Liferay

Alfredo Soto Gómez

Senior Test Engineer Testing Exercise

13/10/2021

INDEX

1. Introduction

For this exercise I would like to explain different situations that I lived during the test.

1. For time reasons I've decided not to write BDD test cases for Test Planning section. In addition, on automated test part I did the same and I write the tests directly. A good practise would be to write BDD scenarios for manual testing and add a cucumber library or BDD library to use BDD scenarios in automated testing too.
   Examples of both cases are provided in their section.
2. If I had to say something, I would say that we are not using BD to store data. It seems that when we click Submit button, a standard message is displayed is all is ok. If we use database, we have to check frontend and backend validation for situations such as special chars (%$') and long lengths (database columns usually has 250chars o a limited chars)
3. We have tested with Javascript enabled.
4. I would implement some tips or placeholders on some fields. For example, I would write the date type allowed (dd-MM-YYYY) and I would have a counter for testing text area to set a limit of chars to send.
5. A good practise with webpages it's to test using Chrome developer console and see for JS error.

1. Test Planning

I have done a .html file which contains several test cases, from positives scenarios to negative and security scenarios. It can be found on ***1_Test Planning Exercise*** folder.

Test cases are written in an informal mode, just to show what we want to test. As I said at Introduction part, I didn't have time to do all scenarios using BDD language. A good practise is to set a classification for each test, such a *Critical, Medium or Low*. This classification can be used for Testing Reports to decide if we go to PROD or we don't.

To see an example of BDD, the positive scenario would be:

Scenario:

Given: Liferay Forms is loaded in <language> language

WHEN I type <name> name

AND I select a <date> from date picker

AND I type a <reason> on testing are

AND I click on <buttonName> button

THEN a <message> is dispayed

Examples:

|language|name|date|reason|buttonName|message|

|en|Alfredo Soto| 12/13/2021|Reason text|Submit|Success|

|pt|Alfredo Soto| 12/13/2021|Reason text|Submeter|Success|

These two rows implies that the test will be executed two times, using different data provided.

Furthermore, I've done some exploratory testing which I consider it's important because with test cases you cannot cover all the combinations. It would be unreal. Combination of both are the ideal scenario for manual testing.

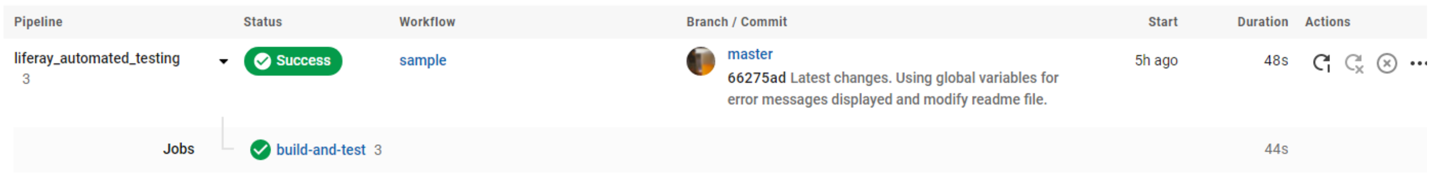
2. Automated Test

We have created two automated test using Selenium and TestNG. We have used CircleCI as CI and testingbot resource to test browsers on cloud. In this case we have selected Windows 10 and Chrome browser, because are the most common nowadays.

For running local test, we can use phantomJS or a headless browser, but in this case we wanted to see all the steps, so chromedriver was the decision.

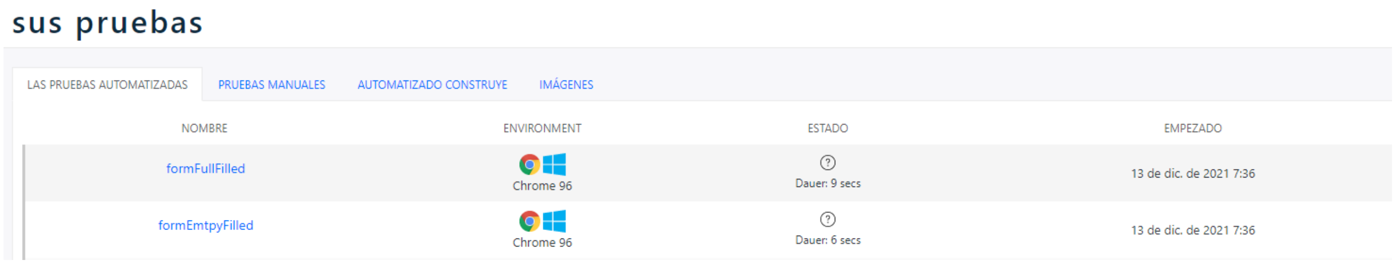
<u>What is the process?</u>

1. Download code from repo **(https://github.com/kokart/liferay_automated_testing)**
2. Commit a change
3. CircleCI job build-and-test starts (**https://circleci.com/)**



Pipeline	Status	Workflow	Branch / Commit	Start	Duration	Actions
liferay_automated_testing 3	Success	sample	master 66275ad Latest changes. Using global variables for error messages displayed and modify readme file.	5h ago	48s	
Jobs	build-and-test 3				44s	

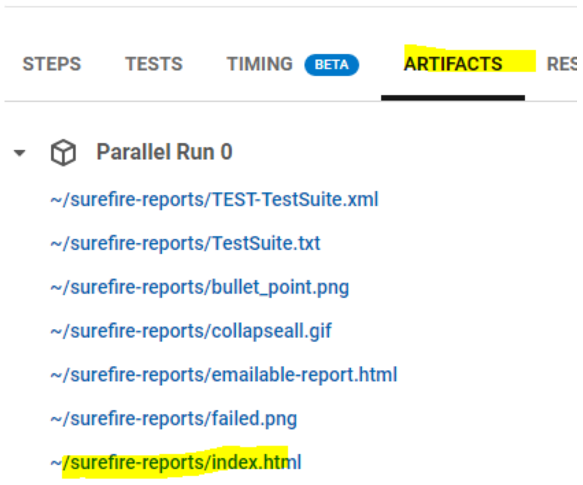
4. Two test are executed on testingbot **(https://testingbot.com/)**



sus pruebas

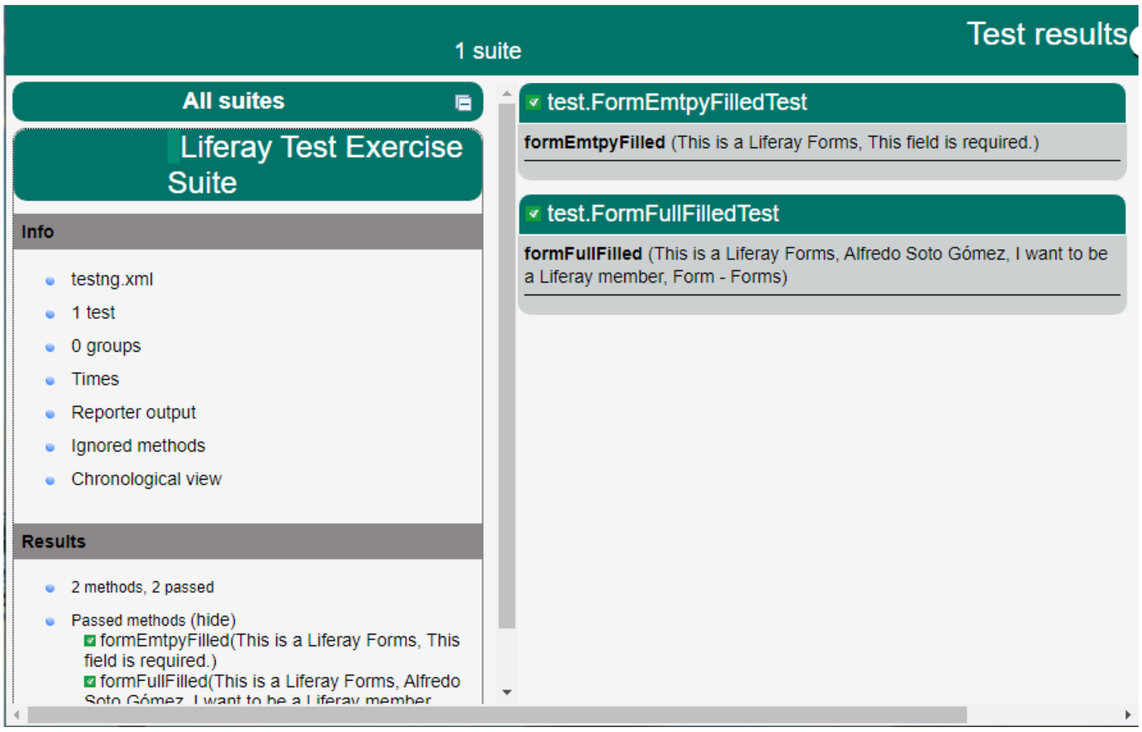
NOMBRE	ENVIRONMENT	ESTADO	EMPEZADO
formFullFilled	Chrome 96	Dauer: 9 secs	13 de dic. de 2021 7:36
formEmtpyFilled	Chrome 96	Dauer: 6 secs	13 de dic. de 2021 7:36

5. Results are stored.



Parallel Run 0

~/surefire-reports/TEST-TestSuite.xml
~/surefire-reports/TestSuite.txt
~/surefire-reports/bullet_point.png
~/surefire-reports/collapseall.gif
~/surefire-reports/emailable-report.html
~/surefire-reports/failed.png
~/surefire-reports/index.html


Test results

1 suite

All suites

Liferay Test Exercise Suite

Info

- testng.xml
- 1 test
- 0 groups
- Times
- Reporter output
- Ignored methods
- Chronological view

Results

- 2 methods, 2 passed
- Passed methods (hide)
  ✓ formEmtpyFilled(This is a Liferay Forms, This field is required.)
  ✓ formFullFilled(This is a Liferay Forms, Alfredo Soto Gómez, I want to be a Liferay member

✓ test.FormEmtpyFilledTest

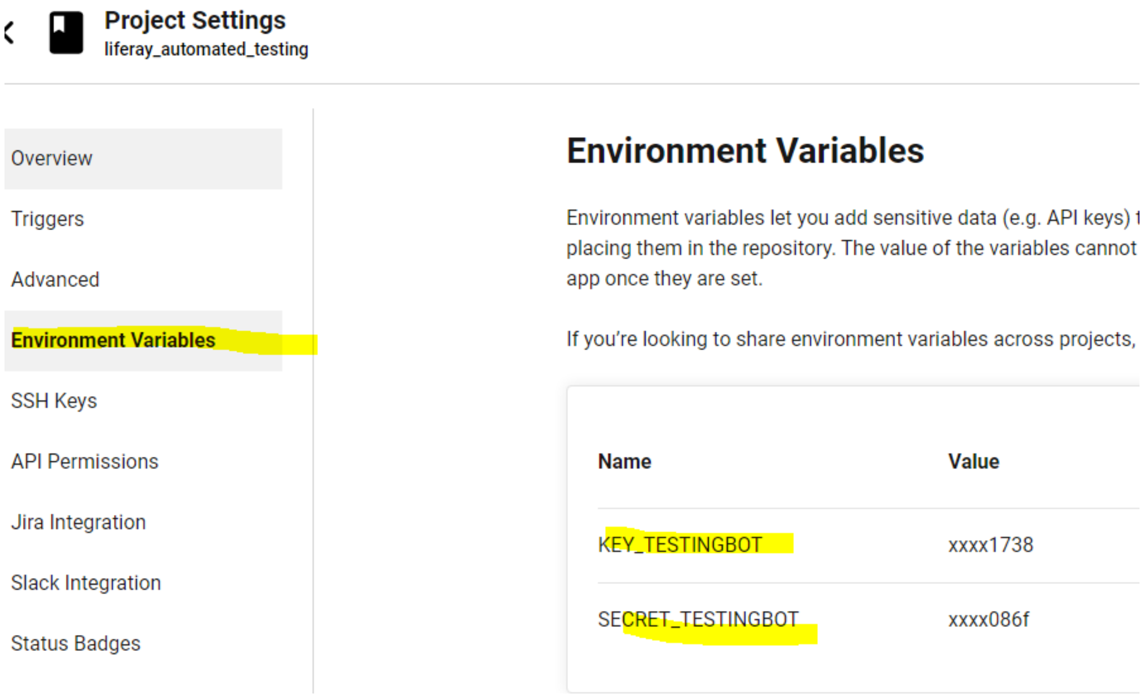
formEmtpyFilled (This is a Liferay Forms, This field is required.)

✓ test.FormFullFilledTest

formFullFilled (This is a Liferay Forms, Alfredo Soto Gómez, I want to be a Liferay member, Form - Forms)

For using testingbot we have created a trial account and we used the secret and key token to launch several browsers.

Secret and Key token provided by testingbot should be configured on CIRCLECI on Environment variables for the project we want to launch:


Project Settings
liferay_automated_testing

Environment Variables

Environment variables let you add sensitive data (e.g. API keys) t placing them in the repository. The value of the variables cannot app once they are set.

If you're looking to share environment variables across projects,

Name	Value
KEY_TESTINGBOT	xxxx1738
SECRET_TESTINGBOT	xxxx086f

Then, we have to configure configure.yml file to build and test.

To use BDD we would have to write different annotation before a method and create .scenario files.

A .txt pointing the repo can be found at ***2_Automated Testing*** folder.

3. Test Strategy

A pdf with the explanation can be found at ***3_Test Strategy*** folder.

4. Bug Reporting

A pdf with several bug defects can be found at ***4_Bug Reporting*** folder.

If we would use a Test Cases tracker or JIRA, we can link test case with defect detected. In this case, I write the number of the test case which detects the bug.

5. Test Report

A pdf with a test report can be found at ***5_Test Report*** folder. As I said on Test Plan section, we can provide two levels of reports:

- A pie chart with the result of the Test Run execution, just shown Passed or Failed test.
- A pie chat displaying the number of Critical tests passed, Medium passed and Low passed.

I considerer the second option the best way to decide, because we can have a lot of low test failed but we can go PROD. They are known as “minor issues”.