**Testing Task:**

* **Report of executed tests**

High Level Test Scenarios:

| Test Scenarios | Test Cases/Steps | Result | Log name |
| --- | --- | --- | --- |
| User must Register Before login | 1. Go to URL 2. Maximize Browser 3. Click Register 4. Input all the fields 5. Click Register | PASS | Log.html |
| User Login | 1. Go to URL 2. Maxmize Browser 3. Click Login 4. Input User name and Password fields 5. Click Login | PASS | Log.html |
|  |  |  |  |
|  |  |  |  |

Negative Test Scenarios :

| Test Scenario | Test Case | Result | Log name |
| --- | --- | --- | --- |
| User must Register Before Login | Registering with already registered user | PASS | Registered\_User\_Log.html |
| User name intelligence tests - Usernames should be starting with a character or number or \_ and not with any special characters | FAIL | User\_name\_int.html |
| Password Intelligence tests - Password should have minimum length check, no of special characters check and numbers check | FAIL | Password\_int.html |
| First name and Family name Intelligence test - First namd and Family name cannot be same | FAIL | First\_Family\_name\_int.html |
| Telephone number intellignece tests - Telephone number field should have a special drop down with the country codes and text field with minimum length of the phone number as per the country specific | FAIL | Phone\_num\_int.html |
| Telephone number should not accept characters but only numbers | FAIL | phn\_num.html |
| Registeration page can add user email address for User convenience | FAIL | Log.html |
| Login Page post Registeration | PASS | Log.html |
| Login Page Test | UI should accept only the registered user to login | PASS | Incorrect\_login.html |
| UI should check if the user is providing the user name registered in UI eventhough the password is wrongly provided - expected output is the provided user is not registered | FAIL | Incorrect\_login.html |
| UI should have a link 'Forgot Password ' to provide user a link to change his/her password based on the registered email or phone number | FAIL | Log.html |
| Cross Page Test | Link to back to the home page test | PASS | Log.html |
| UI Look and Feel Test | Look and feel is pretty much normal and can be more user attention gaining experience by having some good styling features, images and other styling features |  |  |

* **Report of found issues/bugs**

Above table will give the details

* **Answers to the questions related testing and testability**

Can be found in the testing and testability document

* **Exploratory Testing Report**

**Test Charter :**

a. Test automation started with the high level two scenarios mentioned in the above first table

b. Test cases written in such a way to cover the main two scenarios in a single script file

c. Based on the execution I came across different scenarios mentioned in the above second table for both of the main Test scenarios

d. Code refactoring done based on the scenarios to cover all the test cases ( Common test cases are added into a single script and scenario based test cases are added into a single script)

e. High test coverage with more negative test scenarios

* **Tell us what improvement would you propose for the app**

From the UI side below are the improvements that cane be done :

1. Build more intelligence for the Registration and login pages for the above Failed test cases in the second table

2. Look and feel is pretty much normal and can be more user attention gaining experience by having some good styling features, images and other styling features

* **If you would be given a week to do quality assurance for this product, briefly plan the tasks based on your skills, knowledge and expertise**

1. First I tried to understand the requirement

2. Based on the understanding I started writing the test steps in both the UI and API scenarios

3. As both the environments to test are with different tools (Python & RF) I started doing the environment readiness .

4. Started with the draft scripts in both the areas and based on the exploration added few more steps and tried to make the code more better and error catching way

5. Refactored the code and made it more maintainable

6. Executed the tests for all the scenarios planned and found out the issues.