# **The Ultimate Guide to Software Testing**

#### **Presented by:** Kana, Azibabari Lagalo

##### **Date:** 28th April 2022

## **What is Software Testing?**

Software testing is a method of evaluating the quality of software and lowering the chance of a software failure during operation and the practice of reviewing and validating that a software product or application accomplishes what it is designed to do. It is not synonymous with test execution. It is a process that encompasses numerous actions, one of which is test execution. Software testing is also. Software testing is a stage in the Software Development Lifecycle. Software Testers are part of the development team. Software testing can be done by testers, developers and DevOps. It is done in phases which is known as Software Testing Life Cycle

## **Why is Software Testing important?**

Software testing is vital because it aids in the prevention of defects, the reduction of development expenses, and the enhancement of performance. It is important for repairing software not operating properly because failing to do so may result in a slew of issues such as lost money, time, and corporate reputation, as well as injury or death. In essence, software testing is very important for the quality of the product and the satisfaction of the customer (which means that the actual result of the software is the same as the expected result).

## 

## **The Types of Software Testing**

There are different types of software Testing which are majorly categorized into two. They are:

1. Manual Testing: It is a technique of verification and validation to test any application according to the demands of the customer without the use of any automated technology. It is classified into:

* White-Box Testing: The goal of white box testing is to highlight the flow of inputs and outputs over software and improve application security. Here, input is supplied, and the internal structure (source code) of the software is examined. White-box testing is also known as glass-box testing
* Black-Box Testing: Black box testing is the process of verifying an application's functionality in accordance with the needs of the client. It is referred to as black-box testing because the internal structure (source code) is not visible during this testing.
* Grey-Box Testing: A collaboration of both White and Black Box testing.
* Functional Testing: Functional testing is the process of checking all of the components as compared to the requirements specifications in a systematic manner. Component testing is another name for functional testing. E.g: For login pages, the tester tests the login functionality. It is classified into:
* Unit Testing: The primary goal of performing unit testing is to confirm the performance of the unit components. Any component that may be evaluated separately is tested.This test is the developer's responsibility. E.g: Testing a module of codes.
* Integration Testing: It is integrating the modules of codes where we put the data flow between dependent modules or the interface between two features to the test (interaction between components and systems).
* System Testing: The primary purpose is to identify defects and vulnerabilities. The ideal test environment is similar to the final target or product environment. It focuses on the product's behavior and capabilities, taking into account the tasks the system can do as well as the non-functional behavior it shows.
* Non-Functional Testing: This is basically testing the system to verify its non-functional requirement. E.g: Is the login fast enough?. It is classified into:
* Performance Testing: This is testing how the system performs. It involves:
* Testing the response time: E.g: Is the login fast enough?
* Load testing: Adding some load (many users) to the specific application in order to test its performance.
* Stress testing: Testing the system to see if it can handle peak loads that are beyond the limits of its work load.,
* Scalability testing: Testing to find out the ability of a system to meet future efficiency requirements which may be beyond those currently required. E.g: Will our system expand due to new users?
* Stability testing: Testing the system over a time frame specific to the system’s operational context.
* Usability Testing: Testing to analyze if the software can be easily used, and if it has an appealing UI/user experience as well as detecting bugs in the end-user’s interface.
* Compatibility Testing: Testing an application's functionality in specific hardware and software settings.

1. Automated Testing: employs specialized technologies to automate manual design test cases without requiring human intervention using test scripts. These specialized technologies are called automation tools, e.g: Selenium. Some of these tools have tutorials on how to use them. Automation testing is the most effective method for increasing the efficiency, productivity, and coverage of software testing. It is used to re-run test scenarios that were previously conducted manually, fast, and repeatedly.

There are so many other different types of testing such as: retesting, smoke testing, sanity, regression testing, user acceptance testing(alpha and beta), dynamic, exploratory testing,, adhoc, security, localization, penetration testing, API testing, etc.

**Regression Testing:** Testing the whole software after a bug was found and fixed in a component or part of the software.

## **What are the methods of Software Testing?**

There are different software testing methods. These methods are also software testing types:

1. Black-box testing
2. White-box testing
3. Grey-box testing
4. Static Testing: Analyzing the source code and any associated documentation, but not running the program
5. Dynamic Testing: Testing the program is operating, giving in numerous inputs, and testers comparing outputs to expected behavior.
6. Functional testing
7. Non-functional testing

## **What are the approaches to Software Testing?**

The goal of all software testing is to find bugs. Testers must understand how often a bug arises and how it impacts the software.

Running software across actual devices and browsers is the best approach to find any issues. When it comes to a website, make sure it is subject to both manual and automated testing. Automated Selenium testing should be used in conjunction with manual testing to ensure that no issues are missed throughout the Quality Assurance process. Websites and web applications must also go through comprehensive cross-browser testing to ensure that they perform consistently regardless of the browser used to view them.

During testing, keep the risks in mind. Is it possible that tests may cause software to malfunction? For example, if an app is already popular in the digital market, testing for new features or upgrades may cause it to crash or jumble certain functions. In this instance, testers may want to use regression testing. Is the testing plan meeting all of the stakeholders' expectations and needs? Tests should not only determine if all software features work as expected, but also whether they match business needs and really help users.

Is the software being tested compliant with all industry (and digital) regulations? Testers must be familiar with local rules so that the program does not unwittingly violate the law.

## **What are the Levels of Software Testing?**

The major test levels are:

* Unit/Component Testing
* Integration Testing
* System Testing
* Acceptance Testing

## **What are the different testing process models?**

These testing process model are also used in the software development process are:

* Water-fall model: The entire software testing process is divided into discrete parts in "The Waterfall" Model approach. In this approach, the output of one phase serves as the input for the following step in the sequence. The waterfall model consists of four distinct phases: requirement collection and analysis, software design, programmed implementation and testing, and maintenance. All four phases will occur in the above-mentioned order.
* Agile Model: The Agile model relies on the collaboration of many cross-functional teams to produce requirements and solutions. Agile is sometimes referred to as an iterative and incremental methodology. The agile software testing model prioritizes process flexibility and customer satisfaction by delivering functioning software products quickly and breaking the product down into small incremental releases.
* Spiral model: This software testing strategy is comparable to the Agile model, but places a greater emphasis on risk analysis. The spiral model comprises four phases: planning, risk analysis, engineering, and assessment. In this situation, you must collect the requirements and do the risk assessment at the base level, and then each upward spiral builds on it.
* Iterative/Incremental Model: Before beginning a project, the Iterative approach does not require a comprehensive list of requirements. The development process begins with the functional part's requirements, which can be expanded later. The procedure is repeated for each cycle, allowing for new product versions. Every iteration entails the creation of a separated component of the system that is added to the previously built functional.
* V model: The V Model is preferred above the waterfall model. The development and testing operations are carried out side by side in the downhill and uphill shapes in this model. In this case, the testing phases run concurrently. Furthermore, testing begins at the unit level and progresses to the integration of the full system.

## **What is the difference between unit testing and acceptance testing?**

Unit testing is a software development technique that examines the smallest testable elements of a program, known as units, separately and independently for appropriate operation. This testing approach is carried out by software developers and, on occasion, software testers during the development process while Acceptance testing typically focuses on the behavior and capabilities of the whole system. It is mostly done by users or stakeholders. It is divided into 2: Alpha testing(Bringing customers to the company and making them test there) and Beta Testing(Customers testing applications in their own homes/site)

## **What is the difference between software testing and software engineering?**

Software testing is a method that is used to determine the accuracy, completeness, and quality of computer software that has been produced. It is the process of determining if actual results match predicted results and ensuring that the software system is defect-free. It is the process of running software under controlled settings to ensure that it operates as expected, detects mistakes, and validates that what was stated is what the user truly wants while Software engineering is described as the process of evaluating user needs and then designing, constructing, and testing software applications to meet those requirements.

In essence, software engineering is creating software while software testing ensures great software quality. Software testing is part of software engineering but not vice versa.

In essence, software engineering is creating software while software testing ensures great software quality. Software testing is part of software engineering but not vice versa.

## **Do software testers need to know programming languages?**

A good tester should have a basic understanding of programming and scripting languages such as Java, VBScript, JavaScript, and others. Knowing programming languages is not compulsory, however it is desirable.

## **What are the various tools for software testing?**

The software testing tools/frameworks can be divided into:

* Test management tools: For managing test data, test results, test suites, test coverage. Example: TestLink
* Security testing tool. Example: Wfuzz, Wapiti
* Bug tracking tool. Example: Jira
* GUI Testing tool. Example: Katalon Studio
* Automation testing tool. Example: Selenium, Appium
* Performance testing tool. Example: Jmeter
* Cross-browser testing tool Example: TestComplete
* Integration testing tool Example: Cucumber BDD
* Unit testing tool Example: Mockito
* Mobile/android testing tool Example: Calabash

## **What are the principles of software testing?**

In software testing, there are seven principles to follow. They are:

* Testing shows the presence of defects.
* It is not possible to do exhaustive testing.
* Early testing is very necessary.
* Defects clustering
* The Pesticide Paradox
* Context dictates testing
* The fallacy of absence of mistakes.

## **What qualities should you look for in your software testing team?**

For your software testing team, you should look for testers with the following qualities.

* Can understand End User’s point of view
* A Constant Learner
* Able to relate real-time scenarios to software testing
* Flexible to support when needed
* Good at reporting
* Can give negative report in a positive way
* Can analyze data
* Creative with ideas
* Asks Questions
* Recognize Priorities