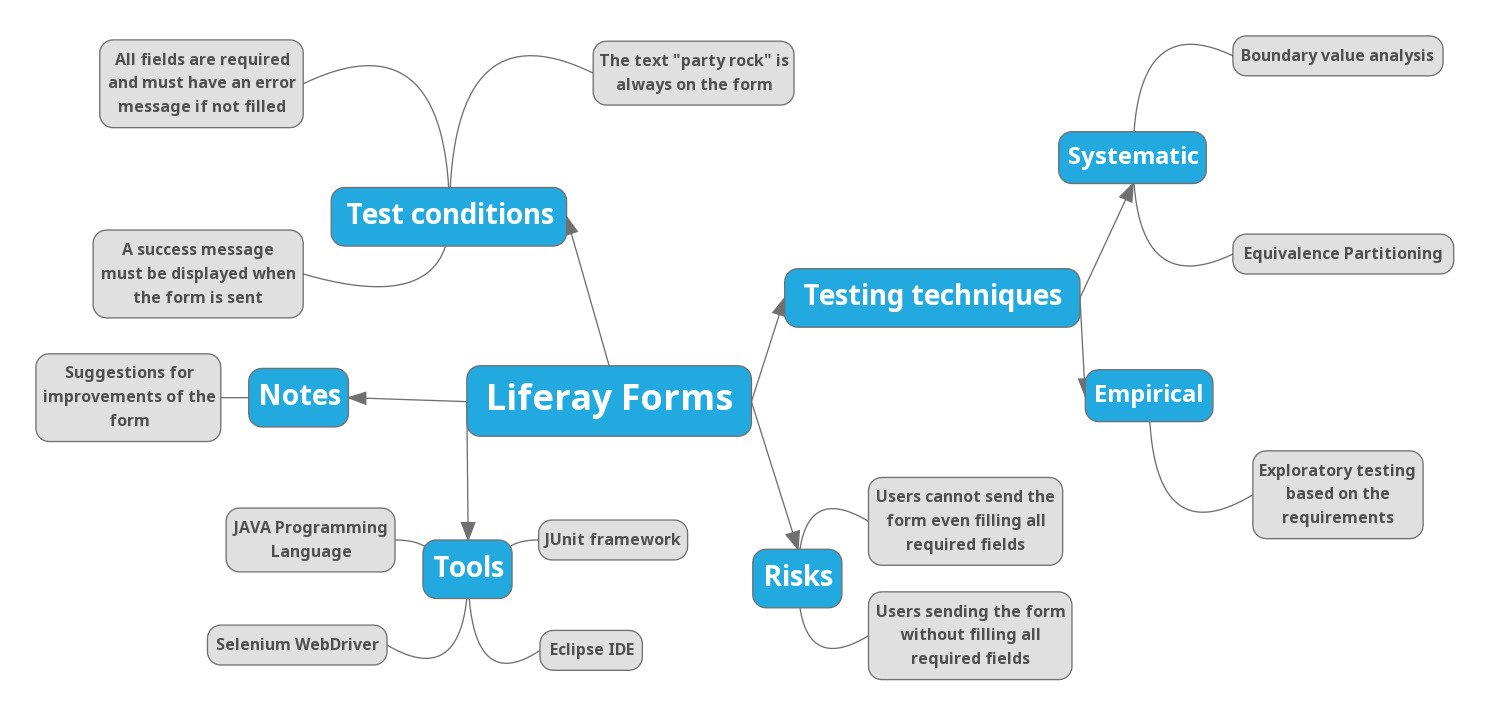
**Liferay Forms Test Report**

This report is a brief description about all the process during Liferay Forms quality analysis. The approach used to execute manual tests, the approach to writing the automation scripts, the bugs found during the execution of both automated and manual tests, the test plan, as well as some suggestions to improve the form are described here.

**Test Plan**

The test plan is described in the following mind map.



**The approach used to test the application manually**

The first step given to test the application manually was looking at the requirements and understand how the application works. After understanding how the form works, a strategy to test was created aiming to cover all the three requirements given and also some alternate paths. Empirical and Systematic testing techniques were used. The Empirical was exploratory testing based on the requisites and the Systematics were equivalence partitioning and boundary value analysis specially to test the “Date of Birth” field.

The first tests executed were to cover the happy path, to make sure that the application was working as expected and all the requisites were being complied. The happy path consisted in checking if the text “party rock” is on the form, regardless of the language of the application (English or Portuguese), filling out all the fields and clicking on “Submit” button to check if a success message is displayed to the user. There were 2 tests to cover the happy path. The first was filling all the fields through the keyboard, clicking on “Submit” button and verifying if the success message is shown. The second was filling the name and text field through the keyboard and selecting the date of birth through the calendar. In this second test, it was not possible to select a year before 2015, in other words, if the user was born before this year, he or she would not be able to select the date of birth through the calendar. In this case, it was not possible to send the form, choosing a date through the calendar before 2015.

Unhappy paths were also tested to check if the form is not sent and if all error messages are displayed bellow each field when one or more are not filled. Besides, tests to make sure that there are no error messages after filling each field and to check if everything was translated correctly (except for “party rock”) were executed.

**The approach used to automate the tests**

The automation was based on a risk analysis of the form, where the scripts were written to cover all the requirements and risks of the application. Only The tests related to the form submission and the field filling were automated, due to it’s not a good practice to automate every single test in the application. The tests to ensure that the form is sent after filling the name and text field through keyboard and the date field through both keyboard and calendar.

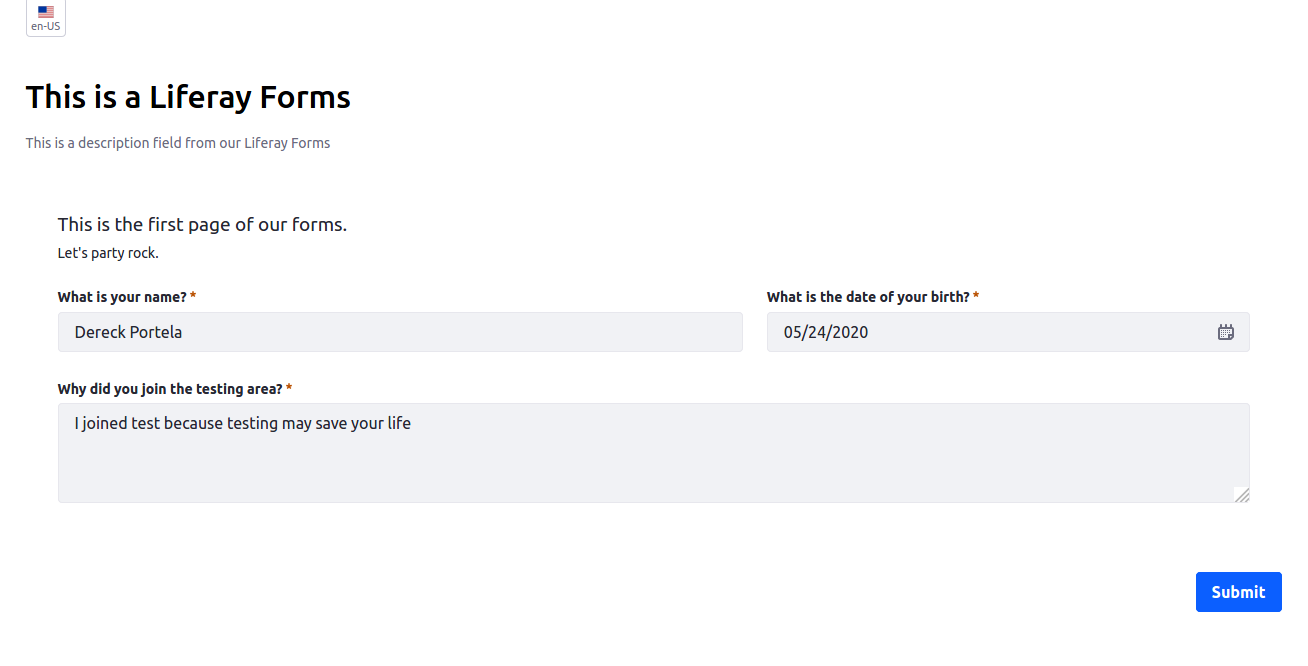
The unhappy path tests were also automated to check if the form is not sent when the fields are not filled out, if after typing something in the field and clicking on another one, no error message should be displayed bellow the filled field. If what was typed by the user is what is being displayed in the field to make sure that there is no script to manipulate the data. Also tests to verify that the text “party rock” is displayed on the form, regardless of the language, were automated too. These tests covered all the risks and if one of them fails, it means that the requirements are not being fulfilled properly.

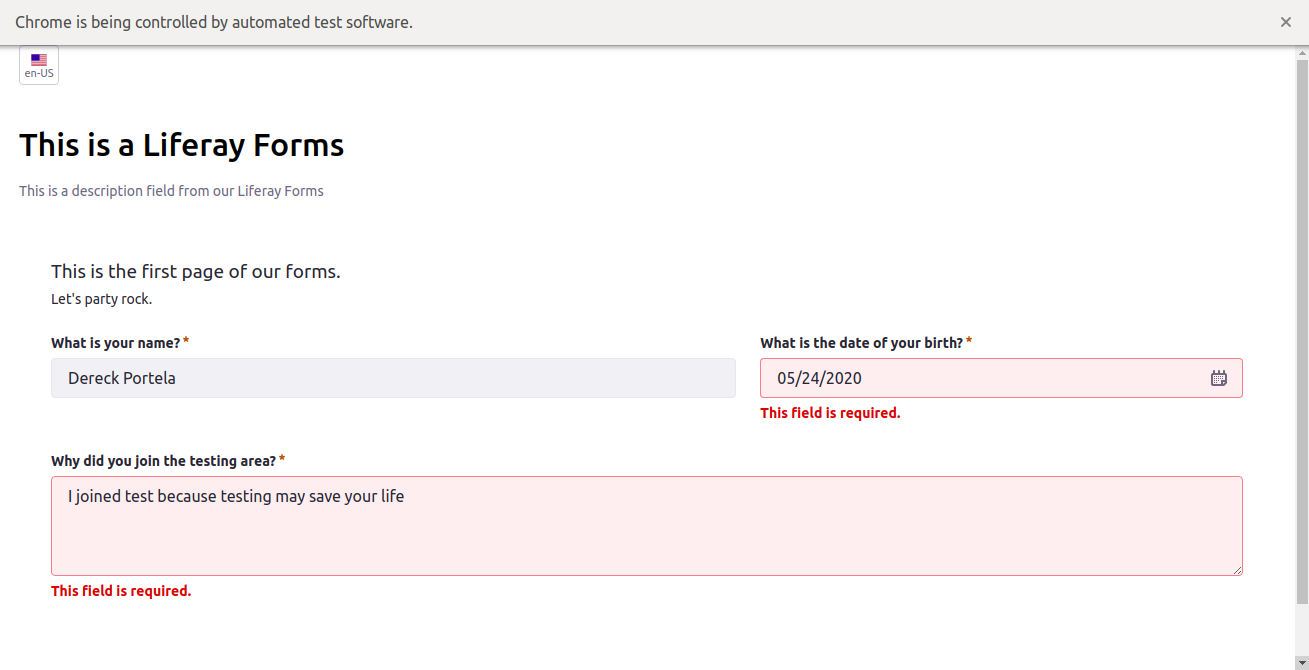
The tools used to write the automation scripts were Java programming language, eclipse IDE, selenium WebDriver and page objects as the pattern.

**Bugs found when running the automation scripts**

When running the automation scripts, was realized that it was not possible to send the form even executing the Happy Path script, which fills all fields and clicks on Submit button. The application does not recognize that the fields are not empty and displays an error message as if they were. It does not matter if the script fills the name, text field and type or choose the date through calendar, it’s not possible to send the form running an automation script.

What’s is interesting is that following the same steps manually, the application works fine. Bellow, are two pictures, the first one the happy path steps were executed manually and the second one, automatically.





It’s easy to see that there is an inconsistency between executing the same test manually and automatically.

Other inconsistencies were found when executing the automated scripts. When the script simulates the user typing the date in the field, it’s not possible to see the typed date, it behaves as if nothing was entered.

When the script types special characters in the field and clicks on another field, no character is entered, which is correct, but the error message is not displayed bellow the date field. If this test is executed manually, the error message appears bellow the date field.

If the script types in a field and after that, clicks on another one, the error message is rapidly displayed bellow the field that was filled but it should not be. It’s hard to simulate this scenario manually, but the script was able to catch it.

These issues may be happening when the automated scripts are run because of its speed. Perhaps goes so fast that the application does not recognize when something is typed.

**Scenarios used to automate the scripts**

The following are BBD scenarios using gherkin language to describe each scenario. Consider the fictional user called John is using the Liferay Forms.

**Scenario 1**: John can successfully see the page title

**GIVEN:** John has just typed the URL and accessed the form page

**WHEN:** The page loads

**THEN:** He can see the title of the page

**Scenario 2**: John can successfully see the text “party rock”

**GIVEN:** John is on the form page

**WHEN:** He looks for the text “party rock”

**THEN:** He can see the text “party rock” on the form

**Scenario 3**: John can successfully see the text “party rock” after changing language

**GIVEN:** John is on the form page

**WHEN:** He changes the language

**AND:** He looks for the text “party rock”

**THEN:** He can see the text “party rock” on the form

**Scenario 4**: John types his name in the “name” field

**GIVEN:** John clicks on the name field

**WHEN:** He types his name

**THEN**: He can see his name in the field and no error message is displayed bellow this field when clicking on another one

**Scenario 5**: John types his date of birth in the “date of birth” field

**GIVEN:** John clicks on the date of birth field

**WHEN:** He types his date of birth

**THEN**: He can see his date of birth in the field and no error message is displayed bellow this field when clicking on another one

**Scenario 6**: John select his date of birth through the calendar

**GIVEN:** John clicks on the “date of birth” field

**WHEN:** He selects the month of birth

**AND:** He selects the year of birth

**AND:** He selects the day of birth

**THEN:** He can see his date of birth in the field and no error message is displayed bellow this field when clicking on another one

**Scenario 7**: John select the current date through the circular button on the calendar

**GIVEN:** John clicks on the “date of birth” field

**WHEN:** He clicks on the circular button on the calendar

**AND:** He selects the current day

**THEN:** He can see the current date in the field and no error message is displayed bellow this field when clicking on another one

**Scenario 8**: John types special characters in the “date of birth” field

**GIVEN:** John clicks on the “date of birth” field

**WHEN:** He types special characters

**THEN**: He must not see any special character and in case of clicking on another field, an error message must be displayedbellow the "date of birth" field

**Scenario 9**: John types why he joined the testing area

**GIVEN:** John clicks on the “text” field

**WHEN:** He types why he joined the testing area

**THEN:** He can see why he joined the testing area and no error message is displayed bellow this field when clicking on another one

**Scenario 10**: John fills all fields and submit the form

**GIVEN:** John clicks on the “name” field

**WHEN:** He types he types his name

**AND:** He clicks on “date of birth” field

**AND:** He types his date of birth

**AND:** He clicks on the “text” field

**AND:** He types why he joined the testing area

**AND:** He clicks on submit button

**THEN:** He is able to submit the form and see a success message

**Scenario 11**: John fills “name”, “text”, select the date of birth through the calendar and submit the form

**GIVEN:** John clicks on the “name” field

**WHEN:** He types he types his name

**AND:** He clicks on “date of birth” field

**AND:** He selects his date of birth through the calendar

**AND:** He clicks on the “text” field

**AND:** He types why he joined the testing area

**AND:** He clicks on the “Submit” button

**THEN:** He is able to submit the form and see a success message

**Scenario 12**: John submit the form without filling any fields

**GIVEN:** John is on the form page

**WHEN:** He clicks on the “Submit” button

**THEN:** He cannot submit the form and error messages are displayed bellow the fields, saying that the field filling is required

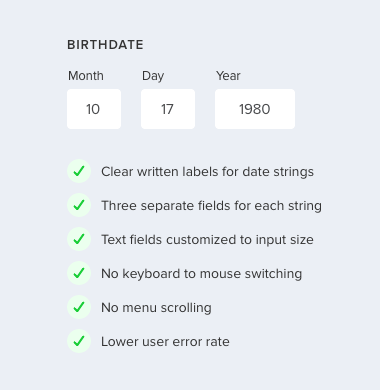
**Suggestions and improvements**

Here go some improvements that can be made in the application. The first one, would be define a minimum and maximum number of characters in “name” and “text” field. For example, in the “name” field, the user can type only the letter “A” and it’s enough to not trigger the field error message but it’s very unlikely that a person is called “A”. The same for a huge number of characters. If the user enters 200 characters, the field has no treatment and accept all of them, but it’s also very unlikely that a person has a 200-character name. The same goes for the “text” field, there should be a minimum and maximum character validation. This feature could be easily done by setting up the min and max length in the HTML tag. Special characters would be fine to accept in the “name” field, given that there are foreigner names.

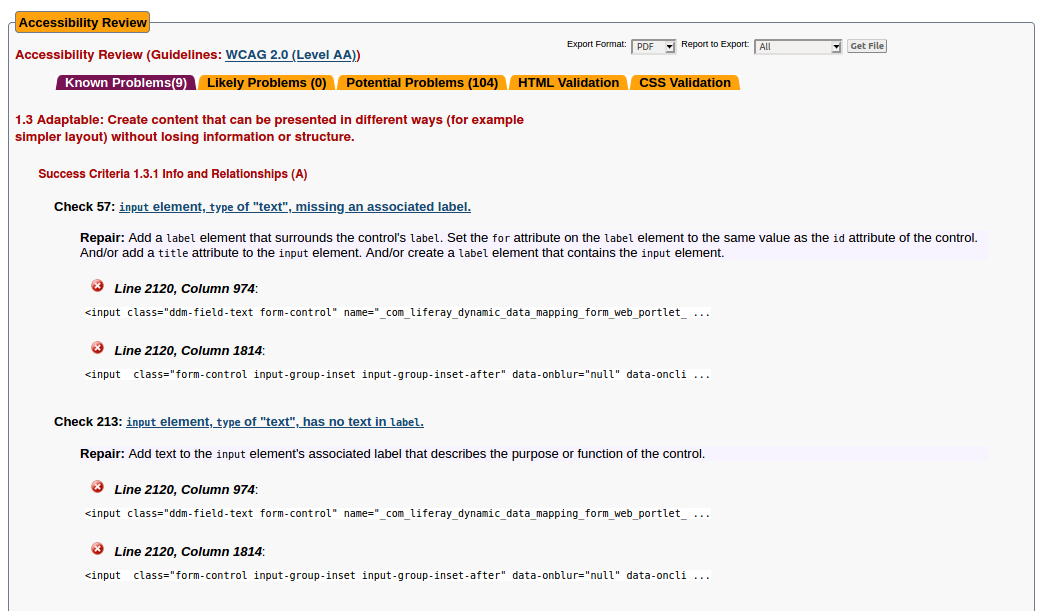
The “date of birth” field could also be improved. The first improvement would be defining a limitation on the calendar dates. It’s possible to type 01010001 (01/01/0001) or 12319999 (12/31/9999) but no one is too old or be born in the future and came back to the past to fill out the form.

Another improvement would be putting a placeholder in the field like “MM/DD/YYYY”. It’s really confusing to the user to enter his date of birth without knowing in what order the date should be entered.

Even though defining a date limitation and putting a place holder would improve the “date of birth” field, it’s not the best way to create this kind of form field. According to [**uxmovement**](https://uxmovement.com/forms/bad-practices-on-birthdate-form-fields/)it’s a bad practice to create these forms using “MM/DD/YYYY” format because it leads the user to stop the flow of filling the form and thinking about how the date should be entered. Calendars are not a good idea too, due to the user has to scroll down or up to find the day, month and year of birth. The best practice to implement this kind of form is like in the picture bellow.



It’s not possible for people with some kind of visual impairment or blindness to know what the application does and what elements are on the screen, using a screen reader. According to [Achecker](https://achecker.ca/checker/index.php), the application does not support accessibility.



Front end rendering performance could also be improved. According to [PageSpeed Insight](https://developers.google.com/speed/pagespeed/insights/?hl=pt-br) the Front end rendering speed is really slow on mobile devices and just okay on desktops.

