System Design: Technical Spec

Now that we know much more clearly what problem we're solving, let's go broad, and build a **technical spec.**

Your goal here is to outline all of the component parts we would need to build in our system. Imagine you will divide up work across a bunch of senior engineers, and you want to find out who will own what.

Here's how we suggest you do this.

Look over each screen, and draw out the systems we would need to support everything.

For example, we'll need a web page, so draw a box for that. We'll need an auth api, so draw a server box, and name the APIs. We'll need to store the data, so draw the database box. We may need caching, so you can draw the redis box.

To do this well, we suggest two paths:

If you are already comfortable with distributed systems — you can go right in and start drawing out the complete system — discussing how you would add redundancy and noting down the kinds of tradeoffs you would make.

Alternatively, you can start off by saying explicitly **let's first build a simple system, and scale it from there.** It's important to explicitly mention this to the interviewer, so you aren't interrupted in the middle. Once you draw a complete simple system, start asking over and over — what will fail, and fix them, **noting down the kinds of tradeoffs you are marking.**

Either way, you will have finished this step once you have outlined broadly a complete system, that would solve our problem and scale.

One of the biggest mistakes people make here is ignoring the big problems that come to their mind.

For example, in Twitter's feed API, how would we load new tweets? We would need to handle merging with old tweets, and based on how we do our pagination, we may end up missing tweets.

This kind of problem is almost **the whole purpose of the interview.** Detecting this problem demonstrates your experience to the interviewer, and coming up with the right tradeoff shows your technical skill.

The key realization is this: you don't need to solve these problems at this moment yet. Instead, say the problem that comes to mind, the interviewer will either clarify what they expect, or you can note it down to handle later when you get to the API.

Once you're done with the technical spec, every screen you drew out should be supported by your system.