Full Stack Developer interview questions

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

What to expect

Full stack developer interviews test breadth as much as depth: expect questions on both front-end and back-end work, plus how you handle the messiness of production systems and team collaboration. Employers want to see that you can move across the stack without losing quality anywhere.

- **Technical:** Questions on specific languages, frameworks, databases and cloud tools, sometimes backed by a live coding exercise or take-home task.
- **Behavioural:** Questions about past project work, teamwork and how you've handled setbacks, usually answered with a real example.
- **Scenario/judgement:** Hypothetical situations, such as a production outage or conflicting priorities, testing how you think under pressure.
- **Process:** Questions asking you to walk through how you approach a task, such as designing a database schema or deploying an application.

Most full stack interviews start with a recruiter screen on experience and salary expectations, followed by a technical round that may include a coding exercise, whiteboard system design or a take-home assignment. A final panel interview usually mixes behavioural questions with a deeper technical discussion, sometimes including the hiring manager and a senior engineer.

1. Walk me through how you'd design a database schema for a new feature, say a booking system.

Why they ask

Tests whether you think about data relationships and query patterns before writing code, not just whether you know SQL syntax.

How to structure your answer

Process walk-through: describe the steps in order, starting from identifying entities and relationships, through normalisation decisions, to how you'd handle indexing and expected query load.

Example answer

“I'd start by listing the core entities: users, bookings, and the resource being booked. I'd sketch the relationships, work out whether it's one-to-many or many-to-many, and normalise the schema to avoid duplicate data. Then I'd think about the queries the application will actually run most often, like checking availability, and add indexes on the columns those queries filter on. I'd also plan for concurrent bookings so two users can't double-book the same slot, usually with a database-level constraint rather than relying on application logic alone.”

2. Tell me about a time you had to debug a production issue under time pressure.

Why they ask

Production debugging is a near-daily reality in this role, and employers want evidence you stay methodical rather than guessing.

How to structure your answer

STAR: situation, task, action, result.

Example answer

“A feature that had worked fine in staging started throwing errors for a subset of users in production. I checked the logs first and narrowed it down to a specific input format we hadn't tested. I reproduced it locally, wrote a fix, and added a test case so it couldn't slip through again. I deployed the fix through

our pipeline and confirmed the error rate dropped back to zero within the hour. I also wrote up what happened so the team could see the gap in our test coverage."

3. How would you approach deploying an application to the cloud for the first time, and what would you check before going live?

Why they ask

Confirms hands-on familiarity with cloud platforms and DevOps pipelines rather than just theoretical knowledge.

How to structure your answer

Process walk-through: sequence the steps from environment setup to go-live checks.

Example answer

"I'd set up the environment on AWS, Azure or GCP depending on what the team already uses, provision the database and any storage the app needs, and containerise the application with Docker so it runs consistently. I'd wire up a CI/CD pipeline so deployments are automated and repeatable rather than manual. Before going live I'd check environment variables and secrets are set correctly, run a smoke test against the production environment, and confirm logging and monitoring are in place so we'd know quickly if something broke."

4. You're told a feature needs to ship this week, but you've found a design flaw that will cause problems later if you don't fix it now. How do you handle it?

Why they ask

Tests judgement under pressure and how you balance delivery deadlines against technical quality.

How to structure your answer

Scenario/judgement: state the immediate decision, then the reasoning and trade-offs you'd weigh.

Example answer

"I'd flag the flaw to the team straight away rather than sitting on it, with a clear explanation of what breaks later if we ignore it. Then I'd look for a middle path: is there a smaller fix that ships this week and buys us time, with the fuller fix logged as follow-up work? If there's no safe shortcut, I'd rather have that conversation with the product owner now than explain a production failure later. Most of the time there's a workable compromise once everyone understands the actual risk."

5. How do you keep front-end and back-end work consistent when you're working across both in the same feature?

Why they ask

Full stack roles live or die on whether the two sides of the stack actually integrate cleanly, so this checks for real habits, not just tool familiarity.

How to structure your answer

Process walk-through: describe your working method and how you verify consistency.

Example answer

"I usually define the API contract first, agreeing on request and response shapes before writing either side properly. That way the front-end can build against a mock while the back-end is still in progress. I write integration tests that hit the actual API rather than just unit tests on each side in isolation, since that's where mismatches usually show up. Version control and clear commit messages help too, so if something breaks I can quickly see whether the change came from the front-end or the API."

6. What's your experience working in an Agile team, and how do you handle changing priorities mid-sprint?

Why they ask

Nearly all full stack roles run on Agile delivery, so employers want to see you can work within sprint cycles without derailing the team.

How to structure your answer

STAR, with an emphasis on the outcome for the team rather than just the individual task.

Example answer

“In my last team we ran two-week sprints with daily stand-ups. Midway through one sprint, a critical bug came in from a client that took priority over a feature I was building. I raised it in stand-up, we agreed as a team to pause the feature, and I fixed the bug and got it deployed the same day. The feature slipped by a couple of days, but the team backlog was updated straight away so there were no surprises at the sprint review. Being upfront about the trade-off mattered more than trying to do both at once.”