithmic techniques (iteration, sorting, divide-and- conquer, memoization, recursion).
- Think about data structures, particularly the ones used most often (Array, Stack / Queue, Object Hash, Tree, Set, Map, etc). While the front-end interview doesn't focus on building them from scratch, we expect that you can improve UI performance by choosing the appropriate data structures.
  You should expect to solve about three small problems in the course of about 45 minutes. When you have a solution, be sure to review it. Make sure that it's correct, that you've considered the edge cases, that it's efficient, and that it clearly reflects the ideas that you're trying to express in your code.

System Design Challenge

What can you expect?

- You will be asked to design a web-scale application. For example, they might ask you to design Instagram widget, Type Ahead widget, Infinite scrolling, Carousel Widget.
- Unlike a coding interview question, System Design Interviews are free-form discussions, and there's no right or wrong answer. Instead, we are trying to evaluate your ability to hold a conversation about the different aspects of the system and assess the solution based on the requirements that might evolve during the conversation.

- You should drive the conversation by asking clarifying questions, discussing the pros and cons of various approaches and describing the utility of each component that you propose to use.
- You have clear understanding of the required data that the application needs and have capacity to design APIs. You create the data communication contracts between Front and Back End.
- You understand how the Web and the DOM work. You understand and apply concepts like asynchronous communication, caching, security, session maintenance. You understand and use client-server communication methods and pick the right technology according to the problem you are solving.

What do we look for?

- **Technical Communication** - Can you articulate your vision and technical ideas clearly? We are assessing your ability to communicate your reasoning as well as understand and address feedback from the interviewer.
- **Improve upon a design** - Think about and review the complex systems you've already designed. What would you change about your approach if you did it all over again? What worked well?
- Explore competing solutions and speak to all their major tradeoffs. Demonstrate that you can make intelligent decisions about how to balance each of those tradeoffs.
- **Scalability**: Start with 1 user, 1 server, 1 app but do emphasis on the next scale curve. For eg - When user base grows to 1 million, what are changes required in your design?
- **Adaptability**: In the real world, changes to requirements can happen suddenly. We look for engineers who can rapidly iterate over a new or existing solution and that can create Web applications that can scale to multiple customers.

How to prepare?

- Do some research. Read engineering blogs about approaches that have worked for big companies along the way. Read about the false-starts too!
- You can practice to divide your interview in following areas -
    - **Understanding the problem and establish the design scope.** One of most important skill as an engineer is to ask the right questions, make proper assumptions, and gather all the information to build the system. For example - What specific features are we you going to build? How many users are going to use the system.
    - **Propose a high-level design and concur with the interviewer.** Discuss as many solutions as you can before you decide to deep dive. Come up with an initial blue print for the design. Treat your interviewer as a teammate and work together.
    - **Design and Deep dive.** At this step you have already pinned down on the feature you will be elaborating and its scope. Proposed a high level blueprint and Obtained a feedback from the interviewer. Now, You can work with your interviewer to pick each component one by one and discuss more on each of them.
    - In the end, it is good to wrap things up. Go through all the components you have designed and trade-offs taken during the deep dive phase. It is helpful to recapitulate as soon as a solution has been reached in case you or the interviewer have overseen any requirement needed to complete the system design question.

Additional Resources:

- Every FEE guide - https://developer.mozilla.org/en-US/

- Practice FEE FAQs - https://bigfrontend.dev/problem, https://css-tricks.com/
- Accessibility guide - https://www.youtube.com/playlist?list=PLNYkxOF6rcICWx0C9LVWWVqvHlYJyqw7g
- Advanced Concepts - https://www.30secondsofcode.org/, https://web.dev/learn/