

Q1. What is software testing?

A1. Software testing is a process that corrects, completes & quality checks the software that is being developed.

Q2. What is Exploratory Testing?

A2. Exploratory testing as the name suggests, is a simultaneous learning, test design, and test execution process. We can say that in this testing test planning, analysis, design and test execution, are all done together and instantly. Exploratory testing allows you to think outside the box and come up with use cases that might not be covered in a test case. For example, you might perform one test and then ask yourself, “What if I tried this? What if I didn't do that?”

Q3. What is traceability matrix?

A3. RTM stands for Requirement Traceability matrix. RTM maps all the requirements with the test cases. By using this document one can verify test cases cover all functionality of the application as per the requirements of the customer. The main purpose of the requirement traceability matrix is to verify that the all requirements of clients are covered in the test cases designed by the testers.

Q4. What is Boundary value testing?

A4. Boundary testing is the process of testing between extreme ends or boundaries between partitions of the input values. So, these extreme ends like Start- End, Lower- Upper, Maximum-Minimum, Just Inside-Just Outside values are called boundary values and the testing is called boundary testing.

Q5. What is Equivalence partitioning testing?

A5. Equivalence partitioning (EP) is a method for testing software programs. In this technique, the data fed into the software to be tested is divided into partitions of equal sizes. From each partition of data, one test case is needed. Each test case is defined specifically to check a specific type of error. This speeds up the error hunting process as fewer test cases are required.

Q6. What is Integration testing?

A6. Integration testing is the second level of the software testing process comes after unit testing. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

Q7. What determines the level of risk?

Project Risk- when a senior member leaves the project abruptly, it is risk to the project.

Product Risk- Product risks result from problems with the delivered product. Product Risks associate with specific quality characteristics of the product.

Q8. What is Alpha testing?

A8. Alpha testing is simulated or real operational testing at an in-house site. It comes after the unit testing, integration testing, etc. Alpha testing is conducted in the organization and tested by a representative group of end-users at the developer's side and sometimes by an independent team of testers.

Q9. What is beta testing?

A9. Beta testing is a type of User Acceptance Testing among the most crucial testing, which performed before the release of the software. Beta Testing is a type of Field Test. This testing performs at the end of the software testing life cycle. Beta testing is the last phase of the testing, which is carried out at the client's or customer's site.

Q10. What is component testing?

A10. A unit is a single testable part of a software system and tested during the development phase of the application software. The purpose of unit testing is to test the correctness of isolated code. Unit testing involves the testing of each unit or an individual component of the software application. It is the first level of functional testing.

Q11. What is functional system testing?

A11. Functional testing is a part of black-box testing as its emphases on application requirement rather than actual code. The test engineer has to test only the program instead of the system. In functional testing, all the components are tested by giving the value, defining the output, and validating the actual output with the expected value. The test engineer will check all the components systematically against requirement specifications is known as functional testing. Functional testing is also known as Component testing.

Types of Functional Testing:

1. Unit Testing
2. Integration Testing
3. System Testing

Q12. What is Non-Functional Testing?

A12. Non-functional testing is a type of software testing to test non-functional parameters such as reliability, load test, performance and accountability of the software. The primary purpose of non-functional testing is to test the reading speed of the software system as per non-functional parameters. Non-Functional testing checks the ability of the software to work in an external environment.

Types of Non-functional Testing:

1. Performance Testing- Load Testing, Stress Testing, Scalability Testing, Stability Testing
2. Usability Testing-
3. Compatibility Testing

Q13. What is GUI Testing?

A13. It is one of the unique types of software testing that is frequently used to check the Graphical user interface features for the application or the software.

Q14. What is Ad hoc testing?

A14. Ad-hoc testing is an informal testing type whose aim is to break the system. This type of software testing is unplanned activity. It does not follow any test design to create the test cases. Ad-hoc testing is done randomly on any part of the application; it does not support any structured way of testing.

Q15. What is white box testing and list the types of white box testing?

A15. In white-box testing, the developer will inspect every line of code before handing it over to the testing team or the concerned test engineers. White box testing is also known as open box testing, glass box testing, structural testing, clear box testing, and transparent box testing. It is based on inner workings of an application and revolves around internal structure testing.

7 Different types of white-box testing

1. Unit Testing
2. Static Analysis
3. Dynamic Analysis
4. Statement Coverage
5. Branch testing Coverage
6. Security Testing
7. Mutation Testing

Q16. What is black box testing? What are the different black box testing techniques?

A16.Black box testing is a process of checking the functionality of an application as per the customer requirement. The source code is not visible in this testing; that's why it is known as **black-box testing.**

Black Box testing techniques are:

1. Boundary Value Analysis.
2. Equivalence partitioning.
3. State Transition Testing.
4. Decision Table Testing.
5. Graph-Based Testing.
6. Error guessing technique.

Q17. Mention what are the categories of defects?

A17.

* Data quality/database defects: deals with improper handling of data in the database.
* Critical functionality defects: the occurrence of these bugs hampers the crucial functionality of the application.
* Functionality defects: these defects affect the functionality of the application
* Security defects: application security defects generally involve improper handling of data sent from the user to the application. These defects are the most severe and given highest priority for a fix
* User interface defects: as the name suggests, these bugs deal with problems related to ui are usually considered less severe.

Q18. Mention what big bang testing is?

A18. In this approach, testing is done via integration of all modules at once. The advantage is that it is convenient for small size software systems. But the disadvantage is that Identification of defects is difficult because finding the error where it came from is a problem, and we don't know the source of the bug and time provided for testing is very less.

Q19. What is the purpose of exit criteria?

A19. Purpose of exit criteria is to define when to STOP testing. Either to stop testing at the end, where product go live or at the end of the phase of testing.

Q20. When should "Regression Testing" be performed?

A20. We do regression testing whenever the production code is modified.

We can perform regression testing in the following scenario, these are:

1. When new functionality added to the application.

Example: A website has a login functionality which allows users to log in only with Email. Now providing a new feature to do login using Facebook.

2. When there is a Change Requirement.

**Example:** Remember password removed from the login page which is applicable previously.

**3. When the defect fixed**

**Example:** Assume login button is not working in a login page and a tester reports a bug stating that the login button is broken. Once the bug fixed by developers, tester tests it to make sure Login Button is working as per the expected result. Simultaneously, tester tests other functionality which is related to the login button.

**4. When there is a performance issue fix**

**Example:** Loading of a home page takes 5 seconds, reducing the load time to 2 seconds.

**5. When there is an environment change**

**Example:** When we update the database from MySQL to Oracle.

Q21. What is 7 key principles? Explain in detail?

A21. The 7 key principles of testing are:

1. **Testing shows presence of defects**- testing helps in showing that there are defects but it can’t prove that there are no defects. It reduces the probability of undiscovered defects. Testing can reduce the number of defects but not remove all defects.

1. **Exhaustive testing is impossible**- the process of testing the functionality of the software in all possible inputs (valid or invalid) and pre-conditions is known as exhaustive testing. Exhaustive testing is impossible means the software can never test at every test case. It can test only some test cases and assume that the software is correct and it will produce the correct output in every test case. If the software will test every Test Case, then it will take more cost, effort, etc., which is impractical.
2. **Early testing**- To find the defect in the software, early test activity shall be started. The defect detected in the early phases of SDLC will be very less expensive. For better performance of software, software testing will start at the initial phase i.e., testing will perform at the requirement analysis phase.
3. **Defect clustering**- Defects are not evenly spread in a system, they are clustered. These are usually confined to a small number of modules.
4. **The Pesticide Paradox-** Repeating the same test cases, again and again, will not find new bugs. So, it is necessary to review the test cases and add or update test cases to find new bugs.
5. **Testing is Content Dependent**- The testing approach depends on the context of the software developed. Different types of software need to perform different types of testing. For example, the testing of the e-commerce site is different from the testing of the Android application.
6. **Absence of Errors Fallacy**- If a built software is 99% bug-free but it does not follow the user requirement then it is unusable. It is not only necessary that software is 99% bug-free but it is also mandatory to fulfill all the customer requirements.

Q22. Difference between QA v/s QC v/s Tester.

|  |  |  |
| --- | --- | --- |
| **Quality Assurance** | **Quality Control** | **Testing** |
| Subset of SDLC | Subset of QA | Subset of QC |
| Process-oriented activities. | Product-oriented activities. | Product-oriented activities. |
| Preventive process | Corrective process | Preventive process |
| Focus on processes | Focus on required quality | Focus on actual testing |
| Verifies the quality | Validates the quality | Validates the quality |
| Whole project team involved | Testing team involved | Testing team involved |
| Makes sure the right things are done | Makes sure the things are done right | Evaluates the result of done things |

Q23. Difference between Smoke and Sanity?

A23.

|  |  |  |
| --- | --- | --- |
| **Sr no.** | **Smoke Testing** | **Sanity Testing** |
| 1 | It is a broad approach to testing where all parts of the application are tested. | It is a narrow approach to testing where specific parts of the application are tested. |
| 2 | It measures the stability of the system by performing rigorous testing. | It measures the rationality of the system by performing rigorous testing. |
| 3 | Smoke testing can be either manual or automated. | Sanity testing can be done without test cases or scripts |
| 4 | It is performed by both testers and developers. | It is performed by only testers. |
| 5 | Smoke testing is documented. | Sanity testing is not documented. |
| 6 | It is used to test End to End function of the application. | It is used to test only modified or defect fixed functions. |
| 7 | It is considered as a subset of acceptance testing. | It is considered as a subset of regression testing |

Q24. Difference between verification and Validation.

These two terms are very confusing for most people, who use them interchangeably. The following table highlights the differences between verification and validation.

|  |  |  |
| --- | --- | --- |
| **Srno.** | **Verification** | **Validation** |
| 1 | Verification addresses the concern: "Are you building it right?" | Validation addresses the concern: "Are you building the right thing?" |
| 2 | Ensures that the software system meets all the functionality. | Ensures that the functionalities meet the intended behavior. |
| 3 | Verification takes place first and includes the checking for documentation, code, etc. | Validation occurs after verification and mainly involves the checking of the overall product. |
| 4 | Done by developers. | Done by testers. |
| 5 | It is an objective process and no subjective decision should be needed to verify a software. | It is a subjective process and involves subjective decisions on how well a software works. |

Q25. Explain types of Performance testing.

A25. Checking the behavior of an application by applying some load is known as performance testing.

Following are the types of performance testing:

* Load testing- The load testing is used to check the performance of an application by applying some load which is either less than or equal to the desired load is known as load testing.
* Stress testing- this testing checks the behavior of an application by applying load greater than the desired load.
* Scalability testing- Checking the performance of an application by increasing or decreasing the load in particular scales (no of a user) is known as scalability testing.
* Stability testing- Checking the performance of an application by applying the load for a particular duration of time is known as Stability Testing.

Q26. What is Error, Defect, Bug and failure?

A26.

* **Error:** An Error is a mistake made in the code due to which compilation or execution fails, Error is mainly done by the developers.
* **Defect:** A Defect is a deviation between the actual and expected output. The defect is identified by The Testers and is resolved by developers in the development phase of SDLC.
* **Bug:** A bug refers to defects which means that the software product or the application is not working as per the requirements. When we have any type of logical error, it causes our code to break, which results in a bug.
* **Failure:** Failure is detected by end-users once they face a particular issue in the software. Failure is the combinations of several defects that leads to Software failure and results in making the system unresponsive.

A simple diagram depicting Bug vs Defect vs Error vs Failure:

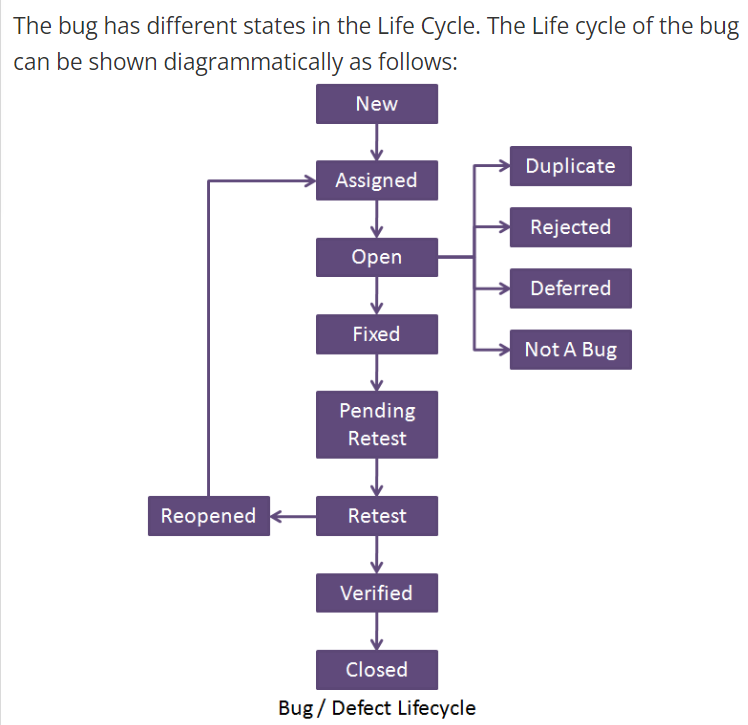
Bug vs Defect vs Fault vs Failures vs Error

Q27. Difference between Priority and Severity.

|  |  |
| --- | --- |
| **Severity** | **Priority** |
| Severity means how severe defect is affecting the functionality. | Priority means how fast defect has to be fixed. |
| Testing engineer decides the severity level of the defect. | Product manager decides the priorities of defects. |
| Severity is related to the quality standard. | Priority is related to scheduling to resolve the problem. |
| Its value does not change from time to time. | Its value changes from time to time. |
| Severity is of 5 types: Critical, Major, Moderate, Minor, and Cosmetic. | Priority is of 3 types: Low, Medium, and High. |

Q28. What is Bug Life Cycle?

A28. Defect life cycle is a cycle which a defect goes through during its lifetime. It starts when defect is found and ends when a defect is closed, after ensuring it is not reproduced. Defect life cycle is related to the bug found during testing.



Q29. Explain the difference between Functional testing and Nonfunctional testing.

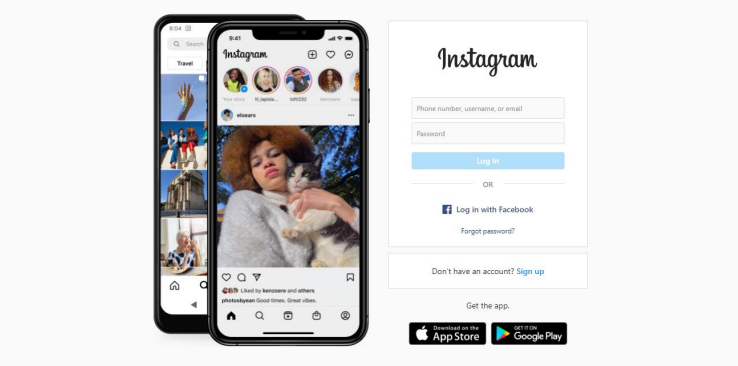
A29.

|  |  |
| --- | --- |
| **Functional Testing** | **Non- Functional Testing** |
| Defines what to test | Defines how to test |
| Tests the functionality | Tests how the functionality is performed |
| It is manual testing | It is automated testing |
| Based on business requirements | Based on customer expectations |
| Checks apps, operations and actions | Checks apps |
| Checks actual vs expected results | Checks response, speed & other requirements |

Q30. To create HLR & Test Case of

1. Instagram, Facebook) only first page

**A30. Uploaded in GitHub.**



1. Facebook Login Page : <https://www.facebook.com/>



Q31. What is the difference between the STLC (Software Testing Life Cycle) and SDLC

(Software Development Life Cycle)?

A31.

|  |  |
| --- | --- |
| **SDLC** | **STLC** |
| Stands for software development life cycle | Stands for software testing life cycle |
| Covers entire lifecycle of the software | Limited only to testing phase |
| SDLC provides a software product | STLC carries out process to check the quality of the product |
| Developers are needed | Testers are needed |
|  |  |
|  |  |

Q32. What is the difference between test scenarios, test cases, and test script?

A32.

|  |  |  |
| --- | --- | --- |
| Test Scenarios | Test Cases | Test Script |
| It is any functionality that can be tested | It is set of actions that are executed to verify functionality or feature | It is set of steps used to test part of functionality of any app or software |
| Is more focused on what to test | Is focused on what to test & how | Is focused on the expected result |
| Needs less time & fewer resources | Needs more resources & time | Needs less time for testing but more resources for script creating & editing |
| Derived from BRS or SRS | Derived from test scenarios | Derived from test cases |

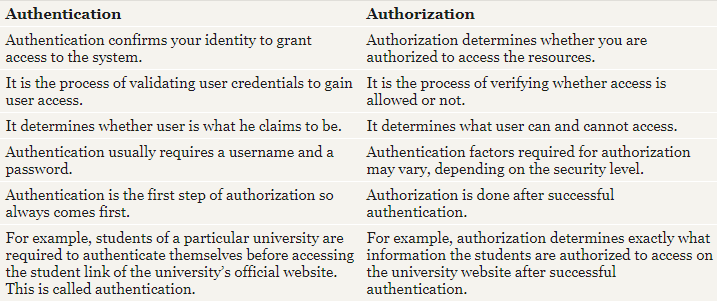
Q33. Explain what Test Plan is? What is the information that should be covered?

A33. A test plan outlines the strategy that will be used to test an application, the resources that will be used, the test environment in which testing will be performed, and the limitations of the testing and the schedule of the testing activities.

The information that should be covered in a test plan is:

* Introduction to the Test Plan document.
* Assumptions while testing the application.
* List of test cases included in testing the application.
* List of features to be tested.
* The sort of approach to be used while testing the software.
* List of deliverables that need to be tested.
* The resources allocated for testing the application.
* Any risks involved during the testing process.
* A schedule of tasks and milestones to be achieved.

Q35. Explain the difference between Authorization and Authentication in Web testing.



Q36. What are the common problems faced in Web testing?

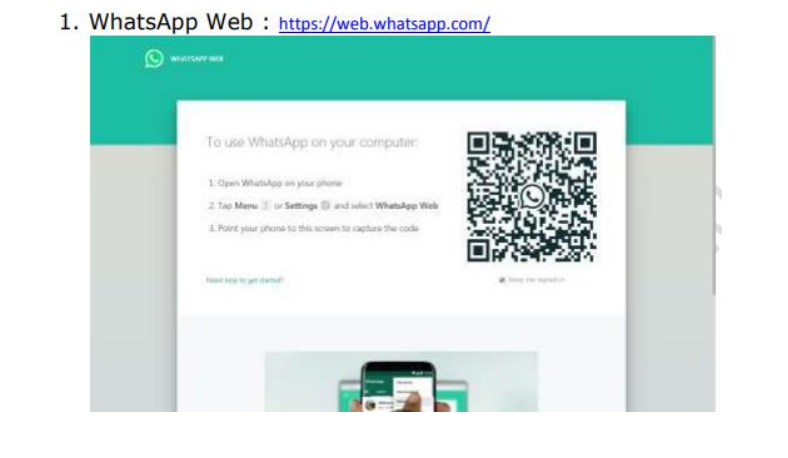
A36. Below are the testing challenges faced by web developers during the development process.

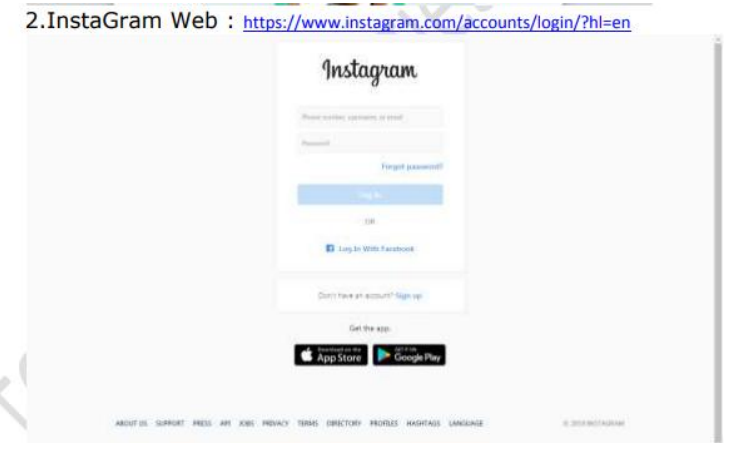
1. Integration. Integration testing exposes problems with interfaces among different program components before deployment. ...
2. Interoperability. ...
3. Security. ...
4. Performance. ...
5. Usability. ...
6. Quality Testing, Exceptional Services.

Q37. To create HLR & Test Case of Web Based (WhatsApp web, Instagram)

**Uploaded in GitHub- whatsapp done, insta done**

1. WhatsApp Web: <https://web.whatsapp.com/>





Q38. To create HLR and Test Case on this Link. <https://artoftesting.com/>

