Software Testing Styles:

1. Unit Testing:

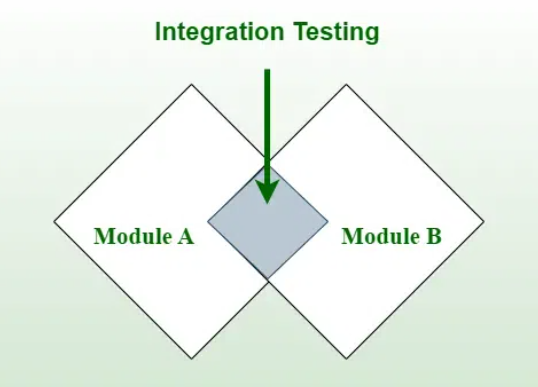
Unit Testing is a software testing technique where individual units or components of a software application are tested in isolation. These units are typically the smallest pieces of code, such as functions or methods, and the goal is to ensure they perform as expected. It helps identify bugs early in the development cycle, enhances code quality, and reduces the cost of fixing issues later.

Unit testing is a key part of software development that improves code reliability by verifying the correctness of each unit independently. Developers write unit tests for these code units and run them automatically whenever changes are made. If a test fails, it quickly points out the issue, making it easier and faster to fix. Unit testing promotes modular design, ensures better test coverage, and supports Test-Driven Development (TDD), encouraging the creation of dependable and maintainable code.

1. Integration Testing:

Integration Testing is a software testing technique that focuses on verifying the interactions and data exchange between different components or modules of a software application. It is performed after unit testing and before system testing to identify any problems or bugs that arise when modules are combined. The main goal of integration testing is to expose faults in the interaction between integrated units and ensure that the overall system functions as intended.

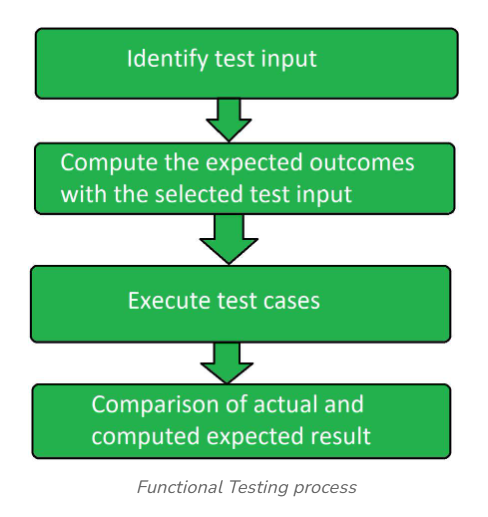
Integration testing is important because it verifies that individual modules work correctly together, helping to identify compatibility or communication issues early in the development cycle. By following a proper sequence when combining modules, developers can avoid missing important integration scenarios and ensure smooth interaction. Detecting and resolving integration problems at this stage improves the overall reliability, performance, and quality of the final software product.



1. Functional Testing:

Functional Testing is a type of software testing that focuses on verifying that the software system works according to the specified requirements. It checks whether the functions of an application — like user login, file uploads, or payment processing — behave as expected. Functional testing mainly involves black-box testing, where the tester does not need to know the internal code structure but only focuses on inputs and expected outputs.

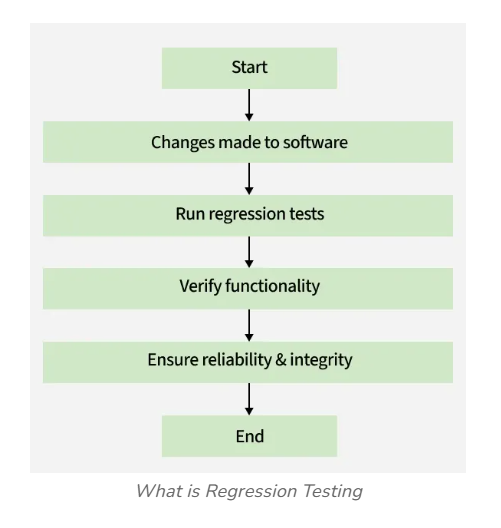
The goal of functional testing is to ensure that the software delivers the correct output for given inputs and that all user requirements are properly met. It helps detect issues like missing functionalities, incorrect system behavior, and user interface problems early in the development cycle. Functional testing is important because it validates the core actions of the application that users will interact with, ensuring the system meets business goals and customer expectations. It can be performed manually or automated, and it usually happens before non-functional testing like performance or security testing.



1. Regression Testing:

Regression Testing is a type of software testing that focuses on verifying that existing functionalities still work correctly after changes like bug fixes, enhancements, or new feature additions. The goal of regression testing is to ensure that recent changes have not introduced new bugs or negatively impacted the existing system. It helps maintain the stability and reliability of the software by repeatedly testing important features whenever the code is modified.

Regression testing is important because even small updates can unintentionally break parts of an application that were previously working. By running regression tests, teams can catch and fix such issues early, saving time and effort in the long run. This testing can be done manually for smaller projects, but for larger applications with frequent changes, automation is often used to quickly re-run a large set of test cases. Effective regression testing ensures that the software continues to meet user expectations and business requirements after every update.



1. Performance Testing:

Performance Testing is a type of software testing that focuses on evaluating the speed, responsiveness, and stability of a software application under a particular workload. The goal of performance testing is to identify and fix performance bottlenecks before the system goes live. It checks how the system behaves under normal conditions, high load, and stress situations to ensure it can handle real-world usage.

Performance testing is important because slow or unstable applications can lead to poor user experience, loss of customers, and damage to a company's reputation. It helps measure system attributes like response time, throughput, resource usage, and scalability. Different types of performance testing include load testing, stress testing, spike testing, and endurance testing. By conducting thorough performance testing, developers and testers can ensure the application remains reliable, efficient, and responsive, even as the number of users or the size of data grows

