Introduction

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not.

Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

According to ANSI/IEEE 1059 standard, Testing can be defined as - A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.

Who does Testing?

It depends on the process and the associated stakeholders of the project(s). In the IT industry, large companies have a team with responsibilities to evaluate the software developed in context of the given requirements. Moreover, developers also conduct testing called Unit Testing. In most cases, the following professionals are involved in testing a system within their respective capacities:

•   Software Tester
•   Software Developer
•   Project Lead/Manager
•   End User

Verification & Validation

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

|	Verification	Validation
	Verification addresses the concern: "Are you building itright?"	Validation addresses the concern: "Are youbuilding the right thing?"
	Ensures that the software system meets all thefunctionality.	Ensures that the functionalities meet theintended behavior.
	Verification takes place first and includes the checkingfor documentation, code, etc.	Validation occurs after verification and mainlyinvolves the checking of the overall product.
	Done by developers.	Done by testers.
	It has static activities, as it includes collecting reviews,walkthroughs, and inspections to verify a software.	It has dynamic activities, as it includes executingthe software against the requirements.
	It is an objective process and no subjective decisionshould be needed to verify a software.	It is a subjective process and involves subjectivedecisions on how well a software works.

Types of Testing

Manual Testing

Manual testing includes testing a software manually, i.e., without using any automated tool or any script. In this type, the tester takes over the role of an end-user and tests the software to identify any unexpected behavior or bug. There are different stages for manual testing such as unit testing, integration testing, system testing, and useracceptance testing.

Testers use test plans, test cases, or test scenarios to test a software to ensure the completeness of testing. Manualtesting also includes exploratory testing, as testers explore the software to identify errors in it.

Automation Testing

Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses anothersoftware to test the product. This process involves automation of a manual process. Automation Testing is usedto re-run the test scenarios that were performed manually, quickly, and repeatedly.
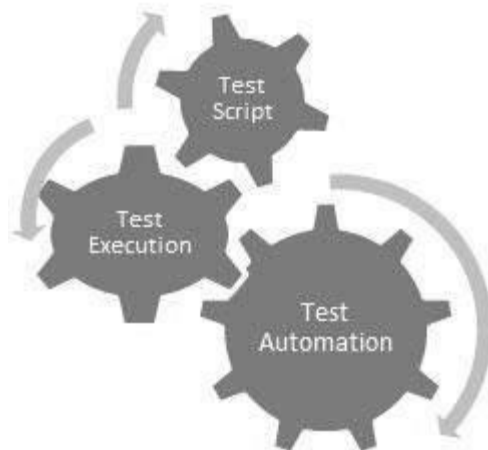
Apart from regression testing, automation testing is also used to test the application from load, performance, andstress point of view. It increases the test coverage, improves accuracy, and saves time and money in comparisonto manual testing.

What is Automate?

It is not possible to automate everything in a software. The areas at which a user can make transactions such asthe login form or registration forms, any area where large number of users can access the software simultaneously should be automated.

Furthermore, all GUI items, connections with databases, field validations, etc. can be efficiently tested byautomating the manual process.

When to Automate?

Test Automation should be used by considering the following aspects of a software:
- Large and critical projects
- Projects that require testing the same areas frequently
- Requirements not changing frequently

- Accessing the application for load and performance with many virtual users
- Stable software with respect to manual testing
- Availability of time

How to Automate?

Automation is done by using a supportive computer language like VB scripting and an automated software application. There are many tools available that can be used to write automation scripts. Before mentioning the tools, let us identify the process that can be used to automate the testing process:

- Identifying areas within a software for automation
- Selection of appropriate tool for test automation
- Writing test scripts
- Development of test suits
- Execution of scripts
- Create result reports
- Identify any potential bug or performance issues

Software Testing Tools

The following tools can be used for automation testing:

- HP Quick Test Professional
- Selenium
- IBM Rational Functional Tester
- SilkTest
- TestComplete
- Testing Anywhere
- WinRunner
- LoadRunner
- Visual Studio Test Professional
- WATIR

Software Testing - Levels

Levels of testing include different methodologies that can be used while conducting software testing. The main levels of software testing are:
- Functional Testing
- Non-functional Testing

A. Functional Testing

This is a type of black-box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional testing of a software is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

There are five steps that are involved while testing an application for functionality.

Steps	Description
I	The determination of the functionality that the intended application is meant to perform.
II	The creation of test data based on the specifications of the application.
III	The output based on the test data and the specifications of the application.
IV	The writing of test scenarios and the execution of test cases.
V	The comparison of actual and expected results based on the executed test cases.

An effective testing practice will see the above steps applied to the testing policies of every organization and hence it will make sure that the organization maintains the strictest of standards when it comes to software quality.

1. Unit Testing

This type of testing is performed by developers before the setup is handed over to the testing team to formally execute the test cases. Unit testing is performed by the respective developers on the individual units of source code assigned areas. The developers use test data that is different from the test data of the quality assurance team.

The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality.

Limitations of Unit Testing

Testing cannot catch each and every bug in an application. It is impossible to evaluate every execution path in every software application. The same is the case with unit testing.

There is a limit to the number of scenarios and test data that a developer can use to verify a source code. After having exhausted all the options, there is no choice but to stop unit testing and merge the code segment with other units.

2. Integration Testing

Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. Integration testing can be done in two ways: Bottom-up integration testing and Top-down integration testing.

Integration Testing Method
Bottom-up integration

This testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules or builds.

Top-down integration

In this testing, the highest-level modules are tested first and progressively, lower-level modules are tested thereafter.

In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic actual situations.

3. *System Testing*

System testing tests the system as a whole. Once all the components are integrated, the application as a whole is tested rigorously to see that it meets the specified Quality Standards. This type of testing is performed by a specialized testing team.

System testing is important because of the following reasons:

•   System testing is the first step in the Software Development Life Cycle, where the application is tested as a whole.

•   The application is tested thoroughly to verify that it meets the functional and technical specifications.

•   The application is tested in an environment that is very close to the production environment where the application will be deployed.

•   System testing enables us to test, verify, and validate both the business requirements as well as the application architecture.

4. *Regression Testing*

Whenever a change in a software application is made, it is quite possible that other areas within the application have been affected by this change. Regression testing is performed to verify that a fixed bug hasn't resulted in another functionality or business rule violation. The intent of regression testing is to ensure that a change, such as a bug fix should not result in another fault being uncovered in the application.

Regression testing is important because of the following reasons:

- Minimize the gaps in testing when an application with changes made has to be tested.
- Testing the new changes to verify that the changes made did not affect any other area of the application.

- Mitigates risks when regression testing is performed on the application.
- Test coverage is increased without compromising timelines.
- Increase speed to market the product.

5. *Acceptance Testing*

This is arguably the most important type of testing, as it is conducted by the Quality Assurance Team who will gauge whether the application meets the intended specifications and satisfies the client's requirement. The QA team will have a set of pre-written scenarios and test cases that will be used to test the application.

More ideas will be shared about the application and more tests can be performed on it to gauge its accuracy and the reasons why the project was initiated. Acceptance tests are not only intended to point out simple spelling mistakes, cosmetic errors, or interface gaps, but also to point out any bugs in the application that will result in system crashes or major errors in the application.

By performing acceptance tests on an application, the testing team will deduce how the application will performin production. There are also legal and contractual requirements for acceptance of the system.

6. *Alpha Testing*

This test is the first stage of testing and will be performed amongst the teams (developer and QA teams). Unittesting, integration testing and system testing when combined together is known as alpha testing. During this phase, the following aspects will be tested in the application:

- Spelling Mistakes
- Broken Links
- Cloudy Directions
- The Application will be tested on machines with the lowest specification to test loading times and any latency problems.

7. *Beta Testing*

This test is performed after alpha testing has been successfully performed. In beta testing, a sample of the intendedaudience tests the application. Beta testing is also known as **pre-release testing**. Beta test versions of software are ideally distributed to a wide audience on the Web, partly to give the program a "real-world" test and partly toprovide a preview of the next release. In this phase, the audience will be testing the following:

- Users will install, run the application and send their feedback to the project team.
- Typographical errors, confusing application flow, and even crashes.
- Getting the feedback, the project team can fix the problems before releasing the software to the actual users.
- The more issues you fix that solve real user problems, the higher the quality of your application will be.
- Having a higher-quality application when you release it to the general public will increase customer satisfaction.

B. Non-Functional Testing

This section is based upon testing an application from its non-functional attributes. Non-functional testinginvolves testing a software from the requirements which are nonfunctional in nature but important such asperformance, security, user interface, etc.

Some of the important and commonly used non-functional testing types are discussed below.

1. *Performance Testing*

It is mostly used to identify any bottlenecks or performance issues rather than finding bugs in a software. Thereare different causes that contribute in lowering the performance of a software:

- Network delay
- Client-side processing
- Database transaction processing
- Load balancing between servers
- Data rendering

Performance testing is considered as one of the important and mandatory testing type in terms of the followingaspects:

- Speed (i.e. Response Time, data rendering and accessing)
- Capacity
- Stability
- Scalability

Performance testing can be either qualitative or quantitative and can be divided into different sub-types such as **Load testing** and **Stress testing**.

(a) Load Testing
It is a process of testing the behavior of a software by applying maximum load in terms of software accessing andmanipulating large input data. It can be done at both normal and peak load conditions. This type of testing identifies the maximum capacity of software and its behavior at peak time.
Most of the time, load testing is performed with the help of automated tools such as Load Runner, AppLoader, IBM Rational Performance Tester, Apache JMeter, Silk Performer, Visual Studio Load Test, etc.
Virtual users (VUsers) are defined in the automated testing tool and the script is executed to verify the load testingfor the software. The number of users can be increased or decreased concurrently or incrementally based upon the requirements.

(b) Stress Testing
Stress testing includes testing the behavior of a software under abnormal conditions. For example, it mayinclude taking away some resources or applying a load beyond the actual load limit.
The aim of stress testing is to test the software by applying the load to the system and taking over the resourcesused by the software to identify the breaking point. This testing can be performed by testing different scenariossuch as:

- Shutdown or restart of network ports randomly
- Turning the database on or off
- Running different processes that consume resources such as CPU, memory, server, etc.

2. Usability Testing

Usability testing is a black-box technique and is used to identify any error(s) and improvements in the software by observing the users through their usage and operation.

According to Nielsen, usability can be defined in terms of five factors, i.e. efficiency of use, learn-ability, memory-ability, errors/safety, and satisfaction. According to him, the usability of a product will be good and thesystem is usable if it possesses the above factors.

Nigel Bevan and Macleod considered that usability is the quality requirement that can be measured as the outcomeof interactions with a computer system. This requirement can be fulfilled and the end-user will be satisfied if theintended goals are achieved effectively with the use of proper resources.

Molich in 2000 stated that a user-friendly system should fulfill the following five goals, i.e., easy to Learn, easy to remember, efficient to use, satisfactory to use, and easy to understand.

In addition to the different definitions of usability, there are some standards and quality models and methods thatdefine usability in the form of attributes and sub-attributes such as ISO-9126, ISO-9241-11, ISO-13407, and IEEEstd.610.12, etc.

UI vs Usability Testing

UI testing involves testing the Graphical User Interface of the Software. UI testing ensures that the GUI functions according to the requirements and tested in terms of color, alignment, size, and other properties.

On the other hand, usability testing ensures a good and user-friendly GUI that can be easily handled. UI testing can be considered as a sub-part of usability testing.

3. Security Testing

Security testing involves testing a software in order to identify any flaws and gaps from security and vulnerability point of view. Listed below are the main aspects that security testing should ensure:

- Confidentiality
- Integrity
- Authentication
- Availability
- Authorization
- Non-repudiation
- Software is secure against known and unknown vulnerabilities
- Software data is secure
- Software is according to all security regulations
- Input checking and validation
- SQL insertion attacks
- Session management issues
- Cross-site scripting attacks
- Buffer overflows vulnerabilities

4. *Portability Testing*

Portability testing includes testing a software with the aim to ensure its reusability and that it can be moved fromanother software as well. Following are the strategies that can be used for portability testing:

- Transferring an installed software from one computer to another.
- Building executable (.exe) to run the software on different platforms.

Portability testing can be considered as one of the sub-parts of system testing, as this testing type includes overalltesting of a software with respect to its usage over different environments. Computer hardware, operating systems,and browsers are the major focus of portability testing. Some of the pre-conditions for portability testing are as follows:

- Software should be designed and coded, keeping in mind the portability requirements.
- Unit testing has been performed on the associated components.
- Integration testing has been performed.
- Test environment has been established.

Testing documentation involves the documentation of artifacts that should be developed before or during the testing of Software.