Top Manual Testing Interview Questions and Answers for 2023

Manual Testing Questions

1. Explain what is software testing.

It is the process of analyzing any given piece of software to determine if it meets shareholders' needs as well as detecting defects, and ascertaining the item's overall quality by measuring its performance, features, quality, utility, and completeness. Bottom line, it's quality control.

2. What is quality control, and how does it differ from quality assurance?

Quality control is the process of running a program to determine if it has any defects, as well as making sure that the software meets all of the requirements put forth by the stakeholders. Quality assurance is a process-oriented approach that focuses on making sure that the methods, techniques, and processes used to create quality deliverables are applied correctly.

3. What exactly is manual software testing, and how does it differ from automated software testing?

Manual software testing is a process where human testers manually run test cases, then generate the resulting test reports. With automation software testing, these functions are executed by automation tools such as test scripts and code. The tester takes the end user's role to determine how well the app works.

4. What are the advantages of manual testing?

Manual testing's strengths are:

- It's cheaper
- You get visual feedback that's accurate and quick
- It's ideal for testing minor changes
- It's perfect for ad hoc testing
- Testers don't have to know anything about automation tools

5. On the other hand, what are the drawbacks to manual testing?

Manual testing's weaknesses are:

- Susceptible to human error
- Some tasks may be difficult to accomplish manually, requiring more time to complete
- The cost adds up, so it's more expensive in the long run
- You cannot record the manual testing process, so it's hard to replicate it

6. What kind of skills are needed for someone to become a software tester?

Software testers need skills such as:

- Problem-solving skills
- Excellent written and verbal communication skills

- Detail-oriented
- Able to handle the pressure
- Can work solo or as a team member equally well
- Organizational skills
- Related technical skills

7. Explain what is SDLC.

This is an acronym for Software Development Life Cycle and encompasses all of the stages of software development, including requirement gathering and analysis, designing, coding, testing, deployment, and maintenance.

8. What is a test case?

Test case is used to check whether an application complies with its requirements. It is a documented set of circumstances including prerequisites, input values, and expected outcomes.

Test case answers the questions for how to test.

9. What is a test scenario?

A test scenario is derived from a use case. It's used to test an application's feature from beginning to end. Multiple test cases can be accommodated by a single test scenario. When there is a time constraint during testing, scenario testing comes in handy.

Test Scenario answers the questions for What to test.

10. What is a test plan?

A test plan is a formal document that specifies the scope of testing, the method to be used, the resources needed, and the estimated time to complete the testing process. It is derived from the specifications (Software Requirement Specifications).

11. What is test data?

Test data is information that is used to test software with various inputs and determine whether the resulting output matches the intended result. This data is generated based on the needs of the company.

12. What is a test script?

An automated test case created in any programming or scripting language is known as a test script. These are essentially a collection of instructions for evaluating an application's functionality.

13. What types of manual testing are there? Break them down.

Manual testing is broken down into:

- Black Box
- White Box
- Unit
- Integration
- System
- Acceptance

14. What is black box testing, and what are the various techniques?

Software testers employ black-box testing when they do not know the internal architecture or code structure. The techniques are:

- Equivalence Partitioning
- Boundary value analysis
- Decision Table testing
- State transition testing

15. What is white box testing and its various techniques?

Unlike black-box testing, white box involves analyzing the system's internal architecture and/or its implementation, in addition to its source code quality. It's techniques are:

- Statement Coverage
- Decision Coverage

16. Explain the difference between alpha testing and beta testing.

Alpha testing is at the developer's site before release. Potential clients conduct beta testing at their websites/environment.

17. What's the difference between verification and validation?

Verification evaluates the software at the development phase (**Static Testing-review documents)**, ascertaining whether or not a product meets the expected requirements. On the other hand, validation evaluates the software after the development phase(**Dynamic testing**), making it sure it meets the requirements of the customer.

18. What's a testbed?

It’s not furniture. A testbed is an environment used for testing an application, including the hardware as well as any software needed to run the program to be tested also called test environment.

19. What is Smoke testing?

Smoke testing is done to test the main functionalities of an application.

Also Read: Sanity Testing Vs Smoke Testing

20. Sanity Testing is **a type of testing which is performed on the stable build of the software**. It is a quick and basic test (or set of tests) to ensure that the code changes made are working properly without any bugs.

21. List the four different test levels

The four levels are:

- Unit/component/program/module testing
- Integration testing
- System testing
- Acceptance testing

22. What’s a bug or defect?

- A bug is the consequence/outcome of a coding fault

- Any mismatched functionality found in an application is called Defect/Bug.

- During test execution Test Engineers are reporting mismatches(**expected result vs actual result**) as defects to developers

23. What about the difference between an error and a failure?

The terms: **error, defect, bug,** and **failure** are closely related in software development.

In a nutshell: **An error leads to a defect, and when these defects go undetected, they lead to failure**. An error is a mistake made by a developer in the code.

So, we can say that a mistake made by humans during coding is called error, an error found during the testing phase is called a defect and when a build does not meet its specifications then it is termed as failure.

.

24. What's GUI testing?

This tests the interface between the software and the end-user. Short for Graphics User Interface.

25. When should testing end?

There are a few criteria for ending testing:

- The bug rate has fallen below an agreed-upon level
- The testing or release deadlines have arrived
- The testing budget is out of funds

- A certain percentage of test cases have passed
- The alpha or beta testing periods have ended
- Code, functionality, or requirements coverage have been met at a declared point

26. Why is Software Testing Required?

Software testing is required to ensure the quality and reliability of a software product.

- Testing helps to uncover any bugs, errors, or other issues in the software so that they can be addressed and fixed before the product is released.
- Testing also ensures that the software meets all the requirements specified by the customer and works as expected.
- Finally, testing helps to ensure that the software is secure and can withstand malicious attacks.

27. What are the different levels of testing?

The different levels of manual testing are:

- Unit Testing
- Integration Testing
- System Testing
- User Acceptance Testing

28. What are different types of testing?

Functional Testing :

- Unit, Integration, System, Regression, Smoke, Sanity, Re-testing etc,

Non-functional testing

- Performance Testing
- Security Testing
- Compatibility Testing
- Usability Testing
- Installation Testing

28. Explain the procedure for manual testing?

Manual testing is a process of identifying bugs and errors in a software product without the use of automated tools. The procedure for manual testing is as follows:

- Identify the scope of testing: The first step of manual testing is to identify the scope of testing. The range could be a specific module, functionality, feature, or end-to-end system.
- Design test cases: The next step is to design test cases based on the identified scope. The test cases should include test scenarios, data, expected results, and all other details necessary to perform the tests.
- Execute the test cases: After designing the test cases, the testers execute them to find any discrepancies between the expected and actual results.
- Record the results: While performing the tests, the testers should record the results for further analysis.

29. What is the test case?

A test case is a set of conditions or variables under which a tester will determine whether a software system or one of its features is working as it was originally established for it to do. It may take the form of input, action, or environmental conditions. In addition, a test case includes requirements, test steps, verification steps, prerequisites, outputs, and actual results.

30. What's the role of documentation in Manual Testing?

Documentation is an integral part of manual testing. It is essential to document all steps taken in the testing process to ensure thorough test coverage and accurate results. Documentation provides an audit trail, which can be used to evaluate past test results and identify areas for improvement. Additionally, it is a reference for other testers who may be unfamiliar with the system or application under test.

31. What are the different types of Software testing?

Software testing is classified into two main categories.

1. Functional testing
2. Non-Functional testing

32. Explain Functional Testing

Functional testing is a type of black-box testing. It focuses on the software's functional requirements rather than its internal implementation. A functional requirement refers to the system's needed behaviour in terms of input and output.

It checks the software against the functional requirements or specification, ignoring non-functional characteristics like performance, usability, and dependability.

The purpose of functional testing is to ensure that the software up to snuff in terms of functionality and to solve the difficulties of its target users.

Some of the types of functional Testing are -

- Unit Testing
- Integration Testing
- Regression Testing
- System Testing
- Smoke Testing

33. Explain Non functional testing

Non-functional testing examines the system's non-functional requirements, which are characteristics or qualities of the system that the client has specifically requested. Performance, security, scalability, and usability are among them.

Functional testing is followed by non-functional testing. It examines aspects that are unrelated to the software's functional requirements. Non-functional testing assures that the program is safe, scalable, and fast, and that it will not crash under excessive pressure.

34. Mention a few advantages of Automated testing.

The following are some major advantages of automated testing -

- Automated test execution is quick and saves a significant amount of time.
- Human mistakes are eliminated during testing when test scripts are carefully prepared.
- CI tools like Jenkins, which may also be set to distribute daily test results to key stakeholders, can be used to schedule test execution for a nightly run.
- Automation testing uses a lot less resources.

35. What is Regression Testing?

Regression Testing is a full or partial selection of already executed test cases that are re-executed to ensure existing functionalities work fine.

36. What is Test Harness?

A test harness is a collection of software and test data used to put a programme unit to the test by running it under various conditions such as stress, load, and data-driven data while monitoring its behaviour and outputs.

37. Differentiate between Positive and Negative Testing

Positive Testing	Negative Testing
Positive testing ensures that your software performs as expected. The test fails if an error occurs during positive testing.	Negative testing guarantees that your app can gracefully deal with unexpected user behaviour or incorrect input.
In this testing, the tester always looks for a single set of valid data.	Testers use as much ingenuity as possible when validating the app against erroneous data.

38. What is a Critical Bug?

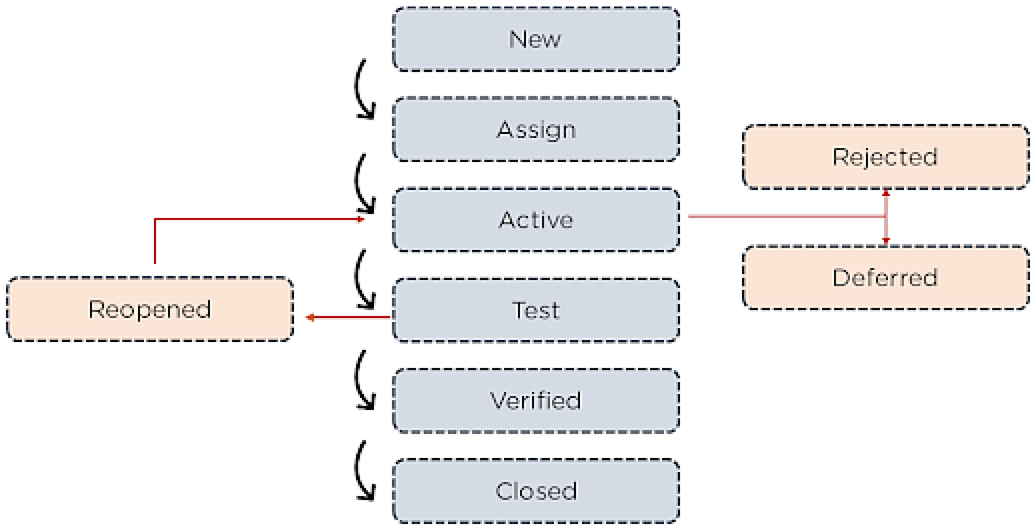
A critical bug is one that has the potential to affect the bulk of an application's functioning. It indicates that a significant portion of functionality or a critical system component is utterly broken, with no way to proceed. The application cannot be delivered to end users until the critical bug has been fixed.

39. What is Test Closure?

Test Closure is a document that summarises all of the tests performed throughout the software development life cycle, as well as a full analysis of the defects fixed and errors discovered. The total number of experiments, the total number of experiments executed, the total number of flaws detected, the total number of defects settled, the total number of bugs not settled, the total number of bugs rejected, and so on are all included in this memo.

40. Explain the defect life cycle.

A defect life cycle is a process by which a defect progresses through numerous stages over the course of its existence. The cycle begins when a fault is discovered and concludes when the defect is closed after it has been verified that it will not be recreated.



41. What is the pesticide paradox? How to overcome it?

According to the pesticide paradox, if the same tests are done repeatedly, the same test cases will eventually stop finding new bugs. Developers will be especially cautious in regions where testers discovered more flaws, and they may overlook Positive and Negative Testing?

other areas. Methods for avoiding the pesticide conundrum include:

- To create a completely new set of test cases to put various aspects of the software to the test.
- To create new test cases and incorporate them into existing test cases.

It is possible to detect more flaws in areas where defect levels have decreased using these methods.

42. What is API testing?

API testing is a sort of software testing that entails evaluating application programming interfaces (APIs) to see if they meet functionality, reliability, performance, and security requirements. Simply put, API testing is designed to detect defects, inconsistencies, or departures from an API's expected behaviour. Typically, applications are divided into three layers:

The user interface is also known as the presentation layer.

For business logical processing, the Business Layer or application user interface is used.

API testing is done at the most vital and important layer of software architecture, the Business Layer, for modelling and manipulating data.

43. What is System testing?

System testing is a type of testing in which the entire software is tested. System testing examines the application's compliance with its business requirements.

44. What is Acceptance testing?

Acceptance testing is a type of testing done by a possible end-user or customer to see if the software meets the business requirements and can be used.

45. Differentiate between bug leakage and bug release

Bug Leakage - When tested software is pushed into the market and the end-user discovers defects, this is known as bug leakage. These are bugs that the testing team overlooked throughout the testing phase.

Bug Release - When a certain version of software is launched into the market with some known bugs that are expected to be fixed in later versions, this is known as a bug release. These are low-priority issues that are highlighted in the release notes when sharing with end-users.

46. What do you mean by Defect Triage?

Defect triage is a procedure in which defects are prioritised depending on a variety of characteristics such as severity, risk, and the amount of time it will take to fix the fault. The defect triage meeting brings together several stakeholders - the development team, testing team, project manager, BAs, and so on – to determine the order in which defects should be fixed.

47. What is Integration Testing? What are its types?

Integration testing is performed after unit testing. We test a group of linked modules in integration testing. Its goal is to identify faults with module interaction.

The following are the types of integration testing -

- Big Bang Integration Testing — After all of the modules have been merged, big bang integration testing begins.

- Top-down Integration Testing — In top-down integration, testing and integration begin at the top and work their way down.
- Bottom-up Integration Testing — In bottom-up integration testing, lower-level modules are tested before moving up the hierarchy to higher-level modules.
- Hybrid Integration Testing — Hybrid integration testing combines top-down and bottom-up integration testing techniques. The integration with this approach starts at the middle layer, and testing is done in both directions.

48. What is a stub?

Many times, when top-down integration testing is performed, lower-level modules are not produced until top-level modules are tested and integrated. Stubs or dummy modules are used in these circumstances to emulate module behaviour by delivering a hard-coded or predicted result based on the input variables.

49. What is code coverage?

The quantity of code covered by the test scripts is referred to as code coverage. It conveys the scope of the test suite's coverage of the application.

50. Explain equivalence class partitioning.

Equivalence class partitioning is a black-box testing technique based on specifications. A set of input data that defines multiple test conditions is partitioned into logically comparable groups in equivalence class partitioning, so that utilising even a single test data from the group for testing can be considered as similar to using all the other data in that group.

52. What is boundary value analysis?

The border values of the classes of the equivalence class partitioning are used as input to the test cases in boundary value analysis, which is a software testing technique for designing test cases.

53. What is your approach towards a severely buggy program? How would you handle it?

In such cases, the best course of action is for testers to go through the process of reporting any flaws or blocking-type issues that arise, with an emphasis on critical bugs. Because this sort of crisis might result in serious issues such as insufficient unit or integration testing, poor design, wrong build or release methods, and so on, management should be contacted and given documentation as proof of the problem.

54. What if an organization's growth is so rapid that standard testing procedures are no longer feasible? What should you do in such a situation?

This is a very prevalent issue in the software industry, especially with the new technologies that are being used in product development. In this case, there is no simple answer; however, you could:

- Hire people who are good at what they do.
- Quality issues should be ‘fiercely prioritised' by management, with a constant focus on the client.
- Everyone in the company should understand what the term "quality" implies to the end-user.

55. When can you say for sure that the code has met its specifications?

Most businesses have coding "standards" that all developers are expected to follow, but everyone has their own opinion on what is best, as well as how many regulations are too many or too few. There are many methods available, such as a traceability matrix, to guarantee that requirements are linked to test cases. And when all of the test cases pass, that means the code satisfies the requirement.

56. What is the difference between manual testing and automation testing?

Manual testing is the process of manually testing software for defects. It requires a tester to manually execute the test steps and compare the actual and expected results. Automation testing uses special software to control the execution of tests and compare the results with the desired results. As a result, automation testing is much faster than manual testing and can reduce the time required to complete a test cycle.

57. When should you opt for manual testing over automation testing?

Manual testing should be used over automation testing when the tests are particular or require human interpretation or when our tests cases cannot be automated.

58. What are the phases involved in the Software Testing Life Cycle?

The phases involved in the Software Testing Life Cycle are:

- Test Planning
- Test Analysis
- Test Design
- Test Implementation
- Test Execution
- Test Results Analysis
- Test Closure

59. What makes a good test engineer?

A good test engineer is detail-oriented and organized, has excellent problem-solving skills, and can produce high-quality work quickly and efficiently. And also should have strong communication and collaboration skills and be an outstanding team player. They also need to be up to date on the latest technologies and software trends and be able to apply them to their testing process.

60. What is the difference between system testing and integration testing?

System testing is a type of software testing that evaluates a complete and fully integrated software product. It verifies that the software meets the requirements specified in the design and the system-level technical specifications. System testing also identifies any weaknesses, errors, or bugs.

Integration testing is software testing that verifies the interactions between two or more system components. It is performed after unit testing and before system testing. It checks how components interact with each other and how they fit together. Integration testing is necessary to ensure that the components of the system work together as expected.

61. What does the term ‘quality' mean when testing?

Quality in testing refers to the degree to which a product meets its intended requirements, as well as the degree to which it satisfies customer needs and expectations. It includes both the functional and non-functional aspects of the product. Quality assurance is ensuring that the product meets its requirements, while quality control focuses on testing to ensure that the product meets its needs.

63. What are the Experience-based testing techniques?

Experience-based testing techniques include:

- Exploratory Testing
- Error Guessing
- Adhoc Testing
- Checklist-based Testing
- etc

64. What is a top-down and bottom-up approach in testing?

A top-down and bottom-up approach in testing refers to the order of testing.

- Top-down testing begins at the highest level and works downward. Thus, each higher-level component is tested in isolation from the lower-level components.
- Bottom-up testing starts at the lowest level and works upward. Thus, each lower-level component is tested in isolation from higher-level components.

65. What is the difference between smoke testing and sanity testing?

- Smoke testing is a high-level test used to ensure the most critical functions of a software system are working correctly. It is a quick test that can be used to determine whether the major functionality is stable or not.
- Sanity testing is a more specific test used to check that recent changes to a system have not caused any new, unwanted behavior. It ensures that basic features are still functioning as expected after minor changes have been made.

66. What is the difference between static testing and dynamic testing?

- Static testing is a type of testing performed without executing the code of a software application. Instead, it includes reviews, inspections, and walkthroughs.
- Dynamic testing is a type of testing that involves executing the code of a software application to determine the results of certain functions and operations. It includes unit testing, integration testing, and acceptance testing.

67. How will you determine when to stop testing?

When testing, it is vital to determine when to stop to prevent wasting resources. When deciding when to stop testing, then you should consider the following criteria:

- Desired levels of quality
- Adherence to timelines and budget

- Number of defects found
- Number of test cases that have been completed
- Risk factors associated with the project

Once these criteria have been met, you can stop your testing.

68. How do you test a product if the requirements are yet to freeze?

When requirements are yet to freeze, the best approach is to use an agile development methodology, such as Scrum.

- The first step would be to hold requirements gathering meetings with all stakeholders to understand the product’s purpose and desired outcomes. The next step would be to break up the project into individual, manageable user stories.
- From there, we would prioritize the user stories and assign them to sprint for development.
- As the project progresses, we continually test the product using techniques such as unit tests, integration tests, user acceptance tests, and system testing. In addition, as requirements change, we will update our tests to ensure the product meets the desired outcomes.

69. What are the cases when you'll consider choosing automated testing over manual testing?

The following steps are the considering cases:

- When the test requires repetitive steps:

Automated testing is ideal for running tests requiring multiple iterations or repeating the same actions repeatedly.

- When the test requires a large amount of data:

Automated testing can quickly insert large amounts of data into the tested system.

- When the test requires multiple environments:

Automated testing can easily be configured to test systems in various domains, such as multiple operating systems, browsers, and devices.

- When the test requires precise timing:

Automated tests can be programmed to run precisely, ensuring that each test step is performed at the exact time it needs to be.

- When the test requires multiple users:

Automated testing can simulate multiple users accessing the system simultaneously, allowing for more realistic testing.

70. What is ‘configuration management'?

Configuration management is managing, tracking, and controlling changes to a system's software, hardware, or network configuration. Configuration management can maintain the integrity of a system and ensure that it is secure, stable, and compliant with organizational policies. The primary goals of configuration management are to ensure system reliability, maintain system availability, and improve system performance.

71. What are some best practices that you should follow when writing test cases?

Here are the top 10 best test case practices:

- Develop test cases that are clear, concise, and to the point.
- Ensure that the test cases challenge the software's functionality in all dimensions.
- Make sure that the test cases cover all the requirements.
- Develop repeatable test cases that can be automated when necessary.
- Develop test cases that are independent of each other.

- Use meaningful and descriptive names for test cases.
- Record the results of test cases for future reference.
- Make sure that the test cases are modular and can be reused.
- Perform reviews of the test cases to ensure accuracy and completeness.
- Document the test cases in a standard format.

73. Why is it that the boundary value analysis provides good test cases?

Boundary value analysis provides suitable test cases because it ensures that the boundaries of input and output values are tested, making it easier to identify edge cases. Testing these edge cases ensures that your system is robust and can handle any unexpected input or output values.

74. Why is it impossible to test a program thoroughly or 100% bug-free?

It is impossible to test a program thoroughly or 100% bug-free because it is impossible to anticipate and test every possible combination of inputs, environments, and states a program might encounter.

75. Can automation testing replace manual testing?

No, automation testing cannot fully replace manual testing. Automation testing is designed to supplement manual testing, not replace it. Automation testing can automate repetitive, tedious test cases and make the testing process more efficient. However, it cannot replace manual testing completely, as some tests can only be performed manually.

For example, exploratory testing and user experience testing are all tasks that require manual testing.