**Software Testing :-**

**Definition :-**

The process or method of finding error/s in a software application or program so that the application functions according to the end user's requirement is called software testing.

**Functional Testing**

**There are four main types of functional testing.**

**1) Unit Testing**

Unit testing is a type of software testing which is done on an individual unit or component to test its corrections. Typically, Unit testing is done by the developer at the application development phase. Each unit in unit testing can be viewed as a method, function, procedure, or object. Developers often use test automation tools such as NUnit, Xunit, JUnit for the test execution.

Unit testing is important because we can find more defects at the unit test level.

**For example**, there is a simple calculator application. The developer can write the unit test to check if the user can enter two numbers and get the correct sum for addition functionality.

**a) White Box Testing**

White box testing is a test technique in which the internal structure or code of an application is visible and accessible to the tester. In this technique, it is easy to find loopholes in the design of an application or fault in business logic. Statement coverage and decision coverage/branch coverage are examples of white box test techniques.

**b) Gorilla Testing**

Gorilla testing is a test technique in which the tester and/or developer test the module of the application thoroughly in all aspects. Gorilla testing is done to check how robust your application is.

**For example**, the tester is testing the pet insurance company’s website, which provides the service of buying an insurance policy, tag for the pet, Lifetime membership. The tester can focus on any one module, let’s say, the insurance policy module, and test it thoroughly with positive and negative test scenarios.

**2) Integration Testing**

Integration testing is a type of software testing where two or more modules of an application are logically grouped together and tested as a whole. The focus of this type of testing is to find the defect on interface, communication, and data flow among modules. Top-down or Bottom-up approach is used while integrating modules into the whole system.

This type of testing is done on integrating modules of a system or between systems. For example, a user is buying a flight ticket from any airline website. Users can see flight details and payment information while buying a ticket, but flight details and payment processing are two different systems. Integration testing should be done while integrating of airline website and payment processing system.

**a) Gray box testing**

As the name suggests, gray box testing is a combination of white-box testing and black-box testing. Testers have partial knowledge of the internal structure or code of an application.

**3) System Testing**

System testing is types of testing where tester evaluates the whole system against the specified requirements.

**a) End to End Testing**

It involves testing a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

**For example**, a tester is testing a pet insurance website. End to End testing involves testing of buying an insurance policy, LPM, tag, adding another pet, updating credit card information on users’ accounts, updating user address information, receiving order confirmation emails and policy documents.

**a) Gray box testing**

As the name suggests, gray box testing is a combination of white-box testing and black-box testing. Testers have partial knowledge of the internal structure or code of an application.

**3) System Testing**

System testing is types of testing where tester evaluates the whole system against the specified requirements.

**a) End to End Testing**

It involves testing a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

**For example**, a tester is testing a pet insurance website. End to End testing involves testing of buying an insurance policy, LPM, tag, adding another pet, updating credit card information on users’ accounts, updating user address information, receiving order confirmation emails and policy documents.

**b) Black Box Testing**

Blackbox testing is a software testing technique in which testing is performed without knowing the internal structure, design, or code of a system under test. Testers should focus only on the input and output of test objects.

Detailed information about the advantages, disadvantages, and types of Black Box testing can be found here.

**c) Smoke Testing**

Smoke testing is performed to verify that basic and critical functionality of the system under test is working fine at a very high level.

Whenever a new build is provided by the development team, then the Software Testing team validates the build and ensures that no major issue exists. The testing team will ensure that the build is stable, and a detailed level of testing will be carried out further.

**For example**, tester is testing pet insurance website. Buying an insurance policy, adding another pet, providing quotes are all basic and critical functionality of the application. Smoke testing for this website verifies that all these functionalities are working fine before doing any in-depth testing.

**d) Sanity Testing**

Sanity testing is performed on a system to verify that newly added functionality or bug fixes are working fine. Sanity testing is done on stable build. It is a subset of the regression test.

**For example**, a tester is testing a pet insurance website. There is a change in the discount for buying a policy for second pet. Then sanity testing is only performed on buying insurance policy module.

**e) Happy path Testing**

The objective of Happy Path Testing is to test an application successfully on a positive flow. It does not look for negative or error conditions. The focus is only on valid and positive inputs through which the application generates the expected output.

**f) Monkey Testing**

Monkey Testing is carried out by a tester, assuming that if the monkey uses the application, then how random input and values will be entered by the Monkey without any knowledge or understanding of the application.

The objective of Monkey Testing is to check if an application or system gets crashed by providing random input values/data. Monkey Testing is performed randomly, no test cases are scripted, and it is not necessary to be aware

of the full functionality of the system.

**4) Acceptance Testing**

Acceptance testing is a type of testing where client/business/customer test the software with real time business scenarios.

The client accepts the software only when all the features and functionalities work as expected. This is the last phase of testing, after which the software goes into production. This is also called User Acceptance Testing (UAT).

**a) Alpha Testing**

Alpha testing is a type of acceptance testing performed by the team in an organization to find as many defects as possible before releasing software to customers.

**For example**, the pet insurance website is under UAT. UAT team will run real-time scenarios like buying an insurance policy, buying annual membership, changing the address, ownership transfer of the pet in a same way the user uses the real website. The team can use test credit card information to process payment-related scenarios.

**b) Beta Testing**

Beta Testing is a type of software testing which is carried out by the clients/customers. It is performed in the Real Environment before releasing the product to the market for the actual end-users.

Beta Testing is carried out to ensure that there are no major failures in the software or product, and it satisfies the business requirements from an end-user perspective. Beta Testing is successful when the customer accepts the software.

Usually, this testing is typically done by the end-users. This is the final testing done before releasing the application for commercial purposes. Usually, the Beta version of the software or product released is limited to a certain number of users in a specific area.

So, the end-user uses the software and shares the feedback with the company. The company then takes necessary action before releasing the software worldwide.

**c) Operational acceptance testing (OAT)**

Operational acceptance testing of the system is performed by operations or system administration staff in the production environment. The purpose of operational acceptance testing is to make sure that the system administrators can keep the system working properly for the users in a real-time environment.

**The focus of the OAT is on the following points:**

* Testing of backup and restore.
* Installing, uninstalling, upgrading software.
* The recovery process in case of natural disaster.
* User management.
* Maintenance of the software.
* Non-Functional Testing
* There are four main types of functional testing.

**1) Security Testing**

It is a type of testing performed by a special team. Any hacking method can penetrate the system.

Security Testing is done to check how the software, application, or website is secure from internal and/or external threats. This testing includes how much software is secure from malicious programs, viruses and how secure & strong the authorization and authentication processes are.

It also checks how software behaves for any hacker’s attack & malicious programs and how software is maintained for data security after such a hacker attack.

**a) Penetration Testing**

Penetration Testing or Pen testing is the type of security testing performed as an authorized cyberattack on the system to find out the weak points of the system in terms of security.

Pen testing is performed by outside contractors, generally known as ethical hackers. That is why it is also known as ethical hacking. Contractors perform different operations like SQL injection, URL manipulation, Privilege Elevation, session expiry, and provide reports to the organization.

**Notes**: Do not perform the Pen testing on your laptop/computer. Always take written permission to do pen tests.

Security Testing is done to check how the software, application, or website is secure from internal and/or external threats. This testing includes how much software is secure from malicious programs, viruses and how secure & strong the authorization and authentication processes are.

It also checks how software behaves for any hacker’s attack & malicious programs and how software is maintained for data security after such a hacker attack.

**a) Penetration Testing**

Penetration Testing or Pen testing is the type of security testing performed as an authorized cyberattack on the system to find out the weak points of the system in terms of security.

Pen testing is performed by outside contractors, generally known as ethical hackers. That is why it is also known as ethical hacking. Contractors perform different operations like SQL injection, URL manipulation, Privilege Elevation, session expiry, and provide reports to the organization.

**Notes**: Do not perform the Pen testing on your laptop/computer. Always take written permission to do pen tests.

**2) Performance Testing**

Performance testing is testing of an application’s stability and response time by applying load.

The word stability means the ability of the application to withstand in the presence of load. Response time is how quickly an application is available to users. Performance testing is done with the help of tools. Loader.IO, JMeter, LoadRunner, etc. are good tools available in the market.

**a) Load testing**

Load testing is testing of an application’s stability and response time by applying load, which is equal to or less than the designed number of users for an application.

**For example**, your application handles 100 users at a time with a response time of 3 seconds, then load testing can be done by applying a load of the maximum of 100 or less than 100 users. The goal is to verify that the application is responding within 3 seconds for all the users.

**b) Stress Testing**

Stress testing is testing an application’s stability and response time by applying load, which is more than the designed number of users for an application.

**For example**, your application handles 1000 users at a time with a response time of 4 seconds, then stress testing can be done by applying a load of more than 1000 users. Test the application with 1100,1200,1300 users and notice the response time. The goal is to verify the stability of an application under stress.

**c) Scalability Testing**

Scalability testing is testing an application’s stability and response time by applying load, which is more than the designed number of users for an application.

**For example**, your application handles 1000 users at a time with a response time of 2 seconds, then scalability testing can be done by applying a load of more than 1000 users and gradually increasing the number of users to find out where exactly my application is crashing.

**Let’s say my application is giving response time as follows:**

1000 users -2 sec

1400 users -2 sec

4000 users -3 sec

5000 users -45 sec

5150 users- crash – This is the point that needs to identify in scalability testing

**d) Volume testing (flood testing)**

Volume testing is testing an application’s stability and response time by transferring a large volume of data to the database. Basically, it tests the capacity of the database to handle the data.

**e) Endurance Testing (Soak Testing)**

Endurance testing is testing an application’s stability and response time by applying load continuously for a longer period to verify that the application is working fine.

**For example**, car companies soak testing to verify that users can drive cars continuously for hours without any problem.

**3) Usability Testing**

Usability testing is testing an application from the user’s perspective to check the look and feel and user-friendliness.

**For example**, there is a mobile app for stock trading, and a tester is performing usability testing. Testers can check the scenario like if the mobile app is easy to operate with one hand or not, scroll bar should be vertical, background color of the app should be black and price of and stock is displayed in red or green color.

The main idea of usability testing of this kind of app is that as soon as the user opens the app, the user should get a glance at the market.

**a) Exploratory testing**

Exploratory Testing is informal testing performed by the testing team. The objective of this testing is to explore the application and look for defects that exist in the application. Testers use the knowledge of the business domain to test the application. Test charters are used to guide the exploratory testing.

**b) Cross browser testing**

Cross browser testing is testing an application on different browsers, operating systems, mobile devices to see look and feel and performance.

Why do we need cross-browser testing? The answer is different users use different operating systems, different browsers, and different mobile devices. The goal of the company is to get a good user experience regardless of those devices.

Browser stack provides all the versions of all the browsers and all mobile devices to test the application. For learning purposes, it is good to take the free trial given by browser stack for a few days.

**c) Accessibility Testing**

The aim of Accessibility Testing is to determine whether the software or application is accessible for disabled people or not.

Here, disability means deafness, color blindness, mentally disabled, blind, old age, and other disabled groups. Various checks are performed, such as font size for visually disabled, color and contrast for color blindness, etc.

**4) Compatibility testing**

This is a testing type in which it validates how software behaves and runs in a different environment, web servers, hardware, and network environment.

Compatibility testing ensures that software can run on different configuration, different databases, different browsers, and their versions. The testing team performs compatibility testing.

**Other Types of Testing**

* **Ad-hoc Testing**

The name itself suggests that this testing is performed on an ad-hoc basis, i.e., with no reference to the test case and also without any plan or documentation in place for this type of testing.

The objective of this testing is to find the defects and break the application by executing any flow of the application or any random functionality.

Ad-hoc testing is an informal way of finding defects and can be performed by anyone in the project. It is difficult to identify defects without a test case, but sometimes it is possible that defect found during ad-hoc testing might not have been identified using the existing test cases.

* **Back-end Testing**

Whenever an input or data is entered on the front-end application, it is stored in the database and the testing of such database is known as Database Testing or Backend Testing.

There are different databases like SQL Server, MySQL, Oracle, etc. Database Testing involves testing of table structure, schema, stored procedure, data structure, and so on. In Back-end Testing, GUI is not involved, the testers are directly connected to the database with proper access and testers can easily verify data by running a few queries on the database.

There can be issues identified like data loss, deadlock, data corruption, etc during this back-end testing and these issues are critical to fixing before the system goes live into the production environment.

* **Browser Compatibility Testing**

This is a sub-type of Compatibility Testing (which is explained below) and is performed by the testing team.

Browser Compatibility Testing is performed for web applications and ensures that the software can run with a combination of different browsers and operating systems. This type of testing also validates whether a web application runs on all versions of all browsers or not.

* **Backward Compatibility Testing**

It is a type of testing that validates whether the newly developed software or updated software works well with the older version of the environment or not.

Backward Compatibility Testing checks whether the new version of the software works properly with the file format created by an older version of the software. It also works well with data tables, data files, and data structures created by the older version of that software. If any of the software is updated, then it should work well on top of the previous version of that software.

* **Black Box Testing**

Internal system design is not considered in this type of testing. Tests are based on the requirements and functionality.

Detailed information about the advantages, disadvantages, and types of Black Box testing can be found here.

* **Boundary Value Testing**

This type of testing checks the behavior of the application at the boundary level.

Boundary Value Testing is performed to check if defects exist at boundary values. Boundary Value Testing is used for testing a different range of numbers. There is an upper and lower boundary for each range and testing is performed on these boundary values.

If testing requires a test range of numbers from 1 to 500, then Boundary Value Testing is performed on values at 0, 1, 2, 499, 500, and 501.

* **Branch Testing**

This is also known as Branch coverage or decision coverage testing. It is a type of white box testing performed at the unit test level. It is done to make sure that each possible path from the decision point is executed at least once for 100% of test coverage.

**Example:**

Read number A, B

If (A>B) then

Print(“A is greater”)

Else

Print(“B is greater”)

Here, there are two branches, one for if and the other for else. For 100% coverage, we need 2 test cases with different values of A and B.

**Test case 1:**

A=10, B=5 It will cover the if branch.

**Test case 2:**

A=7, B=15 It will cover the else branch.

* **Comparison Testing**

Comparison of a product’s strengths and weaknesses with its previous versions or other similar products is termed Comparison Testing.

* **Equivalence Partitioning**

It is a testing technique and a type of Black Box Testing. During this Equivalence Partitioning, a set of groups are selected and a few values or numbers are picked up for testing. It is understood that all values from that group generate the same output.

The aim of this testing is to remove redundant test cases within a specific group that generate the same output but not any defect.

Suppose the application accepts values between -10 and +10, then using equivalence partitioning, the values picked for testing are zero, one positive value, and one negative value. So the Equivalence Partitioning for this testing is -10 to -1, 0, and 1 to 10.

**Example Testing**

Example testing is real-time testing. It includes real-time scenarios and scenarios are based on the experience of the testers.

This type of testing is also known as experience-based testing because it uses the tester’s knowledge of how the application has worked in the past, how to break the application, what kind of errors are common in this type of application.

* **Graphical User Interface (GUI) Testing**

The objective of this GUI Testing is to validate the GUI as per the business requirement. The expected GUI of the application is mentioned in the Detailed Design Document and GUI mockup screens.

GUI Testing includes the size of the buttons and input fields present on the screen, alignment of all text, tables, and content in the tables.

It also validates the menu of the application. After selecting different menu and menu items, it validates that the page does not fluctuate, and the alignment remains the same after hovering the mouse on the menu or sub-menu.

* **Incremental Integration Testing**

Incremental Integration Testing is a Bottom-up approach for testing, i.e continuous testing of an application when new functionality is added.

Application functionality and modules should be independent enough to test separately. This is done by programmers or by testers.

* **Install/Uninstall Testing**

Installation testing is performed to check that the software application is installed properly and working as per expectation. Installation testing is a phase of testing before users interact with the actual application for the first time. Installation testing is also called “Implementation Testing”.

Uninstallation Testing is performed to confirm if all the components or elements of the software are removed from the system or not.

Installation and Uninstallation Testing is done on full, partial, or upgraded install/uninstall processes on different operating systems under different hardware or software environments.

* **Mutation Testing**

Mutation Testing is a type of white box testing in which the source code of one program is changed and verifies whether the existing test cases can identify these defects in the system.

The change in the program source code is very minimal, so it does not impact the entire application, only the specific area having the impact and the related test cases should be able to identify those errors in the system.

* **Negative Testing**

The mindset of the tester is to “Break the System/Application” and it is achieved through Negative Testing.

Negative Testing technique is performed using incorrect data, invalid data, or input. It validates if the system throws an error of invalid input and behaves as expected.

It should not take much time to load any page or system and should be sustained during peak load. Different performance and load tools are used to do this testing.

* **Recovery Testing**

It is a type of testing that validates how well the application or system recovers from crashes or disasters.

Recovery Testing determines if the system can continue its operation after a disaster. Assume that the application is receiving data through a network cable and suddenly that network cable has been unplugged.

Sometime later, plug in the network cable; then the system should start receiving data from where it lost the connection due to the network cable being unplugged.

* **Regression Testing**

Regression testing is testing of unchanged features of the application to make sure that any bug fixes, adding new features, deleting, or updating existing features, are not impacting the working application.

To find out regression scope is an important part in Regression Testing. To find out regression scope, Tester needs to find out the area of application where changes happened and the Impact of those changes on the entire application. It is difficult to cover the whole regression test suite in every release, so Automation Testing Tools are used in regression testing.

* **Risk-Based Testing (RBT)**

For Risk-Based Testing, the functionalities or requirements are tested based on their priority. Risk-Based Testing includes testing of highly critical functionality, which has the highest impact on business and in which the probability of failure is very high.

Priority decisions are based on business needs, so once priority is set for all functionalities, then high priority functionality or test cases are executed first, followed by medium and then low priority functionalities.

Low priority functionality may be tested or not tested based on the available time. Risk-Based Testing is carried out if there is insufficient time available to test the entire software and the software needs to be implemented on time without any delay.

This approach is followed only by the discussion and approval of the client and senior management of the organization.

* **Static Testing**

Static Testing is a type of testing which is done without the execution of any code. Reviews, walkthroughs, and inspections are different methods of performing static testing. Activities like reviewing of requirement documents, customer requirement specification, high level, and low-level design, code syntax, naming standards, etc. come under static testing.

Static Testing also applies to test cases, test plans, test scenarios. Static testing is done to prevent the defect rather than catching the defect at a later stage. That is why static testing is cost-effective.

**For example**, Tester is testing a pet insurance website. The logic for premium calculation is described in requirement documentation. As a part of static testing, testers can review the developer code for premium calculation and compare it with the requirement document to prevent the defect related to premium calculation.

* **Vulnerability Testing**

The testing, which involves identifying weaknesses in the software, hardware, and network, is known as Vulnerability Testing. In malicious programs, the hacker can take control of the system, if it is vulnerable to such kinds of attacks, viruses, and worms.

We need to check if those systems undergo Vulnerability Testing before production. It may identify critical defects and flaws in security. Recommended reading =>> Pilo