Software Quality Assurance Interview Questions

**Question-1. What are the main differences between Quality Assurance, Quality Control, and Software Testing?**

**Ans.**

**Software Quality Assurance** encompasses the whole development process, monitoring/improving the process, ensures the adherence to the agreed-upon standards and procedures. And it guarantees the discovery of bugs and validation of their fixes.

**Quality Control** sets up the process to determine bugs and sharing feedbacks to raise the quality of the application under test. It’s probably the testing team which carries out the quality control process.

**Software Testing** is again a process which begins with discovering bugs and ends with the final validation of the bug-fix. It also confirms the product meets the customer criteria and the procedures set by the organization.

**Question-2. What is the right time to commence the QA activities?**

**Ans.**Ideally, we should begin QA activities as early as the project kicks off. There are many advantages of a timely start like we get more time to set standards for achieving the quality.

Any delay in QA roadmap directly impacts the cost, time, and efforts for a project.

**Question-3. What are the differences between a Test Plan and a Use Case?**

**Ans.** A **Test Plan** comprises of all the elements that form the test strategy. It includes the application getting tested, the scope of testing, the QA activities, roles/responsibilities and deliverables.

Whereas a **Use Case** standardizes the way user reacts in a situation and how the system responds. It results in the creation of diagrams, flowcharts, and the pre/post conditions.

**Question-4. What are the differences between a Test Plan and Test Strategy?**

**Ans.** **Test Strategy** is a high-level plan usually initiated by software quality assurance manager. It captures all the primary testing activities. It is basically for the stakeholders who have high influence but low interest in the project.

Whereas a **Test Plan** provides more details on how the testing should take place in the entire project lifecycle.

**Question-5. How does the severity of a bug differ from its priority?**

**Ans.** The **priority** defines the urgency of a bug to get fixed. There are factors to consider e.g. Fix is affecting some other functionality or the bug is blocking a certain feature.

The **severity** depicts the impact of a defect on the application. It determines the level of damage that could affect the integrity of the data in the system.

[**Back to top**](http://www.techbeamers.com/software-quality-assurance-interview-questions/#top)

**Question-6. What is a good test case?**

**Ans.** A **test case** is a sequence of operations controlled by constraints to determine an application, product or service is working correctly to the expectations.

Make sure it covers all the test conditions and provides a proper definition.

Next, read the below examples and see the difference between a bad and good test case.

* **Bad test case.**
  + Verify if login works.
* **Good test case.**
  + Open the login screen, enter the username – xyz@testmail.com and password as “12345678”.
  + Press the login button.
  + Verify if the user dashboard is visible in not more than 2 seconds.

**Question-7. What are the data points you should add while writing a good test case?**

**Ans.** Following are the qualities that constitute a **good test case**.

1. Test case ID  
2. Test description  
3. Priority  
4. Severity  
5. Environment  
6. Product version  
7. Steps required  
8. Expected output  
9. Actual output

**Question-8. When do you decide to stop testing?**

**Ans.** Usually, it’s the test plan which covers the ballpark estimates of the testing cycles. But applications are complex in nature. And it’s not easy to predict the exact time to complete the testing.

However, below are the indicators that signal to stop or halt testing.

1. Deadlines (Release or Testing timelines)  
2. Test case completion with expected pass percentage.  
3. Testing cost overruns.  
4. Attained the desired coverage of code/functionality/requirements.  
5. Defect discovery rate falls below the threshold.

**Question-9. Why is it not easy to convince the top stakeholders to invest in extending or advancing the Software quality assurance process?**

**Ans.** It’s not easy for the software quality assurance manager to attract more investment for the advancement of the process. The reason is evident, read the below famous one-liner.

*Solving Problem is a high-visibility process; Preventing them is low-visibility.*

There are also other financial constraints that don’t allow spending more funds on the testing front. However, if you can demonstrate the benefits with some real-time data, then it’s possible to make ways.

**Question-10. What if QA starts involving in digging out production issues?**

**Ans.** It’s a good practice for QA to check on production issues. They will get to learn the gaps that could have possibly led the bug to leak through the QA gates.

However, such issues occur mostly due to environment mismatch. A small tweak in registry setting, resetting browser cache or restarting the backend service fixes the problem.

By lending support to production, QA can expect to find out untested areas and add them in their test plan. So it’s a worth the exercise which will ensure quality in the long term.

[**Back to top**](http://www.techbeamers.com/software-quality-assurance-interview-questions/#top)

**Question-11. What are the first few things a QA manager should do after joining a startup?**

**Ans.** A successful QA needs resources be it a human resource or hardware or the funds to acquire them. All of it comes with proper planning and timely execution.

However, things are a bit unorganized, especially in a small-scale startup. And that’s where the real test of a QA manager begins. He has to perform in a short time and with limited resources.

Here are a few action items for the QA manager to start.

* Start work on preparing a solid test plan.
* Outline the test areas, write down high-level use cases.
* Split use cases into test cases, add pass/fail criteria and set priority.
* Use a spreadsheet or any open source tool like TestLink to manage test cases.
* Set up a tool like Bugzilla or JIRA for defect management.
* Split up manual testing and organize efforts.
  + Functional testing
  + Acceptance testing
  + Integration testing
  + System testing
* Consider testing on multiple platforms, browsers, and devices.

**Question-12. Let’s say you have a test plan with hundred of test cases. Then, how would you decide what tests to automate and what to cover in manual testing?**

**Ans.** This type of questions tests a candidate’s decision-making skills. It’s a test whether you can prioritize a test case or assess if it is feasible to automate the test case.

However, answering the following questions could help you take the decision. You’ll probably automate in the below situations.

* Identify cases which are tedious and require a lot of manual efforts.
* Scenarios which you’ve not covered in the past.
* Select cases which need to run on different browsers/devices or platforms.
* Pick those features that don’t change so frequently else you won’t automate.
* Exclude test cases which require explorative testing or verify the user experience.

**Question-13. As a QA tester, who will you report or submit your test cases?**

**Ans.** There are a no. of people involved in the QA process. Also, it’s better for your test cases to cross through many eyes.

Hence, you can refer them to the following.

1. To Test Lead for getting them reviewed.  
2. To BA for making sure the test coverage.  
3. To DEV team for ensuring the functionality coverage.

**Question-14. What if you have to improve the QA process, how will you do it?**

**Ans.** A software quality engineer can take a no. of steps to improve the QA process.

1. Introduce a web-based test management tool like JIRA.  
2. Create an automated process for reviewing the test cases.  
3. Assess test coverage and share with people concerned.  
4. Use statistical formulas to provide bug discovery rate/testing efforts.

**Question-15. What are the documents you may refer while writing the test cases?**

**Ans.** Following is the list of documents that helps in writing test cases.

1. Requirement specs doc  
2. Feature description doc  
3. User stories description  
4. Use cases description  
5. Acceptance criteria

[**Back to top**](http://www.techbeamers.com/software-quality-assurance-interview-questions/#top)

**Question-16. What is the difference between functional and nonfunctional testing?**

**Ans.**

**1. Functional testing** takes care of the validation of features in an application. It ensures that the application is behaving in the expected manner.

It also confirms the application meets the customer requirements. Every such requirement has related test cases and each test case should pass before exiting the functional testing

Regression, Smoke, Integration and System testing are types of Functional testing.

**2. Non-functional testing** focuses on the performance, load and stress testing of the application.

In this, QA gathers the data, assess and find the benchmark values. Sometimes, this data is critical for the release approvals.

**Question-17. What is Negative Testing and how is it different from Positive Testing?**

**Ans.**

**1. Negative Testing** is an approach to find how does the system behave with invalid inputs.

It ensures that the application shows correct error codes or messages. And the system does not crash on feeding invalid data.

**2. Negative Testing** differs from the Positive Testing in the sense as the latter confirms the system works as expected. Whereas the Negative Testing does the opposite.

It is for making sure that the system would work even if a novice handles it without prior knowledge.

**Question-18. What are the leading testing techniques for QA? And what is their purpose?**

**Ans.**

Testing techniques primarily fulfill the following two purposes.

1. To help identify defects.  
2. To reduce the number of test cases.

Here are five of the leading testing techniques that QA can use.

**1. Equivalence partitioning.**

Its purpose – Reduce the number of test cases. It works by identifying different sets of data. Each set has unique data values. One test from each set of data gets executed.

**2. Boundary Value Analysis.**

Its purpose – Check the behavior of the system at the boundaries of allowed data.

**3. State Transition Testing.**

Its purpose – Verify the allowed and disallowed states and transitions from one state to another by various input data.

**4. Decision Table Testing.**

Its purpose – Work with the different combination of inputs and their associated outputs.

It is a black box test design technique to determine the test scenarios for complex business logic.

**5. Pairwise Testing.**

Its purpose – Reduce the number of test cases while increasing the coverage of feature combinations.

[**Back to top**](http://www.techbeamers.com/software-quality-assurance-interview-questions/#top)

**Question-19. What are the leading Testing Approaches for QA?**

**Ans.**

**1. Black box testing.**

In this approach, a tester performs testing without looking into the code. It’s a testing method where we treat the application under test as a black box. The internal behavior of the program is out of scope for this testing. It works upon the external specifications. Alternatively, we call it as behavioral testing as it examines only the external behavior of the program.

**2. White box testing.**

In this approach, a tester looks into the code and performs the testing.

**3. Unit testing.**

In this approach, a tester checks the code whether it is properly working or not as per the requirement. The developers perform this testing.

**4. Integration testing.**

It ensures that all the modules are works together with success.

**5. Functional testing.**

It checks that each and every functionality of the application is working as expected. It covers the majority of the tests, amounts to 80% of the overall efforts.

**6. System testing.**

Confirms the overall requirements specification. And it covers all combined parts of a system. It is one of the black box types of testing.

**7. Stress Testing.**

It tests the application against a heavy load.

**For example –** Complex numerical values or a large number of inputs or queries etc.

It confirms that application can withstand the stress.

**8. Smoke testing.**

Puts an initial set of test to check whether the major functionalities are working or not.

**9. Regression testing.**

It makes sure that the newly added functionality does not break the application. We also do regression testing after the developers fix the bugs.

**10. End-To-End Testing.**

Tests the complete application in a situation that imitates the real-world use.

**For example –** Interacting with a database or using network communications or interacting with other hardware, applications, or systems if appropriate.

**Question-20. What are the different verification techniques available for QA?**

**Ans.** Usually, the QA engineers use the following three verification methods.

**1. Review.**

In this method, a third-party runs the test cases instead of the one who produced them. It is the easiest of verification tools to ensure coverage and quality.

**2. Inspection.**

It’s a way to study and fix the defects in the test artifact or source code. It has various roles to support.

* Moderator – runs the entire inspection meeting.
* Recorder – prepares the MOM (minutes of meeting).
* Reader – reads out the document.
* Producer – updates the document as per the comments.
* Reviewer – the whole team plays the reviewer’s role.

[**Back to top**](http://www.techbeamers.com/software-quality-assurance-interview-questions/#top)

**Question-21. How do you manage if you face doubts during a project?**

**Ans.** If you have doubts, then, first of all, try to sort them out by reading the available documents. If the doubts remain, then discuss with your supervisor or a senior member of your team.

Also, don’t mind going to BA for any assistance. Alternatively, you can contact the Dev team for any doubts.

Finally, meet up with your manager and close all of your doubts.

**Question-22. What approach do you follow to measure the testing efforts in a project?**

**Ans.** The most common technique to measure the testing efforts is to find out the Cyclomatic complexity of a module or a feature.

It helps in calculating the precise estimates to validate the parts of an application.

There are these three questions which you get answers by using this method.

1. Whether the module or the feature is testable?  
2. Whether the module or the feature is understandable for everyone?  
3. Whether the module or the feature is sustainable?

With this tool, a software quality engineer can quickly check the level of complexity level. Please note the following facts about this method.

1. If the result of this tool is a large number, then the code or feature requires a deep level of testing.  
2. If the outcome is a medium number, then you should test to a moderate level.  
3. If the result is a small number, then just runs some basic tests.

**Question-23. What causes a Software to have bugs?**

**Ans.** There are a no. of reasons that could lead a software to have bugs or defects. But a majority of them occurs due to assumptions. Here is a famous quote supporting this fact.

*Most deadly errors arise from obsolete assumptions.*

However, we’ve outlined a few points that lead to defects.

**1. Lack of communication.**

It’s the most obvious cause that makes a software runs into errors. It happens if the Product owner who has all the knowledge of the feature misses to pass it to the developers or QA.

Ideally, everyone working on the application should have the same level of understanding of the product features.

**2. High level of complexity.**

Next disrupter is the complexity (of the software application or the underlying technology) which makes it harder for anyone to understand.

So it’s best to have Architects who have in-depth technical skills to study and define the complexity.

**3. Change in requirements.**

It is something which you can’t control but can manage by working with the customer. However, when a change occurs, it impacts schedule, resource allocation, and delivery.

So it’s one more obvious which can cause tremors in a Software application.

**4. Poor documentation.**

It’s not easy to maintain and modify a code which has weak or no documentation. Hence, the results are a handful of bugs.

Many companies offer incentives or reward programs for developers producing clear, understandable, maintainable code.

**5. Deadlines pressure.**

To meet the schedule, you may decide to skip a part of testing, assuming it was working earlier and would continue so. But untested code may cause havoc in a software application when deployed at customer end.

*Nobody works better under pressure. They just work patiently.*

[**Back to top**](http://www.techbeamers.com/software-quality-assurance-interview-questions/#top)

**Question-24. What is the main difference between Verification and Validation?**

**Ans.**

**1. Verification.**

A process to ensure that the final software matches the original design. It checks whether the software meets the criteria and specification described in the requirement document.

You make sure that the product is right and as per design. It is a simple check done via walkthrough (informal) meetings.

**2. Validation.**

A process to check if the product design fulfills the client’s need. It confirms whether you built the right thing.

The validation includes smoke testing, functional testing, and so on.

**Question-25. What is Agile Testing and how does it differ from the traditional waterfall or the V-model?**

**Ans.** Agile Testing is a dynamic testing model that follows the principles of agile software development.

It demands all members of an agile team to have special skills and expertise. So that, they can ensure to deliver the expected business value at regular intervals.

In Agile environment, testing is not just a phase, but it is an activity that runs parallel to the development. Since it allows small features to deliver frequently, so testing activity should be parallel to the development activity.

Testing time is short as we are only testing small features.

Whereas in the waterfall model, we have the testing phase at the end of the development. So it requires more efforts (resources and time) to test the entire application.