



How to Prepare for Success

Get an overview of some interview preparation guidelines and materials.

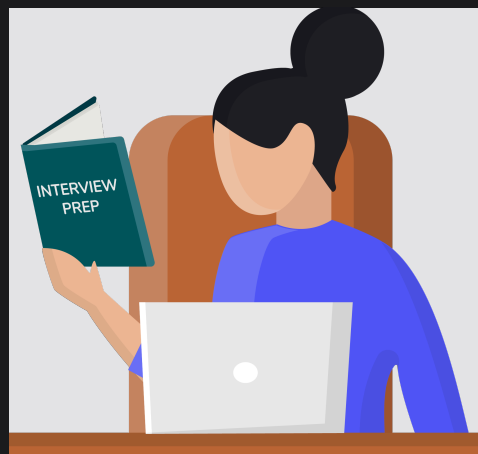
We'll cover the following ^

- This course
- Technical blogs
- Ask why a system works
- The right direction
- Mock interviews

Substantial preparation is necessary to increase our odds of getting the job we apply for.

Depending on a candidate's seniority and proficiency, different candidates need different times for interview preparation.

For an average candidate, three to four months might be required to prepare for a system design interview.



This course



This course helps readers learn or brush up on their system design skills. We've carefully curated some traditional and fresh design problems to cover the substantial depth and breadth of the system design.




The following activities might expedite the preparation and add variety and further depth to a candidate's knowledge.

Technical blogs

Many companies regularly publish the technical details of their significant work in the form of technical blogs.



Why are companies eager to share the technical details of their work?

 Show the reason

Note: There's a fine line between what can be held back by a company due to a competitive edge and what can be made public.



We can study these blogs to get insight into what challenges or problems the company faced and what changes they made in the design to cope with these challenges.

>

Note: Staying informed about these innovations is important for professionals, and it's even more crucial for an applicant.

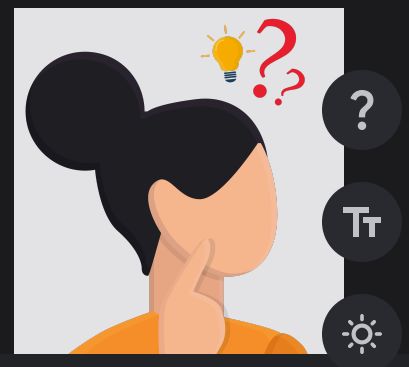
Some important technical blogs are [Engineering at Meta](#), [Meta Research](#), [AWS Architecture Blog](#), [Amazon Science Blog](#), [Netflix TechBlog](#), [Google Research](#), [Engineering at Quora](#), [Uber Engineering Blog](#), [Databricks Blog](#), [Pinterest Engineering](#), [BlackRock Engineering](#), [Lyft Engineering](#), and [Salesforce Engineering](#).

Note: We should always take non-peer-reviewed material with a grain of salt. Think about what blogs say critically and with technical acumen to decide if what they say makes sense or not. If it doesn't make sense, that could be an excellent opportunity to discuss the issue with peers.

Ask why a system works

By asking themselves the right questions, candidates can think through dense and ambiguous situations.

- Learn how popular applications work at a high level—for example, Instagram, Twitter, and so on.
- Start to understand and ask why some component was used instead of another—for example, Firebase versus SQL.
- Build serious side projects. Start with a simple



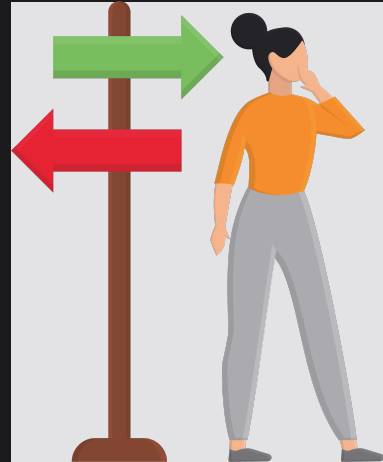
all the processes and details of its construction.

> We can clone a popular application without tutorials.

The right direction

System design deals with components at a higher level, and we need to avoid going into the trenches.

We should focus less on mechanics and more on trade-offs.



For example, discussing whether to use Room library instead of raw SQLite isn't helpful because Room library is a mere wrapper around SQLite. A better discussion might be about using traditional databases like MySQL or using NoSQL stores like MongoDB. This second discussion will help us pinpoint trade-offs between the two.

We should start with high-level stuff because the low-level details will automatically come up.

Mock interviews



>

Mock interviews are a great way to prepare for system design interviews. They involve pairing up with a friend and allowing them to ask questions. If it's not possible to use a friend, another strategy is to record ourselves and play the role of both interviewer and interviewee. With this approach, we can critique ourselves or ask a knowledgeable friend for feedback.



Note: Since no mock interview can fully mimic an actual interview, it's a good idea to take real interviews in companies. Once we've been through an interview or two, we'll be better able to evaluate what went right and what didn't.

