**Manual Testing Vs. Automation Testing**

| **Parameter** | **Automation Testing** | **Manual Testing** |
| --- | --- | --- |
| Definition | Automation Testing uses automation tools to execute test cases. | In manual testing, test cases are executed by a human tester and software. |
| Processing time | Automated testing is significantly faster than a manual approach. | Manual testing is time-consuming and takes up human resources. |
| Exploratory Testing | Automation does not allow random testing | Exploratory testing is possible in Manual Testing |
| Initial investment | The initial investment in the automated testing is higher. Though the ROI is better in the long run. | The initial investment in the Manual testing is comparatively lower. ROI is lower compared to Automation testing in the long run. |
| Reliability | Automated testing is a reliable method, as it is performed by tools and scripts. There is no testing Fatigue. | Manual testing is not as accurate because of the possibility of the human errors. |
| UI Change | For even a trivial change in the UI of the AUT, Automated Test Scripts need to be modified to work as expected | Small changes like change in id, class, etc. of a button wouldn’t thwart execution of a manual tester. |
| Investment | Investment is required for testing tools as well as automation engineers | Investment is needed for human resources. |
| Cost-effective | Not cost effective for low volume regression | Not cost effective for high volume regression. |
| Test Report Visibility | With automation testing, all stakeholders can login into the automation system and check test execution results | Manual Tests are usually recorded in an Excel or Word, and test results are not readily/ readily available. |
| Human observation | Automated testing does not involve human consideration. So it can never give assurance of user-friendliness and positive customer experience. | The manual testing method allows human observation, which may be useful to offer user-friendly system. |
| Performance Testing | Performance Tests like Load Testing, Stress Testing, Spike Testing, etc. have to be tested by an automation tool compulsorily. | Performance Testing is not feasible manually |
| Parallel Execution | This testing can be executed on different operating platforms in parallel and reduce test execution time. | Manual tests can be executed in parallel but would need to increase your human resource which is expensive |
| Batch testing | You can Batch multiple Test Scripts for nightly execution. | Manual tests cannot be batched. |
| Programming knowledge | Programming knowledge is a must in automation testing. | No need for programming in Manual Testing. |
| Set up | Automation test requires less complex test execution set up. | Manual testing needs have a more straightforward test execution setup |
| Engagement | Done by tools. Its accurate and never gets bored! | Repetitive Manual Test Execution can get boring and error-prone. |
| Ideal approach | Automation testing is useful when frequently executing the same set of test cases | Manual testing proves useful when the test case only needs to run once or twice. |
| Build Verification Testing | Automation testing is useful for Build Verification Testing (BVT). | Executing the Build Verification Testing (BVT) is very difficult and time-consuming in manual testing. |
| Deadlines | Automated Tests have zero risks of missing out a pre-decided test. | Manual Testing has a higher risk of missing out the pre-decided test deadline. |
| Framework | Automation testing uses frameworks like Data Drive, Keyword, Hybrid to accelerate the automation process. | Manual Testing does not use frameworks but may use guidelines, checklists, stringent processes to draft certain test cases. |
| Documentation | Automated Tests acts as a document provides training value especially for automated unit test cases. A new developer can look into a unit test cases and understand the code base quickly. | Manual Test cases provide no training value |
| Test Design | Automated Unit Tests enforce/drive Test Driven Development Design. | Manual Unit Tests do not drive design into the coding process |
| Devops | Automated Tests help in Build Verification Testing and are an integral part of DevOps Cycle | Manual Testing defeats the automated build principle of DevOps |
| When to Use? | Automated Testing is suited for Regression Testing, Performance Testing, Load Testing or highly repeatable functional test cases. | Manual Testing is suitable for Exploratory, Usability and Adhoc Testing. It should also be used where the AUT changes frequently. |

## SDLC vs STLC

|  |  |
| --- | --- |
| SDLC | STLC |
| Development Life Cycle | Testing Life Cycle |
| The main object of SDLC life cycle is to complete successful development of the software including testing and other phases. | The only objective of the STLC phase is testing. |
| In SDLC the business analyst gathers the requirements and create Development Plan | In STLC, the QA team analyze requirement documents like functional and non-functional documents and create System Test Plan |
| In SDLC, the development team creates the high and low-level design plans | In STLC, the test analyst creates the Integration Test Plan |
| The real code is developed, and actual work takes place as per the design documents. | The testing team prepares the test environment and executes them |
| SDLC phase also includes post-deployment supports and updates. | Testers, execute regression suits, usually automation scripts to check maintenance code deployed. |

## Static Testing Vs. Dynamic Testing

|  |  |
| --- | --- |
| **Static Testing** | **Dynamic Testing** |
| Testing is done without executing the program | Testing is done by executing the program |
| This testing does the verification process | Dynamic testing does the validation process |
| Static testing is about prevention of defects | Dynamic testing is about finding and fixing the defects |
| Static testing gives an assessment of code and documentation | Dynamic testing gives bugs/bottlenecks in the software system. |
| Static testing involves a checklist and process to be followed | Dynamic testing involves test cases for execution |
| This testing can be performed before compilation | Dynamic testing is performed after compilation |
| Static testing covers the structural and statement coverage testing | Dynamic testing techniques are Boundary Value Analysis & Equivalence Partitioning. |
| Cost of finding defects and fixing is less | Cost of finding and fixing defects is high |
| Return on investment will be high as this process involved at an early stage | Return on investment will be low as this process involves after the development phase |
| More reviews comments are highly recommended for good quality | More defects are highly recommended for good quality. |
| Requires loads of meetings | Comparatively requires lesser meetings |

## Verification vs Validation

|  |  |
| --- | --- |
| **Verification** | **Validation** |
| The verifying process includes checking documents, design, code, and program | It is a dynamic mechanism of testing and validating the actual product |
| It does **not** involve executing the code | It always involves executing the code |
| Verification uses methods like reviews, walkthroughs, inspections, and desk- checking etc. | It uses methods like Black Box Testing, [White Box Testing](https://www.guru99.com/white-box-testing.html), and non-functional testing |
| Whether the software conforms to specification is checked | It checks whether the software meets the requirements and expectations of a customer |
| It finds bugs early in the development cycle | It can find bugs that the verification process can not catch |
| Target is application and software architecture, specification, complete design, high level, and database design etc. | Target is an actual product |
| QA team does verification and make sure that the software is as per the requirement in the SRS document. | With the involvement of testing team validation is executed on software code. |
| It comes before validation | It comes after verification |

## Positive Testing VS Negative Testing

|  |  |
| --- | --- |
| Positive Testing | Negative Testing |
| Checking the application response with the help of valid input data is known as positive testing. | Checking the application response by using the invalid input data set is known as negative testing. |
| Positive testing is implemented only for the expected conditions. | Negative testing is implemented only for unexpected conditions. |
| Positive testing doesn't guarantee a good quality of software product. | Negative testing guarantees to deliver a good quality of software product. |
| The execution of positive testing takes less time as compared to negative testing. | The execution of positive testing takes more time as compared to positive testing. |
| To validate the available set of test conditions, we will consistently implement the Positive testing. | To break the project and product with an unidentified set of test conditions, we will consistently implement Negative testing. |
| The primary purpose of executing Positive testing is to guarantee that the software application always meets the developer's requirements and specifications. | The primary purpose of executing the negative testing is to test a web application's constancy in contradiction to inaccurate validation data sets. |
| Positive testing doesn't encompass all the possible cases. | Negative testing encompasses all the possible cases. |
| It is a process where the system is validated in contradiction of the valid input data. | It is a testing process which contain the validation in contrast to invalid input data. |
| Positive testing is less significant than Negative testing. | Negative testing is more vital than Positive testing. |
| Positive testing can be implemented on every application. | Negative testing can be implemented when the possibilities of unpredicted conditions. |
| The people having less knowledge can execute the positive testing. | The testing professionals can execute the negative testing. |
| It makes sure that the software is standard. | Negative testing makes sure to deliver 100 percent bug-free software. |

## Black Box Testing Vs. White Box Testing

| **Black Box testing** | **White Box testing** |
| --- | --- |
| It is a testing approach which is used to test the software without the knowledge of the internal structure of program or application. | It is a testing approach in which internal structure is known to the tester. |
| It also knowns as data-driven, box testing, data-, and functional testing. | It is also called structural testing, clear box testing, code-based testing, or glass box testing. |
| Testing is based on external expectations; internal behavior of the application is unknown. | Internal working is known, and the tester can test accordingly. |
| This type of testing is ideal for higher levels of testing like [System Testing](https://www.guru99.com/system-testing.html), Acceptance testing. | Testing is best suited for a lower level of testing like [Unit Testing](https://www.guru99.com/unit-testing-guide.html), Integration testing. |
| Programming knowledge is not needed to perform Black Box testing. | Programming knowledge is required to perform White Box testing. |
| Implementation knowledge is not requiring doing Black Box testing. | Complete understanding needs to implement WhiteBox testing. |
| Test and programmer are dependent on each other, so it is tough to automate. | White Box testing is easy to automate. |
| The main objective of this testing is to check what functionality of the system under test. | The main objective of White Box testing is done to check the quality of the code. |
| Testing can start after preparing requirement specification document. | Testing can start after preparing for Detail design document. |
| Performed by the end user, developer, and tester. | Usually done by tester and developers. |
| Granularity is low. | Granularity is high. |
| It is based on trial and error method. | Data domain and internal boundaries can be tested. |
| It is less exhaustive and time-consuming. | Exhaustive and time-consuming method. |
| Not the best method for algorithm testing. | Best suited for algorithm testing. |
| Code access is not required for Black Box Testing. | White box testing requires code access. Thereby, the code could be stolen if testing is outsourced. |
| Well suited and efficient for large code segments. | It allows removing the extra lines of code, which can bring in hidden defects. |
| Low skilled testers can test the application with no knowledge of the implementation of programming language or operating system. | Need an expert tester with vast experience to perform white box testing. |
| Equivalence partitioning is Black box testing technique is used for Blackbox testing.  Equivalence partitioning divides input values into valid and invalid partitions and selecting corresponding values from each partition of the test data.  Boundary value analysis  checks boundaries for input values. | Statement Coverage, Branch coverage, and Path coverage are White Box testing technique.  Statement Coverage validates whether every line of the code is executed at least once.  Branch coverage validates whether each branch is executed at least once  Path coverage method tests all the paths of the program. |
| Update to automation test script is essential if you to modify application frequently. | [Automated test](https://www.guru99.com/automation-testing.html) cases can become useless if the code base is rapidly changing. |

## Functional Testing vs Non Functional Testing

|  |  |
| --- | --- |
| Functional | Non-functional testing |
| It is performed before non-functional testing. | It is performed after the functional testing. |
| It is based on customer’s requirements. | It focusses on customer’s expectation. |
| It is easy to define functional requirements. | It is difficult to define the requirements for non-functional testing. |
| Helps to validate the behavior of the application. | Helps to validate the performance of the application. |
| Carried out to validate software actions. | It is done to validate the performance of the software. |
| Functional testing is carried out using the functional specification. | This kind of testing is carried out by performance specifications |
| Functional testing is easy to execute by manual testing. | It’s very hard to perform non-functional testing manually. |
| It describes what the product does. | It describes how the product works. |
| Check login functionality. | The dashboard should load in 2 seconds. |

### **Monkey Testing vs Gorilla Testing**

|  |  |
| --- | --- |
| Monkey Testing | Gorilla Testing |
| Monkey Testing is performed randomly with no specifically predefined test cases | It is neither predefined nor random |
| Monkey Testing is performed on entire system can have several test cases | Gorilla Testing is performed on specifically few selective modules with few test cases |
| The objective of Monkey Testing is to check for system crash | Objective of Gorilla testing is to check whether the module is working properly or not |

### **Monkey Testing vs Ad-hoc Testing**

|  |  |
| --- | --- |
| Monkey Testing | Ad-hoc Testing |
| Monkey Testing is performed randomly with no specifically predefined test cases | Ad-hoc testing is performed without planning and documentation(test cases and SRS) |
| In Monkey Testing testers may not know what is the system is all about and its purpose | In Ad-hoc Testing tester must understand the system significantly before performing testing |
| The objective of Monkey Testing is to check for system crash | Objective of Ad-hoc testing is to divide the system randomly into subparts and check their functionality |

## Unit Test vs Integration Test

|  |  |
| --- | --- |
| Unit test | Integration test |
| The idea behind Unit Testing is to test each part of the program and show that the individual parts are correct. | The idea behind Integration Testing is to combine modules in the application and test as a group to see that they are working fine |
| It is kind of [White Box Testing](https://www.guru99.com/white-box-testing.html) | It is kind of [Black Box Testing](https://www.guru99.com/black-box-testing.html) |
| It can be performed at any time | It usually carried out after Unit Testing and before [System Testing](https://www.guru99.com/system-testing.html) |
| Unit Testing tests only the functionality of the units themselves and may not catch integration errors, or other system-wide issues | Integrating testing may detect errors when modules are integrated to build the overall system |
| It starts with the module specification | It starts with the interface specification |
| It pays attention to the behavior of single modules | It pays attention to integration among modules |
| Unit test does not verify whether your code works with external dependencies correctly. | Integration tests verify that your code works with external dependencies correctly. |
| It is usually executed by the developer | It is usually executed by a test team |
| Finding errors is easy | Finding errors is difficult |
| Maintenance of unit test is cheap | Maintenance of integration test is expensive |

## Smoke Testing vs Sanity Testing

|  |  |
| --- | --- |
| Smoke Testing | Sanity Testing |
| Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine | Sanity Testing is done to check the new functionality/bugs have been fixed |
| The objective of this testing is to verify the “stability” of the system in order to proceed with more rigorous testing | The objective of the testing is to verify the “rationality” of the system in order to proceed with more rigorous testing |
| This testing is performed by the developers or testers | Sanity testing in software testing is usually performed by testers |
| Smoke testing is usually documented or scripted | Sanity testing is usually not documented and is unscripted |
| Smoke testing is a subset of Acceptance testing | Sanity testing is a subset of [Regression Testing](https://www.guru99.com/regression-testing.html) |
| Smoke testing exercises the entire system from end to end | Sanity testing exercises only the particular component of the entire system |
| Smoke testing is like General Health Check Up | Sanity Testing is like specialized health check up |

## Alpha vs Beta Testing

|  |  |
| --- | --- |
| Alpha Testing | Beta Testing |
| Alpha testing performed by Testers who are usually internal employees of the organization | Beta testing is performed by Clients or End Users who are not employees of the organization |
| Alpha Testing performed at developer’s site | Beta testing is performed at a client location or end user of the product |
| Reliability and [Security Testing](https://www.guru99.com/what-is-security-testing.html) are not performed in-depth Alpha Testing | Reliability, Security, Robustness are checked during Beta Testing |
| Alpha testing involves both the white box and black box techniques | Beta Testing typically uses [Black Box Testing](https://www.guru99.com/black-box-testing.html) |
| Alpha testing requires a lab environment or testing environment | Beta testing doesn’t require any lab environment or testing environment. The software is made available to the public and is said to be real time environment |
| Long execution cycle may be required for Alpha testing | Only a few weeks of execution are required for Beta testing |
| Critical issues or fixes can be addressed by developers immediately in Alpha testing | Most of the issues or feedback is collected from Beta testing will be implemented in future versions of the product |
| Alpha testing is to ensure the quality of the product before moving to Beta testing | Beta testing also concentrates on the quality of the product, but gathers users input on the product and ensures that the product is ready for real time users. |

## Performance Testing vs. Load Testing vs. Stress Testing

|  |  |  |
| --- | --- | --- |
| Performance testing | Load testing | Stress testing |
| Superset of load and stress testing. | A subset of performance testing. | A subset of performance testing. |
| Helps to set the benchmark and standards for the application. | To recognize the upper limit of the system, set SLA of the app and check how the system can handle a heavy load. | To check out how the system behaves under extreme loads and how it recovers from failure. |
| The aim of performance testing is to get an indication of how an application behaves under regular parameters. | Generating increased load on a web application is the main aim of load testing. | Stress testing aims to ensure that under a sudden high load for a considerable duration the servers don’t crash. |
| Resource usage, availability, and reliability of the product are validated under this testing. | The attributes which are checked in a load test are peak performance, server quantity and response time. | This kind of testing checks stability response time, etc. |
| In performance testing, Load limit is both below and above the threshold of a break. | In load testing load limit is a threshold of a break. | In stress testing load limit is above the threshold of a break. |
| Example of performance testing are : Checking concurrent users, HTTP connections or checking Suitable response time. | Example of Load testing are Test of a word processor by make change in the large volume of data,  test a printer by transferring heavy data.  Check mail server with thousands of concurrent users. | Example of Stress testing is casually shut down and restart ports of a large network. |
| Why Performance Testing?  -Validating that application performs properly  -Validating to conform the performance needs of the business  Finding, analyzing and, fixing performance issues  Validating the hardware adequate to handle the expected load.  Doing capacity planning for future demand of the application | Why Load Testing?  -Find bugs which are not possible with any other testing method. Such as memory management memory leaks, bugs, buffer overflows, etc.  -To assure that application able to achieve the performance point recognized during performance testing.  -To determine the operating capacity of an application  -To check out that current infrastructure is sufficient to run the application or not.  -Numbers of concurrent users which an application can support, and scalability to allow more users to access it. | Why Stress Testing?  -It helps the testing unit to test the system in the situation of failures.  -To make sure that the system has saved the data before crashing or not.  -To check whether any unexpected failures do not harm the system security. |

## 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 |
| Defect verification is not the part of Regression Testing | Defect verification is the part of re-testing |
| 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 |
| Regression testing checks for unexpected side-effects | Re-testing makes sure that the original fault has been corrected |
| Regression testing is only done when there is any modification or changes become mandatory in an existing project | Re-testing executes a defect with the same data and the same environment with different inputs with a new build |
| Test cases for regression testing can be obtained from the functional specification, user tutorials and manuals, and defect reports in regards to corrected problems | [Test cases](https://www.guru99.com/test-case.html) for retesting cannot be obtained before start testing. |

## Frontend Testing vs Backend Testing

|  |  |
| --- | --- |
| Frontend testing | Backend testing |
| Frontend testing is always performed on the GUI. | [Back End Testing](https://www.guru99.com/what-is-backend-testing.html) involves databases and business logic testing. |
| The tester must be knowledgeable about the business requirements as well as the usage of the automation frameworks tools. | The tester to be able to perform back-end testing must have a strong background in the database and Structured Query Language (SQL) concepts. |
| GUI is used to perform the Testing | GUI may or may not be used to perform Testing |
| It does not need any information to be stored in a database. | It does need information stored in the database. |
| It is essential to check the overall functionality of the application. | Backend testing is important to check for deadlock, data corruption, data loss, etc |
| Types of Testing done are – Unit Tests, Acceptance Testing, Accessibility Testing, Regression Testing, etc. | Three widely used types of database testing are SQL Testing, [API Testing](https://www.guru99.com/api-testing.html), etc. |

## Globalization Testing vs Localization Testing

|  |  |
| --- | --- |
| Globalization Testing | Localization Testing |
| Globalization testing checks the proper functioning of the product, using every type of international input possible. It ensures that without breaking functionality the code can handle all international support. For example I18N, is the process of planning and implementing products and services so that they can easily be adapted to specific languages and culture. | Localizing testing is done to ensure the quality of a product for a particular target or locale. For example, for French users, the testing product is denoted as L10N. |
| In a globalized product, a code is separated from the messages or information. With the help of globalization, it enables software to be used with different languages without having to redesign the complete software. | This is not necessary for a Localized product |
| Globalization focuses your application’s capabilities on users as the generic user base. | Localization focuses on a subset of users in a given culture or locale. |
| Separation of testers from translators and engineers, ensuring a thorough and impartial approach. | It helps to reduce a time for testing since it’s done for just on locale |
| Formalized [bug reporting](https://www.guru99.com/top-20-bug-tracking-tools.html) | It reduces overall testing and support costs |
| Detect potential problems in application design that could inhibit globalization | Validation of all application resources |
| It ensures that without breaking functionality code can handle all international support | Verification of linguistic accuracy and resource attributes. Check Typographical errors |
| Compatibility tests of hardware and application according to the product’s target region | Confirmation of input and display environment standards, adherence to the system. Usability of User Interface |

## Severity vs Priority

| **Priority** | **Severity** |
| --- | --- |
| Defect Priority has defined the order in which the developer should resolve a defect | Defect Severity is defined as the degree of impact that a defect has on the operation of the product |
| Priority is associated with scheduling | Severity is associated with functionality or standards |
| Priority indicates how soon the bug should be fixed | Severity indicates the seriousness of the defect on the product functionality |
| Priority of defects is decided in consultation with the manager/client | QA engineer determines the severity level of the defect |
| Priority is driven by business value | Severity is driven by functionality |
| Its value is subjective and can change over a period of time depending on the change in the project situation | Its value is objective and less likely to change |
| High priority and low severity status indicates, defect have to be fixed on immediate bases but does not affect the application | High severity and low priority status indicates defect have to be fixed but not on immediate bases |
| Priority status is based on customer requirements | Severity status is based on the technical aspect of the product |
| During UAT the development team fix defects based on priority | During SIT, the development team will fix defects based on the severity and then priority |
| Priority is categorized into three types   * Low * Medium * High | Severity is categorized into five types   * Critical * Major * Moderate * Minor * Cosmetic |