Job Interviews

Linkedin Senior Software Engineer Onsite Interview

B BjVa24
Jun 19

Any tips for prepping for an upcoming Senior Software Engineer Onsite interview at Linkedin? Specific responses about topics , guidance on system design questions would be much appreciated.

LinkedIn Software Engineer

Questions

1. Any tips for prepping for an upcoming Senior Software Engineer Onsite interview at Linkedin? Specific responses about topics , guidance on system design questions would be much appreciated.

50 words minimum.

1 Answer

H help LinkedIn
Jun 20

+ Ask

1. Any tips for prepping for an upcoming Senior Software Engineer Onsite interview at Linkedin? Specific responses about topics , guidance on system design questions would be much appreciated.

I have a whole lot of advice for how to prepare so... get ready and I'll try to break it down with regards to the limited information you provided in your question, and I'm assuming you have passed your TPS already:

You will likely be presented with a system design example and the process tends to be either:

1. Present a prepared system design question and work through it with the candidate.

2. Using your resume/experience, dig deeply into a previous system (I'll dig into this in more detail in a mo

No matter what, your recruiting manager or hiring manager should have given you a bit of information or u are going to meeting with. They should spend a few minutes introducing themselves, and a bit of their background and role. The other interviewers (if they exist) should do the same. Then they'll have you do the same. The reason they go first, is to set an example for what they're looking for in response, and to give you a bit of context.

In both of the above scenarios the goal is to be able to gauge your skill set. Option 1 will help understand your thought process and skills within LinkedIn's domain, but can put you at a disadvantage if your experience doesn't overlap as much as another candidate's. Option 2 will help show you at your best since (hopefully) you know the subject matter well, but it might not show LinkedIn your problem-solving approach as well since a previously-solved project is being discussed. Certainly both scenarios can occur, time permitting.

With regards to Option 1, there isn't too much detail I can provide on what kinds of questions you'll be asked but if you are at the "Senior" level in this space then it is likely that there shouldn't be anything that is going to throw you for a loop. My experience interviewing at LinkedIn never made me feel like they were trying to trick or trap me.

No mater what they'll often have a couple questions about projects mentioned on your resume, but these are mostly just to get a conversation going, and get you comfortable. You'll likely be asked if you have any experience with some of the tools/frameworks that are used and how you've used them. And if you haven't worked with something, that's ok, we can teach it... not a big deal but it might mean you aren't coming in at the level you expect. But all this is for the "big questions" which could be a 2 part thing:

1. Tell us about a system that you were a major contributor on, give us a high level architecture, and how it basically worked. Use the whiteboard if it helps to draw it out (almost everyone did). Then you could be asked some questions about it, so that the interviewers feel that they understood it.

This will show that you can both explain things, and understand them.

2. Were there parts of the system that you wish you'd done differently, and how would you address those? Given no constraints on time/budget/resources, what would you have done differently, knowing what you know now?

This will show if you have some detachment from your code, nothing that is written is written without flaws, there's always a better way... often we make concessions due to time/budget/resources. It also shows that you're thinking at a bigger picture, and not just about the task. This is really the more important part of the discussion. About the only answer that people won't like is "nothing, it's perfect"... especially when the interviewer can see flaws... especially if they say "what about this bit...how was it?" and you say "buggy" ... and still insist the system couldn't be improved.

At a high level, be prepared to talk about the answers to these kinds of questions:

- What's the worst bug you've tracked down?
- In what sort of scenarios do you thrive in compared to your peers? Where might you struggle?
- Assume I am someone fresh out of college. I'd like you to explain <some concept specific to your favorite language>. What does it do? How does it work? Why is it good?
- Specific technical questions related to the team you could be joining to see if you know stuff
- Here is some scenario. These are the constraints. I would like you to design a solution for it. Please ask me if you have questions about requirements, etc.

The biggest thing I want to see in the design question (and in other questions) is an understanding of *implications*. Can the person work out that a constraint means that certain solutions are poor? Do you understand that their design leads to certain benefits/detriments? When I ask follow up questions about "what happens if this thing fails?" and similar, do they understand the implications, is their design resilient or naive?

Ultimately, all of the questions asked should inform the thing that the interviewer needs to know: Can this person (you) do this job (the one you're hoping to get) successfully? If it’s not in service of finding that out, then it probably isn't being asked.

I hope that helps, good luck in your preparations!