Software Testing

The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy and usability. It mainly aims at measuring specification, functionality and performance of a software program or application.

Software testing can be stated as the process of verifying and validating that a software or application is bug free, meets the technical requirements as guided by it's design and development and meets the user requirements effectively and efficiently with handling all the exceptional and boundary cases.

Or

SOFTWARE TESTING is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest.Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements. It can be either done manually or using automated tools. Some prefer saying Software testing as a White Box and Black Box Testing.

Software testing can be divided into two steps:

1. **Verification:** it refers to the set of tasks that ensure that software correctly implements a specific function.

2. **Validation:** it refers to a different set of tasks that ensure that the software that has been built is traceable to customer requirements.

Verification: “Are we building the product right?”

Validation: “Are we building the right product?”

Importance of Software Testing

- ***The testing is important since it discovers defects/bugs before the delivery to the client, which guarantees the quality of the software.***
- ***It makes the software more reliable and easy to use.***
- ***Thoroughly tested software ensures reliable and high-performance software operation.***

For example, assume you are using a Net Banking application to transfer the amount to your friend’s account. So, you initiate the transaction, get a successful transaction message, and the amount also deducts from your account. However, your friend confirms that his/her account has not received any credits yet. Likewise, your account is also not reflecting the reversed transaction. This will surely make you upset and leave you as an unsatisfied customer.

Testing is important because software bugs could be expensive or even dangerous. Software bugs can potentially cause monetary and human loss, and history is full of such examples.

- **In April 2015, Bloomberg terminal in London crashed due to software glitch affected more than 300,000 traders on financial markets. It forced the government to postpone a 3bn pound debt sale.**
- **Nissan cars have to recall over 1 million cars from the market due to software failure in the airbag sensory detectors. There has been reported two accident due to this software failure.**
- **Starbucks was forced to close about 60 percent of stores in the U.S and Canada due to software failure in its POS system. At one point store served coffee for free as they unable to process the transaction.**
- **Some of the Amazon's third party retailers saw their product price is reduced to 1p due to a software glitch. They were left with heavy losses.**
- **Vulnerability in Window 10. This bug enables users to escape from security sandboxes through a flaw in the win32k system.**
- **In 2015 fighter plane F-35 fell victim to a software bug, making it unable to detect targets correctly.**

Software Testing can be broadly classified into two types:

1. 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 user acceptance testing.

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

2. Automation Testing: Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses another software to test the product. This process involves automation of a manual process. Automation Testing is used to 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, and stress point of view. It increases the test coverage, improves accuracy, and saves time and money in comparison to manual testing.

Also,Typically Testing is classified into three categories.

- **Functional Testing**
- **Non-Functional Testing or Performance Testing**
- **Maintenance (Regression and Maintenance)**

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**

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.

St	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.

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.

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.

Sr	Integration Testing Method
1	**Bottom-up integration** **This testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules or builds.**
2	**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.

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.**

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.**

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 reduce how the application will perform in production. There are also legal and contractual requirements for acceptance of the system.

Alpha Testing

This test is the first stage of testing and will be performed amongst the teams (developer and QA teams). Unit testing, 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.**

Beta Testing

This test is performed after alpha testing has been successfully performed. In beta testing, a sample of the intended audience 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 to provide

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.**

Non-Functional Testing

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

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

Performance Testing

It is mostly used to identify any bottlenecks or performance issues rather than finding bugs in a software. There are 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 following aspects −

- **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.

Load Testing

It is a process of testing the behavior of a software by applying maximum load in terms of software accessing and manipulating 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 testing for the software. The number of users can be increased or decreased concurrently or incrementally based upon the requirements.

Stress Testing

Stress testing includes testing the behavior of a software under abnormal conditions. For example, it may include 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 resources used by the software to identify the breaking point. This testing can be performed by testing different scenarios such 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.**

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 the system 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 outcome of interactions with a computer system. This requirement can be fulfilled and the end-user will be satisfied if the intended 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 that define usability in the form of attributes and sub-attributes such as ISO-9126, ISO-9241-11, ISO-13407, and IEEE std.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.

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**
- **Injection flaws**
- **Session management issues**
- **Cross-site scripting attacks**
- **Buffer overflows vulnerabilities**
- **Directory traversal attacks**

Portability Testing

Portability testing includes testing a software with the aim to ensure its reusability and that it can be moved from another 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 overall testing 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.**