from the other side of the table ★

40 Questions I Actually Ask When I Interview QA Engineers

Plus what I am quietly listening for in every single answer.

50+ interviews. 7 rounds. Free.

AssertHired

Aston Cook

Read this first

Sixty seconds, then the questions.

These are real. I have run 50+ QA interviews as the person deciding. Every question here is one I have actually asked, grouped the way a real loop runs.

The answers are not scripts. Memorised answers are obvious in about nine seconds. Each question tells you what I am listening for so you can answer in your own words.

Structure beats knowledge. The candidates who get offers are almost never the ones who know the most. They are the ones who sort a messy question into clear parts before answering.

Say "I don't know" out loud. Then say how you would find out. That answer has never once cost someone an offer. Bluffing has, many times.

Practice these out loud. Reading them is revision. Saying them is preparation. The gap between the two is where interviews are lost.

The Screen

First 20 minutes. I am checking how you think, not what you memorized.

01 Walk me through how you'd test a login page.

Listening for: Do you sort into categories (functional, security, UX, edge) or list random clicks? Structure is the whole answer.

02 What is the difference between severity and priority?

Listening for: Severity is technical impact. Priority is business urgency. A logo typo is low severity, high priority.

03 When would you stop testing?

Listening for: There is no clean answer. I want exit criteria, risk coverage, and a decision, not 'when everything passes'.

04 What is the difference between verification and validation?

Listening for: Verification: did we build it right. Validation: did we build the right thing. Bonus if you connect it to requirements.

05 What was the last thing you learned about testing, and where?

Listening for: Curiosity is the only skill that survives every tool change. Vague answers here predict vague answers everywhere.

Manual & Process

Automation people who skip these fundamentals get caught fast.

06 Smoke, sanity, and regression. What is the difference and when do you run each?

Listening for: Smoke: is the build alive. Sanity: did this specific fix work. Regression: did we break anything else.

07 Write me a bug report out loud, right now.

Listening for: Title, steps, expected, actual, environment, severity. If I cannot reproduce it from your words, it is not done.

08 How do you test something with no requirements?

Listening for: Explore, compare against similar products, ask the users, document assumptions. Never 'I would refuse to test'.

09 What is a test scenario versus a test case?

Listening for: Scenario is the what, case is the how. I am checking vocabulary precision, which predicts written communication.

10 How do you decide what to test first when you have two days?

Listening for: Risk. Highest business impact and highest likelihood of failure. Not 'start at the top of the list'.

11 What is boundary value analysis and why does it matter?

Listening for: Bugs cluster at edges. Test min, min-1, max, max+1. If you also mention equivalence partitioning, good sign.

Automation Strategy

This is where I separate test writers from engineers.

12 How do you decide what to automate?

Listening for: Repetitive, stable, high-value regression paths. Not everything. It is a return-on-investment call, and I want to hear tradeoffs.

13 Explain the test pyramid. Do you actually follow it?

Listening for: Many unit, fewer integration, fewest end to end. The honest answer is 'we drift toward too many E2E' and I respect that.

14 Your suite takes 4 hours. What do you do?

Listening for: Parallelize, push checks down to API and unit layers, cut duplicate coverage, tag a fast smoke subset. Not 'buy a bigger machine'.

15 Explain the Page Object Model and one problem with it.

Listening for: Separates page structure from test logic. The problem: page objects bloat into god classes. Naming the weakness scores higher than praising it.

16 What makes a test suite trustworthy?

Listening for: Deterministic results. If people rerun failures out of habit, the suite is already dead. I want to hear that trust is the product.

17 How do you handle test data?

Listening for: Created and cleaned per test, isolated, never dependent on a shared mutable record. Shared state is the top cause of flake.

18 What is your framework's folder structure and why?

Listening for: Any coherent answer works. I am checking whether you designed it or inherited it and never asked why.

19 When is automation the wrong choice?

Listening for: Constantly changing UI, one-off checks, exploratory work, anything needing human judgment. Saying 'never' is a fail.

Tools In Practice

I do not care which tool. I care that you know it deeply.

20 Which locator strategy do you reach for first, and why?

Listening for: User-facing locators (role, label, text) over brittle CSS chains and XPath. I want the reasoning, not the ranking.

21 A test passes locally and fails in CI. Walk me through it.

Listening for: Environment differences, timing, test data, parallelism, headless rendering. Method matters more than the right guess.

22 Explain auto-waiting and why it changed how we write tests.

Listening for: Modern frameworks retry assertions and wait for actionability, which is why hardcoded sleeps are a smell now.

23 What is a headless browser and when would you not use one?

Listening for: No visible UI, faster in CI. Skip it when debugging visually or testing rendering behavior that differs headed.

24 How do you handle authentication across a large suite?

Listening for: Log in once, reuse the storage state or token. Logging in inside every test is the most common suite-killer I see.

25 How do you run tests in parallel without them colliding?

Listening for: Independent test data, no shared fixtures, no ordering assumptions. If tests must run in order, they are not tests.

API Testing

Half of candidates who claim API testing have only clicked Send in Postman.

26 What do you actually assert on an API response?

Listening for: Status code, schema, payload values, headers, and response time. 'I check it returns 200' is not API testing.

27 Explain idempotency and which methods are idempotent.

Listening for: Same request repeated leaves the same state. GET, PUT, and DELETE are idempotent. POST typically is not.

28 401 versus 403. What is the difference?

Listening for: 401: we do not know who you are. 403: we know, and you still cannot. Small question, very revealing.

29 **How would you test an endpoint that has no documentation?**

Listening for: Inspect real traffic, probe methods, validate against observed behavior, then write the contract yourself.

30 **Why test at the API layer when the UI tests already cover it?**

Listening for: Faster, more stable, and closer to the logic. UI tests should cover journeys, not every rule the API already enforces.

Engineering Practice

The part that turns a tester into an SDET.

31 Walk me through your CI pipeline.

Listening for: Trigger, stages, where tests run, what blocks a merge. If you have never seen the config file, say so honestly.

32 A test fails in the pipeline. What do you do first?

Listening for: Read the actual error and the trace. Rerunning first is the answer I hear most and the one I like least.

33 How do you handle a flaky test?

Listening for: Find the cause (timing, isolation, data), fix at the source, quarantine only as a last resort and always tracked.

34 What Git commands do you use in a normal week?

Listening for: Branch, commit, rebase or merge, and ideally bisect or blame when hunting a regression. This exposes real workflow fast.

Behavioral

Technical skill gets you shortlisted. These decide the offer.

35 **Tell me about a bug you missed that reached production.**

Listening for: I want ownership and a systemic fix. 'I have never missed one' ends the interview faster than any wrong answer.

36 **A developer says your bug is not a bug. What do you do?**

Listening for: Evidence, not volume. Reproduce it, show impact, escalate to the requirement. Conflict handling is the real question.

37 **You are asked to sign off but you are not confident. What do you say?**

Listening for: State the risk plainly, give a recommendation, let the business decide. Quality advocacy without being a blocker.

38 **What is the most useful thing you have automated that was not a test?**

Listening for: Reporting, test data setup, environment checks. Great signal for engineering instinct beyond writing test cases.

39 **Teach me something about testing I probably do not know.**

Listening for: Depth check. Anyone can recite definitions. Very few can teach. Prepare one thing you know unusually well.

40 **What questions do you have for me?**

Listening for: Having none reads as low interest every single time. Ask about the suite, the release process, or where quality hurts today.

5 answers that end interviews

Not because they are wrong. Because of what they reveal.

"I would automate everything."

Tells me you have never maintained a large suite.

"We just rerun the flaky ones."

Tells me your team stopped trusting its own tests.

"I have never missed a bug."

Tells me you are either new or not being honest.

"That is the developer's job."

Tells me you see a wall where there should be a team.

"I don't have any questions."

Tells me you want a job, not this job.

One last thing before your next one.

The best candidate I ever interviewed did not answer the most questions correctly.

She asked me what our flakiest test was, and then asked why we had not fixed it.

We talked for eleven minutes about a problem she did not have to care about yet. That is the moment she got the offer, and she never knew it.

Curiosity is the one thing nobody can memorise. ★

Aston Cook · Senior QA Automation Engineer · AssertHired