Bộ 40 câu hỏi phỏng vấn Manual Tester

Fresher · Junior · Middle · Senior — kèm gợi ý đánh giá

Tác giả: Anh Tester
https://anhtester.com

Phần	Số câu	Đối tượng
Phần 1 — Nền tảng lý thuyết	Câu 1–10	Fresher
Phần 2 — Thiết kế & tình huống cơ bản	Câu 11–20	Junior
Phần 3 — Thực chiến & xử lý tình huống	Câu 21–30	Middle
Phần 4 — Process, leadership, chiến lược	Câu 31–40	Senior

Phần 1 — Fresher (Câu 1–10)

Kiểm tra nền tảng lý thuyết. Ứng viên cần trả lời rõ ràng, đúng định nghĩa.

Câu 1. Phân biệt SDLC và STLC.

Mục tiêu, các giai đoạn, và mối quan hệ giữa hai vòng đời. SDLC là vòng đời phát triển phần mềm; STLC là vòng đời kiểm thử, nằm song song bên trong SDLC.

Câu 2. Verification và Validation khác nhau như thế nào?

Verification: "Are we building the product right?" — kiểm tra quy trình, tài liệu, design có đúng chuẩn không. Validation: "Are we building the right product?" — kiểm tra sản phẩm cuối có đáp ứng nhu cầu người dùng.

Câu 3. Test case chuẩn gồm những trường nào?

ID, Title, Precondition, Steps, Expected Result, Actual Result, Status, Priority. Có thể thêm Postcondition, Test Data, Module, Author.

Câu 4. Phân biệt Smoke Test, Sanity Test, Regression Test.

Smoke: build có chạy được không (rộng và nông). Sanity: tính năng mới/fix mới có hoạt động đúng không (hẹp và sâu). Regression: thay đổi mới có làm hỏng tính năng cũ không.

Câu 5. Severity và Priority khác nhau ra sao? Cho ví dụ Severity cao – Priority thấp và ngược lại.

Severity: mức độ nghiêm trọng về kỹ thuật. Priority: mức độ ưu tiên fix về mặt business. Ví dụ Severity cao – Priority thấp: crash khi nhập 10.000 ký tự vào ô tên (hiếm gặp). Ví dụ Severity thấp – Priority cao: sai logo công ty trên trang chủ.

Câu 6. Bug life cycle gồm những trạng thái nào?

New → Assigned → Open → Fixed → Retest → Verified/Reopened → Closed. Các trạng thái phụ: Deferred, Rejected, Duplicate, Cannot Reproduce.

Câu 7. Black box, White box, Gray box testing khác nhau ở điểm gì?

Black box: không biết code, test theo functional spec (tester thường làm). White box: biết code, test logic bên trong (dev hoặc tester có kỹ thuật). Gray box: biết một phần code/architecture, kết hợp cả hai.

Câu 8. Functional vs Non-functional testing — mỗi loại cho 3 ví dụ.

Functional: login, payment, search. Non-functional: performance, security, usability, compatibility, scalability, reliability.

Câu 9. Một bug report tốt cần có gì?

Title rõ ràng và ngắn gọn, Steps to Reproduce chi tiết, Expected vs Actual Result, Environment (OS, browser, version), Severity/Priority, Attachment (screenshot/video/log). Có thể thêm Workaround nếu có.

Câu 10. Test Plan và Test Strategy khác nhau như thế nào?

Test Plan: tài liệu chi tiết cấp dự án — scope, schedule, resource, deliverables. Test Strategy: tài liệu cấp tổ chức, mang tính nguyên tắc, ít thay đổi. Một strategy có thể áp dụng cho nhiều plan.

Phần 2 — Junior (Câu 11–20)

Kiểm tra khả năng áp dụng lý thuyết vào case cụ thể. Câu 11, 12, 19, 20 là các câu sàng lọc mạnh.

Câu 11. Cho ô input "Tuổi" hợp lệ từ 18–60. Viết test case áp dụng Boundary Value Analysis + Equivalence Partitioning.

BVA: 17, 18, 19, 59, 60, 61. EP: hợp lệ (18–60), không hợp lệ thấp (<18), không hợp lệ cao (>60), không hợp lệ định dạng (chữ, ký tự đặc biệt, âm, decimal). Kiểm tra tư duy thiết kế test, không chỉ test happy case.

Câu 12. Liệt kê test case cho chức năng Đăng nhập.

Valid/invalid credentials, SQL injection, XSS, brute force, remember me, case-sensitive, special character, empty field, max length, session timeout, login concurrent từ 2 device, social login, forgot password flow, account locked sau N lần sai.

Câu 13. Dev nói "anh không reproduce được bug này", bạn xử lý thế nào?

Kiểm tra giao tiếp + debug: cung cấp video, log, env (OS/browser/version), data cụ thể, các bước chính xác, ngồi cạnh dev để reproduce. Nếu vẫn không được, ghi chú "Cannot Reproduce" kèm điều kiện đã thử.

Câu 14. Khi nào bạn dừng testing?

Khi đạt exit criteria: hết thời gian theo plan, đạt coverage mục tiêu, bug rate giảm dưới ngưỡng, đã pass các test case critical, stakeholder đồng ý release. Không bao giờ là "test hết bug" vì điều đó không tồn tại.

Câu 15. Mô tả workflow report bug trên Jira.

Tạo ticket với mô tả đầy đủ, gắn label/component, attach evidence, set Severity/Priority, assign cho dev hoặc lead, link với test case/user story, theo dõi sau khi dev fix, retest và đóng ticket.

Câu 16. Test API cơ bản: với mỗi method GET/POST/PUT/DELETE bạn test gì? Liệt kê các status code 2xx, 4xx, 5xx phổ biến.

GET: response data, query params, pagination. POST: tạo mới, validate body, duplicate. PUT/PATCH: update full/partial, idempotent. DELETE: xóa, xóa không tồn tại. Status code: 200 OK, 201 Created, 204 No Content, 400 Bad Request, 401 Unauthorized, 403 Forbidden, 404 Not Found, 409 Conflict, 422 Unprocessable, 500 Internal Server Error, 502 Bad Gateway, 503 Service Unavailable.

Câu 17. Test responsive web cần test gì? Dùng tool nào?

Breakpoint (mobile/tablet/desktop), layout không bị vỡ, touch event, orientation (portrait/landscape), font size, image scale, hamburger menu, sticky element. Tool: Chrome DevTools, Firefox Responsive Mode, BrowserStack, real device.

Câu 18. Cross-browser testing — tại sao cần? Test trên những browser/version nào và dựa trên cơ sở gì?

Cần vì mỗi browser render khác nhau (CSS, JS engine, font). Chọn dựa trên analytics của sản phẩm + market share thực tế (StatCounter, GA), không test bừa. Ví dụ: Chrome 2 version mới nhất, Safari iOS, Edge, Firefox.

Câu 19. Tình huống: deadline còn 1 ngày, còn 200 test case chưa chạy. Bạn xử lý sao?

Risk-based testing: ưu tiên critical flow, high-risk module, regression của feature mới, smoke test toàn bộ. Báo cáo minh bạch với PM/Lead về scope đã/chưa test, đề xuất hoãn release nếu rủi ro cao. Không im lặng và bỏ qua.

Câu 20. Thiết kế test case cho chức năng "Thêm vào giỏ hàng" của 1 trang e-commerce.

Số lượng (âm, 0, max stock, vượt stock, decimal), sản phẩm hết hàng, user chưa login, giỏ trống/đầy, đồng bộ giữa các tab, áp voucher, tính tổng tiền, xóa item, chuyển trang, performance khi giỏ có 100+ items, tiền tệ/đơn vị, sản phẩm đã ngưng bán, giỏ persist sau khi logout.

Tip: *Câu 13 và 19 nhà tuyển dụng đánh giá thái độ và cách giao tiếp nhiều hơn câu trả lời "đúng". Câu 5, 11, 12, 20 là các câu sàng lọc nhanh giữa ứng viên học vẹt và ứng viên có tư duy thực sự.*

Phần 3 — Middle (Câu 21–30)

Middle là level chuyển tiếp: phải vững basic, làm việc độc lập, xử lý tình huống thực tế, có domain knowledge và bắt đầu hỗ trợ Junior.

Câu 21. Bạn vừa nhận build mới lúc 5h chiều, release lúc 9h sáng hôm sau. Bạn sẽ test gì trong 4 tiếng còn lại?

Smoke test trước để chắc build chạy được, sau đó regression các module bị impact bởi change. Không cố cover hết — báo cáo rõ những gì chưa test để stakeholder quyết định.

Câu 22. Phân biệt Retesting và Regression Testing. Khi suite regression quá lớn, bạn chọn case như thế nào?

Retest: test lại đúng bug đã fix để confirm fix đúng (luôn manual, target). Regression: test lại các chức năng liên quan để chắc fix không gây side effect. Khi suite quá lớn: dùng impact analysis (module nào bị change), risk-based selection (critical flow ưu tiên), automation cho regression ổn định, manual cho phần thay đổi nhiều. Không chạy hết "cho an tâm" — tốn thời gian mà ROI thấp.

Câu 23. Bạn tìm thấy 1 bug nghiêm trọng nhưng PM bảo "không fix, release đi". Bạn xử lý thế nào?

Document rõ risk bằng văn bản (email/Jira), escalate đúng kênh nếu cần (Tech Lead, QA Manager), không tự ý quyết. Ghi nhận quyết định để có evidence sau này. Không cãi gay gắt, không im lặng bỏ qua.

Câu 24. Test 1 form upload file: bạn nghĩ ra bao nhiêu test case?

Mong đợi 15+ case: file đúng định dạng, sai định dạng, file rỗng, file quá lớn/quá nhỏ, tên file có ký tự đặc biệt, tên file dài, file bị corrupt, upload nhiều file cùng lúc, mất mạng giữa chừng, file trùng tên, đường dẫn UNC, file ẩn, file virus (eicar), MIME type giả mạo, đổi extension, upload song song, cancel giữa chừng.

Câu 25. Test case cho ô tìm kiếm (search box) trên 1 trang web.

Empty search, 1 ký tự, ký tự đặc biệt, SQL injection, XSS, từ khóa không tồn tại, từ khóa tiếng Việt có dấu/không dấu, case sensitive, max length, autocomplete, copy-paste, leading/trailing space, recent search, performance với keyword dài, search trong khi đang load.

Câu 26. Bug bạn report bị dev đánh "Not a bug / Rejected". Bạn làm gì?

Đọc lại requirement/spec, hỏi BA để confirm expected behavior. Nếu đúng là bug → reopen kèm bằng chứng và reference đến spec. Nếu sai → học từ đó, không cãi cố. Đôi khi spec chưa rõ → đề xuất clarify với cả team.

Câu 27. Sự khác nhau giữa Test Scenario và Test Case? Khi nào nên viết scenario thay vì test case chi tiết?

Scenario: "what to test" — mức cao, 1 dòng mô tả end-to-end flow. Test Case: "how to test" — chi tiết step-by-step có expected result. 1 scenario thường sinh ra nhiều test case. Khi nên dùng scenario: dự án Agile chạy nhanh, tester có kinh nghiệm và domain, exploratory testing, prototype thay đổi liên tục. Khi cần test case chi tiết: dự án regulated (finance/medical), team có nhiều fresher, cần audit trail, outsource testing.

Câu 28. Bạn được giao test 1 tính năng nhưng không có tài liệu (requirement/spec). Bạn làm gì?

Exploratory testing, hỏi BA/PM/Dev, tham khảo sản phẩm tương tự (competitor), dựa vào common sense + user perspective. Document lại assumption đã đưa ra và confirm với team trước khi bắt đầu test chính thức.

Câu 29. Performance, Load, Stress, Spike Testing khác nhau ra sao? Tool và metric chính bạn quan tâm?

Performance: đo tốc độ/độ ổn định trong điều kiện bình thường. Load: test với lượng user dự kiến để xem có chịu được không. Stress: đẩy quá giới hạn để tìm breaking point. Spike: tăng tải đột ngột rồi giảm, xem hệ thống có ổn định lại không. Tool: JMeter, K6, Gatling, LoadRunner. Metric quan trọng: response time (avg, p95, p99 — không chỉ avg vì avg đánh lừa), throughput (req/s), error rate, CPU/RAM/DB connection pool. Middle cần biết "p95 quan trọng hơn average".

Câu 30. Tình huống: phát hiện bug có thể là lỗ hổng bảo mật (ví dụ thấy được data user khác). Bạn xử lý thế nào?

Không screenshot data thật, không share trên channel công khai. Report riêng cho Security Lead/PM, mô tả bằng dummy data, đánh dấu confidential. Đây là câu test sense về security và ethics — rất quan trọng cho công ty fintech/healthcare.

Phần 4 — Senior (Câu 31–40)

Tư duy hệ thống, leadership, strategic thinking. Câu trả lời quá trơn tru = red flag, cần follow-up bằng ví dụ thực tế.

Câu 31. Bạn được giao build QA process cho 1 startup chưa có quy trình test. Bạn bắt đầu từ đâu?

Audit hiện trạng (dev flow, tool, team size), define test strategy phù hợp scale (không bê nguyên enterprise process vào startup), chọn tool (Jira/TestRail/Zephyr), setup bug workflow, định nghĩa entry/exit criteria, KPI, training team. Quan trọng: biết cân bằng giữa "đúng quy trình" và "đủ dùng".

Câu 32. Định nghĩa Test Strategy cho 1 sản phẩm mobile banking. Bạn sẽ ưu tiên gì?

Risk-based approach: security (OWASP Mobile Top 10), transaction integrity, compliance (PCI-DSS), performance dưới mạng yếu, device fragmentation, offline mode, biometric, rooted/jailbroken detection. Đánh giá khả năng tư duy theo domain.

Câu 33. Làm sao bạn đo lường hiệu quả của team QA? KPI/metric nào dùng và metric nào tránh?

Nên dùng: Defect Leakage, Defect Removal Efficiency (DRE), Test Coverage, Escaped Defects, MTTR. Tránh: số bug raised/tester (sinh ra gaming metric), số test case viết được (số lượng ≠ chất lượng). Senior phải hiểu metric sai → behavior sai.

Câu 34. Shift-left testing là gì? Bạn đã áp dụng trong dự án thực tế thế nào?

Test sớm từ requirement review, tham gia grooming, viết acceptance criteria cùng BA, review design, pair với dev. Mong đợi ví dụ cụ thể, không lý thuyết suông. Lợi ích: phát hiện defect sớm → chi phí fix thấp hơn.

Câu 35. Khi nào nên automate, khi nào nên giữ manual? Cho 1 case cụ thể bạn quyết định KHÔNG automate.

Không automate khi: UI thay đổi liên tục, test chạy 1 lần, exploratory, usability, ROI thấp, chi phí maintain > giá trị. Câu này lọc senior thật vs senior "ngộ nhận" — junior hay nghĩ cái gì cũng nên automate.

Câu 36. Bạn mentor 1 fresher liên tục miss bug ở production. Bạn xử lý thế nào?

Không đổ lỗi: phân tích root cause cùng họ (thiếu kiến thức domain? test design yếu? thiếu checklist? môi trường test khác prod?). Pair testing, review test case của họ, build checklist chung cho team. Đánh giá kỹ năng leadership.

Câu 37. Production có bug nghiêm trọng escape khỏi QA. Bạn làm Root Cause Analysis như thế nào?

5 Whys, không tìm người đổ lỗi mà tìm gap trong process: bug nằm trong scope test không? Test case có cover? Nếu có thì sao không phát hiện? Environment khác prod? Data khác? Sau đó: update test suite, update checklist, share lesson learned cho team.

Câu 38. Bạn đánh giá 1 ứng viên Junior trong phỏng vấn kỹ thuật như thế nào? 3 câu hỏi bạn luôn hỏi?

Đảo vai — xem ứng viên có khả năng tuyển người không. Câu hỏi tốt thường xoay quanh: tư duy thiết kế test case (vd: test cái thang máy), thái độ xử lý conflict với dev, khả năng học (kể về công nghệ mới nhất bạn học).

Câu 39. Estimation: PM hỏi "test feature này mất bao lâu?". Bạn trả lời thế nào và dựa trên cơ sở nào?

Không nói đại 1 con số. Break down: số test case dự kiến × thời gian/case + buffer cho retest/regression + setup environment + report. Dùng historical data của team. Đưa ra range (best/worst case) thay vì 1 số cứng. Communicate assumption rõ ràng.

Câu 40. Bạn và Dev Lead bất đồng việc 1 ticket là bug hay enhancement. Cả hai đều có lý. Bạn xử lý ra sao?

Không phải câu test "ai đúng" mà test khả năng giải quyết conflict: quay về source of truth (requirement, acceptance criteria, user story). Nếu spec không rõ → escalate lên BA/PO để clarify, đề xuất update spec để không lặp lại. Không cá nhân hóa, không win/lose mindset.

Lưu ý khi phỏng vấn Senior: *Câu trả lời "đúng sách" thường là red flag. Senior thật phải có ví dụ cụ thể từ dự án, biết cả mặt trade-off, dám nói "tôi đã sai và học được X". Nếu ứng viên trả lời quá trơn tru như đọc tutorial, đào sâu bằng follow-up: "Cho 1 ví dụ thực tế bạn đã làm việc đó".*