Manual Software Testing

- ✧ Manual testing is a software testing process in which test cases are executed manually without using any automated tool.
- ✧ All test cases executed by the tester manually according to the end user's perspective. It ensures whether the application is working, as mentioned in the requirement document or not.
- ✧ Test cases are planned and implemented to complete almost 100 percent of the software application.
- ✧ Test case reports are also generated manually.
- ✧ Manual Testing is one of the most fundamental testing processes as it can find both visible and hidden defects of the software.
- ✧ The difference between expected output and output, given by the software, is defined as a defect.
- ✧ The developer fixed the defects and handed it to the tester for retesting.
- ✧ Manual testing is mandatory for every newly developed software before automated testing.
- ✧ This testing requires great efforts and time, but it gives the surety of bug-free software.
- ✧ Manual Testing requires knowledge of manual testing techniques but not of any automated testing tool.
- ✧ Manual testing is essential because one of the software testing fundamentals is "100% automation is not possible."

Software Development Life Cycle (SDLC)

- ✧ SDLC is a process that creates a structure of development of software. There are different phases within SDLC, and each phase has its various activities.
- ✧ It makes the development team able to design, create, and deliver a high-quality product.
- ✧ SDLC describes various phases of software development and the order of execution of phases.
- ✧ Each phase requires deliverable from the previous phase in a life cycle of software development.
- ✧ Requirements are translated into design, design into development and development into testing; after testing, it is given to the client.

Phases of SDLC
1.Requirement Phase
2.Design Phase
3.Build /Development Phase
4.Testing Phase
5.Deployment/ Deliver Phase
6.Maintenance

1. Requirement Phase

- ✧ This is the most crucial phase of the software development life cycle for the developing team as well as for the project manager. During this phase, the client states requirements, specifications, expectations, and any other special requirement related to the product or software. All these are gathered by the business manager or project manager or analyst of the service providing company.
- ✧ The requirement includes how the product will be used and who will use the product to determine the load of operations. All information gathered from this phase is critical to developing the product as per the customer requirements.

2. Design Phase

- ✧ The design phase includes a detailed analysis of new software according to the requirement phase. This is the high priority phase in the development life cycle of a system because the logical designing of the system is converted into physical designing. The output of the requirement phase is a collection of things that are required, and the design phase gives the way to accomplish these requirements. The decision of all required essential tools such as programming language like Java, .NET, PHP, a database like Oracle, MySQL, a combination of hardware and software to provide a platform on which software can run without any problem is taken in this phase.
- ✧ There are several techniques and tools, such as data flow diagrams, flowcharts, decision tables, and decision trees, Data dictionary, and the structured dictionary are used for describing the system design.

3. Build /Development Phase

- ✧ After the successful completion of the requirement and design phase, the next step is to implement the design into the development of a software system. In this phase, work is divided into small units, and coding starts by the team of developers according to the design discussed in the previous phase and according to the requirements of the client discussed in requirement phase to produce the desired result.

- ✧ Front-end developers develop easy and attractive GUI and necessary interfaces to interact with back-end operations and back-end developers do back-end coding according to the required operations. All is done according to the procedure and guidelines demonstrated by the project manager.
- ✧ Since this is the coding phase, it takes the longest time and more focused approach for the developer in the software development life cycle.

4. Testing Phase

- ✧ Testing is the last step of completing a software system. In this phase, after getting the developed GUI and back-end combination, it is tested against the requirements stated in the requirement phase. Testing determines whether the software is actually giving the result as per the requirements addressed in the requirement phase or not. The Development team makes a test plan to start the test. This test plan includes all types of essential testing such as integration testing, unit testing, acceptance testing, and system testing. Non-functional testing is also done in this phase.
- ✧ If there are any defects in the software or it is not working as per expectations, then the testing team gives information to the development team in detail about the issue. If it is a valid defect or worth to sort out, it will be fixed, and the development team replaces it with the new one, and it also needs to be verified.

5. Deployment/ Deliver Phase

- ✧ When software testing is completed with a satisfying result, and there are no remaining issues in the working of the software, it is delivered to the customer for their use.
- ✧ As soon as customers receive the product, they are recommended first to do the beta testing. In beta testing, customer can require any changes which are not present in the software but mentioned in the requirement document or any other GUI changes to make it more user-friendly. Besides this, if any type of defect is encountered while a customer using the software; it will be informed to the development team of that particular software to sort out the problem. If it is a severe issue, then the development team solves it in a short time; otherwise, if it is less severe, then it will wait for the next version.
- ✧ After the solution of all types of bugs and changes, the software finally deployed to the end-user.

6. Maintenance

- ✧ The maintenance phase is the last and long-lasting phase of SDLC because it is the process which continues until the software's life cycle comes to an end. When a customer starts using software, then actual problems start to occur, and at that time there's a need to solve these problems. This phase also includes making changes in hardware and software to maintain its operational effectiveness like to improve its performance, enhance security features and according to customer's requirements with upcoming time. This process to take care of product time to time is called maintenance.

Software Testing Life Cycle (STLC)

The procedure of software testing is also known as STLC (Software Testing Life Cycle) which includes phases of the testing process.
The testing process is executed in a well-planned and systematic manner.
All activities are done to improve the quality of the software product.
Software testing life cycle contains the following steps:
1.Requirement Analysis
2.Test Plan Creation
3.Environment setup
4.Test case Execution
5.Defect Logging
6.Test Cycle Closure

Requirement Analysis:

- ✧ The first step of the manual testing procedure is requirement analysis.
- ✧ In this phase, tester analyses requirement document of SDLC (Software Development Life Cycle) to examine requirements stated by the client.
- ✧ After examining the requirements, the tester makes a test plan to check whether the software is meeting the requirements or not.

Test Plan Creation:

- ✧ Test plan creation is the crucial phase of STLC where all the testing strategies are defined. Tester determines the estimated effort and cost of the entire project.
- ✧ This phase takes place after the successful completion of the Requirement Analysis Phase.
- ✧ Testing strategy and effort estimation documents provided by this phase.
- ✧ Test case execution can be started after the successful completion of Test Plan Creation.

Environment setup:

- ✧ Setup of the test environment is an independent activity and can be started along with Test Case Development.
- ✧ This is an essential part of the manual testing procedure as without environment testing is not possible.
- ✧ Environment setup requires a group of essential software and hardware to create a test environment.
- ✧ The testing team is not involved in setting up the testing environment, its senior developers who create it.

Test case Execution:

- ✧ Test case Execution takes place after the successful completion of test planning.
- ✧ In this phase, the testing team starts case development and execution activity.
- ✧ The testing team writes down the detailed test cases, also prepares the test data if required. The prepared test cases are reviewed by peer members of the team or Quality Assurance leader.
- ✧ RTM (Requirement Traceability Matrix) is also prepared in this phase.
- ✧ Requirement Traceability Matrix is industry level format, used for tracking requirements. Each test case is mapped with the requirement specification.
- ✧ Backward & forward traceability can be done via RTM.

Defect Logging:

- ✧ Testers and developers evaluate the completion criteria of the software based on test coverage, quality, time consumption, cost, and critical business objectives.
- ✧ This phase determines the characteristics and drawbacks of the software.
- ✧ Test cases and bug reports are analyzed in depth to detect the type of defect and its severity.
- ✧ Defect logging analysis mainly works to find out defect distribution depending upon severity and types.
- ✧ If any defect is detected, then the software is returned to the development team to fix the defect, then the software is re-tested on all aspects of the testing.
- ✧ Once the test cycle is fully completed then test closure report, and test metrics are prepared.

Test Cycle Closure:

- ✧ The test cycle closure report includes all the documentation related to software design, development, testing results, and defect reports.
- ✧ This phase evaluates the strategy of development, testing procedure, possible defects in order to use these practices in the future if there is a software with the same specification.

Test Case

A TEST CASE is a documented set of preconditions (prerequisites), procedures (inputs / actions) and post-conditions (expected results) which a tester uses to determine whether a system under test satisfies requirements or works correctly. A test case can have one or multiple test scripts and a collection of test cases is called a test suite.
Template for Test Case Writing

1. Test Scenario ID
2. Test Scenario Description
3. Test Case ID
4. Test Case Description
5. Steps To Follow
6. Expected Result
7. Actual Result.

Defect Life Cycle State

The different states of a bug in the bug life cycle are as follows:

New

When a tester finds a new defect. He should provide a proper Defect document to the Development team to reproduce and fix the defect. In this state, the status of the defect posted by the tester is “New”

Assigned
Defects that are in the status of New will be approved (if valid) and assigned to the development team by Test Lead/Project Lead/Project Manager. Once the defect is assigned then the status of the bug changes to "Assigned"

Open
The development team starts analyzing and works on the defect fix

Fixed
When a developer makes the necessary code change and verifies the change, then the status of the bug will be changed as "Fixed" and the bug is passed to the testing team.

Test
If the status is "Test", it means the defect is fixed and ready to do test whether it is fixed or not.

Verified
The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is "verified."

Closed
After verified the fix, if the bug is no longer exits then the status of the bug will be assigned as "Closed."

Reopen
If the defect remains the same after the retest, then the tester posts the defect using the defect retesting document and changes the status to "Reopen". Again the bug goes through the life cycle to be fixed.

Duplicate
If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to "duplicate" by the development team.

Deferred
In some cases, the Project Manager/Lead may set the bug status as deferred.
If the bug found during the end of the release and the bug is minor or not important to fix immediately.
If the bug is not related to the current build.
If it is expected to get fixed in the next release.
The customer is thinking to change the requirement.
In such cases the status will be changed as "deferred" and it will be fixed in the next release.

Rejected
If the system is working according to specifications and the bug is just due to some misinterpretation (such as referring to old requirements or extra features) then the Team lead or developers can mark such bugs as "Rejected"
Some other statuses are:

Cannot be fixed
Technology not supporting, Root of the product issue, Cost of fixing a bug is more

Not Reproducible
Platform mismatch, improper defect document, data mismatch, build mismatch, inconsistent defects

Traceability Matrix (TM)

- ✧ A Traceability Matrix is a document that co-relates any two-baseline documents that require a many-to-many relationship to check the completeness of the relationship.
- ✧ It is used to track the requirements and to check the current project requirements are met.

Requirement Traceability Matrix

- ✧ Requirement Traceability Matrix (RTM) is a document that maps and traces user requirement with test cases.
- ✧ It captures all requirements proposed by the client and requirement traceability in a single document, delivered at the conclusion of the Software devlopement life cycle.

- ✧ 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.

Quality Assurance In Software Testing

Quality Assurance is popularly known as QA Testing, is defined as an activity to ensure that an organization is providing the best possible product or service to customers.

- ✧ It is a procedure that focuses on providing assurance that quality requested will be achieved.
- ✧ QA aims to prevent the defect
- ✧ It is a method to manage the quality- Verification
- ✧ It does not involve executing the program.
- ✧ It's a Preventive technique
- ✧ It's a Proactive measure
- ✧ It is the procedure to create the deliverables
- ✧ QA involves in full software development life cycle
- ✧ In order to meet the customer requirements, QA defines standards and methodologies
- ✧ It is performed before Quality Control
- ✧ It is a Low-Level Activity, it can identify an error and mistakes which QC cannot
- ✧ Its main motive is to prevent defects in the system. It is a less time-consuming activity
- ✧ QA ensures that everything is executed in the right way, and that is why it falls under verification activity
- ✧ It requires the involvement of the whole team
- ✧ The statistical technique applied on QA is known as SPC or Statistical Process Control (SPC)

Quality Control in Software Testing

Quality Control in Software Testing is a systematic set of processes used to ensure the quality of software products or services. The main purpose of the quality control process is ensuring that the software product meets the actual requirements by testing and reviewing its functional and non-functional requirements. Quality control is popularly abbreviated as QC.

- ✧ It is a procedure that focuses on fulfilling the quality requested.
- ✧ QC aims to identify and fix defects
- ✧ It is a method to verify the quality-Validation
- ✧ It always involves executing a program
- ✧ It's a Corrective technique
- ✧ It's a Reactive measure
- ✧ It is the procedure to verify that deliverables
- ✧ QC involves in full software testing life cycle
- ✧ QC confirms that the standards are followed while working on the product
- ✧ It is performed only after QA activity is done
- ✧ It is a High-Level Activity, it can identify an error that QA cannot
- ✧ Its main motive is to identify defects or bugs in the system. It is a more time-consuming activity
- ✧ QC ensures that whatever we have done is as per the requirement, and that is why it falls under validation activity
- ✧ It requires the involvement of the Testing team
- ✧ The statistical technique applied to QC is known as SQC or Statistical Quality Control

KEY DIFFERENCE in Quality Control and Quality Assurance

1)Quality Assurance is aimed to avoid the defect whereas Quality control is aimed to identify and fix the defects.
2)Quality Assurance provides assurance that quality requested will be achieved whereas Quality Control is a procedure that focuses on fulfilling the quality requested.
3)Quality Assurance is done in software development life cycle whereas Quality Control is done in software testing life cycle.
4)Quality Assurance is a proactive measure whereas Quality Control is a Reactive measure.
5)Quality Assurance requires the involvement of all team members whereas Quality Control needs only testing team.
6)Quality Assurance is performed before Quality Control.

Verification in Software Testing

- Verification in Software Testing is a process of checking documents, design, code, and program in order to check if the software has been built according to the requirements or not.
- The main goal of verification process is to ensure quality of software application, design, architecture etc.
- The verification process involves activities like reviews, walk-throughs and inspection.
- Validation in Software Testing
- Validation in Software Testing is a dynamic mechanism of testing and validating if the software product actually meets the exact needs of the customer or not.
- The process helps to ensure that the software fulfills the desired use in an appropriate environment.
- The validation process involves activities like unit testing, integration testing, system testing and user acceptance testing.

KEY DIFFERENCE Between Verification and Validation

- Verification process includes checking of documents, design, code and program whereas Validation process includes testing and validation of the actual product.
- Verification does not involve code execution while Validation involves code execution.
- Verification uses methods like reviews, walkthroughs, inspections and desk-checking whereas Validation uses methods like black box testing, white box testing and non-functional testing.
- Verification checks whether the software confirms a specification whereas Validation checks whether the software meets the requirements and expectations.
- Verification finds the bugs early in the development cycle whereas Validation finds the bugs that verification can not catch.
- Verification process targets on software architecture, design, database, etc. while Validation process targets the actual software product.
- Verification is done by the QA team while Validation is done by the involvement of testing team with QA team.
- Verification process comes before validation whereas Validation process comes after verification.

Types of Manual Testing

There are various methods used for manual testing. Each technique is used according to its testing criteria. Types of manual testing are given below:

1) White Box Testing
2) Black Box Testing
3) Gray Box Testing

White-box testing

The white box testing is done by Developer, where they check every line of a code before giving it to the Test Engineer. Since the code is visible for the Developer during the testing, that's why it is also known as White box testing.

Black box testing

The black box testing is done by the Test Engineer, where they can check the functionality of an application or the software according to the customer /client's needs. In this, the code is not visible while performing the testing; that's why it is known as black-box testing.

Gray Box testing

Gray box testing is a combination of white box and Black box testing. It can be performed by a person who knew both coding and testing. And if the single person performs white box, as well as black-box testing for the application, is known as Gray box testing.

Black Box Testing

- Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding. The primary source of black box testing is a specification of requirements that is stated by the customer.
- In this method, tester selects a function and gives input value to examine its functionality, and checks whether the function is giving expected output or not.

- If the function produces correct output, then it is passed in testing, otherwise failed. The test team reports the result to the development team and then tests the next function.
- After completing testing of all functions if there are severe problems, then it is given back to the development team for correction.

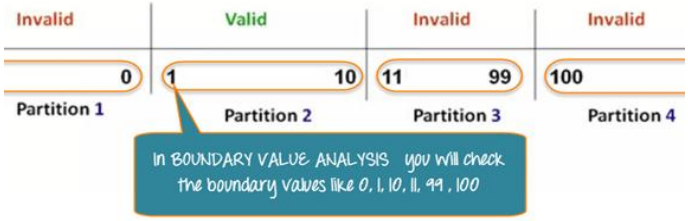
Generic steps of black box testing

- The black box test is based on the specification of requirements, so it is examined in the beginning.
- In the second step, the tester creates a positive test scenario and an adverse test scenario by selecting valid and invalid input values to check that the software is processing them correctly or incorrectly.
- In the third step, the tester develops various test cases such as decision table, all pairs test, equivalent division, error estimation, cause-effect graph, etc.
- The fourth phase includes the execution of all test cases.
- In the fifth step, the tester compares the expected output against the actual output.
- In the sixth and final step, if there is any flaw in the software, then it is cured and tested again.

Techniques of Black Box Testing

The following are the techniques employed while using Black box testing for a software application.

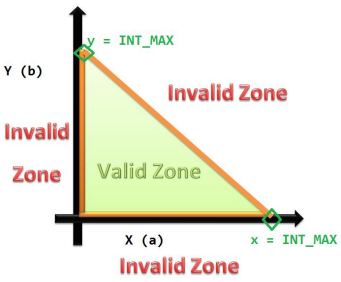
1.BVA or Boundary Value Analysis:



- It is one among the useful and critical Black box testing technique that helps in equivalence partitioning. BVA helps in testing any software having a boundary or extreme values.
- This technique is capable of identifying the flaws of the limits of the input values rather than focusing on the range of input value.
- Boundary Value Analysis also deals with edge or extreme output values.

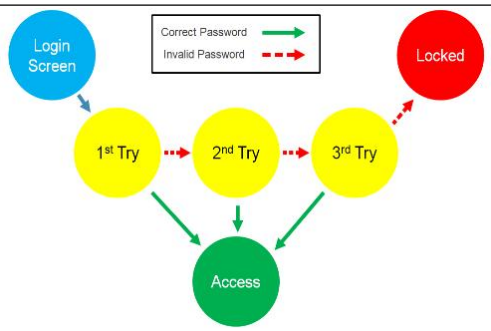
Equivalence Class Partitioning:

- This technique of Black box testing is widely used to write test cases. It can be useful in reducing a broad set of possible inputs to smaller but effective ones.
- It is performed through the division of inputs as classes, and each class is given a value.
- It is applied when the need for exhaustive testing arises and for resisting the redundancy of inputs.



State Transition Testing

- This technique usually considers the state, outputs, and inputs of a system during a specific period.
- Based on the type of software that is tested, it checks for the behavioral changes of a system in a particular state or another state while maintaining the same inputs.
- The test cases for this technique are created by checking the sequence of transitions and state or events among the inputs.
- The whole set of test cases will have the traversal of the expected output values and all states.



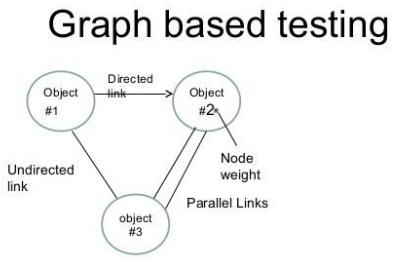
Decision Table Testing:

- ✧ In some instances, the inputs combinations can become very complicated for tracking several possibilities.
- ✧ Such complex situations rely on decision tables, as it offers the testers an organized view about the inputs combination and the expected output.
- ✧ This technique is identical to the graph-based testing technique; the major difference is using tables instead of diagrams or graphs.

	Rule 1	Rule 2	Rule 3	Rule 4
Condition				
End of month	No	Yes	Yes	Yes
Salary Transferred	N/A	No	Yes	Yes
Provident fund	N/A	N/A	No	Yes
Action				
Income tax	No	No	Yes	Yes
Provident fund	No	No	No	Yes

Graph-Based Testing:

- ✧ This technique of Black box testing involves a graph drawing that depicts the link between the causes (inputs) and the effects (output), which trigger the effects.
- ✧ This testing utilizes different combinations of output and inputs.
- ✧ It is a helpful technique to understand the software's functional performance, as it visualizes the flow of inputs and outputs in a lively fashion.

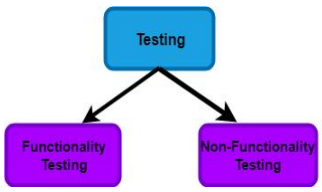


Error Guessing Technique:

- ✧ This testing technique is capable of guessing the erroneous output and inputs to help the tester fix it easily.
- ✧ It is solely based on judgment and perception of the earlier end user experience.
- ✧ Apart from the above-explained popular techniques of this testing, there are few more, such as the fuzzing technique, all pair testing and orthogonal array testing.
- ✧

Types of BlackBox Testing

Software testing is a technique to check whether the actual result matches the expected result and to ensure that the software has not any defect or bug.



Functional Testing

- ✧ It is a type of software testing which is used to verify the functionality of the software application, whether the function is working according to the requirement specification.

- In functional testing, each function tested by giving the value, determining the output, and verifying the actual output with the expected value.
- Functional testing performed as black-box testing which is presented to confirm that the functionality of an application or system behaves as we are expecting.
- It is done to verify the functionality of the application.
- Functional testing also called as black-box testing, because it focuses on application specification rather than actual code.
- Tester has to test only the program rather than the system.

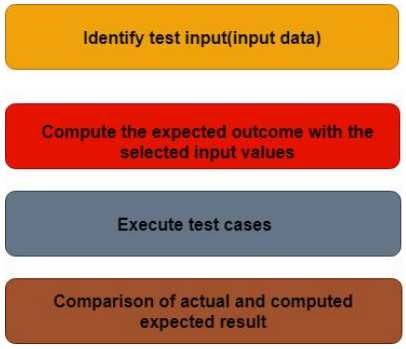
The Process of Functional Testing

- Testers follow the following steps in the functional testing:
- Tester does verification of the requirement specification in the software application.
- After analysis, the requirement specification tester will make a plan.
- After planning the tests, the tester will design the test case.
- After designing the test, case tester will make a document of the traceability matrix.
- The tester will execute the test case design.
- Analysis of the coverage to examine the covered testing area of the application.
- Defect management should do to manage defect resolving.

The complete process to perform functional testing

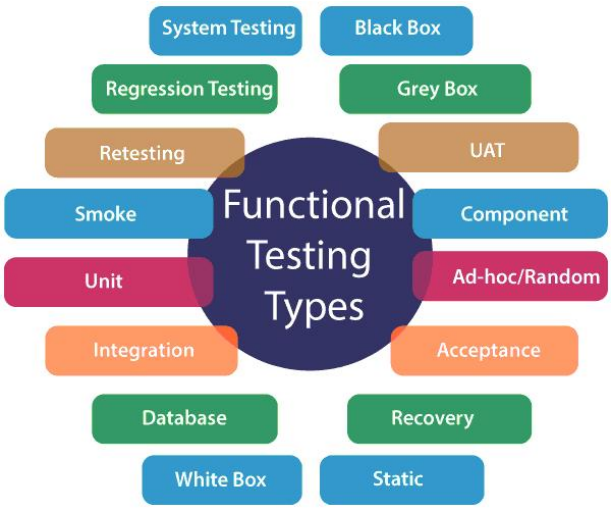
There are the following steps to perform functional testing:

- There is a need to understand the software requirement.
- Identify test input data
- Compute the expected outcome with the selected input values.
- Execute test cases
- Comparison between the actual and the computed result
-



Types of Functional Testing

The main objective of functional testing is to test the functionality of the component.
Functional testing is divided into multiple parts.
Here are the following types of functional testing.



Unit Testing: Unit testing is a type of software testing, where the individual unit or component of the software tested. Unit testing, examine the different part of the application, by unit testing functional testing also done, because unit testing ensures each module is working correctly.
The developer does unit testing. Unit testing is done in the development phase of the application.

Smoke Testing: Functional testing by smoke testing. Smoke testing includes only the basic (feature) functionality of the system. Smoke testing is known as "Build Verification Testing." Smoke testing aims to ensure that the most important function work.
For example, Smoke testing verifies that the application launches successfully will check that GUI is responsive.

Sanity Testing: Sanity testing involves the entire high-level business scenario is working correctly. Sanity testing is done to check the functionality/bugs fixed. Sanity testing is little advance than smoke testing.
For example, login is working fine; all the buttons are working correctly; after clicking on the button navigation of the page is done or not.

Regression Testing: This type of testing concentrate to make sure that the code changes should not side effect the existing functionality of the system. Regression testing specifies when bug arises in the system after fixing the bug, regression testing concentrate on that all parts are working or not. Regression testing focuses on is there any impact on the system.

Integration Testing:
Integration testing combined individual units and tested as a group. The purpose of this testing is to expose the faults in the interaction between the integrated units.
Developers and testers perform integration testing.

User Acceptance Testing:
It is a type of testing performed by the client to certify the system according to requirement. The final phase of testing is user acceptance testing before releasing the software to the market or production environment. UAT is a kind of black-box testing where two or more end-users will involve.

Retesting:
Retesting is a type of testing performed to check the test cases that were unsuccessful in the final execution are successfully pass after the defects fixed. Usually, tester assigns the bug when they find it while testing the product or its component. The bug allocated to a developer, and he fixes it. After fixing, the bug is assigned to a tester for its verification. This testing is known as retesting.

Database Testing:
Database testing is a type of testing which checks the schema, tables, triggers, etc. of the database under test. Database testing may involve creating complex queries to load/stress test the database and check its responsiveness. It checks the data integrity and consistency.
Example: let us consider a banking application whereby a user makes a transaction. Now from database testing following, things are important. They are:
Application store the transaction information in the application database and displays them correctly to the user.
No information lost in this process
The application does not keep partially performed or aborted operation information.
The user information is not allowed individuals to access by the

Ad-hoc Testing:
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.

Recovery Testing:
Recovery testing is used to define how well an application can recover from crashes, hardware failure, and other problems. The purpose of recovery testing is to verify the system's ability to recover from testing points of failure.

Static Testing:
Static testing is a software testing technique by which we can check the defects in software without actually executing it. Static testing is done to avoid errors in the early stage of the development as it is easier to find failure in the early stages. Static testing used to detect the mistakes that may not found in dynamic testing.
Static testing helps to find the error in the early stages. With the help of static testing, this will reduce the development timescales. It reduces the testing cost and time. Static testing also used for development productivity.

Component Testing:
Component Testing is also a type of software testing in which testing is performed on each component separately without integrating with other parts. Component testing is also a type of black-box testing. Component testing also referred to as Unit testing, program testing, or module testing.

Non-Functional Testing

- ✧ Non-Functional testing is a software testing technique that verifies the attributes of the system such as memory leaks, performance or robustness of the system.
- ✧ Non-Functional testing is performed at all test levels.
- ✧ It covers all the areas that are not covered in functional testing.
- ✧ It checks the attributes such as memory leaks, performance, or robustness of the system.
- ✧ In simple words, how well the system performs is non-functionality testing.

Non-functional Testing Parameters

Non Functional Testing Parameters

1) Security:
The parameter defines how a system is safeguarded against deliberate and sudden attacks from internal and external sources. This is tested via Security Testing.

2) Reliability:
The extent to which any software system continuously performs the specified functions without failure. This is tested by Reliability Testing

3) Survivability:
The parameter checks that the software system continues to function and recovers itself in case of system failure. This is checked by Recovery Testing

4) Availability:
The parameter determines the degree to which user can depend on the system during its operation. This is checked by Stability Testing.

5) Usability:
The ease with which the user can learn, operate, prepare inputs and outputs through interaction with a system. This is checked by Usability Testing

6) Scalability:
The term refers to the degree in which any software application can expand its processing capacity to meet an increase in demand. This is tested by Scalability Testing.

7) Interoperability:
This non-functional parameter checks a software system interfaces with other software systems. This is checked by Interoperability Testing.

8) Efficiency:
The extent to which any software system can handles capacity, quantity and response time.

9) Flexibility:
The term refers to the ease with which the application can work in different hardware and software configurations. Like minimum RAM, CPU requirements.

10) Portability:
The flexibility of software to transfer from its current hardware or software environment.

11) Reusability:
It refers to a portion of the software system that can be converted for use in another application.

Non-Functional Testing Types:

1. Compatibility Testing
2. Install Testing
3. Load Testing
4. Performance Testing
5. Security Testing
6. Stress Testing
7. Usability Testing
8. Volume Testing

Compatibility Testing
- ✧ It is a non-functional testing conducted on the application to evaluate the application's compatibility within different environments. It can be of two types - forward compatibility testing and backward compatibility testing.
- ✧ Operating system Compatibility Testing - Linux , Mac OS, Windows
- ✧ Database Compatibility Testing - Oracle SQL Server
- ✧ Browser Compatibility Testing - IE , Chrome, Firefox
- ✧ Other System Software - Web server, networking/ messaging tool, etc.

Installation Testing:
It is performed to verify if the software has been installed with all the necessary components and the application is working as expected.
This is very important as installation would be the first user interaction with the end users.

Security Testing:
Security Testing is the process which checks whether the confidential data stays confidential or not (i.e. it is not exposed to individuals/ entities for which it is not meant for) and the users can perform only those tasks that they are authorized to perform.
For Example, a user should not be able to deny the functionality of the website to other users or a user should not be able to change the functionality of the web application in an unintended way etc.

Load Testing
- ✧ Load testing is one of the type of performance testing using which we evaluate the performance of an application under expected real-world load.
- ✧ Virtual users are created to simulate a load of multiple concurrent users accessing the application.
- ✧ After subjecting application to the virtual user load, we eventually measure the different performance attributes along with identifying performance bottlenecks.
- ✧ The virtual user creation is performed by tools like – JMeter and LoadRunner.

- ✧ These performance testing tools allow us to create scripts that makes different requests to the server(just like a real-world user), along with various configurations like a number of threads or virtual users, duration of load test, performance attributes & graphs that we want to analyze etc.
- ✧ After the script creation and setting up of test configuration, we can run the load test and analyze the test results.

Performance Testing :

Performance Testing is the process of analyzing the quality and capability of a product.
It is a testing method performed to determine the system performance in terms of speed, reliability and stability under varying workload.
Performance testing is also known as Perf Testing

Stress Testing :

Stress testing a Non-Functional testing technique that is performed as part of performance testing.
During stress testing, the system is monitored after subjecting the system to overload to ensure that the system can sustain the stress.
The recovery of the system from such phase (after stress) is very critical as it is highly likely to happen in production environment.

Usability Testing :

Usability testing is a method of testing the functionality of a website, app, or other digital product by observing real users as they attempt to complete tasks on it.
The users are usually observed by researchers working for a business.

Volume Testing

- ✧ Volume Testing, aka Flood Testing, is a non-functional test used to see the software or application's performance when introduced to a high volume of data.
- ✧ The volume here refers to the size of the database or the file subjected to the test.
- ✧ Under the volume testing, the developers will keep adding data until the database reaches its threshold.
- ✧ Then the system will be analyzed for its response.
- ✧ For example, you want to add 1000 new products under the "TV" category on your eCommerce site.
- ✧ Before adding those entries into the database, you must ensure whether your site can handle such an extensive database.
- ✧ This is where Volume Testing can help.