## **Purpose of this document**

Main purpose of this doc is to create a best practice of creating Test Case (Steps) Style Guide. Investing time and effort in the creation and maintenance of good test cases makes testing more effective and ensures the success of the testing plan. Many software companies and organizations using a different style and techniques to create a test case, and specially how to create and what need to be presented in execution steps, links and descriptions..etc

Test cases are sets of conditions or variables used by software testers to determine if a system works correctly as expected. Test cases help in validating the test coverage of a software application. A test case contains essential fields that provide information about the test case, the activity involved in execution, and the results expected. These fields include a unique name, any preconditions involved, detailed steps, input conditions, and expected results for a particular app function.

## **Essentials of a Good Test Case**

* It is easy to understand and execute ( for first time or new tester as well)
* It is accurate with a specific objective ( function)
* It is easy to trace as per requirements ( and key words)
* It is repeatable and can be used to test again and again ( for different versions or testing environments)
* It saves time and money by avoiding unnecessary steps ( necessary info, links or screenshots will be provided)
* It has clear short description with point to user story or functions will be tested ( include project name and special conditions if required)
* it will show preconditioning to prepare environment or tool for execution of test case
* it will provide easy to read and follow execution test steps
* it will provide an expected result with required content , screenshots etc.
* it will provide additional necessary information may need to execute / finish test case ( links of relative test cases, attachments, design docs or demo etc

## **Best Practices for Effective Test Case Writing**

Good test cases are easy to execute and maintain. They make the testing process more effective by saving time and effort.

### **1. Keep It Simple and Easy to Understand**

A good test case that is well-written and simple is easy for testers to understand and execute. Organize test cases according to specific categories or related areas of the application. Test cases can be grouped based on their user story or modules like browser specific behaviors, etc. This makes it easier to review and maintain the test document. Information given in the test cases should be clear to other testers, developers, and other stakeholders involved in the project.

### **2. Include End User Perspective**

Take into account the end user perspective before drafting a test case. Think like an end user who is the key stakeholder for whom the app is intended. Understand the requirement, the functionality aspects to be covered, and the end user perspective. This will help in identifying test scenarios that arise in real life conditions, and also in coming up with test cases that are most critical to business.

### **3. Use Correct Naming Conventions**

Name test cases in a way that makes it easy for stakeholders to identify it and understand its objective. You can name the test cases according to the module or the functional area that is included for testing.

### **4. Provide Test Case Description**

A proper test case description will allow users to understand what is being tested and how. Provide relevant details like the test environment and any other specific information. Mention the testing data to be used and the testing tools, if any, to be employed for executing the tests.

### **5. Include Assumptions and Preconditions**

Include all the assumptions and conditions that are applicable to the test case. Provide details of the test environment, any special set up for execution of the test case, etc.

### **6. Give the Steps Involved**

Include the actual steps involved in the execution of the test cases. Do not miss out on any step. Ensure that all the test case verification steps are covered. Include relevant documents or screenshots that can help in the execution of the steps given in the test design.

### **7. Give Details of the Test Data**

Provide the details of the test data for execution of the test case especially in cases where the same data can be reused. This helps in saving time for the creation of the test data for each cycle to be run. Specify the value range, if any, for the respective fields.

### **8. Make It Reusable and Modular**

As far as possible, ensure that there is no dependency or conflict among test cases. In case there are test cases that are inter-dependent or batched, mention them clearly in the test document and create a link / relationship.

### **9. Assign Testing Priority**

Assign testing priority to each test case based on the feature or component involved. This will ensure that during execution, high priority test case are executed first.

### **10. Provide the Expected Result and Post Conditions**

Include the expected result for every step of the test case. You can also include screenshots and relevant documents for reference. Mention the post-conditions or things to be verified after the execution of the test case.

# 

# **Maintenance of Tests Cases**

Change is constant for any software development project. As apps evolve, new upgrades and changes are introduced in the system. Test cases must be maintained and updated to reflect these new changes that are made.

### **1. Review Test Cases After Documentations**

Get the test cases reviewed by relevant stakeholders to ensure that it is easy to understand and does not include any unnecessary steps. A periodic review and revision of test cases will help in updating changes if any that are applicable.

### **2. Update Existing Test Cases Before Creating New Ones**

Update the existing test cases as and when new changes are introduced in the app. Include updates in order to accommodate changes to any existing user story or functionality.

### **3. Remove (Archive) Test Cases That Are Obsolete and Unnecessary**

Remove (Archive) test cases that are no longer valid for existing app features. Exclude duplicate test cases as well from the test suite.

### **4. Add Relevant Test Cases as Per Requirements**

Create and add test cases that are required as per new upgrades or features that are introduced in the software. Add any additional test case that is relevant, including test cases for defects that were found by end users, post-production(adding additional case from customer support requires to be a part of main regression test suite)

Test case creation can be made effective and simple with good knowledge of the technology used, the end user perspective, and the objective of the software application. Investing time and effort in the creation and maintenance of good test cases can make the entire testing process more effective and help ensure the success of the testing plan.

# **Template of Standard Test Cases**

### **Title:**

(Casablanka\_V.2.3.4) Login screen properly appeared under vertical and horizontal mode with iPad4 (Retina Screen)

### **Pre-conditions:**

1. Go to App Store ( apple application store, and search for Support App) when click on product icon and install latest production version of Support App

2. Click to start on app icon after installation complete

3. Application should start

4. Login /Register link visible and active (clicable)

**Steps To Reproduce:**

1. Start application

2. Verify first start screen appeared and all tabs and links are visible

3. Click on Login Link to verify it brings login screen

4. Verify login screen appeared and aligned to the center of the screen ( on horizontal mode)

5. Turn tablet to vertical position and verify app react and changed position for vertical view

6. Turn back to vertical mode and verify app return to vertical mode with required alignment

7. Repeat steps 5&6 few times to make sure app reacts properly and no visible issues found.

**Expected Result:**

1. Application appeared as required

2. Application react to change a view mode from horizontal to vertical and back with no issues

3. All required elements and menu are appeared as designed.

**Other (attachments, specific conditions etc):**

1. Verification must be done on iPad 4 with retina screen

2. Screen resolutions must be set to support retina.

**Meta Data:**

1. Key Words

2. Estimated time to complete

3. Location ( if required )

4. TBD