**Testing levels**

* **Unit Testing**
* **Integration testing**
* **System testing**
* **User acceptance testing**

**Unit testing**

The most basic type of testing is unit, or component testing.

Unit testing aims to verify each part of the software by isolating it and then perform tests to demonstrate that each individual component is correct in terms of fulfilling requirements and the desired functionality.

While Coding programmer performs some tests on that unit of program to know if it is error free.

The advantage of detecting any errors in the software early in the day is that by doing so the team minimises software development risks, as well as time and money wasted in having to go back and undo fundamental problems in the program once it is nearly completed.

**Integration testing**

Integration testing aims to test different parts of the system in combination in order to assess if they work correctly together. By testing the units in groups, any faults in the way they interact together can be identified.

There are many ways to test how different components of the system function at their interface; testers can adopt either a bottom-up or a top-down integration method

In bottom-up integration testing, testing builds on the results of unit testing by testing higher-level combination of units (called modules) in successively more complex scenarios.

It is recommended that testers start with this approach first, before applying the top-down approach which tests higher-level modules first and studies simpler ones later.

**System testing**

The next level of testing is system testing. As the name implies, all the components of the software are tested as a whole in order to ensure that the overall product meets the requirements specified.

System testing enables testers to ensure that the product meets business requirements, as well as determine that it runs smoothly within its operating environment. This type of testing is typically performed by a specialized testing team.

**Acceptance testing**

Finally, [acceptance testing](http://reqtest.com/testing-blog/a-guide-to-excellent-acceptance-testing/) is the level in the software testing process where a product is given the green light or not. The aim of this type of testing is to evaluate whether the system complies with the end-user requirements and if it is ready for deployment.

By performing acceptance tests, the testing team can find out how the product will perform when it is installed on the user’s system.

**Testing**

**Static Testing**

Static Testing, a software testing technique in which the software is tested without executing the code.

Static testing is a [software testing](https://whatis.techtarget.com/definition/software-testing) method that involves examination of the [program's](https://searchsoftwarequality.techtarget.com/definition/program) code and its associated [documentation](https://searchsoftwarequality.techtarget.com/definition/documentation) but does not require the [program](https://searchsoftwarequality.techtarget.com/definition/program) be executed.

It starts early in the Life cycle and so it is done during the verification process.

Static testing is performed due to the following reasons

* Early defect detection and correction
* Reduced development timescales
* Reduced testing cost and time
* For improvement of development productivity
* To get fewer defect at a later stage of testing

**Types of defects that are easier to find during the static testing are**

Deviation from standards, missing requirements, design defects, non-maintainable code and inconsistent interface specifications.

It has two parts as listed below:

* + Review
  + Static analysis

REVIEWS

During reviews participants question development decisions recommend improvements, and examine work products to determine status and conformance to requirements. The review is an aid to quality and determines status.

During the Review process four types of participants that take part in testing are:

* **Moderator**: leads the review process,his role is to determine the type of review ,scheduling meetings,distribute documents to other participants,coaching team member.
* **Author**: Takes responsibility for fixing the defect found and improves the quality of the document.He is the writer of the document under review .
* **Scribe**:He is responsible to record each defect found and any suggestions given in the meeting for process improvement. It does the logging of the defect during a review and attends the review meeting
* **Reviewer**: Check material for defects and inspects

Types of reviews

* **Informal Review**

In informal review the creator of the documents put the contents in front of audience and everyone gives their opinion and thus defects are identified in the early stage.

* **Walkthrough**

It is basically performed by experienced person or expert to check the defects so that there might not be problem further in the development or testing phase.

* **Peer Review**

Peer review means checking documents of one-another to detect and fix the defects. It is basically done in a team of colleagues.

* **Inspection**

Inspection is basically the verification of document the higher authority like the verification of software requirement specifications (SRS).

**Activities of Formal review**

* Planning
  + Define review criteria
  + Select reviewer
  + Define roles
  + Define entry and exit criteria
  + Select review product
* Kickoff
  + Distribute documents
  + Explain objectives
  + Process and document
* Individual preparation
* Preparing by reviewing products
* Review Meeting
  + Logging document results
  + Noting defects
  + Making recommendations about defect handling
  + Making recommendations about the defect
* Rework
  + Defect fix
  + Update defect status
* Follow up
  + Checking that defects has been addressed
  + Gathering metrics
  + Checking an exit criteria

**Success factors of review**

* Clear Objective
* Right people are involved
* Involve Testers as reviewers
* Defects found are expressed objectively
* People issues needs to be considered
* Use right review techniques based on the objective of review
* Train resources in formal review techniques
* Management supports a good review process

**Static analysis**

The code written by developers are analyzed (usually by tools) for structural defects that may lead to defects.

Following are the types of defects found by the tools during static analysis.

* A variable with an undefined value
* Inconsistent interface between modules and components
* Variables that are declared but never used
* Unreachable code or dead code
* Programming standard violations
* Security vulnerabilities
* Syntax violations

**Dynamic Testing**

Dynamic testing is software testing technique where testing is carried out with executing the code. This type of testing comes under Validation.

The main purpose of the dynamic test is to ensure consistency to the software.

Consistency is not only limited to the functionality it also refers to different standards like performance, usability, compatibity etc, hence it becomes very important to perform Dynamic Testing.

**Structure based testing technique(whitebox testing technique)**

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application.

The main aim of White Box testing is to check on how System is performing based on the code. It is mainly performed by the Developers or White Box Testers who has knowledge on the programming.

There are a number of coverage criteria. Let’s discuss Statement, Decision (Branch) and Path coverage, and understand how to calculate, with examples.

**For e.g.:**

Read X

Read Y

IF X+Y > 100 THEN

Print “Large”

ENDIF

If X > 50 THEN

Print “X Large”

ENDIF

For calculating Statement, Decision (Branch) and Path coverage

*Coverage - Flow Chart*

* Nodes ( , ) represent statement of code [E.g.: entry, exit, decisions]
* Edges ( ) represent links between nodes

Statement Coverage

Statement coverage is a whitebox testing technique technique where the all the statements at the source code are executed at least once. To calculate Statement Coverage, find out the shortest number of paths following which all the nodes will be covered.

In the above example, in case of “Yes”, while traversing through each statement of code and the traversing path (A1-B2-C4-5-D6-E8), all the nodes are covered. So by traveling through only one path all the nodes (A, B, C, D and E) are covered.

Statement coverage (SC) =1

**Branch/Decision Coverage**

Branch coverage covers both ways (true and false). It covers all the possible outcomes of each condition at least once. Branch coverage is a whitebox testing method that ensures that every possible branch from each decision point in the code is executed at least once. To calculate Branch coverage, find out the minimum number of paths which ensure covering of all the edges.

In the above example, in case of traversing through a ‘Yes’ decision, path (A1-B2-C4-5-D6-E8), maximum numbers of edges (1, 2, 4, 5, 6 and 8) are covered but edges 3 and 7 are left out. To cover these edges, we have to follow (A1-B3-5-D7). So by travelling through two paths (Yes, No), all the edges (1, 2, 3,, 5, 6, 7, 8) are covered.

Branch Coverage /Decision Coverage (BC) = 2

**Path Coverage**

It is executed in such a way that every path is executed at least once. It ensures that every statement in the program is guaranteed to be executed at least one time. Path Coverage ensures covering all the paths from beginning to end, in the above example. All the possible paths are:

A1-B3-5-D7  
A1-B2-C4-5-D6-E8  
A1-B2-C4-5-D7  
A1-B3-5-D6-E8  
Path coverage (PC) = 4

**Black box testing techniques (Specification based techniques)**

Also known as specification-based test design technique, the black box test design technique uses the software’s external descriptions such as the technical specifications, design, customer requirements, etc.  This implies that a tester who does not have any knowledge about the code or internal structure can also perform the test.

* **Equivalence partitioning**

The purpose of this type of test design method is to reduce the number of tests by dividing the different types of test. Once the tests have been divided, the system will behave in a similar way for the different tests with equivalence partition.

* **Boundary value analysis**

The best method of the black box design technique, boundary value analysis comprises testing the input values at the boundaries. Generally, the input values are put to test at the initial stages to reduce the chances of causing errors

* **Decision table testing**
* **Use case testing**
* **State transition testing**

**Experience-Based Test Design Technique:**

As the name suggests, the experience-based technique neither involves internal, nor the external structure, but is based on experience. Some of the methods followed are:

* + ***Exploratory Testing:***

This method, generally conducted by business analysts and experts, is followed to test the applications without any documentation.

* + ***Fault Attack:***

One of the widely used method of the experience based test design technique, fault attack involves testers anticipating the errors, defect data availability, etc., as per their experience

**Testing types**

**Smoke Testing**

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 ensures that build is stable and a detailed level of testing is carried out further.

If testers find that the major critical functionality is broken down at the initial stage itself then testing team can reject the build and inform accordingly to the development team. Smoke Testing is carried out to a detailed level of any functional or regression testing.

**Sanity Testing**

Sanity testing is the subset of regression testing and it is performed when we do not have enough time for doing testing.

Sanity testing is the surface level testing where QA engineer verifies that all the menus, functions, commands available in the product and project are working fine.

**Regression testing**

Regression Testing is defined as a type of software testing to confirm that a recent program or code change has not adversely affected existing features. Regression Testing is nothing but full or partial selection of already executed test cases which are re-executed to ensure existing functionalities work fine.

For example, in a project there are 5 modules: login page, home page, user's details page, new user creation and task creation.

Suppose we have a bug in the login page: the login page's username field accepts usernames which are shorter than 6 alphanumeric characters, and this is against the requirements, as in the requirements it is specified that the username should be at least 6 alphanumeric characters.Now the bug is reported by the testing team to the developer team to fix it. After the developing team fixes the bug and passes the app to the testing team, the testing team also checks the other modules of the application in order to verify that the bug fix does not affect the functionality of the other modules.

**Difference between Regression and Retesting**

1-Retesting is done to make sure that bug is fixed and failed functionality is working fine or not, This is kind of verification method followed in testing field for the fixed bugs. Whereas, Regression is re-execution of the test cases for unchanged part to see that unchanged functionality is working fine or not.

2- Retesting is a planned testing while Regression is known as the generic testing.

3- Retesting is only done for failed Test cases while Regression is done for passed test cases.

4- We should always keep this in mind, Re-testing has higher priority than the regression testing. But in bigger projects Retesting and Regression is done in parallel effort. But never forget importance of both in the success of the project.

**Usability Testing**

In usability testing basically the testers test the ease with which the user interfaces can be used. It tests that whether the application or the product built is user-friendly or not. Usability testing also reveals whether users feel comfortable with your application or Web site according to different parameters - the flow, navigation and layout, speed and content - especially in comparison to prior or similar applications.

**Negative Testing**

Negative testing is performed to ensure that the product or application under test does NOT fail when an unexpected input is given. The purpose of Negative testing is to break the system and to verify the application response during unintentional inputs. Negative Testing is carried out to spot the faults that can result in significant failures. It is performed to expose the software weakness and potential for exploitation.

**Recovery Testing**

Recovery testing is the activity of testing how well an application is able to recover from crashes, hardware failures and other similar problems. Recovery testing is the forced failure of the software in a variety of ways to verify that recovery is properly performed. It is basically done in order to check how fast and better the application can recover against any type of crash or hardware failure etc.

**Performance testing**

Software performance testing involves testing software applications to ensure they will perform well under their expected workload. Features and Functionality supported by a software system is not the only concern. A software application's performance like its response time, do matter. The goal of performance testing is not to find bugs but to eliminate performance bottlenecks.

The focus of Performance Testing is checking a software program's

* Speed - Determines whether the application responds quickly
* Scalability - Determines maximum user load the software application can handle.
* Stability - Determines if the application is stable under varying loads

[**Load Testing**](https://www.guru99.com/load-testing-tutorial.html)

Testing technique that puts demand on a system or device and measures its response. It is usually conducted by the performance engineers.

[**Stress Testing**](https://www.guru99.com/stress-testing-tutorial.html):

Testing technique which evaluates a system or component at or beyond the limits of its specified requirements. It is usually conducted by the performance engineer.

[**Endurance Testing**](https://www.guru99.com/endurance-testing.html)

Type of testing which checks for memory leaks or other problems that may occur with prolonged execution. It is usually performed by performance engineers.it is also called soak testing.

Endurance testing involves testing a system with a significant load extended over a significant period of time, to discover how the system behaves under sustained use.

For example, in software testing, a system may behave exactly as expected when tested for 1 hour but when the same system is tested for 3 hours, problems such as memory leaks cause the system to fail or behave randomly.

**Installation Testing**

Installation testing is check that software application is successfully installed & is working as expected after installation. This is testing phase prior to end users will firstly interact with the actual application. Installation testing is also called as “Implementation Testing”.

**Compatibility Testing**

Compatibility testing is conducted on the application to evaluate the application's compatibility with the computing environment.

Compatibility Testing is a type of Software testing to check whether your software is capable of running on different hardware, operating systems, applications, network environments or[mobile](https://www.guru99.com/mobile-testing.html)devices.

[**End-to-end Testing**](https://www.guru99.com/end-to-end-testing.html)

Similar to system testing, involves testing of 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. It is performed by QA teams.

**Security Testing**

Security Testing is a type of Software Testing that uncovers vulnerabilities of the system and determines that the data and resources of the system are protected from possible intruders. It ensures that the software system and application are free from any threats or risks that can cause a loss.

[**Penetration Testing**](https://www.guru99.com/learn-penetration-testing.html)

Testing method which evaluates the security of a computer system or network by simulating an attack from a malicious source. Usually they are conducted by specialized penetration testing companies.

**Gorilla Testing**

Gorilla Testing is a Software testing technique wherein a module of the program is repeatedly tested to ensure that it is working correctly and there is no bug in that module.

In Gorilla testing test case and test data are not required. It uses random data and test cases to perform testing of application.It is called frustrated testing.

[**Exploratory Testing**](https://www.guru99.com/exploratory-testing.html)

Black box testing technique performed without planning and documentation. It is usually performed by manual testers.

Exploratory testing, the application is tested while learning it.Testers can increase their knowledge by testing and exploring.

**Adhoc Testing**

Testing performed without planning and documentation - the tester tries to 'break' the system by randomly trying the system's functionality. It is performed by the testing team.

Ad-hoc testing means learn the application and then test it. Ad-hoc tester should have complete knowledge about the requirements of the system.

Before going for ad-hoc testing, the tester should have done sufficient testing of the system.

**Benefits Realization tests**

The benefits realization test is a test or analysis conducted after an application is moved into production in order to determine whether the application is likely to deliver the original projected benefits. The analysis is usually conducted by the business user or client group who requested the project and results are reported back to executive management.

**API TESTING**

API testing is a software testing type that validates Application Programming Interfaces (APIs). The purpose of API Testing is to check the functionality, reliability, performance, and security of the programming interfaces.

**What is an API?**

API (**A**pplication **P**rogramming **I**nterface) enables communication and data exchange between two separate software systems. A software system implementing an API contains functions/sub-routines which can be executed by another software system.

API tests are very different from GUI Tests and won't concentrate on the look and feel of an application. It mainly concentrates on the business logic layer of the software architecture.

**WEB APPLICATION TESTING**

Web Testing in simple terms is checking your web application for potential bugs before its made live or before code is moved into the production environment.

Some or all of the following testing types may be performed depending on your web testing requirements.

* Functionality testing
* Usability testing
* Interface testing
* Database Testing
* Compatibility Testing
* Performance
* Security

**1. Functionality Testing:**

This is used to check if your product is as per the specifications you intended for it as well as the functional requirements

**Test**all **links**in your webpages are working correctly and make sure there are no broken links. Links to be checked will include -

* Outgoing links
* Internal links
* Broken Links
* MailTo Links

**Test Forms**are working as expected. This will include-

* Scripting checks on the form are working as expected. For example- if a user does not fill a mandatory field in a form an error message is shown.
* Check default values are being populated
* Once submitted, the data in the forms is submitted to a live database or is linked to a working email address
* Forms are optimally formatted for better readability

**Test Cookies** are working as expected. Cookies are small files used by websites to primarily remember active user sessions so you do not need to log in every time you visit a website. Cookie Testing will include

* Testing cookies (sessions) are deleted either when cache is cleared or when they reach their expiry.
* Delete cookies (sessions) and test that login credentials are asked for when you next visit the site.

**2. Usability testing:**

Usability Testing has now become a vital part of any web based project. It can be **carried out by testers** like you **or a small focus group** similar to the target audience of the web application.

**Test**the site **Navigation**:

* Menus, buttons or Links to different pages on your site should be easily visible and consistent on all webpages
* Content should be legible with no spelling or grammatical errors.
* Images if present should contain an "alt" text

**3.Interface Testing:**

Interface testing is to test the interface between the web server and application server, application server and database server have proper interaction or not. It ensures a positive user experience. It includes verifying the communication processes as well as making sure that error messages are displaying correctly.Three areas to be tested here are - Application, Web and Database Server

* **Application:** Test requests are sent correctly to the Database and output at the client side is displayed correctly. Errors if any must be caught by the application and must be only shown to the administrator and not the end user.
* **Web Server**: Test Web server is handling all application requests without any service denial.
* **Database Server:**Make sure queries sent to the database give expected results.

**Database Testing:**

Database is one critical component of your web application and stress must be laid to test it thoroughly. Testing activities will include-

* Test if any errors are shown while executing queries
* Data Integrity is maintained while creating, updating or deleting data in database.
* Check response time of queries and fine tune them if necessary.
* Test data retrieved from your database is shown accurately in your web application

Compatibility Test

Compatibility tests ensures that your web application displays correctly across different devices. This would include-

**Browser Compatibility Test**: Same website in different browsers will display differently. You need to test if your web application is being displayed correctly across browsers, JavaScript, AJAX and authentication is working fine. You may also check for Mobile Browser Compatibility.

The rendering of web elements like buttons, text fields etc. changes with change in **Operating System**. Make sure your website works fine for various combination of Operating systems such as Windows, Linux, Mac and Browsers such as Firefox, Internet Explorer, Safari etc.

**Performance Testing:**

This will ensure your site works under all loads. Testing activities will include but not limited to

* Website application response times at different connection speeds
* Load test your web application to determine its behavior under normal and peak loads
* Stress test your web site to determine its break point when pushed to beyond normal loads at peak time.
* Test if a crash occurs due to peak load, how does the site recover from such an event
* Make sure optimization techniques like gzip compression, browser and server side cache enabled to reduce load times

**Security testing:**

Security Testing is vital for e-commerce website that store sensitive customer information like credit cards. Testing Activities will include-

* Test unauthorized access to secure pages should not be permitted
* Restricted files should not be downloadable without appropriate access
* Check sessions are automatically killed after prolonged user inactivity
* On use of SSL certificates, website should re-direct to encrypted SSL pages.