How to Maintain a Robust and Efficient Regression Suite

By Jacqueline Walton

# Problem:

Too many test cases are redundant, inflating the time it takes to execute a regression test. In order to test the test cases below, following the steps of each, would require a user to partially or completely sign In 19 times.

## Example:

* TC 1: On the Header, verify the Sign In button displays
* TC 2: On the Header, verify that clicking the Sign In button redirects users to the Sign In page
* TC 3: On the Header, verify the username displays in the header after sign in.
* TC 4: On the Header, verify the Sign In button is clickable
* TC 5: On the Sign-In Page, verify the username field displays
* TC 6: On the Sign-In Page, verify the username field can be filled in
* TC 7: On the Sign-In Page, verify the password field displays
* TC 8: On the Sign-In Page, verify the password field can be filled in
* TC 9: On the Sign-In Page, verify the Remember Me checkbox displays
* TC 10: On the Sign-In Page, verify a user can click the Remember Me checkbox
* TC 11: On the Sign-In Page, verify that when a user clicks the Remember Me checkbox and signs in, they are able to close the browser and reopen it and still be logged in.
* TC 12: On the Sign-In Page, verify when a user clicks the Submit button the user is signed in
* TC 13: On the Sign-In Page, verify the Connect with Facebook button displays
* TC 14: On the Sign-In Page, verify the Connect with Facebook button is clickable
* TC 15: On the Sign-In Page, verify when a user clicks the Connect with Facebook button, a Facebook login modal appears
* TC 16: On the Facebook Modal, verify the username and password fields display.
* TC 17: On the Facebook Modal, verify a user can fill in the password
* TC 18:  On the Facebook Modal, verify a user can fill in the username
* TC 19:  On the Header, verify that after a user logs in with Facebook, the username is displayed in the header.

# Solution

Eliminate test cases that are redundant by creating a test case that covers all **required** actions since those steps have to be done in order to complete the functionality, and then add test cases to cover non-required actions (deviations, permutations, or options). Also separate test cases from different pages.

## Example:

**TC 1: On the Header, verify that clicking the Sign In button redirects users to the Sign In page  this is functionality**

This covers the part of the process in the Header and obviously proves the sign in button displays and is clickable. There could be other test cases that test redirection links to the Sign In page and **should** be separate test cases, such as clicking on the Sign In link on the Rewards Enrollment Page, or the contact us page.

**TC 2: On the Sign-In Page, verify a user is able to Sign In. this is functionality**

This test case covers only the steps once a user is on the sign in page. This process does not need to be completed for every link that leads us to this page. These steps will cover all **required** actions such as filling in username and password and clicking the Submit button and if they can do this, then they were obviously there and obviously able to be filled in and the submit button works. This also proves that a user can sign in **without** clicking the Remember Me checkbox so there is no need for another test case for that.

Checking for the username showing up in the header should be in the **expected results** of every type of sign in (there are a few different ways to sign in) But this is an **expected result** of all, not separate test cases. ** this is NOT functionality. This is a state of the application.**

**TC 3: On the Sign-In Page, verify that when the Remember Me checkbox is checked and the Sign in form is submitted, the user is able to close the browser and reopen it and still be logged in.  this is functionality**

These steps covers the ‘Remember Me’ step which is not a required action to sign in, and obviously the fact that it was there and could be clicked. Yes the steps will repeat the Sign In process, but only once instead of several.

**TC 4: On the Sign In page, verify a user can login via the Connect with Facebook button** ** this is functionality**

Again, this covers all required steps to login via Facebook.

# Conclusion:

**The first 19 test cases was reduced to 4 test cases and still ensures 100% coverage.**

* Rule #1: A test STEP is not a test case. It is just a step in verifying some functionality.
* Rule #2: An EXPECTED RESULT is not a test case. It is a result of a test case.
* Rule #3: User or application functionality IS a test case.
* Rule #4: Just because we need to verify a requirement does not necessarily means it is a test case, it could fall under a test step or expected result. It is still verified.
* Rule #5: Every test step should be some sort of user interaction
* Rule #6: No test result should include user interaction, only the final state of the application or object after the steps are complete.

**KNOW YOUR TEST SUITE!**

Don’t just add new test cases! Make sure the functionality is not already covered. It may JUST need updating!