TESTER ASSIGNMENTS

They may be involved in or even be the primary people identifying test conditions and creating test designs, test cases, test procedure specifications and test data, and may automate or help to automate the tests.

Tester assignments typically refer to the process of assigning individuals or teams to test specific components, features, or aspects of a product or system. Testing is an essential part of software development, quality assurance, and product validation. Assigning testers to different tasks helps ensure thorough and efficient testing coverage. Here are some considerations for tester assignments:

Test plan and strategy: Begin by creating a test plan and strategy that outlines the testing objectives, scope, and approach. This will help identify the different types of tests required and the specific areas that need to be covered.

Test case creation: Testers can be assigned to create test cases based on the test plan and requirements. Test cases should cover different scenarios and use cases to ensure comprehensive testing.

Testing types: Identify the different types of testing needed, such as functional testing, performance testing, security testing, usability testing, etc. Assign testers with the relevant expertise and skills for each type of testing.

Test environment: Assign testers to set up and configure the test environment, including any necessary hardware, software, or network configurations. This ensures that the testing environment accurately reflects the production environment.

Test execution: Assign testers to execute the test cases and document the results. Testers should follow the test plan and report any issues or bugs they encounter during the testing process.

Bug reporting and tracking: Assign testers to report identified bugs or issues in a structured manner, including detailed descriptions, steps to reproduce, and any supporting documentation. Testers may also be responsible for tracking the status and resolution of reported issues.

Regression testing: After bug fixes or changes are made, assign testers to perform regression testing to ensure that the fixes or changes did not introduce new issues or break existing functionality.

Collaboration and communication: Assign testers to collaborate with developers, product managers, and other stakeholders to ensure clear communication and understanding of testing requirements, priorities, and progress.

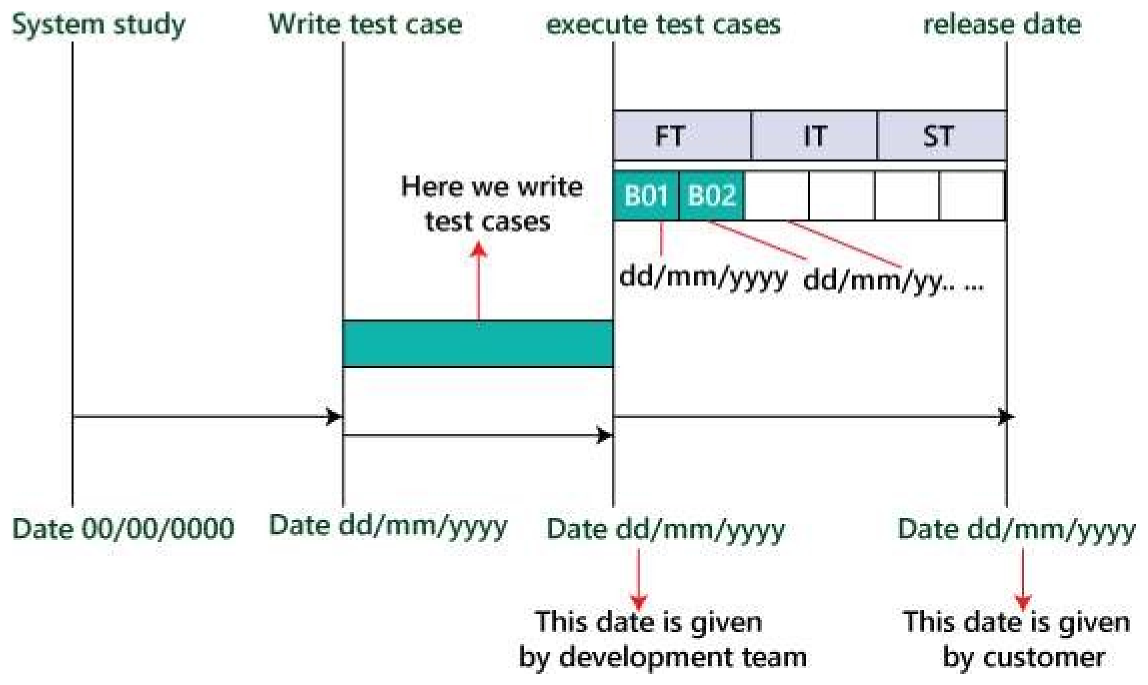
Test coverage analysis: Assign testers to analyze the test coverage and identify any gaps or areas that require additional testing. This helps ensure that all critical features and functionalities are adequately tested.

Test automation: Assign testers with automation skills to develop and maintain automated test scripts, which can help increase testing efficiency and coverage.

It's important to consider the skills, experience, and availability of testers when making assignments. Regular communication and coordination among testers and other team members are crucial to ensure effective testing and timely feedback.

TEST SCHEDULE

It is used to explain the timing to work, which needs to be done or this attribute covers when exactly each testing activity should start and end? And the exact data is also mentioned for every testing activity for the particular date.



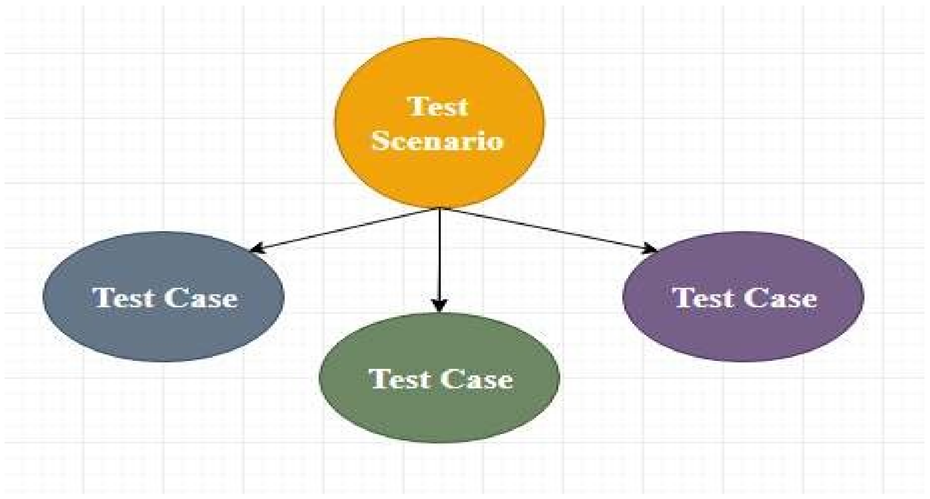
Therefore ,as we can see in the below image that for the particular activity, there will be a startingdate and ending date; for each testing to a specific build, there will be the specified date.

For example

- Writing test cases
- Execution process

TEST CASES

The test case is defined as a group of conditions under which a tester determines whether a software application is working as per the customer's requirements or not. Test case designing includes preconditions, case name, input conditions, and expected result. A test case is a first level action and derived from test scenarios.



It is an in-details document that contains all possible inputs (positive as well as negative) and the navigation steps, which are used for the test execution process. Writing of test cases is a one-time attempt that can be used in the future at the time of regression testing.

Test case gives detailed information about testing strategy, testing process, preconditions, and expected output. These are executed during the testing process to check whether the software application is performing the task for that it was developed or not.

Test case helps the tester in defect reporting by linking defect with test case ID.

Detailed test case documentation works as a full proof guard for the testing team because if developer missed something, then it can be caught during execution of these full-proof test cases

To write the test case, we must have the requirements to derive the inputs, and the test scenarios must be written so that we do not miss out on anyfeatures for testing. Then we should have the test case template to maintain the uniformity, or every test engineer follows the same approach to prepare the test document.

Generally, we will write the test case whenever the developer is busy in writing the code.

When do we write a test case?

We will write the test case when we get the following:

- When the customer gives the business needs then, the developer starts developing and says that they need 3.5 months to build this product.
- And In the meantime, the testing team will **start writing the test cases**.
- Once it is done, it will send it to the Test Lead for the review process.
- And when the developers finish developing the product, it is handed over to the testing team.
- The test engineers never look at the requirement while testing the product document because testing is constant and does not depends on the mood of the person rather than the quality of the test engineer.

Why we write the test cases?

We will write the test for the following reasons:

- **To require consistency in the test case execution**
- **To make sure a better test coverage**

- **It depends on the process rather than on a person**
- **To avoid training for every new test engineer on the product**

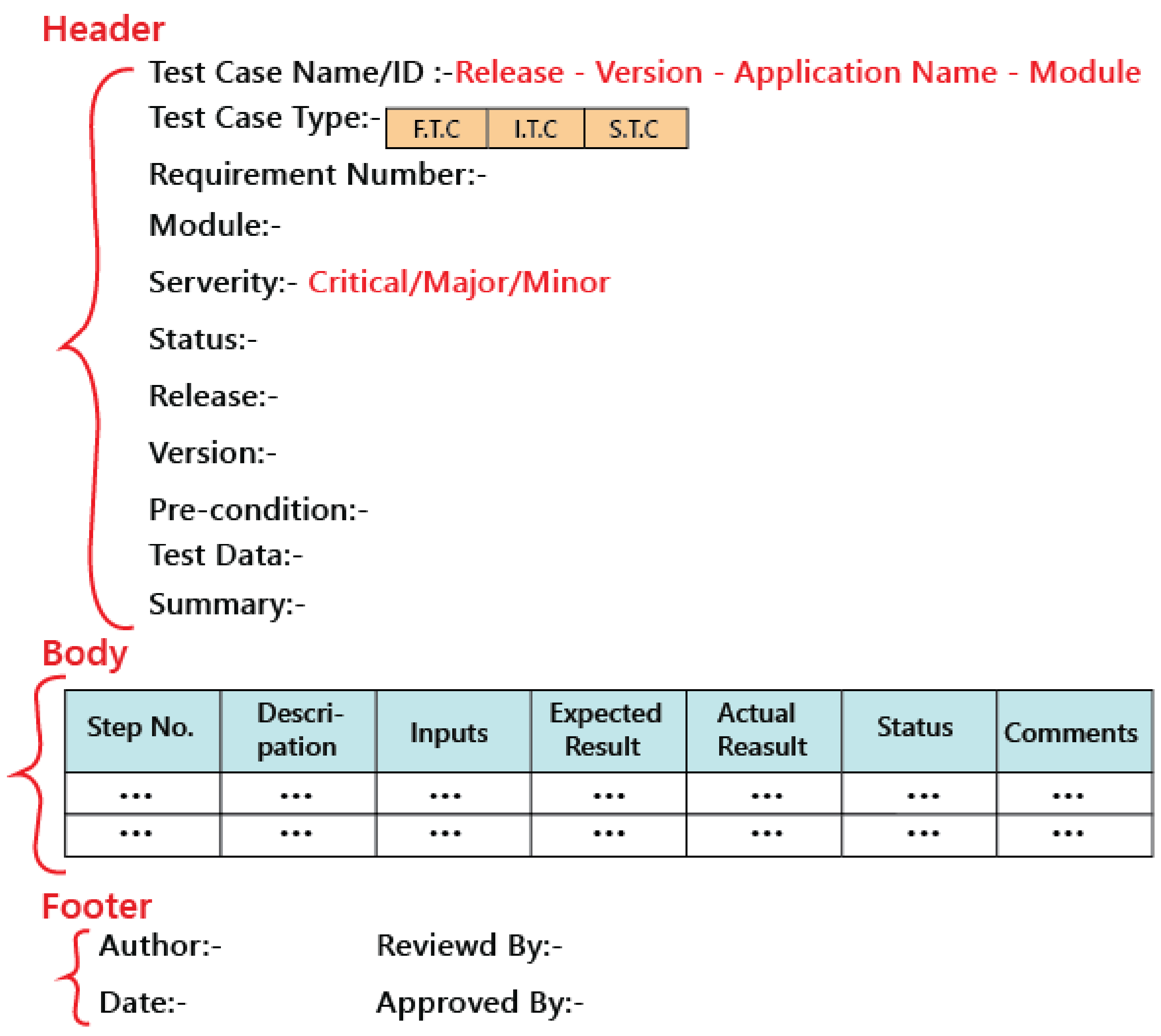
To require consistency in the test case execution: we will see the test case and start testing the application.

To make sure a better test coverage: for this, we should cover all possible scenarios and document it, so that we need not remember all the scenarios again and again.

It depends on the process rather than on a person: A test engineer has tested an application during the first release, second release, and left the company at the time of third release. As the test engineer understood a module and tested the application thoroughly by deriving many values. If the person is not there for the third release, it becomes difficult for the new person. Hence all the derived values are documented so that it can be used in the future.

To avoid giving training for every new test engineer on the product: When the test engineer leaves, he/she leaves with a lot of knowledge and scenarios. Those scenarios should be documented so that the new test engineer can test with the given scenarios and also can write the new scenarios.

Test case template



Header

Test Case Name/ID :-Release - Version - Application Name - Module

Test Case Type:- F.T.C | I.T.C | S.T.C

Requirement Number:-

Module:-

Serverity:- Critical/Major/Minor

Status:-

Release:-

Version:-

Pre-condition:-

Test Data:-

Summary:-

Body

Step No.	Descri-pation	Inputs	Expected Result	Actual Reasult	Status	Comments
...	...	...	...	...	...	...
...	...	...	...	...	...	...

Footer

Author:- Reviewd By:-

Date:- Approved By:-

The primary purpose of writing a test case is to achieve the efficiency of the application.

As we know, the **actual result** is written after the test case execution, and most of the time, it would be same as the **expected result**. But if the test step will fail, it will be different. So, the actual result field can be skipped, and in **the Comments** section, we can write about the bugs.

And also, the **Input field** can be removed, and this information can be added to the **Description field**.

The above template we discuss above is not the standard one because it can be different for each company and also with each application, which is based on the test engineer and the test lead. But, for testing one application, all the test engineers should follow a usual template, which is formulated.

The test case should be written in simple language so that a new test engineer can also understand and execute the same.

In the above sample template, the header contains the following:

Step number

It is also essential because if step number 20 is failing, we can document the bug report and hence prioritize working and also decide if it's a critical bug.

Test case type

It can be functional, integration or system test cases or positive or negative or positive and negative test cases.

Release

One release can contain many versions of the release.

Pre-condition

These are the necessary conditions that need to be satisfied by every test engineer before starting the test execution process. Or it is the data configuration or the data setup that needs to be created for the testing.

For example: In an application, we are writing test cases to add users, edit users, and delete users. The per-condition will be seen if user A is added before editing it and removing it.

Test data

These are the values or the input we need to create as per the per-condition.

For example, Username, Password, and account number of the users.

The test lead may be given the test data like username or password to test the application, or the test engineer may themselves generate the username and password.

Severity

The severity can be **major, minor, and critical**, the severity in the test case talks about the importance of that particular test cases. All the text execution process always depends on the severity of the test cases.

We can choose the severity based on the module. There are many features include in a module, even if one element is critical, we claim that test case to be critical. It depends on the functions for which we are writing the test case.

For example, we will take the Gmail application and let us see the severity based on the modules:

Modules	**Severity**
Login	Critical
Help	Minor
Compose mail	Critical
Setting	Minor
Inbox	Critical
Sent items	Major
Logout	Critical

And for the banking application, the severity could be as follows:

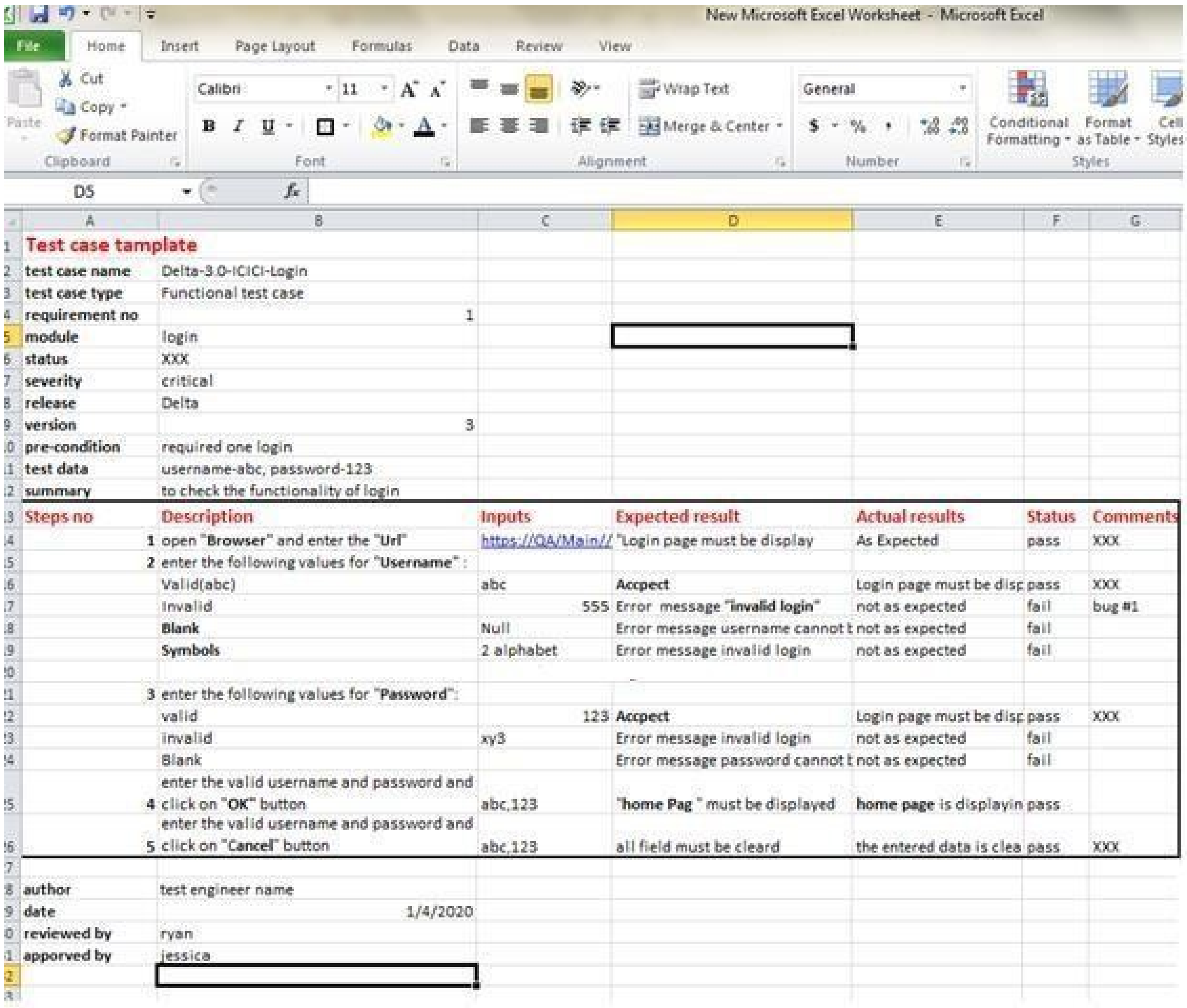
Modules	Severity
Amount transfer	critical
Feedback	minor

Brief description

The test engineer has written a test case for a particular feature. If he/she comes and reads the test cases for the moment, he/she will not know for what feature has written it. So, the brief description will help them in which feature test case is written.

Example of a test case template

Here, we are writing a test case for the **ICICI application's Login** module:



Test case template						
test case name	Delta-3.0-ICICI-Login					
test case type	Functional test case					
requirement no	1					
module	login					
status	XXX					
severity	critical					
release	Delta					
version	3					
pre-condition	required one login					
test data	username-abc, password-123					
summary	to check the functionality of login					
Steps no	Description	Inputs	Expected result	Actual results	Status	Comments
1	open "**Browser**" and enter the "**Url**"	https://QA/Main//	"Login page must be display	As Expected	pass	XXX
2	enter the following values for "**Username**" :					
	Valid(abc)	abc	**Accpect**	Login page must be disp	pass	XXX
	Invalid	555	Error message "**invalid login**"	not as expected	fail	bug #1
	Blank	Null	Error message username cannot b	not as expected	fail	
	Symbols	2 alphabet	Error message invalid login	not as expected	fail	
3	enter the following values for "**Password**":					
	valid	123	**Accpect**	Login page must be disp	pass	XXX
	invalid	xy3	Error message invalid login	not as expected	fail	
	Blank		Error message password cannot b	not as expected	fail	
4	enter the valid username and password and click on "**OK**" button	abc,123	"**home Pag** " must be displayed	**home page** is displayin	pass	
5	enter the valid username and password and click on "**Cancel**" button	abc,123	all field must be cleard	the entered data is clea	pass	XXX
author	test engineer name					
date	1/4/2020					
reviewed by	ryan					
apporved by	jessica					

TYPES OF TEST CASES

We have a different kind of test cases, which are as follows:

- **Function test cases**
- **Integration test cases**
- **System test cases**

The functional test cases

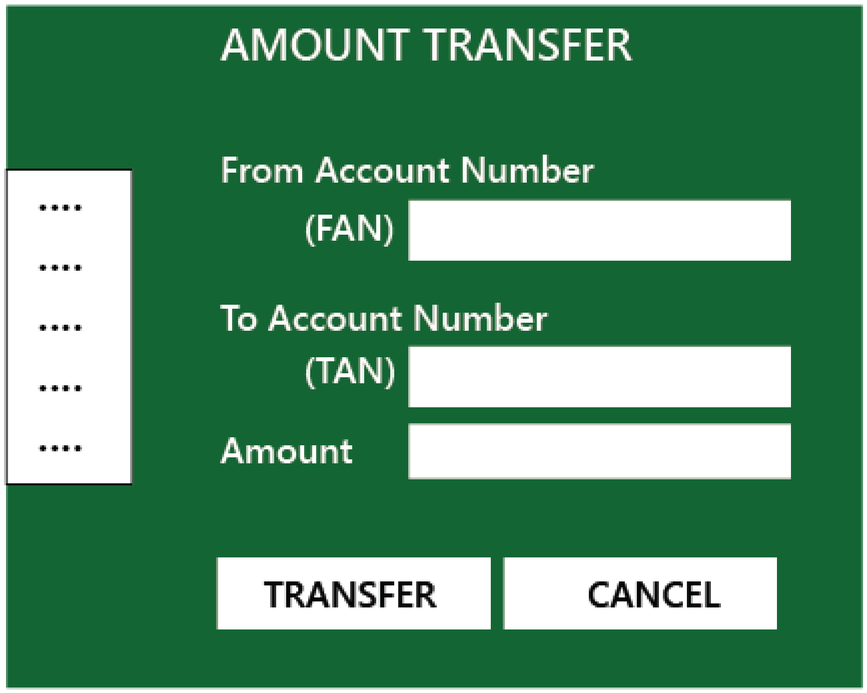
Firstly, we check for which field we will write test cases and then describe accordingly.

In functional testing or if the application is data-driven, we require the input column else; it is a bit time-consuming.

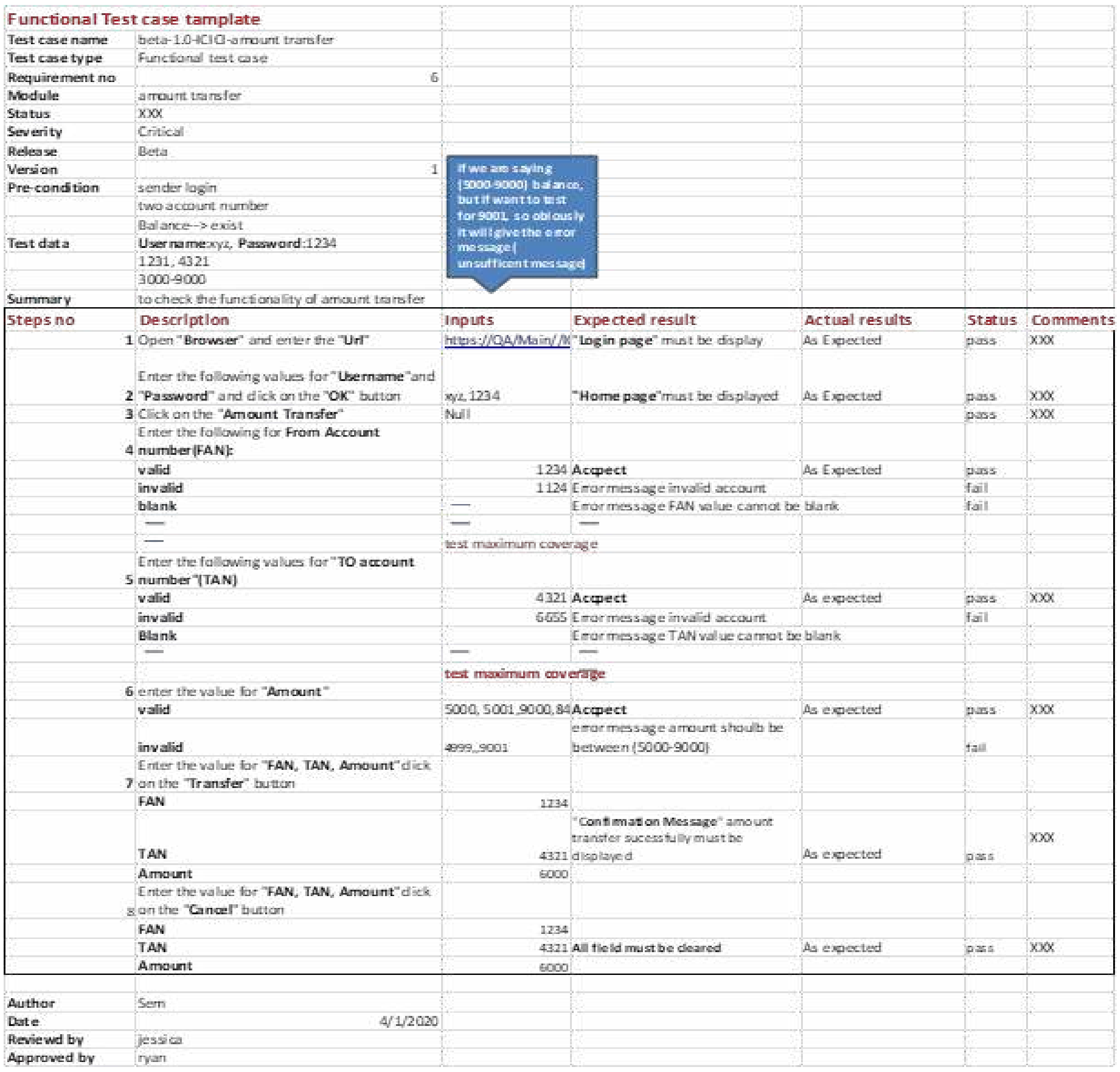
Rules to write functional test cases:

- In the expected results column, try to use **should be** or **must be**.
- Highlight the Object names.
- We have to describe only those steps which we required the most; otherwise, we do not need to define all the steps.
- To reduce the excess execution time, we will write steps correctly.
- Write a generic test case; do not try to hard code it.

Let say it is the amount transfer module, so we are writing the functional test cases for it and then also specifies that it is not a login feature.



The functional test case for amount transfer module is in the below Excel file:



Functional Test case tamplate						
Test case name	beta-1.0-ICICI-amount transfer					
Test case type	Functional test case					
Requirement no	6					
Module	amount transfer					
Status	XXX					
Severity	Critical					
Release	Beta					
Version	1	If we are saying (3000-9000) balance, but if want to test for 9001, so oblously it will give the error message(unsufficent message)				
Pre-condition	sender login					
	two account number					
	Balance--> exist					
Test data	**Username**:xyz, **Password**:1234					
	1231, 4321					
	3000-9000					
Summary	to check the functionality of amount transfer					
Steps no	**Description**	**Inputs**	**Expected result**	**Actual results**	**Status**	**Comments**
1	Open "**Browser**" and enter the "**Url**"	https://QA/Main/ /K	"**Login page**" must be display	As Expected	pass	XXX
2	Enter the following values for "**Username**" and "**Password**" and click on the "**OK**" button	xyz, 1234	"**Home page**" must be displayed	As Expected	pass	XXX
3	Click on the "**Amount Transfer**"	Null			pass	XXX
4	Enter the following for **From Account number(FAN)**:					
	valid	1234	**Acqect**	As Expected	pass	
	invalid	1124	Error message invalid account		fail	
	blank	—	Error message FAN value cannot be blank		fail	
	—	—	—			
	—	test maximum coverage				
5	Enter the following values for "**TO account number**"(**TAN**)					
	valid	4321	**Acqect**	As expected	pass	XXX
	invalid	6655	Error message invalid account		fail	
	Blank		Error message TAN value cannot be blank			
	—	—	—			
		test maximum coverage				
6	enter the value for "**Amount**"					
	valid	5000, 5001,9000,84	**Acqect**	As expected	pass	XXX
	invalid	4999,,9001	error message amount shoulb be between (5000-9000)		fail	
7	Enter the value for "**FAN, TAN, Amount**" click on the "**Transfer**" button					
	FAN	1234				
	TAN	4321	"**Confirmation Message**" amount transfer sucessfully must be displayed	As expected	pass	XXX
	Amount	6000				
8	Enter the value for "**FAN, TAN, Amount**" click on the "**Cancel**" button					
	FAN	1234				
	TAN	4321	**All field must be cleared**	As expected	pass	XXX
	Amount	6000				
Author	Sem					
Date	4/1/2020					
Reviewd by	jessica					
Approved by	ryan					

INTEGRATION TEST CASE

In this, we should not write something which we already covered in the functional test cases, and something we have written in the integration test case should not be written in the system test case again.

Rules to write integration test cases

- Firstly, understand the product
- Identify the possible scenarios
- Write the test case based on the priority

When the test engineer writing the test cases, they may need to consider the following aspects:

If the test cases are in details:

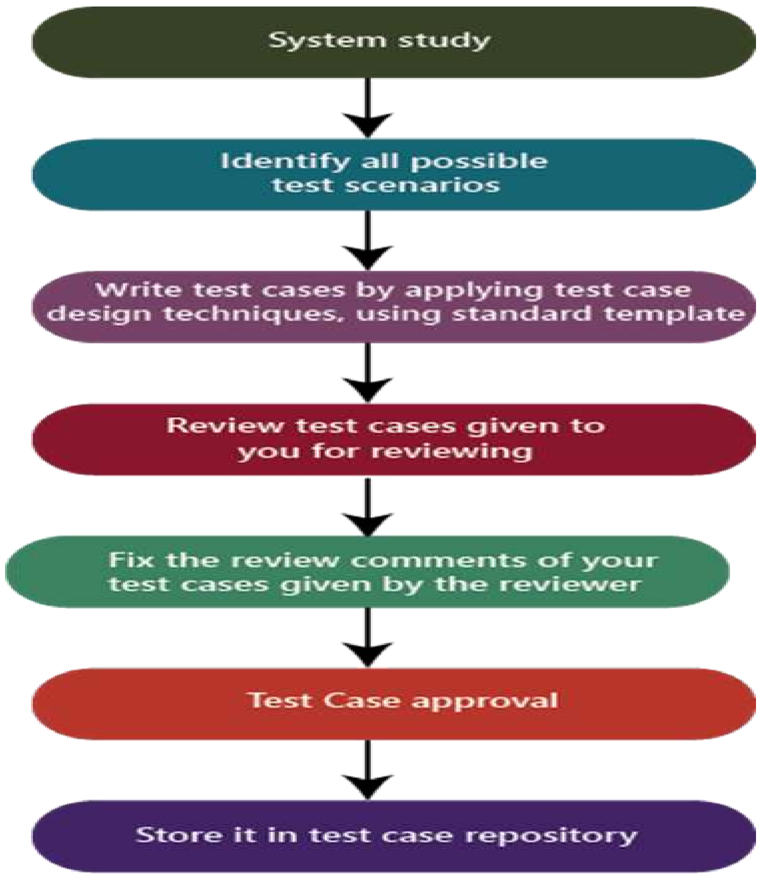
- They will try to achieve maximum test coverage.
- All test case values or scenarios are correctly described.
- They will try to think about the execution point of view.
- The template which is used to write the test case must be unique.

SYSTEM TEST CASES

We will write the system test cases for the end-to-end business flows. And we have the entire modules ready to write the system test cases.

The process to write test cases

The method of writing a test case can be completed into the following steps, which are as below:


System study
Identify all possible test scenarios
Write test cases by applying test case design techniques, using standard template
Review test cases given to you for reviewing
Fix the review comments of your test cases given by the reviewer
Test Case approval
Store it in test case repository

System study

In this, we will understand the application by looking at the requirements or the SRS, which is given by the customer.

Identify all scenarios:

- When the product is launched, what are the possible ways the end-user may use the software to identify all the possible ways.
- I have documented all possible scenarios in a document, which is called test design/high-level design.
- The test design is a record having all the possible scenarios.

Write test cases

Convert all the identified scenarios to test claims and group the scenarios related to their features, prioritize the module, and write test cases by applying test case design techniques and use the standard test case template, which means that the one which is decided for the project.

Review the test cases

Review the test case by giving it to the head of the team and, after that, fix the review feedback given by the reviewer.

Test case approval

After fixing the test case based on the feedback, send it again for the approval.

Store in the test case repository

After the approval of the particular test case, store in the familiar place that is known as the test case repository.