I Can do all things through Christ who strengths me!

MANUAL TESTING

WHAT IS SOFTWARE TESTING:

**Software testing** is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not

To identify the defects to ensure that the product is defect-free in order to produce a quality product.

Software’s correctness, completeness, and quality will be checked.

**WHY SOFTWARE TESTING IS NEEDED:**

If there are any bugs or errors in the software, it can be identified early and can be solved before delivery of the software product.

**NEED OF TESTING:**

* Starbucks was forced to close about 60 percent of stores in the U.S and Canada due to software failure in its POS system. At one point, the store served coffee for free as they were unable to process the transaction.

Testing types: Manual and Automation

Testing Method: Static and Automation

Testing Approaches: White box/Glass box/Structural, Black box/Specification-based/Behavioural and Grey Box

Testing Levels: Unit,Integration,Sytem,Acceptance

Types of Black box Testing: Functional and Non-functional

**Types of Software Testing**

1. Functional Testing

* [Unit Testing](https://www.guru99.com/unit-testing-guide.html)
* [Integration Testing](https://www.guru99.com/integration-testing.html)
* Smoke
* Sanity
* UAT (User Acceptance Testing)
* Localization
* Globalization
* Interoperability

1. Non-Functional Testing

* Performance
* Endurance
* Load
* Volume
* Scalability
* Usability

1. Maintenance

* Regression
* Maintenance

SOFTWARE DEVELOPMENT LIFECYCLE

1.Requirements Gathering

Business analyst collects the requirement from the customer/client as per the clients business needs and documents the requirements in the business requirement specification

2.Analaysis

Once the requirements are gathered and analysed, the next step is to define and document the product requirements and get them approved by the customer. This is recorded through the SRS (Software Requirement Specification) document.

3.Design

1. **HLD** –*High-Level Design* – It gives the architecture of the software product to be developed and is done by architects and senior developers
2. **LLD** – *Low-Level Design* – It is carried out by senior developers. It determines about how each and every feature in the product should work and how every component should work.

4.Coding

Developers of all levels (seniors, juniors, freshers) are involved in this phase. This is the phase where you start building the code for the software.

5.Testing

When the software is ready, it is sent to testing department to test it thoroughly for different defects. Testing of a software is carried out either manually or using automated testing tools.

Ensure each and every component of the software works fine. Once the software is error-free, it goes to the next stage, which is Implementation.

6.Deployment/Maintenance phase.

Deployment: Once the product is error free, it is delivered/deployed to the customer for their use. Deployment is done by the Deployment/Implementation engineers.

Maintenance: As the customers start it, then the actual problems come up and needs to be solved from time to time. Detecting and solving these issues found by the customer comes in the maintenance phase.

### **What’s the difference between verification and validation in testing?**

|  |  |
| --- | --- |
| **Verification** | **Validation** |
| It is a static analysis technique.  Here, testing is done without executing the code.  Examples include – Reviews, Inspection, and walkthrough.  Are we building the product right? | It is a dynamic analysis technique  testing is done by executing the code.  Examples include functional and non-functional testing techniques.  Are we building the right product? |

### **INSPECTION: (FORMAL REVIEW)**

Minimum 3 or more people

Intimated well in advance.Start time and end time I mentioned

Meeting will happen in scheduled venue.Outcome will be shared with upper level management

Members present:

Reader: Who conducts and reads the agenda(Proj.architect)

Writer/Recorder: Who takes notes in the meeting.(SSE/STE)

Moderator: Who ensures the meeting is going on smooth(PM,TM)

Interested Parties: Who simply joins the meeting(TE,TL,DL,Trainee Engg)

WALKTHROUGH: (FORMAL REVIEW)

Occurs at peer level.Meeting takes place at developer’s or Tester’s place

No prior intimation.Duration is not fixed.Outcome is not shared to high level

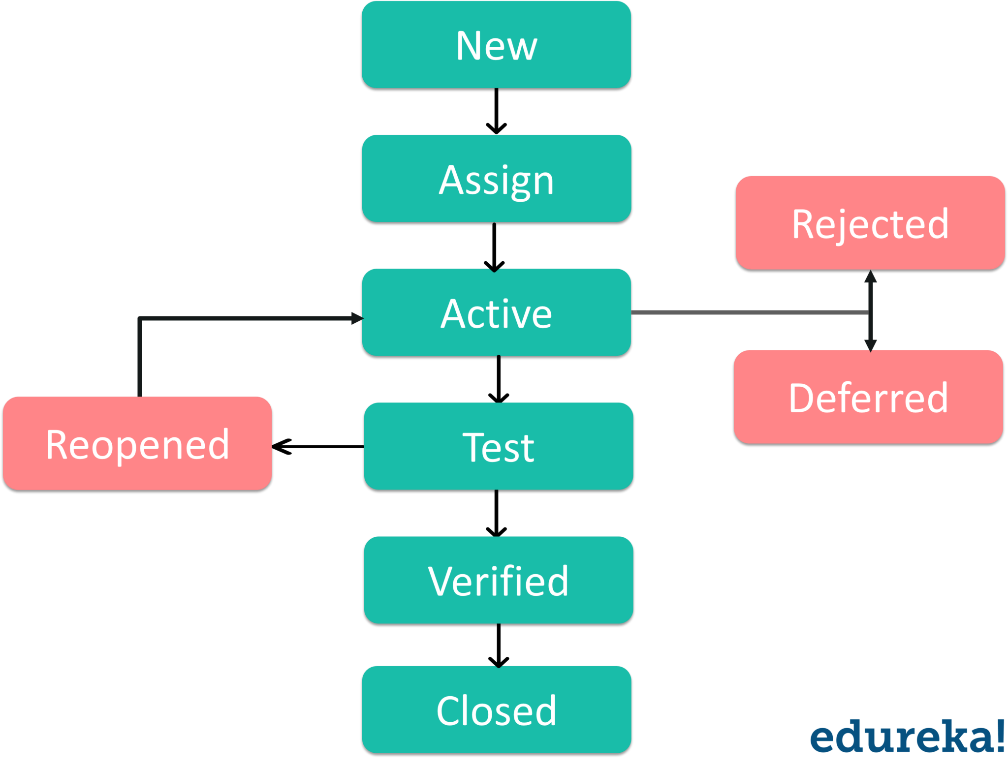
### **What are the phases involved in Software Testing Life Cycle?**

**Software Testing Life Cycle: It is a sequence of activities conducted during the testing process to ensure software quality goals are met. It includes both verification and validation activities**

|  |  |
| --- | --- |
| **Phases** | **Explanation** |
| **Requirement Analysis** | QA team understands the requirement in terms of what we will testing & figure out the testable requirements. |
| **Test Planning** | In this phase, the test strategy is defined. Objective & the scope of the project is determined. |
| **Test Case Development** | Here, detailed test cases are defined and developed. The testing team also prepares the test data for testing. |
| **Test Environment Setup** | It is a setup of software and hardware for the testing teams to execute test cases. |
| **Test Execution** | It is the process of executing the code and comparing the expected and actual results. |
| **Test Cycle Closure** | It involves calling out the testing team member meeting & evaluating cycle completion criteria based on test coverage, quality, cost, time, critical business objectives, and software. |

**Bug life cycle**

A**defect life cycle** is a process in which a defect goes through various phases during its entire lifetime. It starts when a defect is found and ends when a defect is closed, after ensuring it’s not reproduced.



1. **New:** In this step, if a defect is logged and posted for the first time, its state is given as new.
2. **Assigned:** After the tester has posted the bug, the lead of the tester approves that the bug is genuine and he assigns the bug to a corresponding developer and the developer team. It’s state given as assigned.
3. **Open: In** this state, the developer has started analyzing and working on the defect fix.
4. **Fixed:**As the developer makes necessary code changes and verifies the changes then he/she can make bug status as ‘Fixed’ and the bug is passed to the testing team.
5. **Test:** At this stage, the tester does the testing of the changed code which the developer has given back to him to check whether the defect has been fixed or not.
6. **Verified:**Here, the tester tests the bug again after it got fixed by the developer. If there is no bug in the software, he approves that the bug is fixed and changes the status to “verified”.
7. **Reopen:**In case if the bug still exists even after the bug is fixed by the developer, the tester changes the status to “reopened”. In this state, the bug goes through the life cycle once again.
8. **Closed:**As soon as the bug is fixed, it is tested by the tester. In case, if the tester feels that the bug no longer exists in the software, he changes the status of the bug to “closed”. It implies that the bug is fixed, tested and approved.
9. **Duplicate:** In the bug life cycle, if the bug is repeated twice or the two bugs mention the same concept of the bug, then one bug status is changed to “duplicate”**.**
10. **Rejected:** If in case the developer feels that the bug is not genuine, he rejects the bug. Then the state of the bug is changed to “rejected”.
11. **Deferred:** If the bug is changed to deferred state means the bug is expected to be fixed in next releases.

WATERFALL MODEL:

The waterfall model is the earliest SDLC approach that are used for software development

This model illustrates the software development process in a linear sequential flow.This means that any phase in the development process begins only if the previous phase is complete.In this model,the phases do not overlap.

V-MODEL:

It is also known as Verification and Validation model

The V-model is an SDLC model where execution of process happens in a sequential manner in V-shape. The next phase starts only after the completion of the previous phase.

It is based on the association of a testing phase for each corresponding development stage.

For every single phase in the development cycle,there is a directly associated testing p`hase.

**Difference between QC and QA**

|  |  |
| --- | --- |
| **Quality Control** | **Quality Assurance** |
| Quality control is a product-oriented approach of running a program to determine if it has any defects, as well as to make 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. |

**What is Requirement Traceabality Matrix?**

* **RTM** is a document that maps user requirement with test cases.
* It captures all requirements proposed by the client.
* Requirement traceability in a single document, delivered at the conclusion of SDLC.
* The main purpose of Requirement Traceability Matrix is to validate that all requirements are checked via test cases such that no functionality is unchecked during Software testing.

### **What different types of manual testing are there?**

* + Black Box Testing
  + White Box Testing
  + Unit Testing
  + System Testing
  + Integration Testing
  + Acceptance Testing

### **What are the different levels of manual testing?**

* [**Unit testing**](https://www.edureka.co/blog/what-is-unit-testing) – It is a way of testing the smallest piece of code referred to as a **unit** that can be logically isolated in a system. It is mainly focused on the functional correctness of the standalone module.
* [**Integration Testing**](https://www.edureka.co/blog/what-is-integration-testing-a-simple-guide-on-how-to-perform-integration-testing/) – It is a level of software testing where individual units are combined and tested to verify if they are working as they intend to when integrated. The main aim here is to test the interface between the modules.
* **System Testing** – In system testing all the components of the software are tested as a whole in order to ensure that the overall product meets the requirements specified. There are dozens of types of system testing, including usability testing, regression testing, and functional testing.
* **User Acceptance Testing** – The final level, acceptance testing, or UAT (user acceptance testing), determines whether or not the software is ready to be released.

### **Explain the difference between alpha testing and beta testing.**

* **Alpha Testing** – It is a type of software testing performed to identify bugs before releasing the product to real users or to the public. It is a type of user acceptance testing. It is done at developer’s site by developer, tester and client or end user.
* **Beta Testing** – It is performed by real users of the software application in a real environment. It is also a type of user acceptance testing. It is done at client site by the end user.

**What is grey box testing?**

It is a combination of black box and white box testing. Tester who works on this type of testing need to have access to design documents, this helps to create better test cases.

This technique involves testing the software application/product with partial knowledge of internal structure(code) of the application

Example: If tester is testing a website, and encountered a problem with links, he can directly make changes in HTML code and check in real time.

### **What is the difference between manual testing and automation testing?**

|  |  |
| --- | --- |
| Manual Testing | Automation Testing |
| Accuracy, and reliability of test cases are low, as manual tests are more prone to human error. | Automated testing,is more reliable as tools and scripts are used to perform tests. |
| The time required for manual testing is high as human resources perform all the tasks. | The time required is comparatively low as software tool execute the tests |
| Investment cost is low, but Return of Investment(ROI) is low as well. | Investment cost and Return of Investment, both are high. |
| IT is preferred when the test cases are run once or twice. | It is preferred for Regression Testing, Performance Testing, Load Testing or highly repeatable functional test cases |
| Allows for human observation to find out any glitches. Therefore, helps in improving the customer experience. | No human observation involved, there is no guarantee of positive customer experience. |

### Difference Between Functional and Non-Functional Testing

| **Functional Testing** | | **Non Functional Testing** | | |
| --- | --- | --- | --- | --- |
| It tests ‘What’ the product does. It checks the operations and actions of an Application. | | It checks the behaviour of an Application. | | |
| Functional testing is done based on the business requirement. | | Non- functional testing is done based on the customer expectation and Performance requirement. | | |
| It tests whether the actual result is working according to the expected result. | | It checks the response time, and speed of the software under specific conditions. | | |
| It is carried out manually. Example: Black box testing method. | | It is more feasible to test using automated tools. Example: Loadrunner. | | |
| It tests as per the customer requirements. | | It tests as per customer expectations. | | |
| Customer feedback helps in reducing the risk factors of the product. | | Customer feedback is more valuable for non- functional testing as it helps to improve and lets the tester to know the expectation of the customer. | | |
| It is testing the functionality of the software. | | It is testing the performance of the functionality of the software. | | |
| Functional testing has the following types: •Unit testing •Integration testing •System Testing •Acceptance Testing | | Non functional testing includes: •Performance testing •Load Testing •Stress testing •Volume testing •Security testing •Installation testing •Recovery testing | | |
| Example: A Login page must show textboxes to Enter the username and password. | |  | | |
|  | **Performance testing** | | **Load testing** | **Stress testing** |
| **Domain** | Superset of load and stress testing | | A subset of performance testing. | A subset of performance testing. |
| **Scope** | Includes - Load Testing, Stress Testing, capacity testing, volume testing, endurance testing, spike testing, scalability testing and reliability testing, etc. | | Includes volume testing and endurance testing. | Includes soak testing and spike testing. |
| **Major goal** | To set the benchmark and standards for the application. | | To identify the upper limit of the system, set SLA of the app and see how the system handles heavy load volumes. | To identify how the system behaves under intense loads and how it recovers from failure. Basically, to prepare your app for the unexpected traffic spike. |
| **Load Limit** | Both – below and above the threshold of a break. | | Till the threshold of break | Above the threshold of break |
| **Attributes studied** | Resource usage, reliability, scalability, resource usage, response time, throughput, speed, etc. | | peak performance, server throughput, response time under various load levels **(below the threshold of break),** adequacy of H/W environment, the number of user app can handle, load balancing requirements, etc. | Stability beyond bandwidth capacity, response time **(above the threshold of break),** etc. |
| **Issues identified through this testing type** | All performance bugs including runtime bloat, the scope for optimization, issues related to speed, latency, throughput, etc. Basically – anything related to performance! | | Load balancing problems, bandwidth issues, system capacity issues, poor response time, throughput issues, etc. | Security loopholes with overload, data corruption issues at overload situation, slowness, memory leaks, etc. |

**What is blocker?**

A blocker is a bug of high priority and high severity. It prevents or blocks testing of some other major portion of the application.

**What are Cosmetic Bugs?**

Bugs found in the GUI of the application are called as cosmetic bugs. These bugs do not affect the processing of the application, but can have great impact in some contexts and most of the time they are very much visible..

For example: spelling mistake, grammar mistake, font size, colours, tab order, text overlapping etc.

What is Show Stopper?

A critical bug is a show stopper which means a large piece of functionality or major system component is completely broken and there is no workaround to move further.

For example, due to a bug in one module,we cannot test the other modules because the blocker bug has blocked the other modules

Eg: “Sign in” Button is not working in gmail apps.Gmails users are blocked to login to their accounts

Difference between Priority and severity?

|  |  |
| --- | --- |
| Priority | Severity |
| Priority is the order in which the developer should resolve a defect | Severity is the degree of impact that a defect has on the operation of the product |
| Priority is categorized into: Low, Medium and High | Severity is categorized into: Major, Minor, Moderate, Critical and Cosmetic |
| Priority indicates how soon the bug should be fixed | Severity indicates the seriousness of the defect |
| Priority status is based on customer requirements | Severity status is based on technical aspect of the product |
| Priority is driven by business value | Priority is driven by functionality |
| Priority value is subjective-can be changed over a period of time depending on change in project situation | Priority value is objective-less likely to change |
| High Priority and low severity: defect have to be fixed on immediate basis, but doesn’t affect the application | High Severity and low priority: defect have to be fixed but not on immediate basis |

What is pilot testing?

Pilot testing is a type of Software testing which is performed by a group of end-users prior to the deployment of the software in production.

The component of the system or the complete system is tested and verified in the real-time scenario.

Pilot Testing comes in between the User Acceptance test and Production deployment.

**What is test data?**

Test data is the input given to the software program during test execution.

Test data for positive testing, to verify that functions produce expected results for given input

Test data for negative testing, to test software ability to handle unusual or unexpected inputs.

**What are Stubs and Drivers?**

**Note: Both Stubs and Drivers are dummy modules used to simulate the working of modules by providing hardcoded or expected output based on input values**

|  |  |
| --- | --- |
| Stubs | Drivers |
| Stubs are used in Top-Down Integration testing | Drivers are used in Bottom-Up Integration testing |
| It is used when lower-level of modules are missing or in partially developed phase | It is used when higher-level of modules are missing or in partially developed phase |
| The stubs are taken into concern if testing of upper-levels of modules are done, and lower-level modules are under developing process | The stubs are taken into concern if testing of lower-levels of modules are done, and higher-level modules are under developing process |
| Stubs are basically known as “Called programs”. | Stubs are basically known as “Calling programs”. |

### **What is a testbed?**

The testbed is an environment configured for testing an application. It consists of hardware, software, network configuration, an application under test, other related software needed to run the program to be tested.

Eg: For testing an Web Application,

WebServer-ITS/Apache

Database-My Sql

OS- Windows/Linux

Browser- IE/Firefox

Java Version- Version 6

**What is Test Scenario?**

* Test Scenario is any functionality that can be tested.
* Test Scenario is derived from test artifacts like BRS and SRS.
* It is focused on What to test.
* Test Scenarios are high level actions
* It requires few resources and less time for execution.
* It includes end-to-end functionality to be tested.

### **What is the test case?**

* A[**test case**](https://www.edureka.co/blog/test-case-in-software-testing/) is a document that has a set of conditions or actions that are performed on the software application in order to verify the expected functionality of the feature.
* Test Case is derived from Test Scenarios
* It is focused on What to be test and how to test
* Test cases are low-level actions
* It requires more resources and more time for execution.
* It includes test steps, test data, expected results etc..

### **What’s the difference between a bug, defect and Error?**

A [bug](https://www.edureka.co/blog/bugs-in-software-testing/#softwaretestingbugs) is a fault in the software that’s detected during testing time.

A defect is a variance between expected results and actual results, detected by the developer after the product goes live.

**Error** – An error is a mistake, misunderstanding, or misconception, on the part of a software developer. When error arises in software, it leads to a change in the functionality of the program.

### **What is regression testing? When to apply it?**

“Testing of a previously tested program to ensure that defects have not been introduced or uncovered in unchanged areas of the software, as a result of the changes made is called [*Regression Testing*](https://www.edureka.co/blog/regression-testing).”

A regression test is a system-wide test whose main purpose is to ensure that a small change in one part of the system does not break existing functionality elsewhere in the system. It is recommended to perform regression testing on the occurrence of the following events:

* When new functionalities are added
* In case of change requirements
* When there is a defect fix
* When there are performance issues
* In case of environment changes
* When there is a patch fix

### **What is the difference between system testing and integration testing?**

|  |  |
| --- | --- |
| **System Testing** | **Integration Testing** |
| System Testing tests the software application as a whole to check if the system meets the user requirements | Integration testing tests the interface between modules of the software application |
| Involves both functional and non-functional testing like sanity, usability, performance, stress and load | Only functional testing is performed to check whether the two modules when combined give the right outcome |
| It is high-level testing performed after integration testing | It is low-level testing performed after unit testing |

### **What is test closure?**

Test Closure is a document which gives a summary of all the tests conducted during the software development life cycle and also gives a detailed analysis of the bugs removed and errors found.

**What is test strategy?**

It is a high-level document, usually prepared by project manager.

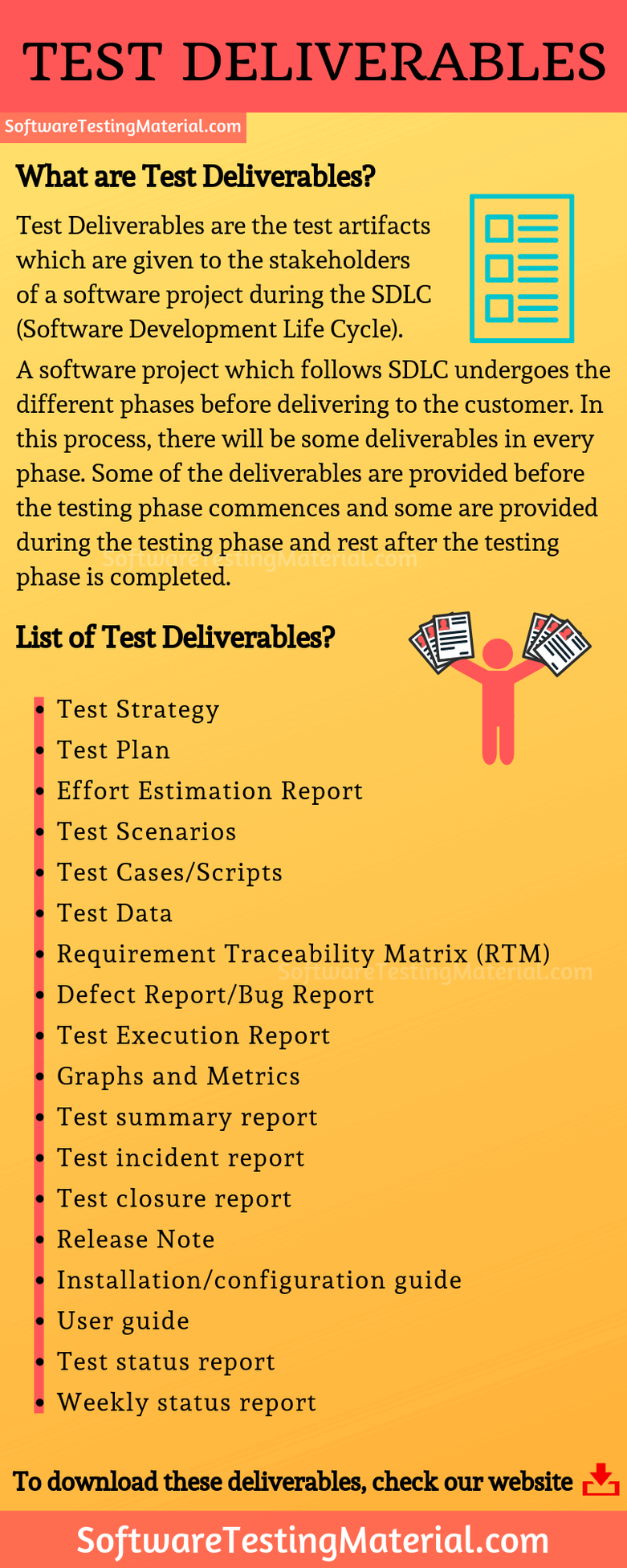
It defines the approach of how to test the product and the means to achieve them.

### **What is the difference between Test Plan and Test Strategy?**

|  |  |
| --- | --- |
| **Test Plan** | **Test Strategy** |
| It describes scope, objective and s/w testing tasks | It describes how testing needs to be done |
| It is used at the project level | It is used at the organization level |
| Primary goal: How to test, when to test and who will verify | What technique to follow and which module to check |
| Test plan can be changed | Test strategy can’t be changed |
| It is carried out by test manager | It is carried out by project manager |

### **What is the difference between Positive and Negative Testing?**

|  |  |
| --- | --- |
| **Positive Testing** | **Negative Testing** |
| **Positive testing** determines that your application works as expected. If an error is encountered during positive testing, the test fails | **Negative testing** ensures that your application can gracefully handle invalid input or unexpected user behaviour |
| In this testing, tester always check for an only valid set of values | Testers apply as much creativity as possible and validating the application against invalid data |



What is Ad hoc Testing?

Ad hoc Testing is an unstructured way of testing and it is randomly done on any part of the application. Adhoc testing can be achieved with the Software testing technique called Error Guessing that is usually an unplanned activity that does not follow any documentation and test design techniques to create test cases.

### **Define what is a critical bug.**

A critical bug is a [bug](https://www.edureka.co/blog/bugs-in-software-testing/) that has got the tendency to affect a majority of the functionality of the given application. It means a large piece of functionality or major system component is completely broken and there is no workaround to move further. Application cannot be distributed to the end client unless the critical bug is addressed.

What is severity and Priority

**Severity** tells us how bad a defect is:” How much damage can it cause”?

**Priority** tells “how soon it is desired to fix the defect”…Should we fix this if so, by when?

**Example for Priority and Severity**

|  |  |  |
| --- | --- | --- |
| Priority | Severity | Severity |
|  | **Critical** | **Non-Critical** |
| **Urgent** | Key feature doesn’t work | Company logo is in wrong colour |
| **Low** | Feature that is rarely used doesn’t work | The caption on an image is written in the wrong font |

What is Risk Analaysis?

While a test plan is being created, risks involved in testing the product are to be taken into consideration along with possibility of their occurrence and the damage they may cause along with solutions; if any. Detailed study of this is called as Risk Analysis.

Three perspectives of Risk Assessment: Effect, Cause and Likelihood.

- To assess risk by Effect, identify a condition, event or action and try to determine its impact.

- To asses risk by Cause is opposite of by Effect. Begin by stating an undesirable event or condition and identify the set of events that could have permitted the condition to exist.

- To asses risk by Likelihood is to determine the probability that a requirement will not be satisfied.

Indicators of risk: High, Medium and low

High: means the effect of the risk would be very high and non-tolerable. Company may face severe loss and its reputation is at risk.Must be tested.

Medium: Tolerable but not desirable.Company may suffer financially but there is limited liability or loss of reputation. Should be tested

Low: tolerable. Little or no external exposure. Little or no financial loss. Company’s reputation unaffected. Might be tested.

Some risks could be:

* new Hardware
* New technology
* New automation tool
* Availability of test resources
* Time automation
* Limited Budget

Difference between release and build?

Build:

* A build is given by the dev team to the test team.
* A build when tested and certified by the test team is given the customers as “release”.
* A build can be rejected by test team if any of the tests fail/or doesn’t meet certain requirements

Release:

* A formal release of the product to its customers
* A release can have several builds associated with it.

### **What is black box testing, and what are the various techniques?**

[Black-Box Testing](https://www.edureka.co/blog/software-testing-methodologies-and-techniques/#BlackBoxTechniques), also known as specification-based testing, analyses the functionality of a software/application without knowing much about the internal structure/design of the item. The purpose of this testing is to check the system’s functionality as a whole to ensure that it works correctly and meets user demands. Various black-box testing techniques are:

* Equivalence Partitioning
* Boundary Value Analysis
* Decision Table Based Technique
* Cause-effect Graphing
* Use Case Testing

### **What is white box testing, and what are the various techniques?**

[White-Box Testing](https://www.edureka.co/blog/software-testing-methodologies-and-techniques/#WhiteBoxTechniques) also known as structure-based testing, requires a profound knowledge of the code as it includes testing of some structural part of the application. The purpose of this testing is to enhance security, check the flow of inputs/outputs through application and to improve design and usability. Various white-box testing techniques are:

* Statement Coverage
* Decision Coverage
* Condition Coverage
* Multiple Condition Coverage

### **What is Experience-based testing techniques?**

[Experienced-based testing](https://www.edureka.co/blog/software-testing-methodologies-and-techniques/#ExperienceBasedTechniques) is all about discovery, investigation, and learning. The tester constantly studies and analyzes the product and accordingly applies his skills, traits, and experience to develop test strategies and test cases to perform necessary testing. Various experience-based testing techniques are:

* Exploratory Testing
* Error Guessing

### **What is the difference between smoke testing and sanity testing?**

|  |  |  |
| --- | --- | --- |
| **Features** | **Smoke Testing** | **Sanity Testing** |
| **Coverage** | It verifies the entire system from end to end (Stability) | It verifies only a particular component in which code changes have been made (Rationality) |
| **Motive of Testing** | It Verifies the critical functionalities of the system | It verifies the new functionality like bug fixes |
| **Subset of?** | Is a subset of acceptance testing | Is a subset of regression testing |
| **Documentation** | Involves documentation and scripting work | Doesn’t emphasize any sort of documentation |
| **Test Coverage** | Shallow & wide approach to include all the major functionalities without going too deep | Narrow & deep approach involving detailed testing of functionalities and features |
| **Type of Testing** | It can be done either manually or by using automation tools | It is commonly executed manually rather than automation |
| **Performed By?** | Executed by developers or testers | Executed by testers |

### **What is the difference between static testing and dynamic testing?**

|  |  |
| --- | --- |
| **Static Testing** | **Dynamic Testing** |
| Static Testing is a white box testing technique, it includes the process of exploring the records to recognize the imperfections in the very early stages of SDLC. | Dynamic testing includes the process of execution of code and is done at the later stage of the software development lifecycle. |
| Static Testing is implemented at the verification stage. | Dynamic testing starts during the validation stage. |
| Static testing is performed before the code deployment. | Dynamic testing is performed after the code deployment |
| Executing the code is not a concern in this type of testing. | Execution of code is necessary for dynamic testing. |

**What are Entry Criteria, Exit Criteria and Suspension Criteria?**

**Test Criteria: A standard on which a test procedure can be based.**

**Entry Criteria:** It describes when to start testing. It gives the prerequisite items that must be completed before testing can begin.

**Exit Criteria:** It describes when to stop testing. It defines the items that must be completed before testing can be concluded

**Suspension Criteria:** It describes when to stop testing temporarily. If suspension criteria are met during testing, the active test cycle will be suspended util the criteria are resolved.

**What are Run Rate and Pass Rate?**

**Run rate** = Number of test cases executed/total test cases. [Note: 100% is mandatory unless clear vision is given]

**Pass rate** = Number of test cases passed/test cases executed [Note: It depends on project scope but achieving high pass rate is our goal]

**How to calculate Test Coverage**

**1)** The total lines of code in the piece of software quality you are testing

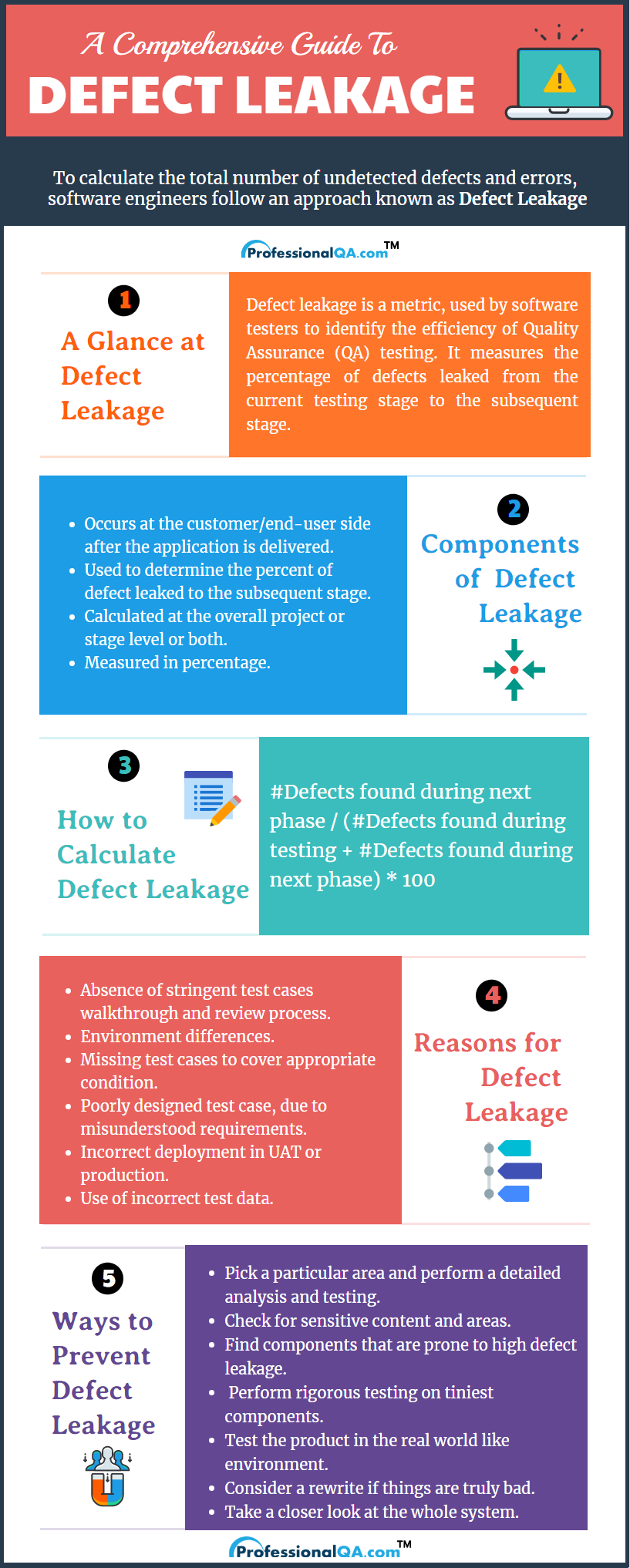
**2)**The number of lines of code all test cases currently execute

Now, you need to find (X divided by Y) multiplied by 100. The result of this calculation is your test coverage %.

**For example:**

If the number of lines of code in a system component is 500 and the number of lines executed across all existing test cases is 50, then your test coverage is:

(50 / 500) \* 100 = 10%



## Retesting vs Regression Testing

|  |  |
| --- | --- |
| **Regression Testing** | **Re-testing** |
| [Regression Testing](https://www.guru99.com/regression-testing.html) is carried out to confirm whether a recent program or code change has not adversely affected existing features | Re-testing is carried out to confirm the test cases that failed in the final execution are passing after the defects are fixed |
| The purpose of Regression Testing is that new code changes should not have any side effects to existing functionalities | Re-testing is done on the basis of the[Defect](https://www.guru99.com/defect-management-process.html)fixes |
| Based on the project and availability of resources, Regression Testing can be carried out parallel with Re-testing | Priority of re-testing is higher than regression testing, so it is carried out before regression testing |
| You can do automation for regression testing, [Manual Testing](https://www.guru99.com/manual-testing.html) could be expensive and time-consuming | You cannot automate the test cases for Retesting |
| Regression testing is known as a generic testing | Re-testing is a planned testing |
| Regression testing is done for passed test cases | Retesting is done only for failed test cases |

***REAL TIME QUESTIONS***

### **What are the drawbacks of manual testing?**

* Highly susceptible to human error and are risky
* Test types like Load testing and Performance testing are not possible manually
* Regression tests are really time-consuming if they are done manually
* Not suitable in very large organizations and time-bounded projects
* More expensive to test manually in the long run

### **What are some best practices that you should follow when writing test cases?**

* Prioritize which test cases to write based on the project timelines and the risk factors of your application.
* Remember the 80/20 rule. 20% of your tests should cover 80% of your application.
* Don’t try to write the test cases in one attempt instead improvise them as you progress.
* List down your test cases and classify them based on business scenarios and functionality.
* Make sure test cases are modular and test case steps are as granular as possible.
* Write test cases in such a way that others can understand them easily & modify if required.
* Always keep end-users’ requirements in the back of your mind because ultimately the software designed is for the customer
* Actively use a test management tool to manage stable release cycle.
* Monitor your test cases regularly. Write unique test cases and remove irrelevant & duplicate test cases.

### **What are the cases when you’ll consider to choose automated testing over manual testing?**

* When tests require periodic execution
* Tests include repetitive steps
* Tests need to be executed in a standard runtime environment
* When you have less time to complete the testing phase
* When there is a lot of code that needs to be repeatedly tested
* Reports are required for every execution

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

It is impossible to build a software product that is 100% bug-free. We can only minimize the error or fault in a computer program or system that causes it to produce an incorrect or unexpected result.

Here are the two principal reasons that make it impossible to test a program entirely.

* Software specifications can be subjective and can lead to different interpretations.
* A software program might require too many inputs, outputs, and path combinations to tests

### **Can automation testing replace manual testing?**

Automation testing isn’t a replacement for manual testing. No matter how good automated tests are, you cannot automate everything.

Manual tests play an important role in software development and come in handy whenever you have a case where you cannot use automation.

Automated and manual testing each have their own strengths and weaknesses.

Manual testing helps us understand the entire problem and more flexibly explore other angles of tests.

On the other hand, automated testing helps save time in the long run by accomplishing a large number of tests in a short time.

### **How will you determine when to stop testing?**

Some common factors in deciding when to stop testing are:

* Deadlines (release deadlines, testing deadlines, etc.)
* Test cases completed with certain percentage passed
* When the test budget is depleted
* Coverage of code or functionality or requirements reaches a specified point
* Bug rate falls below a certain level
* When Beta or alpha testing period ends

### **What makes a good test engineer?**

A **software** **test engineer** is a professional who determines how to create a process that would best test a particular product in the software industry.

* A good test engineer should have a ‘test to break’ attitude, need to think from customer point of view.
* Strong desire for quality and attention to minute details
* Need to maintain a cooperative relationship with developers
* Ability to communicate with both technical (developers) and non-technical (customers, management) people
* Prior experience in the software development industry is always a plus
* Need to make important decisions to test high-risk areas of an application when time is limited

### **When should you opt for manual testing over automation testing?**

* **Short-time projects:** Automated tests are aimed at saving time and resources yet it takes time and resources to design and maintain them. For example, if you are building a small promotional website, it can be much more efficient to rely on manual testing.
* **Ad-hoc Testing:** In ad-hoc testing, there is no specific approach. Ad-hoc testing is a totally unplanned method of testing where the understanding and insight of the tester is the only important factor. This can be achieved using manual testing.
* **Exploratory Test:** This type of testing requires the tester’s knowledge, experience, analytical, logical skills, creativity, and intuition. So human involvement is important in exploratory testing.
* **Usability Testing:** When performing usability testing, the tester needs to measure how user-friendly, efficient, or convenient the software or product is for the end-users. Human observation is the most important factor, so manual testing would be the appropriate one.

**List the items that cannot be automated using selenium?**

* Windows Apps/Mobile Apps/Mac Apps/Smart TV Apps
* Captcha Verification/Barcode/QR code verification/2 factor Authentication
* Performance testing/Security testing/API testing
* File Upload/File Download
* Exploratory/Usability/Accessibility

**Is Captcha can be automated in Selenium?**

No Captcha cannot be automated in Selenium

* Disable Captcha in test environments
* Making Captcha field as an optional field ONLY in test environment, so that if the Captcha step is skipped, test remains unimpacted

BEST PRACTICE TO HANDLE CAPTCHA

If you want to test the application as it is, i.e. without making any modifications to the testing environment for the purpose of automation, then separate all test scripts that involve captcha validation and package them as a single test suite. Run this test suite **with human intervention**. I know this is partial automation but it is not always possible to automate everything under a given set of circumstances. To get something you should be ready to forego something.

* 1. Entire test case except captcha can be automated.
  2. An implicit or an explicit wait can be used and user can be promped to enter the displayed captcha.
  3. When the test script is run, each step will be executed as usual. Once the prompt command is reached, a pop up appears on the browser. User enters the captcha as displayed on the screen.
  4. Once the captcha is entered manually, the test case execution resumes and the subsequent steps are run on the browser.

Reference for handling Captcha: https://javabeginnerstutorial.com/selenium/9q-webdriver-handling-captcha/