**SOFTWARE** :- A set of instruction or data or programs used to operate computers and executes specific tasks.

**MANUAL TESTING**

Software consists of computer programs that instruct the execution of a computer.

Software also includes design documents and specifications.

**TYPES OF SOFTWARE**:-

* **SYSTEM SOFTWARE** :- A software that provides a platform for other software.

A type of computer program that runs a computer’s hardware and application programs.

e.g.:-device driver, operating system, servers, utilities etc.

**device driver** :-A software program that allows a computer’s operating system to communicate with a connected device.

**Operating system** :-Manage all of the software and hardware on the computer.

**Server**:-A computer program or device that provides a service to another computer program and its user, also known as the client.

**Utility**:-A type of system software that helps improve a computer’s performance and speed.

* **PROGRAMMING SOFTWARE** :-programming software is used to write code, which are instructions that computers follow to perform tasks.

Also known as a code editor or integrated development environment (IDE),is used to write code in one or more programming languages to create programs that computers can follow to perform tasks.

e.g.:-compilers, debuggers, interpreters etc.

**Compiler:**-A computer program that translates computer code written in one programming language into another language.

It converts the source code into machine code.

**Debuggers:**-A computer program used to test and debug other programs.

**Interpreters:**- An interpreter translates code line -by-line as the code runs.

* **APPLICATION SOFTWARE**.:-A computer programs that helps users perform specific tasks such as personal ,educational or business tasks.

When a user interacts directly with a piece of software, it is called application software.

e.g.:- web application, mobile application, desktop application.

**HARDWARE** :-the external and internal devices and equipment that enable you to perform major functions such as input, output, storage, communication, processing, and more.

**SOFTWARE TESTING :-**

* It helps to ensure that software works as intended, and avoid costly errors.
* A part of software development process.
* An activity to detect and identify the defects/bug in the software.
* The objective of testing is to release quality product to the client.
* Improve performance of application.
* Better user experience.
* Save time and money.

**WHAT IS SOFTWARE QUALITY**:-

It can include usability, security, performance, reliability, compatibility, and efficiency.

Within budget

Delivered on time

Bug free

Meets requirements or expectations

maintainable

**SOFTWARE QUALITY**

**PROJECT VS.PRODUCT**

PRODUCT:- Developed for multiple customers based on market requirement then it is called product.

PROJECT:- Developed for specific customer based on the requirement then it is called project.

* WHY THE SOFTWARE HAS BUGS?

1. Miscommunication or no communication.
2. Software errors.
3. Changing requirements.
4. Lack of skilled testers.

* **ERROR :-** Human mistakes made in the code or design that led to incorrect results. like syntax mistake, logical mistake.

Example:-A developer mistype a mathematical formula in the code.

* **BUG:-** An error found during testing before the product is released.

Example:-during testing, the login feature fails because of the incorrect formula in the code.

* **FAILURE**:-When a system or component does not perform as expected or intended.

OR, failure occurs when the software does not perform as required in the live environment due to an unsolved defect.

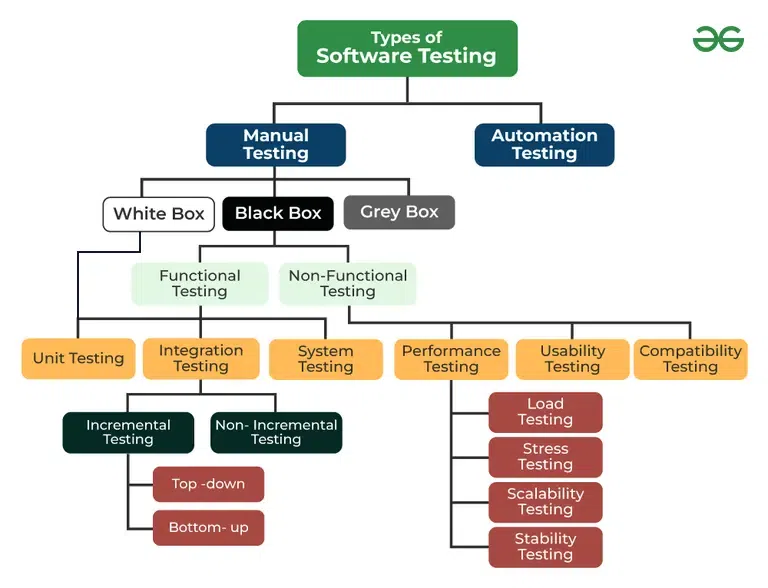
Example:-A user is unable to login because the defect was not fixed before release.

* **FAULT**:-A defect in the system’s code that, when executed, causes the system to behave unexpectedly.

Example:-the incorrect formula causes the application to miscalculate values.

* **DEFECT:**-A discrepancy between expected and actual results in the software, reported by the tester.

Example:- the tester logs a defect in the system because the login feature does not work as expected.



**VERIFICATION AND VALIDATION**

Verification Validation

1: Are you building it right?

**SOFTWARE/SYSTEM DEVELOPMENT** **LIFE CYCLE**

(SDLC)

* It is a process that helps to development teams design, build, test and deploy software.
* The goal of SDLC is to create high quality software that meets customer expectations, is completed on time, and is within budget.
* SDLC is a process followed for software building within software organization. It consists of a precise plan that describes how to develop, maintain. Replace, and enhance specific software.

SRS:- (software requirement specification) A complete specification or description of requirement of the software that need to be fulfilled for the successful development of the software system.

FRS:- (functional requirement specification)A document that describes the functions a system or component must perform.

BRS:- (business requirement specification)A document that outlines the requirements for a project from a business perspective.

SOW:- (signature of work) in e-business a signature can be mark or symbol that both parties accept to show approval, intent, or responsibility for a document.

The SDLC model involves 7 phases or stages while developing any software.

**Stage 1: Project planning**:- it is a vital role in the software delivery lifecycle since this is the part where the team estimates the cost and defines the requirements of the new software.

**Stage 2:-Gathering requirements and analysis**:-Discuss each detail and specification of the product with the customer.

The main goal of this stage is that everyone understands even the small details of the requirement. Hardware, operating systems, and security are to name of the few requirements.

**Stage 3:-Design**:- The program developer scrutinizes whether the prepared software suffices all the requirements of the end-user.

Additionally ,if the project is feasible for the customer technologically, practically, and financially.

**Stage 4:-Coding or Implementation**:- Once the developer decides on the best design approach, he then selects the programming languages like java, python etc.

In this stage translating the design to a computer-legible language. tasks are divided into modules or units and assigned to various developers.

The developer can show the work done to the business analysts in case if any modifications or enhancement required.

**Stage 5:-Testing**:-Once the developer builds the software, then it is deployed in the testing environment. then the testing team tests the functionality of the entire system.

After testing the QA and Testing team might find some bugs or defects and communicate the same with the developers. the development team then fixes the bugs and send it to QA for re-test.

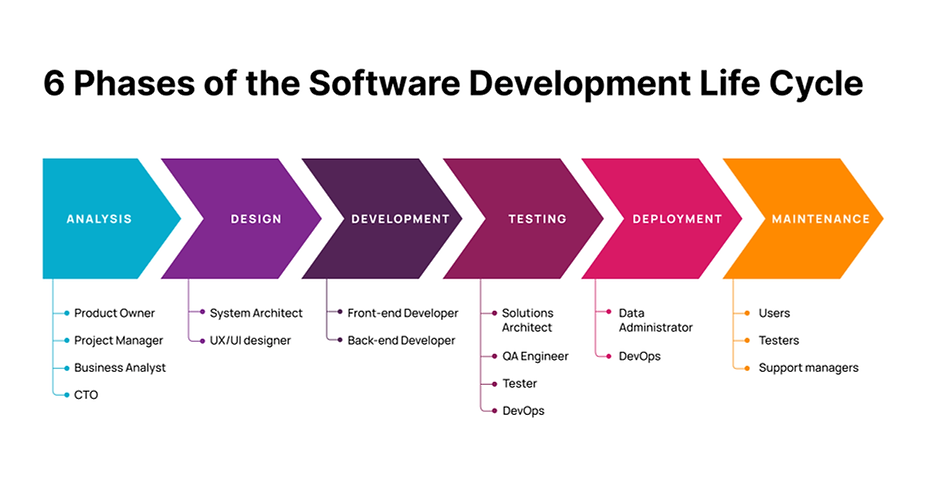
This process goes on until the software is stable, bug-free and working according to the business requirement of that system.

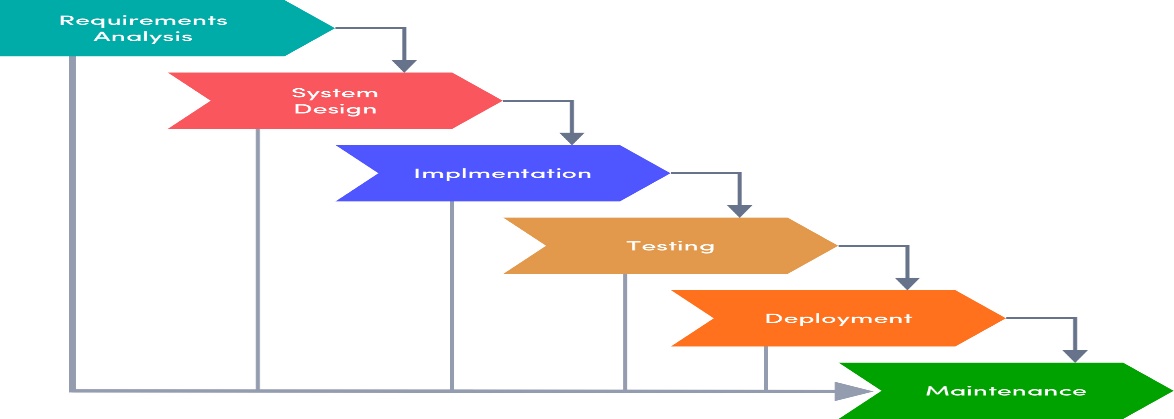
**Stage 6:-Deployment**:- Once the testing is done, and the product is ready for deployment.

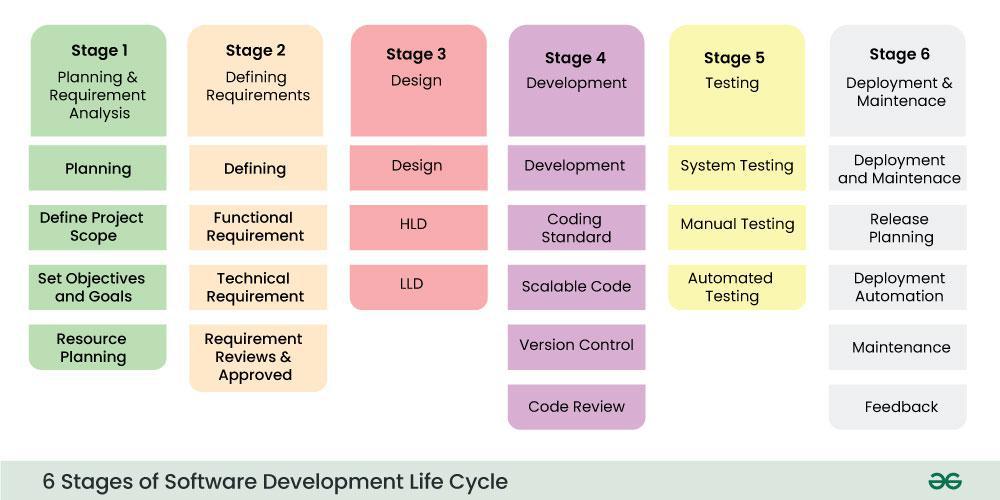
It is released for customers to use. the size of the project determines the complexity of the deployment.

**Stage 7:- Maintenance**:- The actual problem starts when the customer actually starts using the developed system and those needs to be solved from time to time.

According to the changing user end environment or technology the software is updated timely.







**SOFTWARE TESTING LIFE CYCLE(STLC)**

A systematic method that guides us through the several stages of software testing is called the software testing life cycle.it meets the requirements and is free of defects.

The main goal of the STLC is to identify and document any defects or issues in the software application as early as possible in the development process.

It is a process that follows a series of steps or phases.

**PHASES**:-

**Phase 1:-Requirement and analysis**:-In which test team studies the requirements from a testing point of view to identify testable requirements and the QA team may interact with various stakeholders to understand requirements in details.

Requirements could be either functional or non-functional.

e.g.:-project name, project module, SRS/FRS file

**Phase 2:-Test planning**:- it is the most efficient phase of the software testing life cycle where all testing plans are defined.in this phase manager of the testing team calculates the estimated effort and cost for the testing work.

Selecting the testing methods and techniques that will be used. Identifying the testing cases that will be executed and the test data that will be used.

Estimating the time and cost required for testing. Assigning role and responsibility to the testing team .

**Phase 3:- Test case development**:-In this phase testing team notes down the detailed test cases.

The testing team also prepares the required test data for the testing.

Identifying the test cases that will be developed. Write test cases that are clear ,concise test data and test. Creating test data and test scenarios that will be used in the test cases.

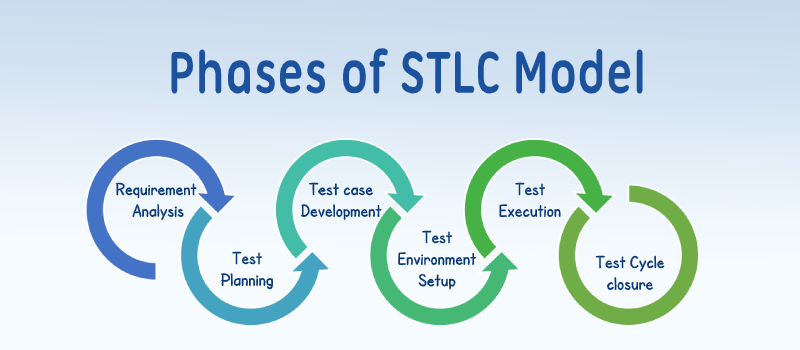
Identifying the expected results for each test case. Reviewing and validating the test cases. Updating the requirement traceability matrix(RTM) to map requirement to test cases.

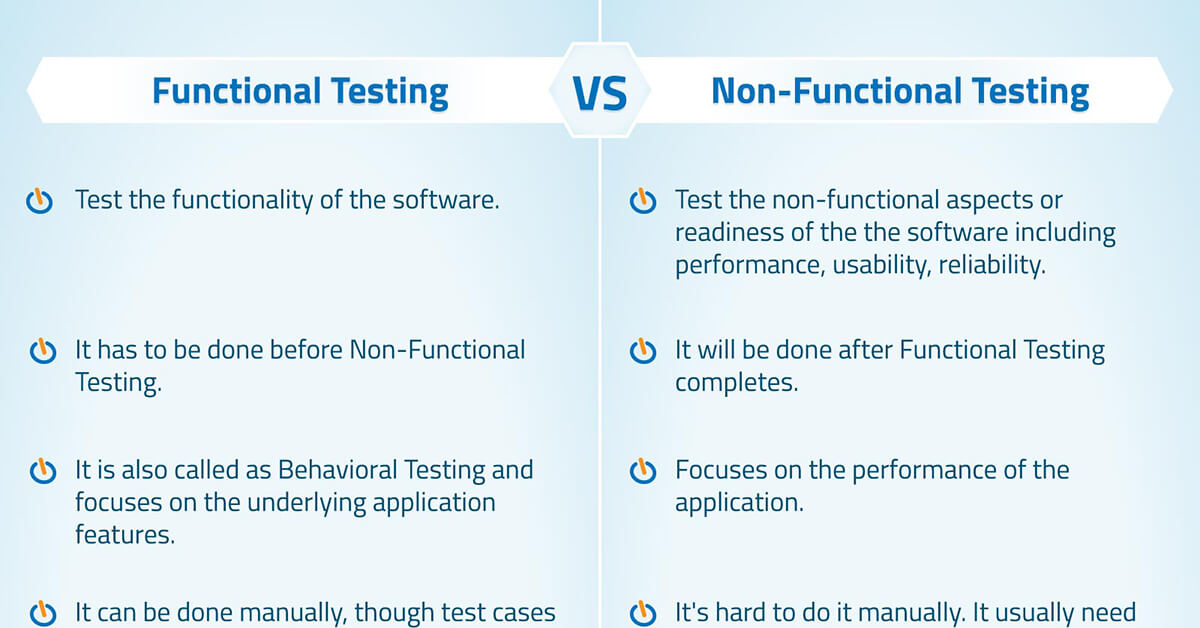
**Phase 4:- Test case environment**:-test environment decides the conditions on which software is tested e.g.:- chrome

**Phase 5:- Test case execution**:-testing team starts executing test cases based on prepared test cases in the earlier step.it is important to note that test execution is an iterative process and may need to be repeated multiple times until all identified defects are fixed and the software is deemed fit for release.

**Phase 6:-Test closure**:-the main objective of the test closure stage is to ensure that all testing related activities have been completed and that the software is ready for release.

The main activities that take place during the test closure stage include:-test summery report, defects tracking, test environment clean up, test closure report, feedback and improvements.





NOTES:-module:- number of pages or number of tasks.

Sprint:- time of duration.

Standard sprint time:-2week-4week

**BUG LIFE CYCLE**

The specific set of states that defect or bug goes through in its entire life. the purpose of bug life cycle is to easily coordinate and communicate current status of defect which changes to various assignees and make the defect fixing process systematic and efficient.

**PHASES:-**

**New**:-when a new defect is logged and posted for the first time,it is assigned a status as NEW.

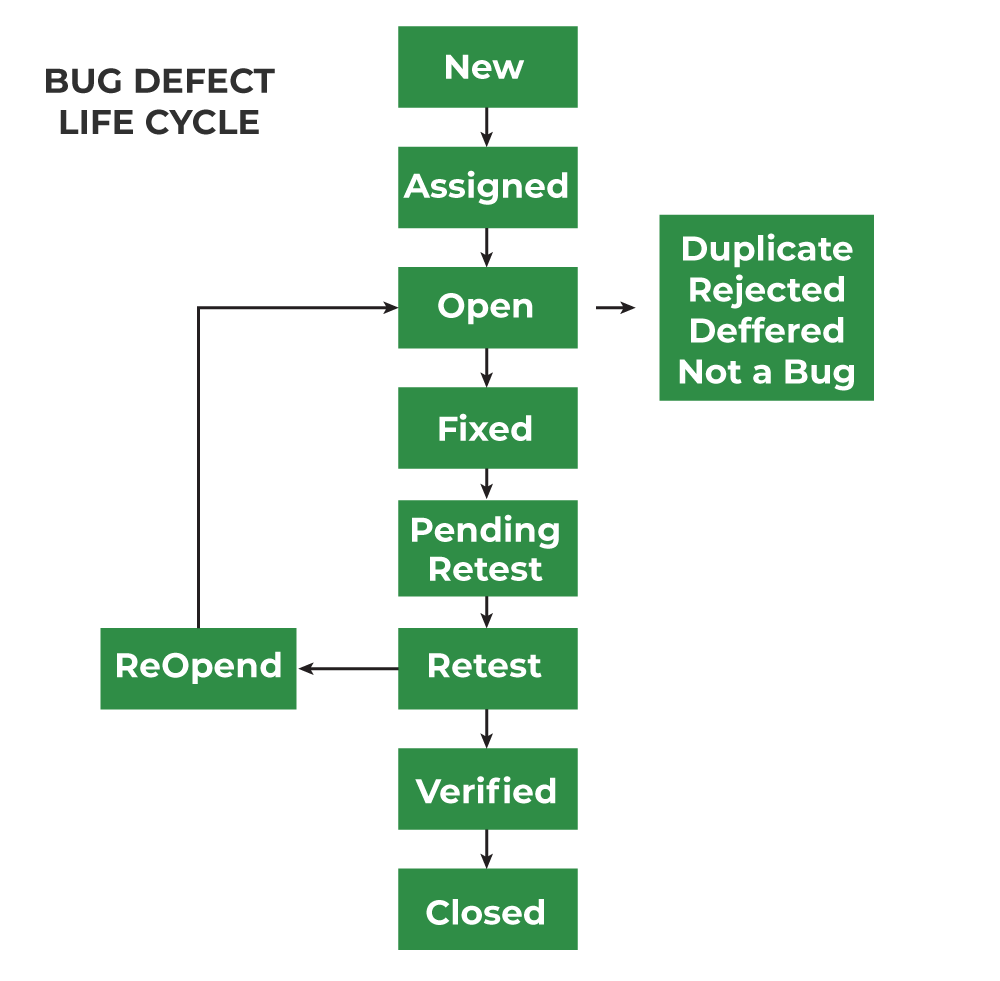
**Assigned**:-Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to the developer team.

**Open**:-the developer starts analyzing and works on the defects fix.

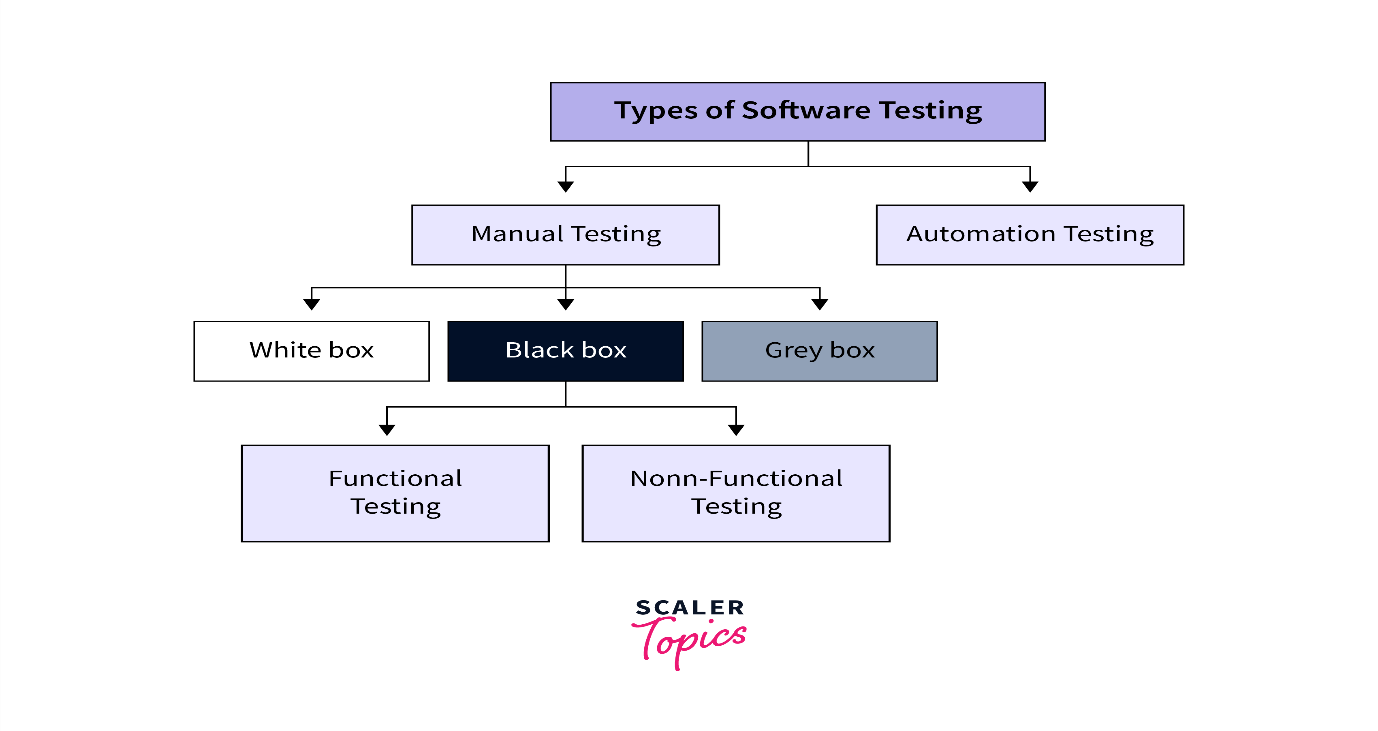
**Fixed**:- when a developer makes a necessary code change and verifies the change, he or she can make bug status as “FIXED”.

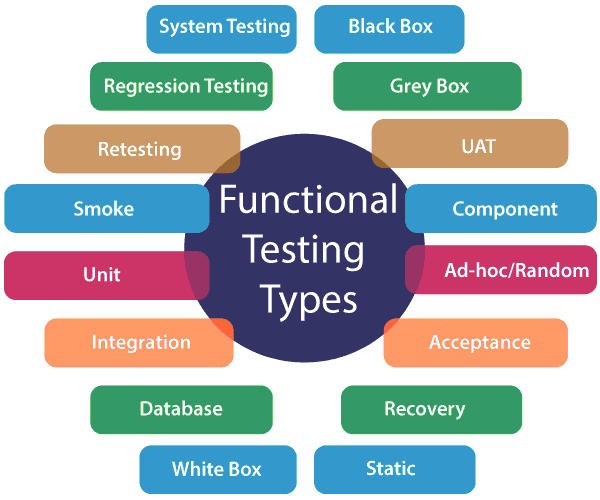
**Pending retest**:- Once the defect is fixed the developer gives a particular code for retesting the code to the tester. Since the software testing remains pending from the testers end, the status assignedis “pending retest”.

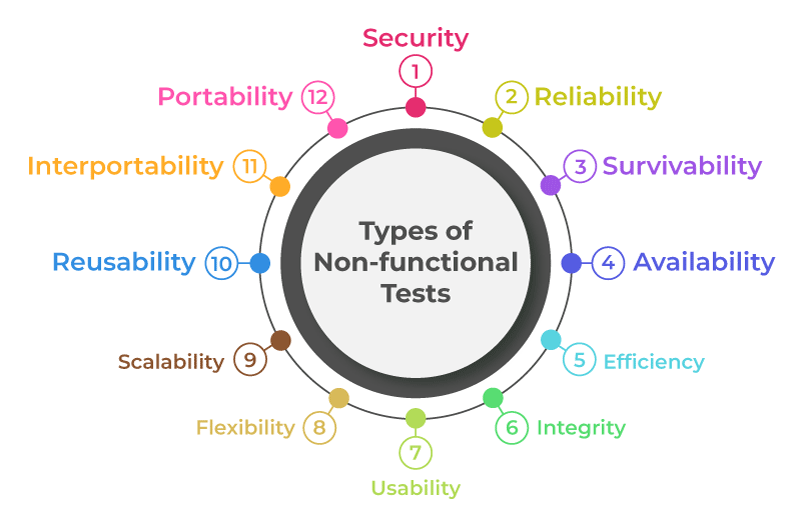
**Retest:**-tester does the retesting of the code at this stage to check whether the defect is fixed by the developer or not and changes the status to “re-test”.



**TYPES OF TESTING**







* **MANUAL TESTING**:-without any code and without any tools, we are testing any application that is called manual testing.

Manually reviewing and testing a software application, simulating the behavior of a real user, to identify errors, bugs, and other issues without using automated tools.

* **Automation testing**:-the process of using specialized software to automatically test and validate a software product.

Tester has the complete knowledge of the application being tested, including access to source code and design documents. tester know the coding syntax.

* **Black box testing**:-A software testing method that evaluates an application’s functionality without knowing its internal code or structure.
* **White box testing**:-A software testing method that allows testers to inspect and verify the internal working of a software system.
* **Gray box testing**:-A software testing technique that combines white box and black box testing. the tester has some knowledge of the system’s internals, but not complete knowledge.
* **Functional testing**:- A type of testing that seeks to establish whether each application feature works as per the software requirements.
* **Non-functional**:-A type of software testing that evaluates how a system operates, rather than the specific functions of the system.

**FUNCTIONAL TESTING TYPES**:-

1. **UNIT TESTING**:-small piece of code tested. A software development process that involves testing the smallest functional units of code in an application.
2. **INTEGRATION TESTING**:- Interconnection between two pages or modules or components.

Or, A type of software testing that verifies that multiple parts of a software system work together as expected.

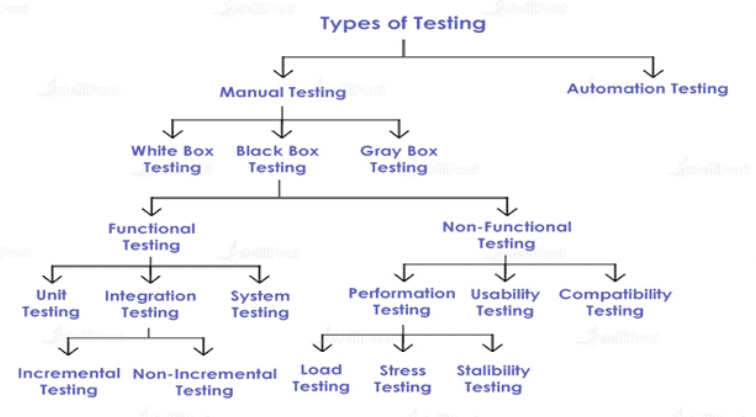
1. **SYSTEM INTEGRATION TESTING**:- A software testing phase that verifies that all the components of a system work together as intended. End to end testing(start to end testing)

Unit testing + integration testing=system integration testing.

**End to End testing**:-A software testing techniques that simulates real- world user scenarios to validate an application’s functionality and performance from start to finish.

1. **MAINTENANCE TESTING**:-A process that involves evaluating how changes to a system affects its performance and functionality.

**Penetration testing**:-A security exercise that simulates a cyberattack to identify weakness in a system’s defenses.





**TESTING TECHNIQUE**

The method applied to evaluate a system or a component with a purpose to find if it satisfies the given requirements. OR,

Software testing techniques are methods to check if a software program works properly, meets its goals, and assesses the quality of software.

1. Boundary value analysis 2) equivalence partitioning
2. Decision table 4) state transition

**BOUNDARY VALUE ANALYSIS**:-BVA is one of the functional testing.

Boundary value analysis is based on testing the boundary values of valid and invalid partitions. the behavior at the edge of the equivalence partition is more likely to be incorrect than the behavior within the partition, so boundaries are an area where testing is likely to yield defects.

It checks for the input values near the boundary that have chance of error. Every partition has its maximum and minimum values are the boundary values of a partition.

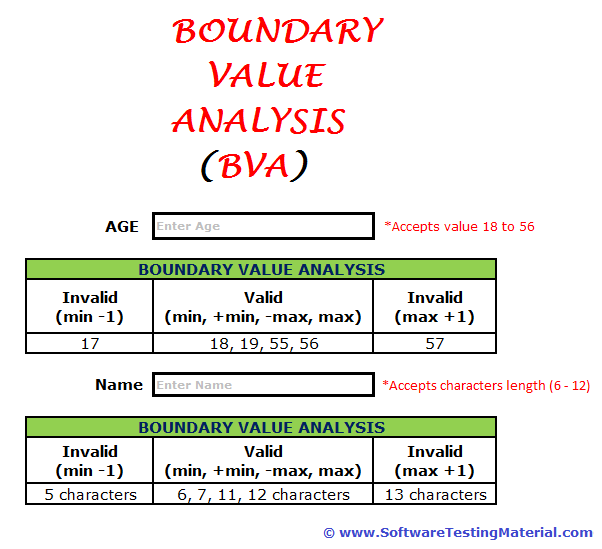
Example:-consider a system that accepts age from 18 to 56.

Boundary value analysis(age accepts 18 to 56)

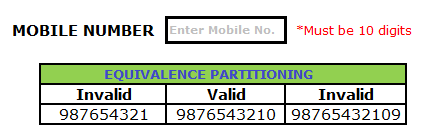
|  |  |  |
| --- | --- | --- |
| invalid | valid | invalid |
| (min-1) | (min,( min+1), nominal, (max-1), max) | (max+1) |
| 17 | 18, 19, 37, 55, 56 | 57 |

Valid test cases:- any value entered greater than 17 and less than 57.

Invalid test cases:-when any value less than 18 and greater than 56 is entered.



**EQUIVALENCE PARTITIONING**:- Also known as equivalence class partitioning(ECP) . it is a software testing technique or black box testing. it divides input data into partitions of equivalent values and then tests representative values from each partition.



**DECISION TABLE:-**this testing is a very effective tool in testing the software and its requirement managements.

A black box test design technique used where different combinations of test input conditions result in different outcomes.

E.g.:-

Next

Password

Email

DECISION TABLE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Testcase 1 | Testcase 2 | Testcase 3 | Testcase 4 |
| Email | v | in | v | in |
| Password | v | v | in | in |
| Outcome | p/f | p/f | p/f | p/f |

E.g.:-

next

Email id

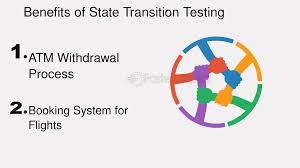
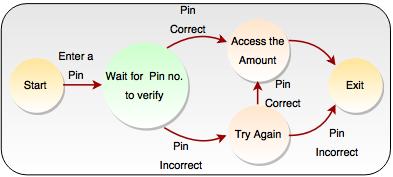
password

username

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Field | Tc1 | Tc2 | Tc3 | Tc4 | Tc5 | Tc6 | Tc7 | Tc8 |
| Field | TC1 | TC2 | TC3 | TC4 | TC5 | TC6 | TC7 | TC8 |
| username | V | v | v | v | in | in | in | in |
| Email id | V | v | in | in | v | v | in | in |
| password | V | in | v | in | v | in | v | in |
| outcomes | p/f | p/f | p/f | p/f | p/f | p/f | p/f | p/f |

**STATE TRANSITION**:- state transition testing is a type of software testing which is performed to check the change in the state of the application under varying input. The condition of input passed is changed and the change in state is observed.

DECISION TABLE

A directed relationship between two states, defining a trigger, condition, and behavior that lead to a change in state within a state machine.

**ERROR GUESSING**:-A software testing technique that uses a tester’s experience and intuition to identify and fix potential errors in a program.

NOTE:- Penetration testing:- A security exercise where a cyber security expert simulates a cyberattack to find vulnerabilities in a computer system.

**ALPHA TESTING**:-A stage in software development where internal teams test a product for bugs and issues before releasing it to the public.

The first end-to-end testing of a product to ensure it meets the business requirements and functions correctly.

**BETA TESTING**:-Beta testing is the process of testing a software product or service in a real- world environment before its official release. Beta testing performed by real users of the software application in a real environment. Beta testing is one of the types of user acceptance testing. A beta version of the software, whose feedback is needed, is released to a limited number of end-users of the product to obtain feedback on the product quality.

Beta testing is performed by clients or users who are not employees of the company, most important before release.

**TEST SCENARIOS**:-A high-level description of a user’s objective when using a program or system. A test scenario is a document that outlines the conditions under which a test will be performed.

How to write test scenario:-

1. Serial number
2. Test scenario id
3. Test scenario title/description/name
4. Reference
5. Total number of test cases

PROJECT DETAILS

E.g.:-

|  |  |
| --- | --- |
| Project name | E-commerce |
| Module name | Register module |
| Assign date | 21-Nov |
| Assign by | TL/PM |

TEST SCENARIOS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Serial number | TS\_ID | TS Title | Reference | Total NO. of test cases |
| 1 | TS\_01 | To verify user able to test **Register** **Page** test **Register Page** functionality | SRS/FRS | 16 |
|  |  |  |  |  |
| 2 | TS\_02 | Test **Login page** functionality | SRS/FRS | 5 |
|  |  |  |  |  |
| 3 | TS\_03 | Test **Logout** functionality | SRS/FRS | 1 |
|  |  |  |  |  |
|  |  |  |  |  |

**TEST CASE**:-A detailed document that outlines the steps, inputs, and expected results for testing a specific feature of a software application or system. Test cases are designed to help testers understand how to test a specific feature, compare expected results to real results, avoid missing important steps, and be assigned to the right workload.

How to write a test cases:-

1. Serial number 2) test case ID 3) test scenario ID 4) test case title 5) Test scenario title
2. Pre-condition 7) Test steps 8)Test data 9) Actual result 10) Expected result 11)Status

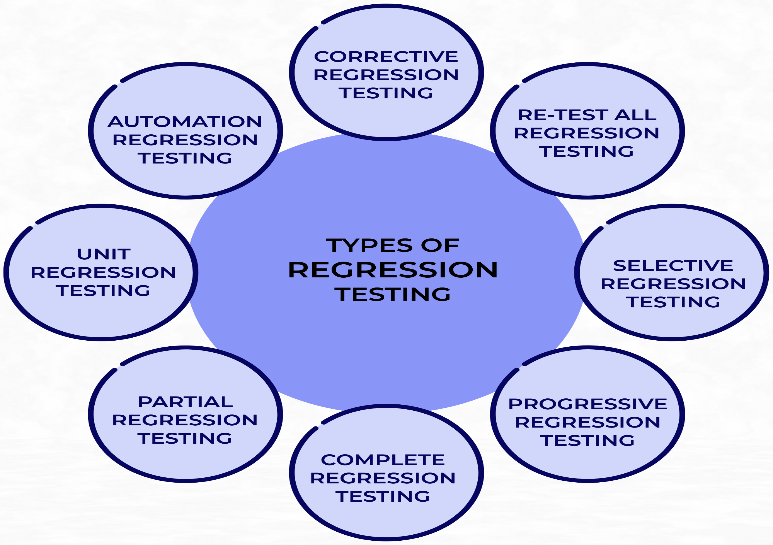
**SMOKE TESTING**:-Also known as “Build Verification Testing” or “Build Acceptance Testing”, or “Confidence Testing ”is a type of software testing that is typically performed at the beginning of the development process to ensure that the most **critical** functions of a software applications are working correctly.

**SANITY TESTING**:-Also known as surface level testing. A software testing technique that verifies that minor changes to a software application do not introduce new defects.

Only New feature tested. Subset of regression testing.

**REGRESSION TESTING**:-A type of software testing that ensure that new changes to a software product do not break existing features.

Old features+ New features= Both are tested.

-

**Fig:-Types of Regression testing**

1. **Complete regression testing**:- Test start to end. Test the entire system, which is often used when a major change has been made. this type of testing is time-consuming and is used to establish the software’s stability.
2. **Partial regression testing**:-Tests the modified part of the software and any related components. This type of testing is used to ensure that the new code hasn’t disrupted the application’s performance.
3. **Unit regression testing**:-Tests a specific unit of code, such as a function or method, to ensure it still works as expected.
4. **Corrective regression testing**:-Runs all the test cases before making changes to the code. This type of testing is used when there are no updates or changes to the source code.
5. **Progressive regression testing**:-Tests new features while ensuring that older features still work. This type of testing is used when new system components are introduces.
6. **Selective regression testing**:-Tests specific test cases that are likely to be affected by the change. This type of testing saves time and resources by avoiding testing the entire test suite.
7. **Non-functional regression testing**:- Test non-functional aspects of the system, such as performance and usability.

**AD-HOC TESTING**:-Also known as random testing or sudden testing or monkey testing or informal testing. A software testing technique that involves exploring an application without a formal test plan or predefined steps. This testing can only be conducted by those testers who have a good and thorough knowledge of working of the application.



Types of Ad-hoc testing:-1: Buddy testing 2: Pair testing 3:Monkey testing.

**BUDDY TESTING**:-Buddy testing is a unique software testing approach involving two team members one from the development team(developer) and one from the testing team. this collaborative effort allows for a more comprehensive evaluation of the application, as both individuals work together on the same module, sharing ideas and uncovering defects and bugs.

**MONKEY TESTING**:-This testing involves testers randomly performing action on the software application to see how the application responds to unexpected inputs. e.g.:-A tester randomly enters text into the search bar of an e-commerce site and then stops it to see how the site responds to unexpected stops.

**PAIR TESTING**:-In the pair testing approach two testers are assigned to tackle a module, brainstorm brilliant ideas, and team up on the same machines to uncover any aggravating defects.

One person takes the lead as the tester, while the other steps in as the writer, taking notes on the finding.

**Difference between Monkey testing and Gorilla testing**:-gorilla testing tests particular modules, whereas monkey testing tests the system as a whole.

**BUG REPORT**:-A document that details a software issue or defect, and is used to inform developers and other stakeholders so that the problem can be resolved.

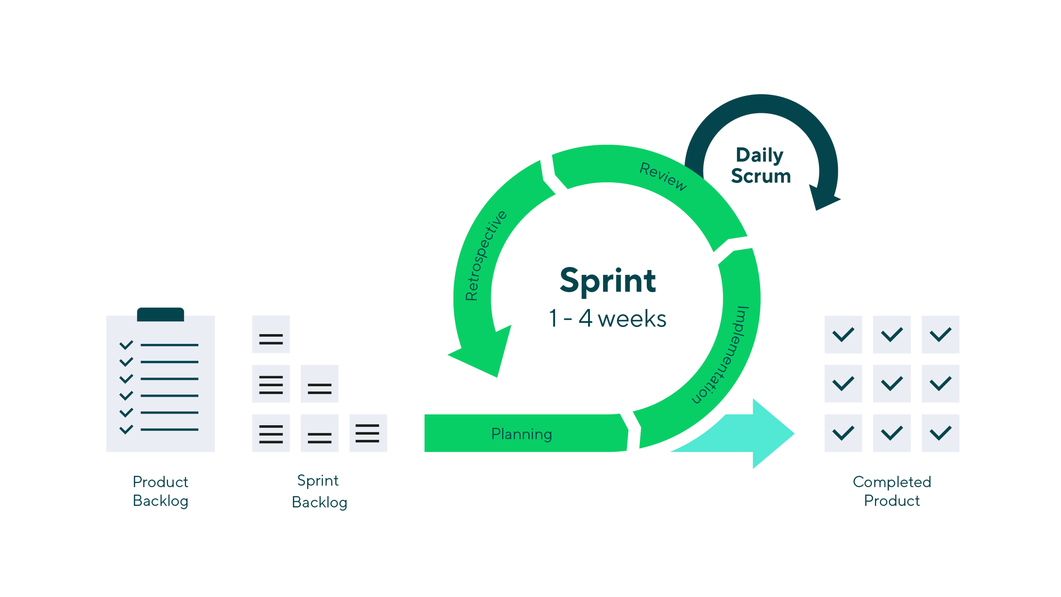
**How to write bug report:-**

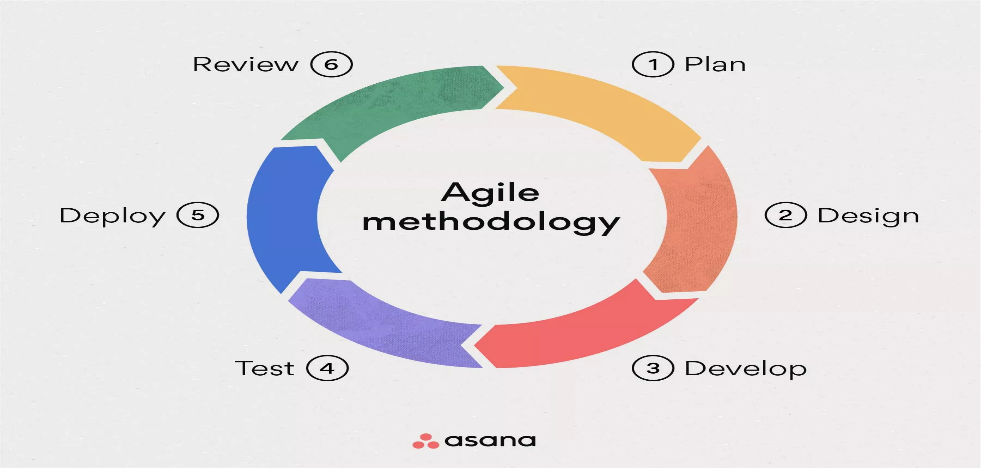
1. Serial number
2. Bug id
3. Bug Title
4. Module Name
5. Build number
6. Test case id
7. Test case Title
8. Test steps
9. Test data
10. Priority/Severity
11. Screenshot

**SOFTWARE DEVELOPMENT LIFE CYCLE(SDLC) MODELS:-**

1. Agile model
2. Iterative model
3. Incremental model
4. Waterfall model
5. V-model
6. Big Bang model
7. Spiral model
8. Prototype model

**AGILE METHODOLOGY**:-





The Agile methodology is a project management and software development approach or process that emphasizes flexibility, collaboration and customer centricity. It is the latest model used by major companies today like Facebook, google etc. it follows the **Iterative** as well as **Incremental** approach that emphasizes the importance of delivering of working product very quickly. It focuses on delivering smaller piece of work regularly instead of one big launch. Agile is not free.

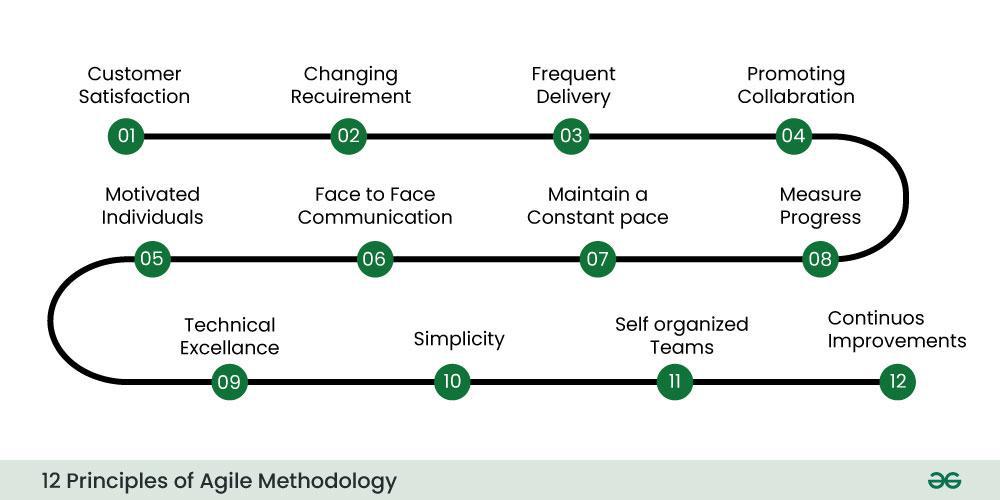
**NOTE**:-Scrum Meeting:-when the scrum master, product owner, and development team meet to plan work, discuss work in progress, gather feedback, and more.

**Four pillars of Agile**:-

1. Individuals and Interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Customer collaboration over contract negotiation.
4. Responding to change over following a plan.

**12 principles of Agile**:-

1. Make customers happy through early and continuous delivery of useful software.
2. Embrace changing requirements, even in later stages.
3. Deliver work frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale.
4. Stakeholders and developers must work together daily throughout the project.
5. Build projects around motivated individuals, giving them the environment and support they need, and trusting.
6. Face-to-Face conversations are the most effective method of communication.
7. The Main measure of progress is working software.
8. The working pace should be constant yet sustainable.
9. Pay continuous attention to technical excellence and good design.
10. Keep things as simple as possible.
11. The best results come from self-organizing teams.
12. The team reflects on how to become more effective at regular intervals, adjusting behavior accordingly.



**NOTE**:- **Backlog:**- A prioritized list of tasks that a product team needs to complete to deliver a product.

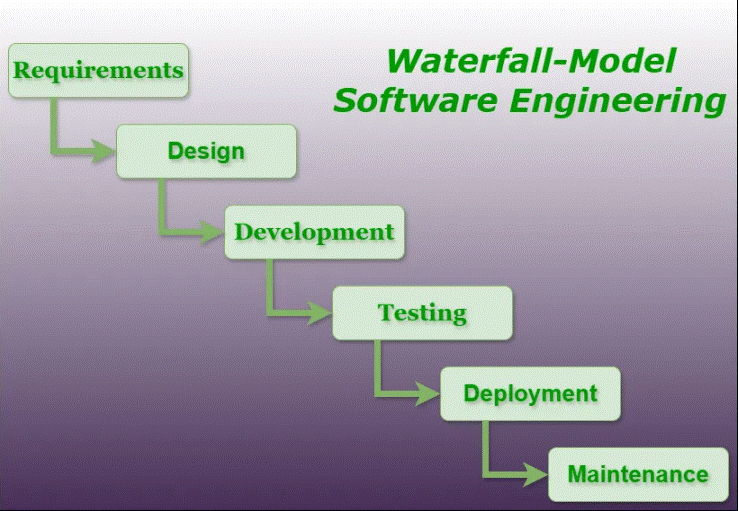
**Types of Agile methodologies**:-

1. **Scrum**:- it is deals for projects with rapidly changing requirements, using short sprints.
2. **Kanban:**- it visualizes project progress and is great for tasks requiring steady output.
3. **Lean**:- it streamlines processes, eliminating waste for customer value.
4. **Extreme programming(XP)**:- it enhances software quality and responsiveness to customer satisfaction.
5. **Adaptive project Framework(APF)**:- it works well for projects with unclear details, as it adapts to constantly evolving client needs.

**JIRA**:-A project management and issue/bug tracking tool from Atlassian that helps teams plan, track, and release software.

* **Agile project management**:- Jira supports agile methodologies like scrum and kanban, and can be used to manage software development projects.
* **Issue tracking**:- Jira can be used for bug tracking. Test management, and more.
* **Collaboration**:- Jira provides a centralized platform for teams to collaborate and stay aligned.
* **Integration:**-Jira integrates with a variety of quality assurance apps.
* **Customization:-**Jira allows for customizable fields, workflows, and screens.

WATERFALL MODULE:-It is a software development methodology. First introduce by Winston w. Royce in 1970. It is a-- linear and sequential approach .The waterfall model is useful in situations where the project requirements are well defined and the project goals are clear.



**Advantage**:-

* Easy to understand.
* Individual processing.
* Properly Defined.
* Properly Documented.
* Working for smaller project.

**Disadvantage:**-

* No feedback path/ Backtracking not possible.
* Difficult to accommodate change request.
* No overlapping of phases.
* Limited flexibility.
* Late defect detection.
* Lengthy development cycle.

SPIRAL MODEL:-The spiral model is a combination of the waterfall model and iterative model. First introduced by Barry Bohem. It provides support for **Risk Handling**. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called Phases.

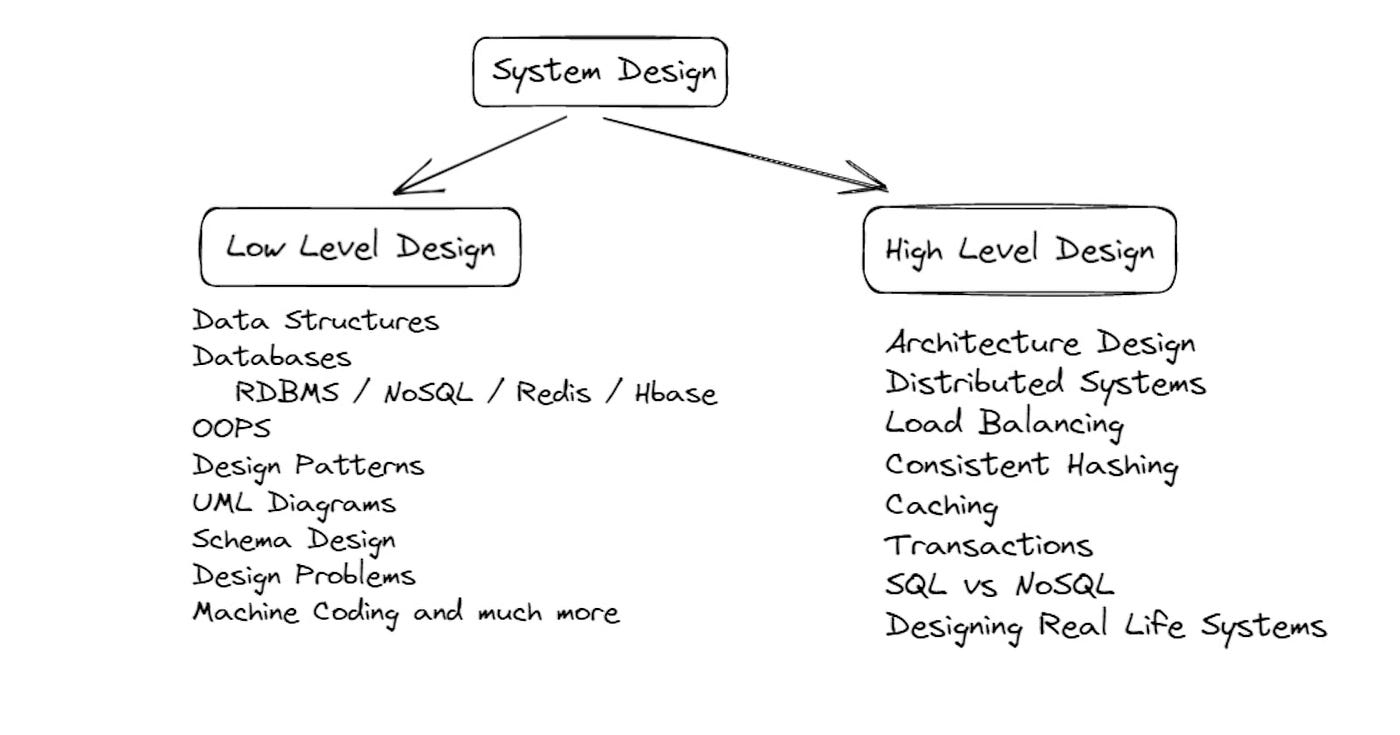


**NOTE**:-The radius of the spiral at any point represents the expenses (cost) of project so far, and angular dimension represents the progress made so far in the current phase.

**Phases**:-

* Identifications
* Design
* Construct or build
* Evaluation and Risk analysis

V -Model:-Also known as verification and validation model.



Iterative Model:-it involves breaking down projects into smaller, more manageable parts, called iterations.

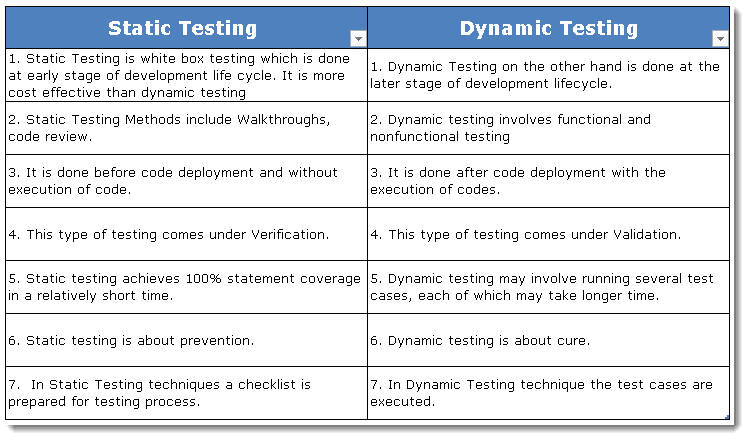


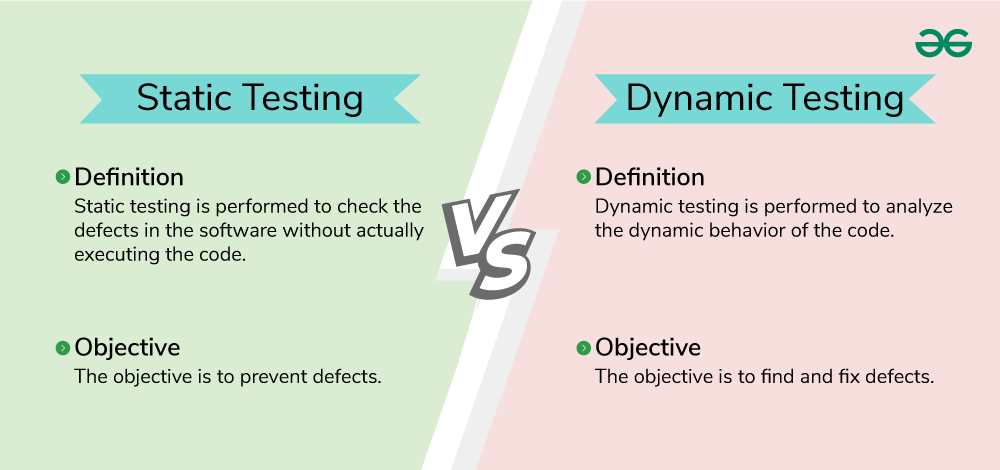
Incremental Model:-A software testing method that involves testing individual modules or segments of software as they are developed.

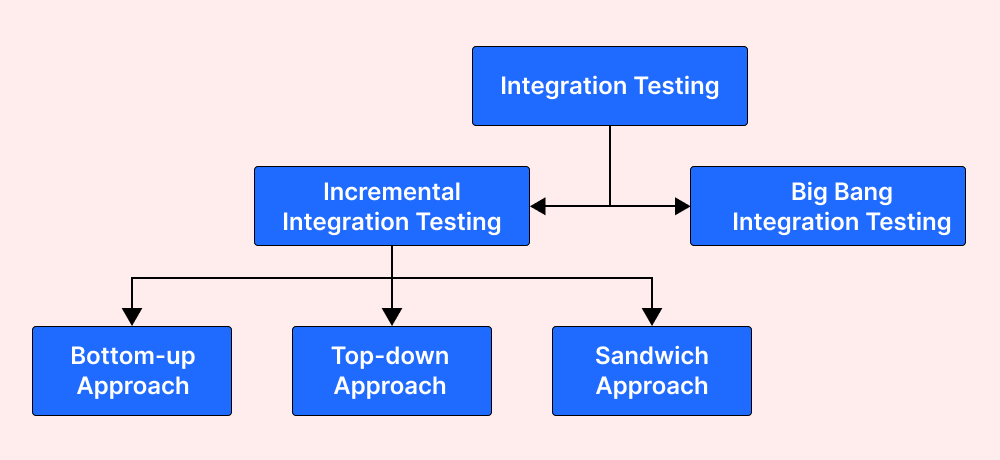
**Non- functional testing**:-

* Load testing
* Stress or soak testing
* Compatibility testing
* Stable testing
* Scabble testing
* Hybrid testing
* Usability testing
* Penetration testing
* Stability testing

**Static testing vs Dynamic testing**:-







Top to bottom testing:-Testing the highest- level modules of a program first, and then moving down to the lower -level components.

Bottom to top testing:-A method that involves testing individual components of a program first, and then gradually combining them to create larger parts until the entire system is tested.

Mutation testing:- A software testing technique that involves intentionally modifying a program’s source code to assess that quality of the software’s test suite.

Use case testing:-A software testing method that analyzes and validates a system’s behavior by testing scenarios, or use cases, to ensure it functions as intended.

Fuzz testing:-A software testing technique that involves feeding a computer program invalid, unexpected, or random data to uncover bugs and security vulnerabilities.

Requirement traceability matrix:-A document that shows the relationship between project requirements and other artifacts, such as tests, test results, and issues.