

- Why System Design Interviews?
- System Design Basics
- Load Balancing
- Caching
- Sharding or Data Partitioning
- Indexes
- Proxies
- Queues
- Redundancy and Replication
- SQL vs. NoSQL
- CAP Theorem
- Consistent Hashing
- Long-Polling vs WebSockets vs Server-Sent Events (*New*)



Why System Design Interviews?

1. What are system design interviews?

In system design interviews, candidates are required to show their ability to design a large system. Designing software systems is a very broad topic and even a person of experience at a top software company may not claim to be an expert on system design. In interviews, candidates have 30 to 40 minutes to answer questions like, “How to design like Dropbox?” or “How to design a search engine” etc. In real life, companies hire and hire a big team of software engineers to build such systems. Given this, the question is, how can a candidate answer such a question in 40 minutes? Moreover, there is no set pattern of such questions. They are unpredictable, usually open-ended, and have no standard or squarely correct answers.

Unlike coding interviews where problem solving ability of the candidates is evaluated, in system design interviews, the questions consist of complicated and fuzzy questions which aim at testing the candidate's ability to design a complicated problem, their compatibility with building large systems and how they communicate.



These days, companies are less worried about your pedigree, where you studied, but surely concerned about what you can do on the job. For them, the most important thing is your process and your mindset to look into and handle problems. For these counts, the questions are more about your process of the design interviews. But despite all this, I believe there is no need of scaring yourself. The companies want to know about you during these 40 minutes, which is basically about your “problem-solving strategy to handle a problem” and how organized, disciplined, systematic, and communicative you are. What is your capacity to analyze an issue and your level of professional maturity?

In short, system design interview is, just understanding it from interviewer's perspective. In the end of the process, it is your discussion with the interviewer that is of core importance.

2. How to give system design interview?

There is no strictly defined process to system design interview. Secondly, there is no clear-cut answer about large systems that without clarifying at least a few of them in the question, it is impossible to go for a solution. Any candidate who does not realize this fact will be jumping onto finding a solution.

For instance, the questions can be like:

- Design a URL shortening service like TinyURL.



- How to design a ride-sharing service like Uber, which connects passengers who do not have a car to those who have a car.

Any candidate who does not have experience in building systems might think that these questions are too broad. On top of that, there generally isn't any one correct answer to such questions. The question would sufficiently tell upon your professional competence and background, which is the thing which the interviewer will evaluate you on.

Since the questions are intentionally weakly defined, jumping into designing a solution without understanding them properly is liable to get you in trouble. Spend a few minutes with the interviewer to comprehend the full scope of the system. Never assume things. For instance, the “URL shortening service” could be serving just a few thousand users or millions of URLs. It could also mean to handle millions of clicks on the shortened URLs. The service may also require providing extensive statistics about each shortened URL (like total data size), or statistics may not be a requirement at all. Therefore, don't forget to ask clarifying questions as the interviewer would not be listing them out for you in advance.

The point I want to make is that the main difference between design interview and coding interview is that you are not presented with the full detail of the problem at the outset. Rather you are given a high-level overview of the problem.



problem presented go a long way in evaluating your ability and competence at solving a problem.

In design and architecture interviews the problems presented are quite big. They are designed to be solved in 40 minutes' time implying that the objective is to test the technical depth and breadth of your knowledge. The interviewer invokes during the interview. That also speaks strongly for your would be level in the company should come from your analytical ability to sort out the problem, your ability to work in a team (your behavioral and background side of the interview), and your ability to be a strong technical leader. In a nutshell, the basic idea of hiring at a level is to scale your contribution to contribute value to the company's wants and needs. For that, you must exhibit a reasonable technical breadth.

Try to learn from the existing systems: How have these been designed? Another important thing in mind is that, the interviewer expects that candidate's analytical ability and knowledge should be comparable to his/her experience. If you have a few years of software development experience, you are expected to have certain knowledge and should avoid divulging into asking basic questions which have been appropriate coming from a fresh graduate. For that, you should prepare yourself by working on real projects and practices, well in advance of the interview as most of the questions are real-life products, issues and challenges.

Leading the conversation: It is not the ultimate solution to the problem, rather it is a conversation.



of solving the problem by communicating with him/her step by step as you move forward.

Solving by breaking down: Design questions at first might look complex and difficult. To tackle the complexity level of the problem, a top-down and modularization approach can be used. Break the problem into smaller problems. Therefore, you should break the problem into modules and then tackle each module. Subsequently, each component can be solved as a sub problem by reducing it to a simpler problem. This strategy will not only make the design much clearer to you and the interviewer, but also make the evaluation much easier for the interviewer. However, while doing so, keep track of the progress. The problems presented in high skill design interviews don't have the solutions. They are designed to test the way how you make progress tackling the problem, and the strategies you make while solving the problem.

Dealing with the bottlenecks: Working on the solution, you might confront some bottlenecks. It is normal. When confronting bottlenecks, your system might require a load balancer to handle the user requests or the data might be so huge that you need to store it on multiple servers. It might also be possible that the interviewer wants to take the solution in a different direction. If that is the case, you are supposed to move in that direction and shift your previous solution aside. If you feel stuck somewhere, you can ask for a hint so that you may move forward. Each solution is a kind of trade-off; hence, changing something may worsen some other thing. The most important thing is your ability to talk about these trade-offs and to measure the impact of each change. Keeping all the constraints and use cases in mind. After finishing with your high-level design, you can move on to the detailed design.



system design.

3. Summary

Solving system design questions could be broken down into three steps:

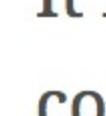
- Scoping the problem:** Don't make assumptions; Ask clarifying questions and use cases.
- Sketching up an abstract design** Illustrating the building blocks of the system and how they interact between them.
- Identifying and addressing the bottlenecks** by using the fundamental principles of system design.

4. Conclusion

Design interviews are formidable, open-ended problems that cannot be solved by a single answer. You should try to understand what your interviewer intends to focus on and should be well aware of the fact that the discussion on system design problem could go in many directions on the preferences of the interviewer. The interviewers might be unwilling to discuss the system architecture covering all aspects of the system or they could be interested in looking at a specific part of the system.



lack the ability to focus on the right things while discussing the problem.



It is also advisable to arrange discussions and even mock interviews with experienced people or companies.

Remember there is no ONE right answer to the question because any system could be designed in many ways.

The only thing that is going to be looked into is your ability to rationalize it.

✔ Mark as completed

Next →
System Design Basics