Scenario-Based Manual Testing Q&A

(30 Questions & Answers for 10+ Years Experience)

Scenario-Based Manual Testing Q&A

Q1: 1. You are testing a banking application and the application crashes when you try to transfer money above 10 lakhs. How will you handle and report this issue?

A: As a senior tester, the first step is to reproduce the issue consistently and gather all details such as test data, environment, and steps to reproduce. I would check whether the business requirement allows transactions above 10 lakhs.

- If the requirement supports transactions above 10 lakhs, then it is a critical defect because it impacts real business flow. I would report it as 'High Severity & High Priority' defect with detailed logs, screenshots, and possibly database entries if relevant.
- If the requirement restricts it, then the application should ideally show a user-friendly error message instead of crashing. In that case, I would raise a defect related to 'Error Handling & User Experience'.

Finally, I would escalate it in defect triage and suggest improvement in validation to avoid application crash.

Q2: 2. In a production release, you find that a core feature is failing, but it was not covered in test cases. How will you address this?

A: First, I would analyze why the test case was missed - whether due to requirement gap, oversight in test design, or unclear specifications. I would immediately log the defect and inform stakeholders with its business impact.

As a corrective action, I would add regression scenarios to cover the missed functionality. I would also recommend strengthening requirement traceability by maintaining a Requirement Traceability Matrix (RTM) to ensure all requirements are mapped to test cases. Additionally, I'd propose a review process improvement so that critical features are not missed in future.

Q3: 3. If developers reject your bug saying 'works as designed' but you believe it impacts end-users, how will you proceed?

A: I would not stop at the rejection. I would revisit the requirements, acceptance criteria, and business cases. If the defect genuinely impacts usability, I would provide practical end-user scenarios, attach supporting evidence like screenshots, videos, or logs, and re-justify the defect with business perspective.

If there's still disagreement, I would escalate in defect triage with Product Owner/Business Analyst involvement, because customer experience is critical. In my 10 years of experience, I have learned that defects must be validated from both technical and business viewpoints, and escalation with proper evidence usually resolves conflicts.

Q4: 4. You are leading a QA team and deadlines are very tight. Some features cannot be tested completely. How do you decide what to test and what to skip?

A: In such cases, I apply risk-based testing. I evaluate business-critical functionalities, areas with high customer impact, and modules that underwent maximum recent code changes. Those get highest priority.

For low-risk areas, I would perform smoke or sanity checks only. I would communicate clearly to stakeholders about what areas are tested thoroughly and what areas have limited coverage. This transparency helps manage expectations.

Additionally, I would encourage automation or parallel testing where possible to maximize coverage within timelines.

Q5: 5. A defect is reported from production that you already tested and passed in QA. How will you investigate this?

A: First, I would gather information on the production environment: configuration, data, user roles, and exact steps. Many times, defects arise due to environment differences or missing scenarios.

I would try to reproduce the issue in QA with production-like data. If reproducible, it indicates a test gap, and I would immediately create new test cases to cover it. If not reproducible, I would compare configurations and identify mismatches.

As preventive action, I would recommend environment alignment, exploratory testing, and production-like data setup in pre-production. I would also stress the importance of negative and boundary testing to avoid such misses.

Q6: 6. How do you test an application with incomplete or unclear requirements?

A: When requirements are unclear, I rely on exploratory testing, domain knowledge, and discussions with stakeholders. I create assumption-based test cases and validate them with the Product Owner or BA.

I also use techniques like 'Error Guessing' and 'Ad hoc Testing' to uncover hidden issues. Over time, I update the test cases as requirements get clarified. This ensures that testing continues in parallel without waiting, while also keeping flexibility to adapt to requirement changes.

Q7: 7. A new build is delivered just before release and testing time is limited. What testing strategy will you apply?

A: I would first perform a smoke test to ensure core functionalities are working. Next, I would prioritize regression testing for high-risk and high-impact areas. I would involve the team in parallel testing to maximize coverage.

Where possible, I would apply automation for repetitive checks. I would clearly communicate to stakeholders about areas covered and uncovered due to time limits. My strategy will always balance speed with risk mitigation.

Q8: 8. How would you test an e-commerce website's checkout process under heavy load manually?

A: Though performance testing is usually automated, manual testers can simulate basic load testing. I would involve multiple testers to place orders simultaneously with different products and payment methods.

I would observe response times, failures, or incorrect transactions. I would also test edge scenarios like network interruptions, session expiry, and duplicate order attempts. Issues observed are logged with detailed steps and data for performance testing team to reproduce with tools.

Q9: 9. During regression testing, you find that fixing one defect has broken another module. How do you approach this?

A: This is a classic regression failure. I would log the defect and provide detailed information about the broken functionality.

Next, I would analyze the impacted areas and expand regression coverage to similar modules. I would also suggest root cause analysis with the dev team to check why code changes were not isolated.

For future prevention, I would recommend impact analysis, modular coding practices, and better regression automation coverage.

Q10: 10. How do you handle a situation where UAT testers raise issues that were not found during QA testing?

A: I would first review the UAT issues and verify whether they were covered in QA test cases. If not, it indicates a test gap, so I would enhance QA coverage. If already covered but missed, I would analyze execution gaps and strengthen peer reviews.

Sometimes UAT uses different test data or configurations, so I would validate environment parity.

My main focus would be on root cause analysis and process improvements, ensuring such issues are caught earlier in QA next time.

Q11: 11. How would you test a login feature where user enters wrong password multiple times?

A: I would test for account lockout functionality. For example: after 3 or 5 failed attempts, the system should lock the account temporarily or permanently as per requirements. I would also check whether proper error messages are displayed.

Additionally, I would validate security aspects: prevention of brute force, captcha appearance, and reset password options. These scenarios ensure both usability and security requirements are met.

Q12: 12. What steps will you take if defects keep reopening after being marked as 'Fixed' by developers?

A: I would review the communication gap first. Sometimes the defect description is not detailed enough. I would ensure that defects have clear reproduction steps, expected vs actual results, screenshots, and logs.

If reopening continues, I would collaborate closely with developers to reproduce issues together. I would also recommend unit test strengthening and peer code reviews to avoid frequent reopenings.

Q13: 13. How do you ensure maximum coverage when testing a complex application manually?

A: I use multiple approaches:

- Requirement Traceability Matrix (RTM) to ensure all requirements are covered.

- Equivalence partitioning and boundary value analysis for input-based testing.

- Exploratory testing for unexpected scenarios.

- Risk-based prioritization for critical areas.

I also encourage peer reviews of test cases and continuous updates to the regression suite.

Q14: 14. How would you test a mobile application where network connectivity is unstable?

A: I would simulate poor or no network conditions by toggling Wi-Fi, switching to airplane mode, or using weak signal zones. I would check whether the app handles it gracefully by showing proper error messages, retry options, or offline mode.

I would also test scenarios like session persistence after reconnecting. This ensures a smooth user experience in real-world conditions.

Q15: 15. How do you verify data migration from one system to another manually?

A: I would first identify the mapping of source and target fields. Then, I would select sample data sets and validate whether they migrated correctly. I would check counts, null values, formatting, and integrity constraints.

For critical data like financial records, I would verify calculations and totals before and after migration. I would document mismatches and provide evidence with SQL queries, screenshots, or reports.

Q16: 16. If test execution is blocked due to environment issues, what will you do?

A: I would immediately raise an environment blocker and inform stakeholders. Meanwhile, I would utilize time for activities like test case review, preparing new test data, exploratory testing on available modules, or documentation updates.

I would also collaborate with DevOps/support teams to resolve the issue quickly. As a senior tester, it's important to ensure no idle time is wasted.

Q17: 17. How do you test an API manually without automation tools?

A: I can use tools like Postman or even browser-based methods to send API requests manually. I would validate response codes, response time, headers, and payload.

For negative testing, I would send invalid requests, missing parameters, or incorrect methods (GET instead of POST).

I would also cross-verify API responses with database records or UI results for consistency.

Q18: 18. How do you ensure testing quality when requirements keep changing frequently?

A: Frequent changes can destabilize testing. I would maintain a flexible test plan and prioritize testing based on finalized requirements. I would adopt Agile practices like maintaining a living regression suite and updating test cases iteratively.

Communication with stakeholders is key: I would document changes and highlight impact on timelines and coverage. Exploratory testing and automation help in adapting quickly.

Q19: 19. How do you test a payment gateway manually?

A: I would validate different payment modes - credit card, debit card, net banking, UPI, wallets. I would test both successful and failed transactions (wrong CVV, expired card, insufficient balance). I would also verify session expiry during payment, handling of duplicate payments, and refund flows. Security aspects like masking card details are also critical.

Q20: 20. How do you convince stakeholders about defect severity when they see it as low priority?

A: I would demonstrate real business impact with examples. For instance, a UI defect may seem minor, but if it confuses users during checkout, it directly affects revenue.

I would provide screenshots, videos, and user journey impact to make the case. If still not convinced, I would document the risk acceptance decision from stakeholders. This ensures accountability and clarity.

Q21: 21. How do you approach testing when a release involves integration with a third-party service?

A: I would identify integration points and validate request/response formats, error handling, and timeouts. I would use test stubs or mocks if the third-party is unavailable.

I would also test failure scenarios, such as when the third-party service is down. Data consistency and contract validation are critical in such integrations.

Q22: 22. How would you test an application with multiple user roles and permissions?

A: I would prepare a role matrix to ensure all permissions are tested. I would validate both positive (access granted) and negative (access denied) cases.

I would also test scenarios where a user has multiple roles, ensuring least privilege principle is respected. This prevents unauthorized access.

Q23: 23. How do you test a system that heavily depends on configurations?

A: I would create test cases for different configuration combinations. I would validate system behavior when configurations are updated dynamically.

I would also test invalid or missing configurations to ensure the system defaults gracefully instead of crashing.

Q24: 24. How do you test search functionality in an application?

A: I would test different search inputs: valid, invalid, empty, long text, and special characters. I would also validate sorting, filtering, and pagination of results.

Additionally, I would test performance for large result sets and verify relevancy of results.

Q25: 25. How do you ensure test data is realistic and effective?

A: I would use production-like anonymized data wherever possible. I would also generate edge case data for boundary conditions.

I ensure diversity in data sets to cover valid, invalid, and extreme cases. Data preparation is as critical as test design.

Q26: 26. How do you test a reporting module manually?

A: I would validate report generation with different filters and date ranges. I would cross-check report data with database queries.

I would also test export formats like PDF, Excel, and CSV. Performance of report generation is another key aspect.

Q27: 27. How do you test notifications in an application (email/SMS/push)?

A: I would verify trigger conditions, message content, and delivery. For emails, I would test different clients (Gmail, Outlook).

For SMS and push, I would validate delivery on multiple devices and OS versions. I would also check for duplicate notifications.

Q28: 28. How do you test file upload functionality manually?

A: I would test with valid and invalid file types, large files, and corrupted files. I would validate error messages, progress indicators, and security checks (e.g., executable file upload should be blocked).

I would also verify maximum file size limits and simultaneous uploads.

Q29: 29. How do you test a localization feature manually?

A: I would switch application language and validate translations, alignment, and currency/date formats. I would test right-to-left languages like Arabic.

I would also check if untranslated strings exist and whether language settings persist across sessions.

Q30: 30. How do you test a scheduler or cron-based functionality manually?

A: I would configure tasks to run at specific times and verify execution. I would test edge cases like daylight savings time, leap year, and server restarts.

I would also check failure retries and logging of scheduled tasks.