**Testing Guidelines Document**

**Revision History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Revision | Date | Createde By | Approved By | Description |
| 1.a | 19.06.2012 | Anju |  | Testing\_Guidelines\_v\_1a |
| 1.b | 02.01.2013 | Niranjana Mishra |  | Testing\_Guidelines\_v\_1b |
| 1.c | 02.01.2017 | Sudhanshu Saxena |  | Testing\_Guidelines\_v\_1c |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Contents

[1. Introduction- 4](#_Toc376443246)

[2. Add-ons To Be Used- 4](#_Toc376443247)

[2.1. Firebug: 4](#_Toc376443248)

[2.2. YSlow: 4](#_Toc376443249)

[2.3. Wink: 4](#_Toc376443250)

[2.4. Selenium: 4](#_Toc376443251)

[2.5. PageSpeed: 5](#_Toc376443252)

[2.6. Spell Checker: 5](#_Toc376443253)

[2.7. GTmetrix: 5](#_Toc376443254)

[3. Project Management Tools To Be Used: 5](#_Toc376443255)

[3.1. GITHUB: 5](#_Toc376443256)

[4. Test Case Creation: 7](#_Toc376443259)

[4.1. TEST STEP – 8](#_Toc376443260)

[4.2. TEST CASE – 8](#_Toc376443261)

[4.3. TEST SCENARIO – 8](#_Toc376443262)

[5. These are the Steps Test Engineer Need to Follow in Creating Test Case: 9](#_Toc376443263)

[5.1. Understand the product- 9](#_Toc376443264)

[5.2. Understand the customer issue-. 9](#_Toc376443265)

[5.3. Cases of functionality TESTING-. 9](#_Toc376443266)

[5.4. Check previously automated high quality TCMs- 9](#_Toc376443267)

[5.6. Creating A Test Case- 9](#_Toc376443268)

[5.6.1. Excel Sheet- 9](#_Toc376443269)

[5.6.2. Microsoft Test Manager- 9](#_Toc376443270)

[6. Step by step creation of Test case: 11](#_Toc376443271)

[6.1. Requirement Document- . 11](#_Toc376443272)

[6.2. Summary- 11](#_Toc376443273)

[6.3. preconditions- 11](#_Toc376443274)

[6.4. Test steps- 11](#_Toc376443275)

[6.4.1 When creating test case 11](#_Toc376443276)

[6.4.2. Always include Login information to application. 11](#_Toc376443277)

[6.4.3. Make sure the steps are precise and correct. 11](#_Toc376443278)

[6.4.4. Always number the test case steps (never leave the numbers off). 11](#_Toc376443279)

[6.4.5. Avoid ambiguous and long steps. 11](#_Toc376443280)

[6.4.6. Have the number 11](#_Toc376443281)

[6.4.7. Each test 11](#_Toc376443282)

[6.5. Validation Steps- 11](#_Toc376443283)

[6.6. Test suite- 12](#_Toc376443284)

[6.7. WINK RECORDING: 12](#_Toc376443285)

[7. Send Test Case for Review: 12](#_Toc376443286)

[8. Scenario Generator Best Practices: 12](#_Toc376443287)

[8.1. Best Practices: For a Test case to be considered of great quality it should meet the following basic criteria: 12](#_Toc376443288)

[8.2. Writing Guidelines for Creating Test Case: 13](#_Toc376443289)

# Introduction-

The document is about how Quality Assurance team works in Teramatrix organization. Document describes the different processes and tools used by the QA group to test different projects of the organization.

# Add-ons To Be Used-

|  |
| --- |
| http://www.sqatester.com/images/pix.gif |

* 1. Firebug:Firebug is a web development tool that facilitates the debugging, editing, and monitoring of any website's CSS, HTML, DOM, XHR, and JavaScript. Go through the below link for more information.

<https://addons.mozilla.org/en-US/firefox/addon/firebug/>

* 1. YSlow: YSlow analyses web pages and suggests ways to improve their performance based on a set of rules for high performance web pages. Go through the below link for more information.

<https://addons.mozilla.org/en-US/firefox/addon/yslow/>

* 1. Wink:Wink is a freeware screen capture and tutorial-creation program. It supports many features such as input-based capture and text annotations. Navigation buttons can be added to any presentation and the bitmaps for all presentation controls can be edited. Go through the below link for more information.

<http://freekje.home.xs4all.nl/Wink/index.html>

* 1. Selenium: Selenium is an open source automation testing tool for web based applications. It runs directly on browser and supports almost all available browsers such as Firefox, chrome, IE, Opera, Safari etc. It runs on all platforms such as Windows, Linux and Macintosh. It’s a very useful tool for System functional testing and browser compatibility testing. It is really strong as compare to other available automation tools and is very flexible and simple to use. Go through the below link for more information.

<http://seleniumhq.org/projects/ide/>

<http://knorrium.info/2010/05/19/a-java-approach-to-selenium/>

* 1. PageSpeed: Page Speed speeds up your site and reduces page load time. This open-source webserver module automatically applies [web performance best practices](https://developers.google.com/speed/docs/best-practices/rules_intro) to pages and associated assets (CSS, JavaScript, images) without requiring that you modify your existing content or workflow. Go through the below link for more information.

<https://developers.google.com/speed/pagespeed/module>

* 1. Spell Checker:It is an [application program](http://en.wikipedia.org/wiki/Application_software) that flags words in a document that may not be spelled correctly. Spell checkers may be stand-alone, capable of operating on a block of text, or as part of a larger application, such as a [word processor](http://en.wikipedia.org/wiki/Word_processor), [email client](http://en.wikipedia.org/wiki/Email_client), electronic [dictionary](http://en.wikipedia.org/wiki/Dictionary), or [search engine](http://en.wikipedia.org/wiki/Search_engine). Go through the below link for more information.

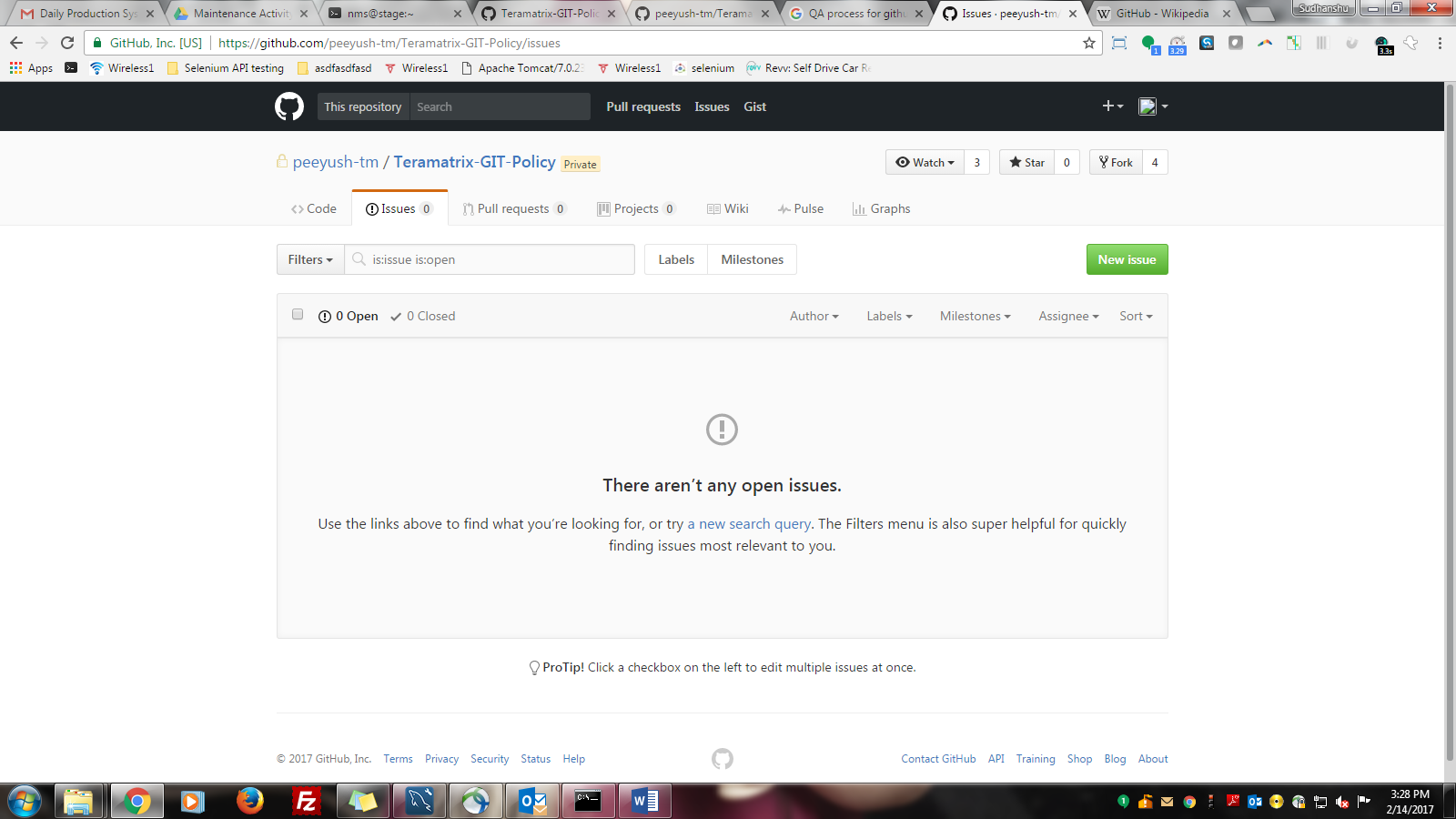
<http://en.wikipedia.org/wiki/Spell_checker>

* 1. GTmetrix: GTmetrix is a free tool that analyzes page speed performance. Using Page Speed and YSlow, GTmetrix generates scores for your pages and offers actionable recommendations on how to fix them.

# Project Management Tools To Be Used:

* 1. Github:GitHub is a [opensource](http://en.wikipedia.org/wiki/Microsoft) web application which provides for [source code management](http://en.wikipedia.org/wiki/Revision_control) (either via Team Foundation Version Control or [Git](http://en.wikipedia.org/wiki/Git_(software))), reporting, requirements management, project management (for both [agile software development](http://en.wikipedia.org/wiki/Agile_software_development) and [waterfall teams](http://en.wikipedia.org/wiki/Waterfall_model)), automated builds, lab management, testing and release management capabilities. It covers the entire end-to-end software development process. Go through the below link for more details.

https://en.wikipedia.org/wiki/GitHub



As a QA team member, we are using TFS to create bugs as per the appropriate related field and priority. Below is the screenshot of the new bug creation page and their required fields –

# Test Case Creation:

A test case has components that describe an input, action or event and an expected response, to determine if a feature of an application is working correctly.” Test case is a description, a recipe for tester, dealing with how to test a given functionality and how to check whether this works correctly. There is relationship between different terminologies used in test creation process-

* 1. TEST STEP – Specifies an action to perform, and the expected response of the test application. For example: Action- Type the password in the password box, Expected result: Your password should be dotted / hidden.
  2. TEST CASE – A list of test steps. Also defines the environmental situation and may link to related bugs, requirements etc.
  3. TEST SCENARIO – Usually comes directly from business requirements or user story. Management tools often ignore test scenario for linkage with a list of the requirements. Scenario contains a list of test cases and often their sequence.

# 5. These are the Steps Test Engineer Need to Follow in Creating Test Case:

## 5.1. Understand the product- In order to create high quality test case one must familiarize with the product that is being tested and have some knowledge of how the product operates.

## 5.2. Understand the customer issue- Test cases can be created by anyone provided they understand the product and the customer issue. A test engineer must understand the customer issue before attempting to create a test case. If you understand the customer issue then test case creation becomes an easy process.

## 5.3. Cases of functionality TESTING- Test engineer must understand the functionality of a particular module in order to create a test case.

## 5.4. Check previously automated high quality TCMs- It is always advisable to look at other previously created test case that are of high quality so as to emulate the same in creation of new test case. This helps the test engineer to have an idea of what is expected in the test case they create.

## 5.6. CREATING A Test Case- Test cases are managed in two manners in the organization:

* + 1. Excel Sheet- For manual use, we create test cases in the excel sheet. Please follow link-

Url: to get the format of excel sheet.

* + 1. Microsoft Test Manager- TM is a tool provided by Visual Studio 2012. TM provides a platform for managing test cases of different projects and of different area belongings in a well-managed manner. Hierarchy used in TM is:

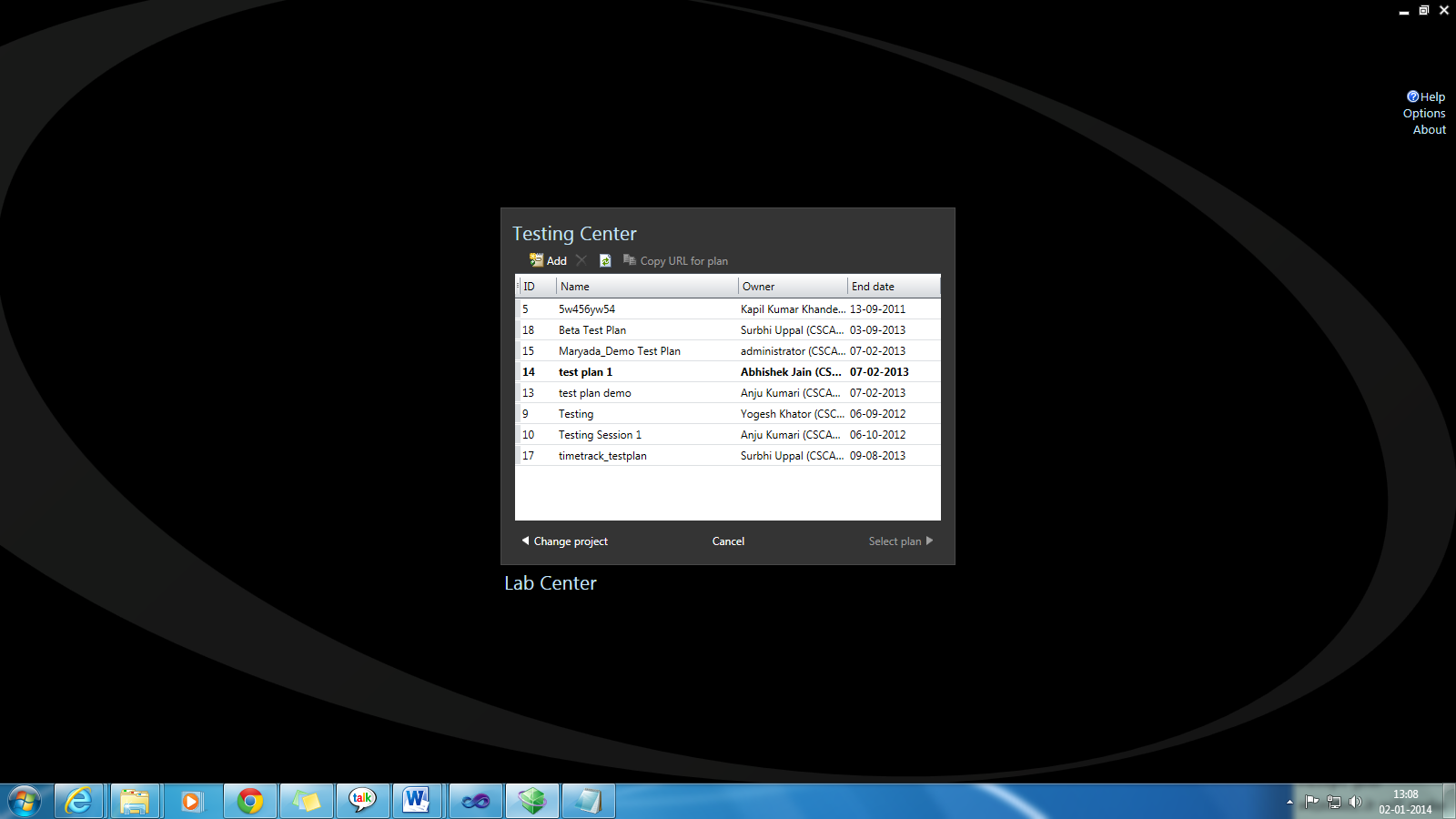
Team Projects

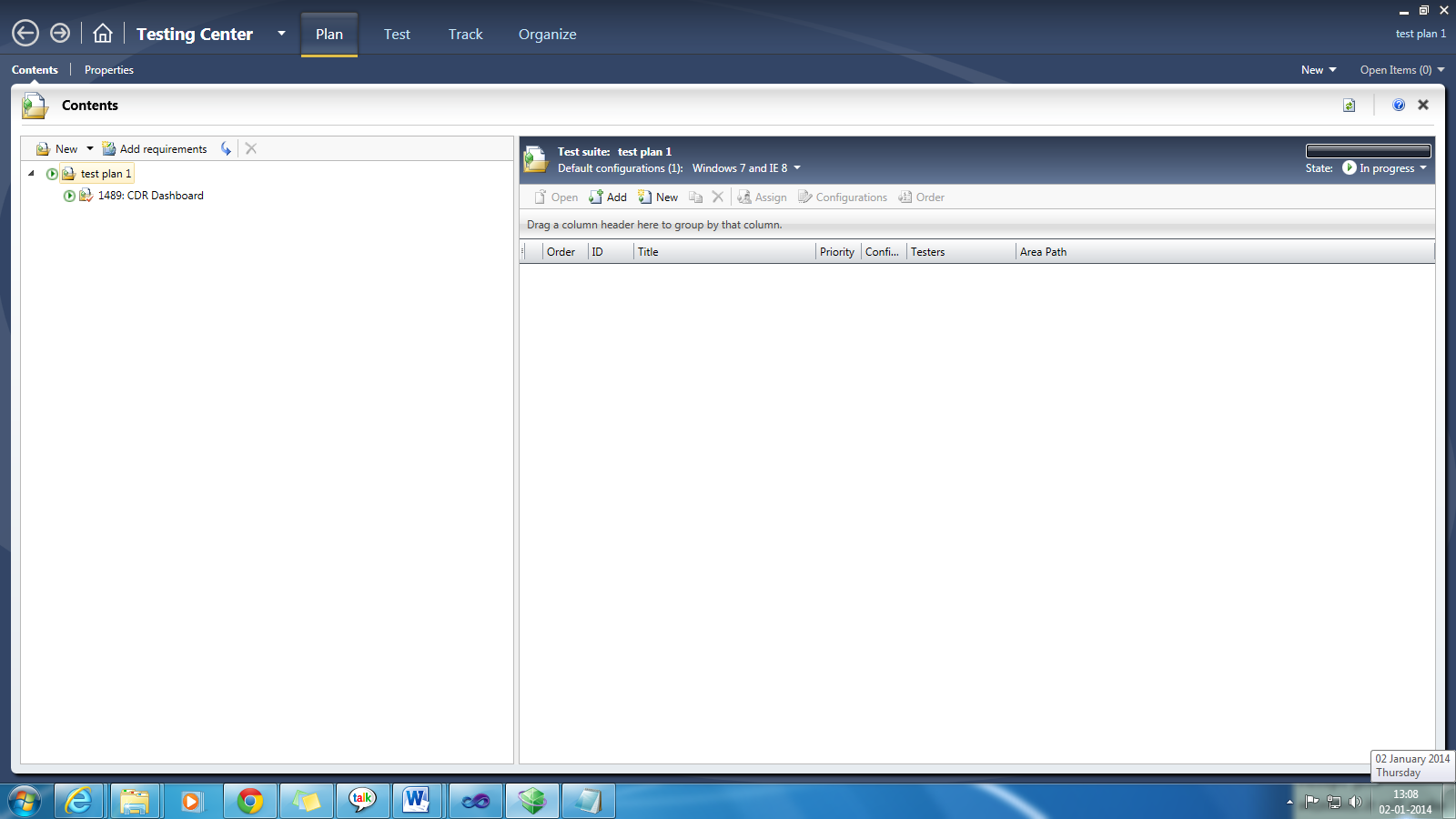
Master plans.

Suits.

Sub-suits.

Here are the screenshots of Master plans and different suits-





# Step by step creation of Test case:

## Requirement Document- Test Engineer should thoroughly go through the requirement of the project/module for which they want to create test cases. Test cases are always created on the basis of the requirement document.

## Summary- In the summary field you need to write the title of the Test case.

## preconditions- These are the necessary operations that need to be performed to execute the test. Try to have a common setup of pre-conditions and clean up actions for a bunch of test-cases.

## Test steps-

### When creating test cases, make sure you create solid steps so that the person running the test case will fully understand how to run the test case

### Always include Login information to application.

### Make sure the steps are precise and correct.

### Always number the test case steps (never leave the numbers off).

### Avoid ambiguous and long steps.

### Have the number of test steps matching the validation steps. If multiple steps are needed for one validation, write them down under a single step.

### Each test step should be easy to execute by anyone without requiring any product business knowledge.

### Validation Steps- Each test step must have its own corresponding validation step. These validation steps must also be precise. For validation step used to show fixed bugs, the validation step must verify the bug has been fixed.

## Test suite- It is important to indicate under which suit the created test case will be placed.

## WINK RECORDING: Upload the wink of the manual execution of the test case with annotations for validations wherever applicable. Annotate the wink with the validations needed to be performed.

# Send Test Case for Review:

Once the test case definition is complete, the user can perform 'send for review' workflow and assign it to the 'product-expert' for that product. In the case where the user is the product-expert himself, he need not reassign. The product expert's job is to make sure that the Test Case is complete and the appropriate validations are present in the Test Case definition. The product expert can either 'accept' or 'reject' the test based on the review.

In case of 'Accept' - assign it to test case executor. In case of 'Reject' - assign it back to the creator with comments on what additional needs to be done and the flow goes back to previous step.

# Scenario Generator Best Practices:

The main goal to give an overview of the best practices and a guide to Scenario Generators in Creating High Quality Test case that can be automated by anyone who may have little or no knowledge of the product. For starters, before starting to create a Test case, a scenario Generator is expected to study the product in order to have a good understanding of the product.

## Best Practices: For a Test case to be considered of great quality it should meet the following basic criteria:

* + 1. The test case should be clear and as precise as possible. This means that every test step should be presented if possible by a singular step like 'click', 'enter', 'input' but not like 'Go to Settings' or 'open the page'. Such steps should be avoided or split into small parts. Avoid ambiguous steps.
    2. If there are a set of operations or initial conditions that need to be executed before performing the test they should be clearly indicated.
    3. Every precise test step should have its own precise validation step.
    4. In cases where something has to be selected, then the test case should provide the information of what is to be selected and how.
    5. If something needs to be entered, the value to be entered should be provided.
    6. The wink recording should have the test steps and validation steps.
    7. Bug fix test cases - the bug fix test case has to be a test case that validates that the bug is fixed, not just reproduces the issue. If a developer checks in code that brings the bug back (causes a "regression") then the test case that you are creating needs to actually fail.
    8. Do not have one test case dependent on other test cases.
    9. When test case is complete and ready to be reviewed, click "Send for Review" button - this will verify that all fields have been entered and wink has been attached.

## Writing Guidelines for Creating Test Case:

* + 1. The title should be clear, concise, and descriptive of the actual test to be performed.
    2. Use proper capitalization. Acronyms are allowed to be capitalized.
    3. Use proper sentence construction to make the title understandable.
    4. Always indicate the steps on how to start the application.

Ensure that the test case ends with the application being shut-down or closed properly.

Teramatrix Technologies Pvt Ltd

**Infinite Possibilities**