CloudWay Digital Inc.
cloudwaydigital.com

Mastering the Interview for Staff & Principal Engineers and Software Architects



The Insider's Guide

Copyright Information

Copyright © 2026 CloudWay Digital Inc.
cloudwaydigital.com

All rights reserved.

This publication and its contents are the intellectual property of **CloudWay Digital Inc.** and are protected under applicable copyright laws. The content was created by the author and is published and distributed by CloudWay Digital Inc.

No part of this guide may be reproduced, distributed, transmitted, or used in any form or by any means — electronic or mechanical — without prior written permission from the copyright owner, except as permitted under applicable copyright law.

For licensing inquiries, please contact: licensing@cloudwaydigital.com

Corporate Use
For corporate, team-wide, or organizational use, please contact:
info@cloudwaydigital.com

Credit: Cover image generated by ChatGPT.

Disclaimer

This guide is provided for informational and educational purposes only. It is intended to help readers prepare for senior technical interviews, think more deliberately about the interview process, and consider perspectives, questions, and approaches they may not have previously encountered.

The content reflects the author's professional experience and perspectives gained over many years working in software architecture, software engineering, and technical leadership, including experience both interviewing candidates and being interviewed for senior technical roles.

However, no interview process is universal. Every organization, team, role, interviewer, and candidate is different. Hiring practices, expectations, evaluation criteria, interview formats, and individual circumstances can vary significantly. The approaches, examples, strategies, and recommendations described in this guide may therefore be more applicable in some situations than in others.

Reading, following, or applying the material in this guide does not guarantee that you will pass an interview, receive a job offer, advance in a hiring process, or achieve any particular career or employment outcome. Interview results depend on many factors outside the scope or control of this publication.

The examples and techniques presented throughout the guide are intended to encourage thoughtful preparation, reflection, and professional judgment rather than provide prescribed answers or formulas that should be followed in every situation. Readers should evaluate and adapt the material based on their own experience, circumstances, and the specific organization and role for which they are interviewing.

While reasonable efforts have been made to ensure that the information is accurate and relevant, no guarantees are made regarding its completeness, accuracy, reliability, or applicability to any particular individual, organization, role, or situation.

The author and publisher disclaim liability for any direct or indirect loss, damage, or consequences arising from the use or application of the information contained in this guide. By using this guide, you acknowledge that you remain responsible for your own decisions, actions, preparation, and results.

This publication is not intended to provide legal, financial, employment, organizational, or other professional advice.

Table of Contents

[Copyright Information](#)
[Disclaimer](#)
[Table of Contents](#)
Who is This Guide For?
📓 Notes
    Technology Agnostic
    Soft Skills Matter More Than Hard Skills
    Role Overlap
Role Types
    Common Role Types
        Staff Software Developer/Engineer
        Principal Software Developer/Engineer
        Other Engineering Roles (Senior Principal/Distinguished/Fellow)
        Technical Lead
        Solutions Architect (and Its Various Subtypes)
Types of Architect Roles
    Solutions Architect
    Enterprise Architect
    Technical Architect
    Cloud Architect
    Integration Architect
    Infrastructure Architect
    Business Architect
    Application Architect
    Data Architect
    Network Architect
    Domain Architect
    Security Architect
    Roles Summary
The Difference Between Staff & Principal Software Engineers and Software Architects
        Staff Software Engineer/Developer
        Distinguished or Principal Software Engineer/Developer

Technology or Software Architect
The Interview
General Mindset
The Questions and Their Breakdown
General Structure
Projecting into the Future 💻🤖🚀🤝
Wrapping Up
Bonus Guide: How to Approach an Interview When Your Interviewer is Less Experienced Than Yourself
Mindset
Mindset #1
Mindset #2
Types of Questions You May be Asked
Bonus #2: The New Baseline Skill - AI Architectural Sense Checking

The Questions

"How would you design [ABC]?" 💻

Technical candidates of the Staff/Principal, Architect, and Lead levels need to work and think widely. They should be able to have a strong grasp of complex system design and architecture topics, alternatives, tools, and technologies.

Most importantly, they should also be aware of the limitations of their knowledge and know how to answer questions that they might not have an answer to at the moment.

They need to be comfortable enough to say:

> **"I have never dealt with this before, but here is how I would go about it."**

In other words, in this role, you need to be an avid learner and have the ability (and desire) to dive as deep and as wide into new topics as needed. Bonus points if you are also passionate about teaching these topics to others.

Senior folks, architects especially, need to foresee problems and pitfalls where others do not.

This type of question provides fertile ground for exploring your experience, abilities, and a variety of skill sets. Most importantly, it works on several dimensions as it demonstrates how well the candidate can adapt to new information, their problem-solving process, collaboration, and communication skills.

The way to ask this type of a question is to slowly build upon a foundation and expand from there.

For example, you are often asked to design some generic system or to design one of your own systems - one that is currently in existence in your current organization.

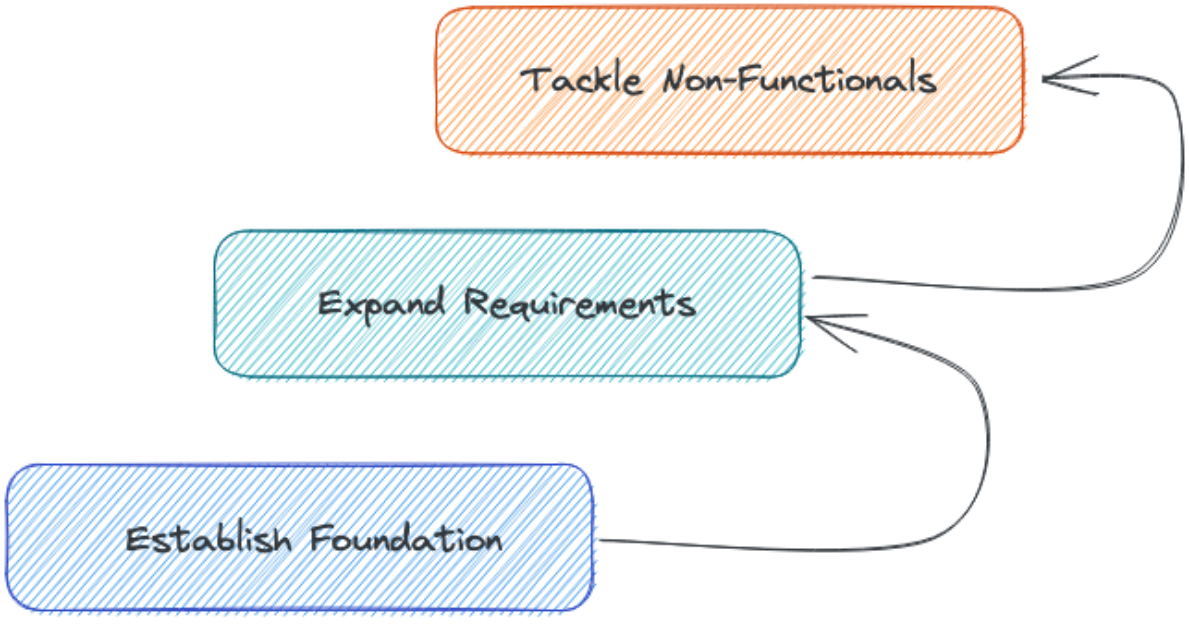
You obviously should not break any NDAs here or share more than you can about your organization's specifics. However, it helps to generalize things or say something along the lines of "***While I cannot divulge any information from my current employer, this is how I would go about designing such and such application***".

The key is to start answering in simpler terms and to add complexity to the question as you go.

The following illustration is one framework that facilitates this approach.

It consists of three stages:

- ✔ Establish Foundation
- ✔ Expand Requirements
- ✔ Tackle Non-Functionals



Example:

Establish Foundation

☑️ "**First Question:** You are building an application that will connect to a set of financial APIs for a particular user account and retrieve the financial details for that user from each API. How would you architect that kind of a system?"

What the interviewer is seeking is for the candidate to start breaking this generic question down into logical parts.

What they are seeking is for the candidate to bring up things such as

✅ Volume of requests and transactions per second

✅ Security and auditing of actions

✅ Synchronous vs asynchronous ways to integrate with these APIs

✅ Ways to integrate and types of third-party APIs

✅ How often is the integration planned to occur?

✅ Architectural tradeoffs in designing the system in this or that way

Expand Requirements

☑️ "**follow-up question:** Let's say that we also need to expose those aggregations to the end user. What would you need to add or change in the application's architecture? What if the application needs to scale to millions of users and handle hundreds of transactions per second?"

The interview will then often expand on the foundation built previously with more detail and specific questions. The candidate should be able to adapt to these new requirements and bring up ways to address them.

Ideally, they will be building incrementally on what they were describing prior. Here, the interviewer is often paying particular attention to whether the candidate has awareness of how decisions made earlier can impact the product or team in the future.

Tackle Non-Functionals

☑️ "**follow-up question No. 2:** How would you ensure the security of such an application? How would you deploy it and what kind of infrastructure would you use? How would you make it resilient and highly available?"

After addressing specific functional requirements, we're heading into an area of NFRs (or non-functional requirements and architectural trade-offs). This part is vital for distinguishing less experienced architects from more experienced ones.

This type of question measures:

- ✔ How well can the candidate build upon a foundation?
- ✔ Is the candidate thinking of limitations and edge cases?
- ✔ How does the candidate tackle "unknown unknowns"?
- ✔ How much technical grasp does the candidate have of these topics?
- ✔ Are you only superficially familiar and masking your own lack of knowledge by technical jargon and buzzwords or do you possess depth of knowledge?
- ✔ If only vaguely familiar - are you being upfront and transparent about that?

"How do you learn a new technology, tool, or technical concept?" 💻🤖

Here is a further expansion on this question.

"Let's say we are exploring a new avenue of leveraging the serverless paradigm and Functions as a Service (FaaS). You know little about that but need to make a recommendation about whether it is a good fit for our solutions. How would you go about making this type of recommendation having little knowledge of the subject?"

What we are typically looking for here is how the candidate approaches that kind of an open-ended problem that has a significant "**unknown unknown**" component to it. The interviewer is also looking for both breadth and depth.

Does the candidate have an organized plan and strategy for learning new technologies? Are they paying attention to both the upsides and downsides?

Is the candidate driving innovation without sacrificing practicality?

Is the candidate willing to introduce new technologies and open to doing so? Are they keen on relaying what they have learned to others?