Site Reliability Engineer interview questions

careertips.com.au: common questions and what they're really testing

What to expect

Interviews for Site Reliability Engineer roles mix hands-on technical questions with incident narratives, because the job is judged as much on how someone behaves during an outage as on what they can build day to day. Expect the panel to probe both your systems knowledge and your track record of reducing manual toil and repeat failures.

- **Process:** Questions asking you to walk through how you'd set up monitoring, automation or infrastructure-as-code from scratch.
- **Behavioural:** Past-experience questions about incidents, automation projects or cross-team collaboration, usually answered with a specific example.
- **Scenario:** Live or hypothetical outage situations testing judgement, prioritisation and communication under pressure.
- **Technical:** Direct questions on tools such as Kubernetes, Terraform and Prometheus, and how you'd apply them to a given problem.

Most SRE interviews start with a recruiter screen, followed by a technical round covering tooling and systems design, then a behavioural or incident-focused round with a hiring manager or senior engineer. Some employers add a scenario-based or whiteboard session where you talk through an outage in real time, and a final panel round to check team fit and on-call expectations.

1. Walk me through how you'd design a monitoring and alerting system for a new production service.

Why they ask

This checks whether you think about observability from first principles rather than just bolting on a dashboard, and whether you understand alert fatigue and signal-to-noise.

How to structure your answer

Answer as a step-by-step walk-through: what you'd instrument first, how you'd choose alert thresholds, and how you'd avoid noisy or duplicate alerts.

Example answer

“I'd start by identifying the service's key indicators, latency, error rate and saturation, rather than instrumenting everything at once. I'd set up Prometheus to scrape those metrics and build dashboards around them before writing a single alert rule. For alerting, I'd tie thresholds to what actually affects users, using error budgets rather than arbitrary numbers, and route anything non-urgent to a ticket queue instead of paging someone at 3am. I'd also review alert volume after a few weeks and tune out anything that fires without needing action.”

2. Tell me about a time you led a post-incident review after a major outage.

Why they ask

Post-incident reviews are core to the role, and the panel wants to see whether you focus on system fixes rather than blame.

How to structure your answer

Use STAR: describe the situation and outage, your task in the review, the actions you took to find root cause and assign fixes, and the result in terms of prevented recurrence.

Example answer

“A deployment change caused a cascading failure that took a service offline for close to an hour. I ran the post-incident review, keeping the discussion focused on the timeline and contributing factors

rather than who pushed the change. We traced it to a missing health check in the rollout pipeline. I assigned owners to each preventative action, added the health check gate to the deployment process, and followed up two weeks later to confirm it had shipped. We didn't see that failure mode again."

3. You get paged at 2am for a service outage affecting customers. Talk me through how you'd approach it.

Why they ask

This tests judgement under pressure: whether you stabilise first, communicate clearly and escalate appropriately rather than trying to fix everything alone.

How to structure your answer

Answer as a judgement-under-pressure walk-through: immediate triage actions, who you'd notify and when, and how you'd decide between a quick mitigation and a full fix.

Example answer

"First I'd check the dashboards to confirm scope and impact, and acknowledge the page so the team knows it's being handled. If there's an obvious mitigation, like rolling back a recent deployment, I'd do that before digging into root cause, since restoring service matters more than understanding it at that point. I'd post a short status update to the incident channel and pull in another engineer if it's not resolving quickly. Once service is stable, I'd hand over notes for the post-incident review rather than trying to close everything out at 3am."

4. How do you decide on service-level objectives with a development team, and what happens when they're breached?

Why they ask

SLOs sit at the centre of the SRE role and this checks whether you can translate reliability into terms a product team will actually accept.

How to structure your answer

Explain your reasoning as a framework: how you'd set the target, negotiate trade-offs and respond to a breach.

Example answer

"I'd start from what users actually notice, like latency or availability, rather than an arbitrary uptime figure, and set the SLO with the product team so it reflects a shared risk tolerance. If the error budget gets breached, that's a signal to slow down feature releases and prioritise reliability work until we're back within budget. I've found this works better than treating reliability as a separate team's problem, because the trade-off is visible and agreed upfront rather than argued about after an outage."

5. Describe a time you automated a manual process to reduce toil.

Why they ask

Reducing repetitive manual work is a stated focus of the role, and this checks for genuine hands-on automation experience.

How to structure your answer

Use STAR: the manual process and its cost, your role in automating it, the tools you used, and the measurable reduction in effort or errors.

Example answer

"Provisioning a new environment used to involve a checklist of manual steps that took most of a day and occasionally got done out of order. I wrote Terraform modules to codify the setup and hooked them into our Jenkins pipeline so a new environment could be requested and built without anyone touching the console. It cut setup time down to under an hour and removed the configuration drift we used to see between environments."

6. What's your experience with infrastructure-as-code tools like Terraform, and how do you manage state and rollback risk?

Why they ask

This is a direct technical check on a tool named in the role, and probes whether you understand the failure modes of IaC, not just the syntax.

How to structure your answer

Give a technical explanation grounded in a concrete example: the tool, how you structured it, and how you handled risk.

Example answer

“I've used Terraform to manage cloud infrastructure across multiple environments, keeping state in a remote backend with locking so two people can't apply changes at once. For rollback risk, I favour small, reviewed changes over large ones, and I always run a plan and have someone else check it before applying anything to production. Where the blast radius is large, like network changes, I'll stage the change in a lower environment first and keep the previous state file backed up in case a rollback is needed.”