**TEST CASES**

A test case refers to the actions required to verify a specific feature or functionality in software testing. The test case details the steps, data, prerequisites, and postconditions necessary to verify a feature.

The Objective of Writing Test Cases in Software Testing

* To validate specific features and functions of the software.
* To guide testers through their day-to-day hands-on activity.
* To record a catalog of steps undertaken, which can be revisited in the event of a bug popping up.
* To provide a blueprint for future projects and testers so they don’t have to start work from scratch.
* To help detect usability issues and design gaps early on.
* To help new testers and devs quickly pick up testing, even if they join in the middle of an ongoing project.

Standard Test Case Format

* Test Case ID
* Test Scenario
* Test Steps
* Prerequisites
* Test Data
* Expected/Intended Results
* Actual Results
* Test Status – Pass/Fail

How to write Test Cases (Test Case Example)

Let’s build a test case example based on a specific scenario. Here is a sample case.

* **Test Case ID**: #BST001
* **Test Scenario**: To authenticate a successful user login on Gmail.com
* **Test Steps:**
  + The user navigates to Gmail.com.
  + The user enters a registered email address in the ’email’ field.
  + The user clicks the ‘Next’ button.
  + The user enters the registered password.
  + The user clicks ‘Sign In.’
* **Prerequisites**: A registered Gmail ID with a unique username and password.
* **Browser**: Chrome v 86. Device: Samsung Galaxy Tab S7.
* **Test Data:** Legitimate username and password.
* **Expected/Intended Results:** Once username and password are entered, the web page redirects to the user’s inbox, displaying and highlighting new emails at the top.
* **Actual Results:** As Expected
* **Test Status – Pass/Fail**: Pass

**Common Features of Test Cases**

* **Likely to be revised and updated regularly:** Software requirements can change depending on business priorities or customer preferences. If requirements change, test cases will have to be altered accordingly. The detection of bugs and debugging steps may also require test cases to be changed.
* **Likely to involve clustering:** Test cases in a single test scenario usually have to be run in a specific sequence or in a group. In this case, particular prerequisites of one test case will apply to other cases in the same sequence.
* **Likely to be interdependent:** Often, test cases can depend on each other. This is especially true for layered applications with multi-tier business logic.
* **Likely to be used by testers and developers:**Test cases are helpful for developers and testers. For example, when devs fix bugs, test cases can be pretty valuable to replicate the said bug. In Test-Driven Development (TDD), devs create test cases to craft business logic, cover multiple test scenarios, and start writing code.

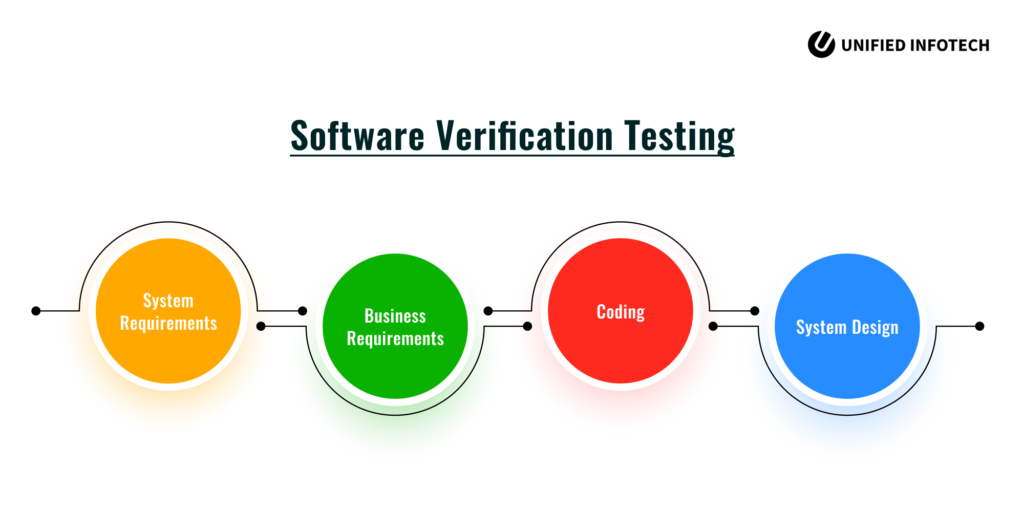
##### **When Should Test Cases Be Written?**

Different circumstances are used when writing test cases:

* **Before Development:** Before beginning the actual coding, test cases might be created to help establish the needs of the product or program. The tests could then be run once the product or software has been produced.
* **After Development:** When necessary to test the functionality of a particular feature, test cases are also produced immediately after generating a product or piece of software, or after developing the feature but before the product or piece of software is released.

**VERIFICATION AND VALIDATION**

**VERIFICATION**



It is the process of verifying whether a software product fulfills the objectives behind its creation and is devoid of bugs. Verification examines every predetermined software specification – the plan, code, design, and documents – to ensure the product meets them.

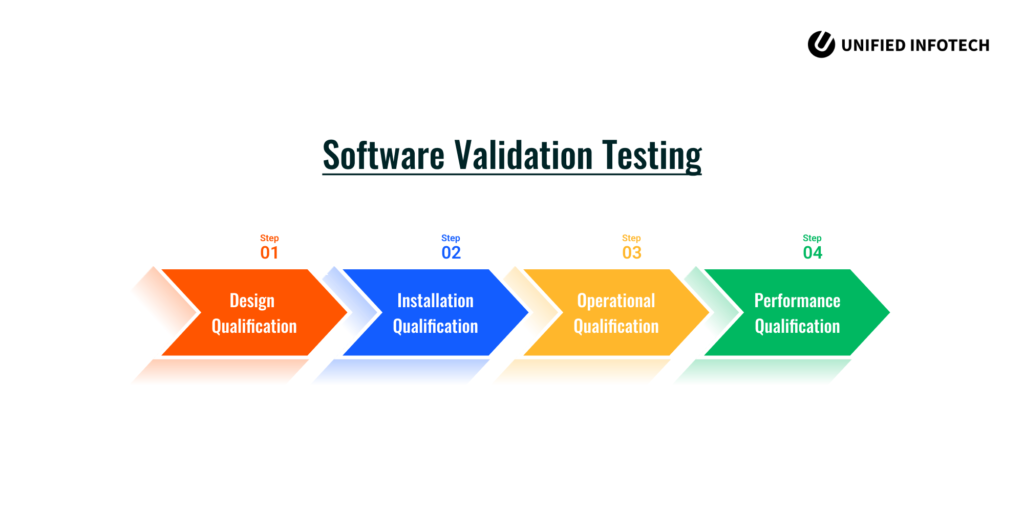
It includes vital activities like checking the system requirements, business requirements, code walkthrough, and design review. These aim to ensure quality in every aspect of the software, from architecture to design.

## **How is Verification Testing Performed?**

The following 4 methods are commonly used:

* Peer Reviews – It involves providing the software product to others and asking them to review it. They can share their views about the product’s quality and help identify flaws.
* Inspections – A particular team inspects the software product and helps find potential faults and critical sections.
* Walkthrough – It involves demonstrating the software to a group of people who are free to ask their doubts. This method can help find potential problems.
* Desk-Checking – It is the process of reviewing a software’s source code. The development team usually desk-checks a program to ensure the algorithms work fine

**VALIDATION**



It involves checking whether the final software product is up to the mark and meets business requirements. The validation activity comprises smoke testing, functional testing, integration testing, and other viable techniques.

This software testing procedure helps ensure that the final product fulfills the desired use under the right conditions and environment.

## **How is Validation Testing Performed?**

It involves the following 4 phases:

* Design Qualification – It involves designing a validation testing plan based on the client’s requirements.
* Installation Qualification – It involves following the validation testing plan and installing the software accordingly.
* Operational Qualification – It involves testing the software using unit testing, integration testing, and more.
* Performance Qualification – It involves verifying whether the software can work in the real world per the requirements.