**Test Cases**

A **TEST CASE** is a set of actions executed to verify a particular feature or functionality of your software application. A Test Case contains test steps, test data, precondition, postcondition developed for specific test scenarios to verify any requirement. The test case includes specific variables or conditions, using which a testing engineer can compare expected and actual results to determine whether a software product is functioning as per the requirements of the customer.

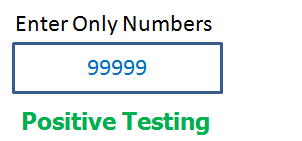
There are two types of test cases:

* Positive test cases
* Negative test cases

**Positive Testing**

**Positive Testing** many times referred to as “Happy path testing”, is a type of testing which is performed on a software application by providing the valid data sets as an input. It checks whether the software application behaves as expected with positive inputs or not. Positive testing is performed in order to check whether the software application does exactly what it is expected to do.

For example –There is a text box in an application which can accept only numbers. Entering values up to 99999 will be acceptable by the system and any other values apart from this should not be acceptable. To do positive testing, set the valid input values from 0 to 99999 and check whether the system is accepting the values.



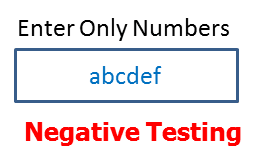
**Negative Testing**

**Negative Testing** is a testing method performed on the software application by providing invalid or improper data sets as input. It checks whether the software application behaves as expected with the negative or unwanted user inputs. The purpose of negative testing is to ensure that the software application does not crash and remains stable with invalid data inputs.

For example - Negative testing can be performed by entering characters A to Z or from a to z. Either software system should not accept the values or else it should throw an error message for these invalid data inputs.

In both the testing, the following needs to be considered:

* Input data
* An action which needs to be performed
* Output Result



Test Cases and Test Scenarios

#### What is a Test Case?

Test cases are the set of positive and negative executable steps which has a set of pre-conditions, test data, expected result, post-conditions and actual results.

Test Case answers “**How to be tested**”

Assume that we need to test the functionality of a login page of Gmail application. Test cases for the above login page functionality as follows:

**Test Case Examples:**

**Test Case 1:** Enter valid User Name and valid Password  
**Test Case 2:** Enter valid User Name and invalid Password  
**Test Case 3:** Enter invalid User Name and valid Password  
**Test Case 4:** Enter invalid User Name and invalid Password

#### What is a Test Scenario?

Test Scenario gives the idea of what we have to test. Test Scenario is like a high-level test case. High level Test Case.

Test Scenario answers “**What to be tested**”

Assume that we need to test the functionality of a login page of Gmail application. Test scenario for the Gmail login page functionality as follows:

**Test Scenario Example:** Verify the login functionality

| **Test Scenario Name** | **test Case Id** | **Test Case Name** | **Step #** | **Step Description** | **Test Data** | **Expected Result** | **Actual Result** | **Result-passed/failed** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Check Customer Login | ID-001 | Check Customer Login with valid data | 1 | Open the gmail web site; | [www.gmail.com](http://www.gmail.com)  www.guru99.com | Gmail web site is displayed |  |  |
|  |  | 2 | Enter a valid user name | serj12 |  |  |  |
|  |  | 3 | Enter a valid user password | 12345 |  |  |  |
|  |  | 4 | Click “Login” button |  | Account page is displayed and “Log out” btn could be clicked |  | passed  (or failed if expected and actual result does not match ) |

## Best Practice for writing good Test Case Example.

**1. Test Cases need to be simple and transparent:**

Create test cases that are as simple as possible. They must be clear and concise as the author of the test case may not execute them.

Use assertive language like go to the home page, enter data, click on this and so on. This makes understanding the test steps easy and tests execution faster.

**2. Create Test Case with End User in Mind**

The ultimate goal of any software project is to create test cases that meet customer requirements and is easy to use and operate. A tester must create test cases keeping in mind the end user perspective

**3. Avoid test case repetition.**

Do not repeat test cases. If a test case is needed for executing some other test case, call the test case by its test case id in the precondition column

**4. Do not Assume**

Do not assume functionality and features of your software application while preparing a test case. Stick to the Specification Documents.

**5. Ensure 100% Coverage**

Make sure you write test cases to check all software requirements mentioned in the specification document. Use[Traceability Matrix](https://www.guru99.com/traceability-matrix.html)to ensure no functions/conditions is left untested.

**6. Test Cases must be identifiable.**

Name the test case id such that they are identified easily while tracking defects or identifying a software requirement at a later stage.

**7. Implement Testing Techniques**

It's not possible to check every possible condition in your software application. Software Testing techniques help you select a few test cases with the maximum possibility of finding a defect.

8. **Self-cleaning**

The test case you create must return the[Test Environment](https://www.guru99.com/test-environment-software-testing.html)to the pre-test state and should not render the test environment unusable. This is especially true for configuration testing.

9. **Repeatable** **and self-standing**

The test case should generate the same results every time no matter who tests it

**10. Peer Review.**

After creating test cases, get them reviewed by your colleagues. Your peers can uncover defects in your test case design, which you may easily miss.

**Test Case Management Tools**

Test management tools are the automation tools that help to manage and maintain the Test Cases. Main Features of a test case management tool are

1. **For documenting Test Cases:**With tools, you can expedite Test Case creation with use of templates
2. **Execute the Test Case and Record the results:**Test Case can be executed through the tools and results obtained can be easily recorded.
3. **Automate the Defect Tracking:**Failed tests are automatically linked to the bug tracker, which in turn can be assigned to the developers and can be tracked by email notifications.
4. **Traceability:**Requirements, Test cases, Execution of Test cases are all interlinked through the tools, and each case can be traced to each other to check test coverage.
5. **Protecting Test Cases:** Test cases should be reusable and should be protected from being lost or corrupted due to poor version control. Test Case Management Tools offer features like

* Naming and numbering conventions
* Versioning
* Read-only storage
* Controlled access
* Off-site backup

Popular Test Management tools are: [HP ALM](https://www.guru99.com/hp-alm-free-tutorial.html)and [JIRA](https://www.guru99.com/jira-tutorial-a-complete-guide-for-beginners.html)