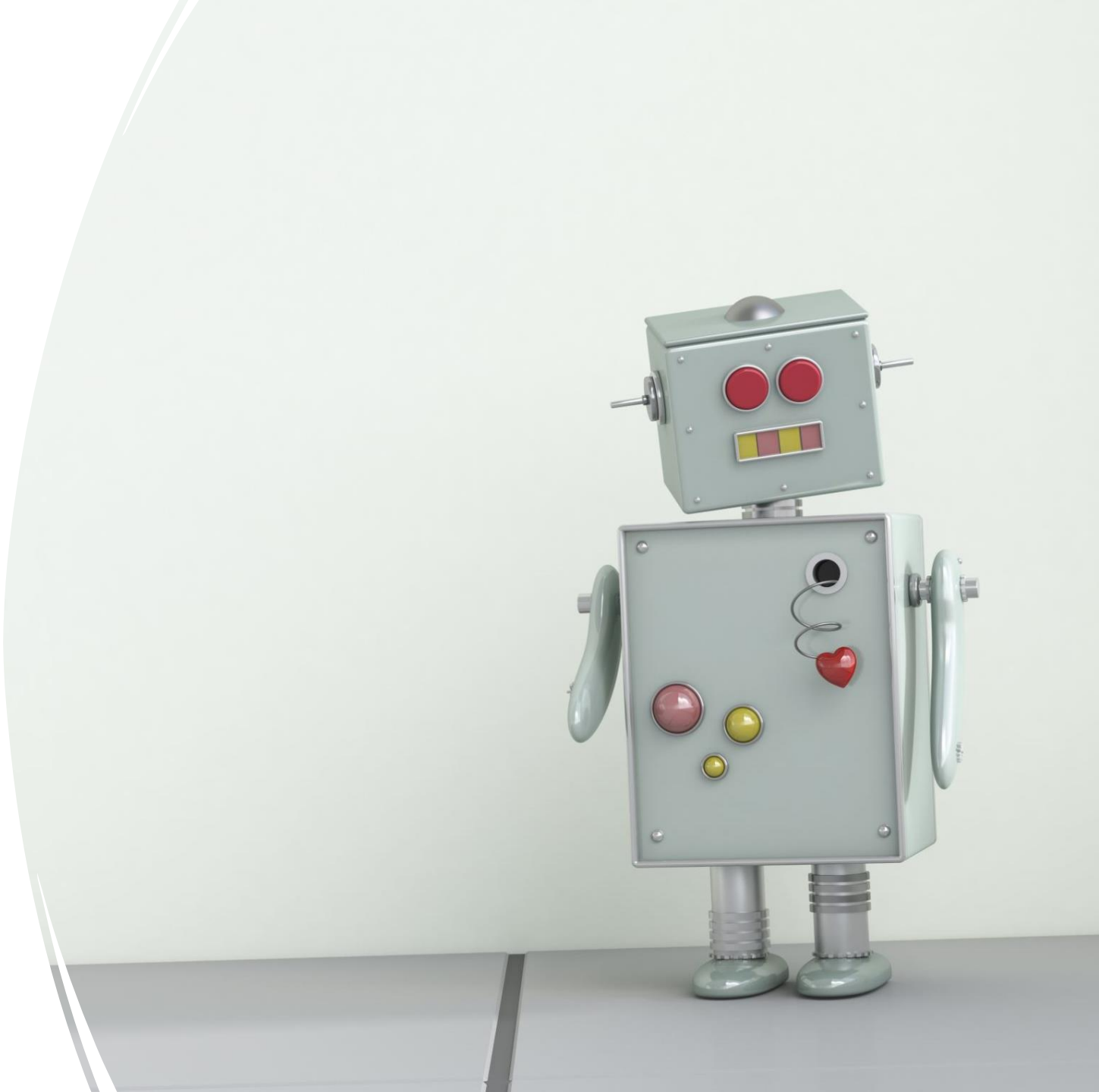
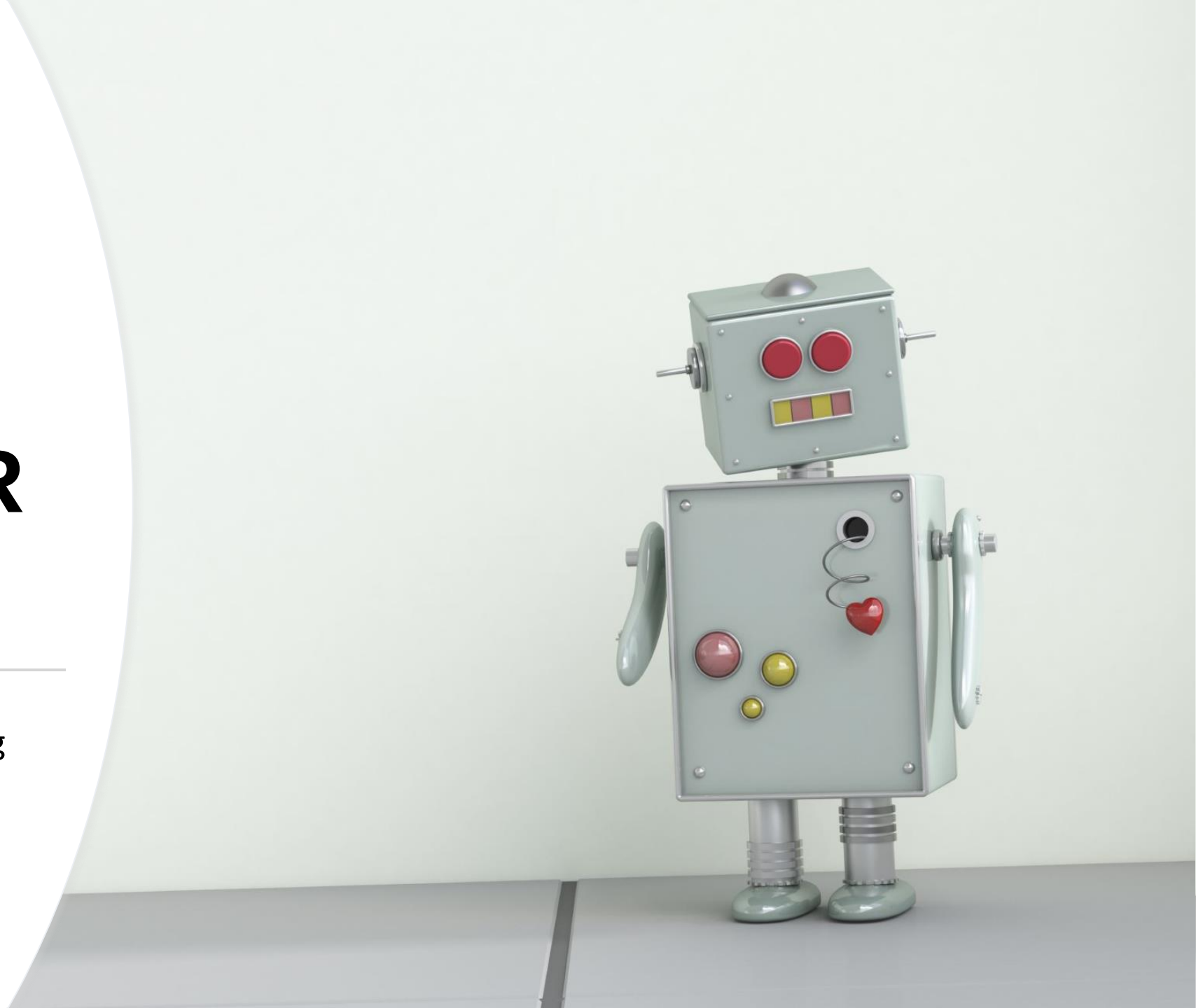
GROUP 2
ROBOT FRAMEWORK

- MOHD SAFWAN
- MUHD SYAZWAN
- ANIS AMIRA



ROBOT FRAMEWORK

Building Automation Testing Using Robot Framework



ROBOT FRAMEWORK

Introduction robot framework

Introduction Robot Framework

generic **open source automation framework**

is a Python-based

extensible keyword-driven automation framework for acceptance testing

behavior driven development (BDD)

robotic process automation (RPA)

Introduction Robot Framework

free to use without licensing costs

easy syntax, utilizing human-readable keywords

Its capabilities can be extended by libraries implemented with Python, Java or many other programming languages

Features of Robot Framework

Keywords (Exp Open browser, Close browser, Click element, Input text)

Variables (scalar, list and dict)

Libraries (Selenium library)

Resources (imported using the *Resource* setting in the Settings section)

Data driven test cases(works with high-level keyword used as a template to the test suite and the test cases)

Test Case Tagging(helps when we want to run only a group of test cases or skip them)

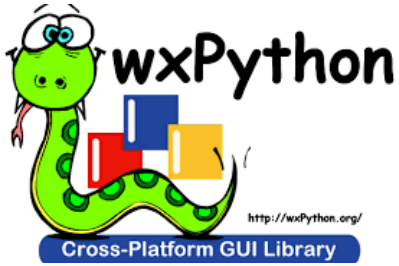
Reports and Logs(provides all the details of test suite, test case execution in the form of report and logs)

Limitation robot framework for automation testing

- Robot framework does not support parallel execution
- Hard to customize html report
- Robot framework is hard to maintain
- Some error are difficult to debug
- Robot framework has strict indentation rules

PRE-REQUISITES

You need to complete these before



Selenium Library

Installing robot framework

- Robot Framework is implemented with Python, so you need to have Python installed.
- On Windows machines, make sure to add Python to PATH during installation.

Install python

Files

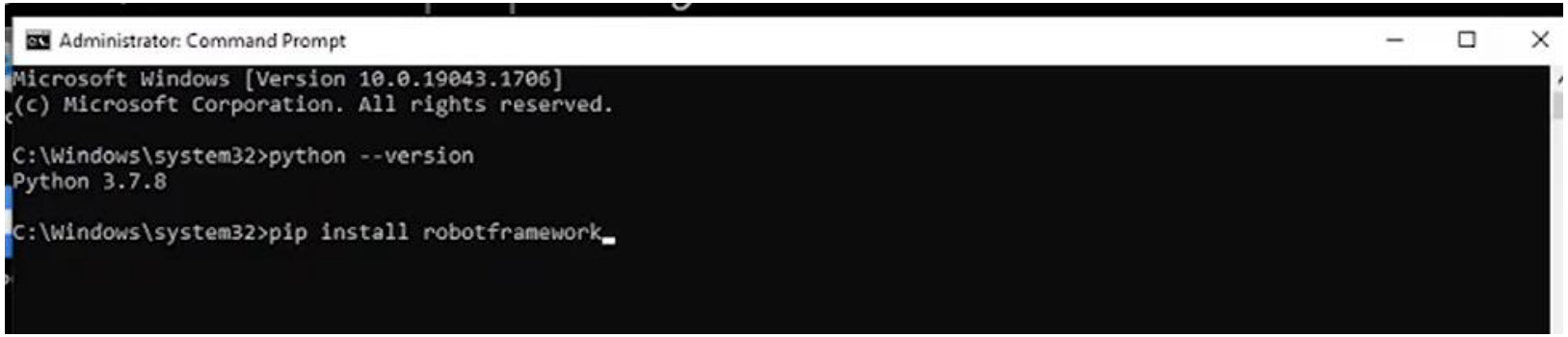
Version	Operating
Gzipped source tarball	Source rel
XZ compressed source tarball	Source rel
macOS 64-bit installer	macOS
Windows help file	Windows
Windows x86-64 embeddable zip file	Windows
Windows x86-64 executable installer	Windows
Windows x86-64 web-based installer	Windows
Windows x86 embeddable zip file	Windows
Windows x86 executable installer	Windows
Windows x86 web-based installer	Windows

Check python use command prompt

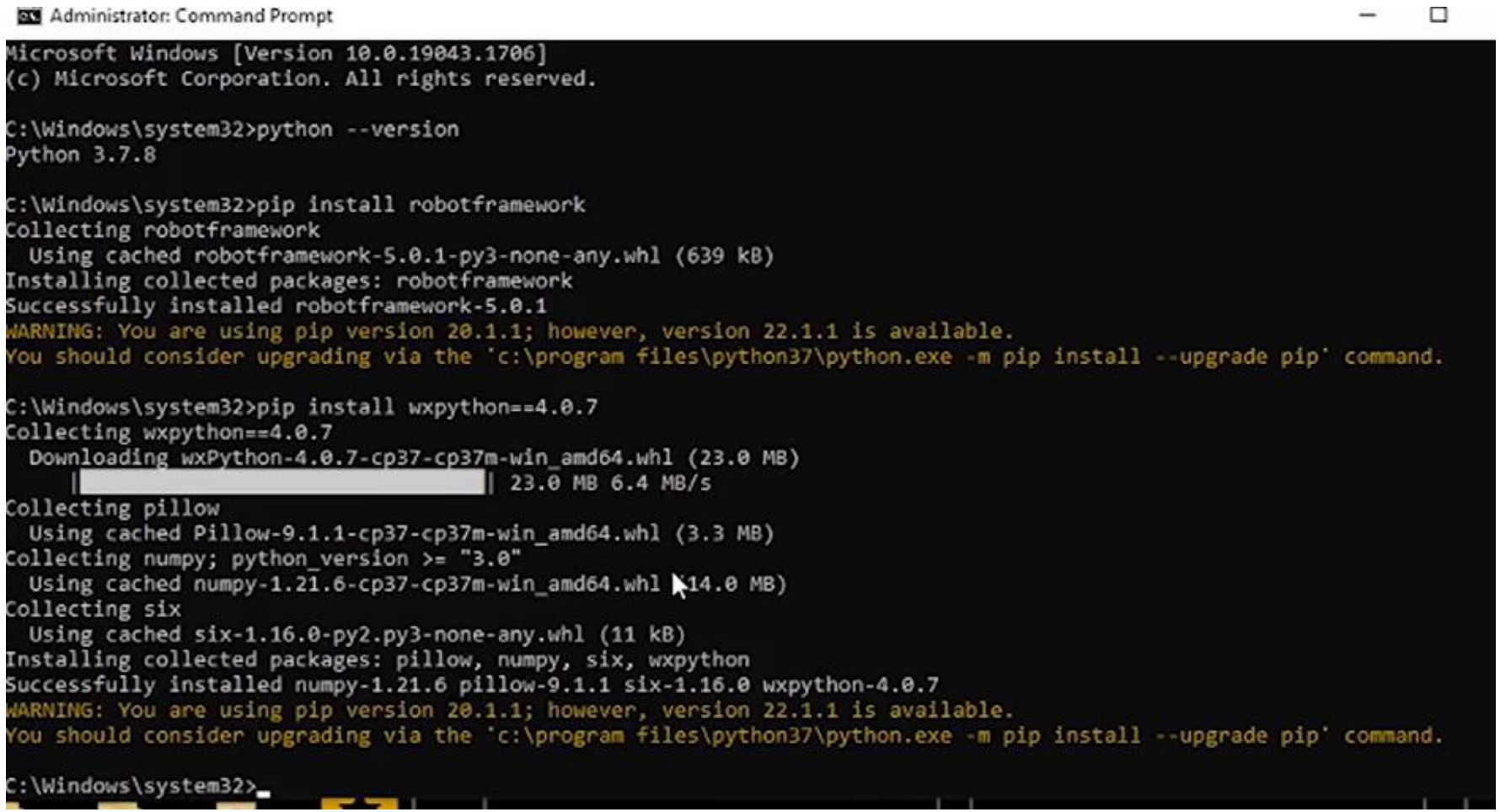
Open cmd

Type Python --version

Install robot framework

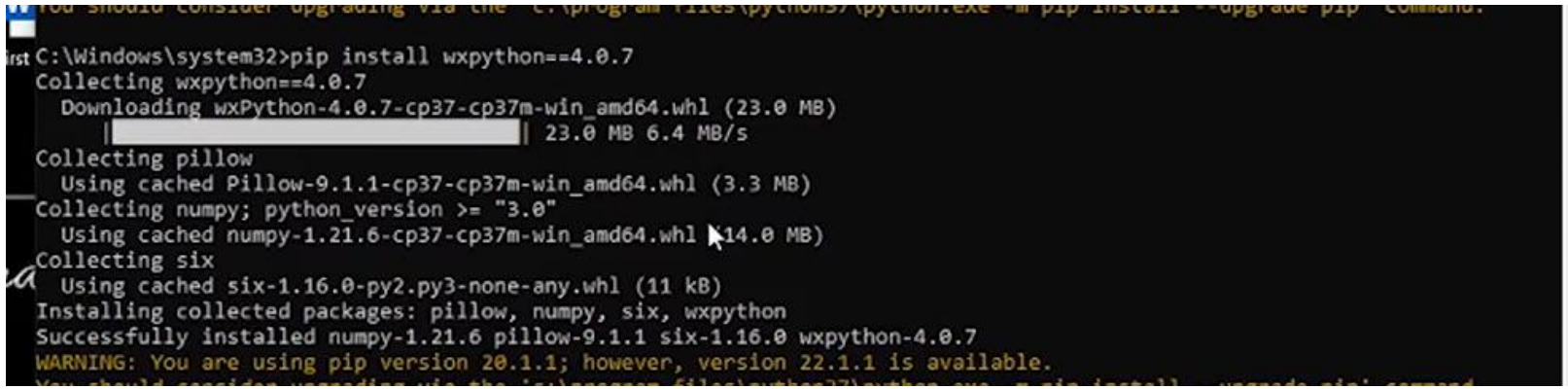


Type pip install robotframework


Administrator: Command Prompt
Microsoft Windows [Version 10.0.19043.1706]
(c) Microsoft Corporation. All rights reserved.
C:\Windows\system32>python --version
Python 3.7.8
C:\Windows\system32>pip install robotframework
Collecting robotframework
Using cached robotframework-5.0.1-py3-none-any.whl (639 kB)
Installing collected packages: robotframework
Successfully installed robotframework-5.0.1
WARNING: You are using pip version 20.1.1; however, version 22.1.1 is available.
You should consider upgrading via the 'c:\program files\python37\python.exe -m pip install --upgrade pip' command.
C:\Windows\system32>pip install wxpython==4.0.7
Collecting wxpython==4.0.7
Downloading wxPython-4.0.7-cp37-cp37m-win_amd64.whl (23.0 MB)
23.0 MB 6.4 MB/s
Collecting pillow
Using cached Pillow-9.1.1-cp37-cp37m-win_amd64.whl (3.3 MB)
Collecting numpy; python_version >= "3.0"
Using cached numpy-1.21.6-cp37-cp37m-win_amd64.whl (14.0 MB)
Collecting six
Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: pillow, numpy, six, wxpython
Successfully installed numpy-1.21.6 pillow-9.1.1 six-1.16.0 wxpython-4.0.7
WARNING: You are using pip version 20.1.1; however, version 22.1.1 is available.
You should consider upgrading via the 'c:\program files\python37\python.exe -m pip install --upgrade pip' command.
C:\Windows\system32>

Install wxpython

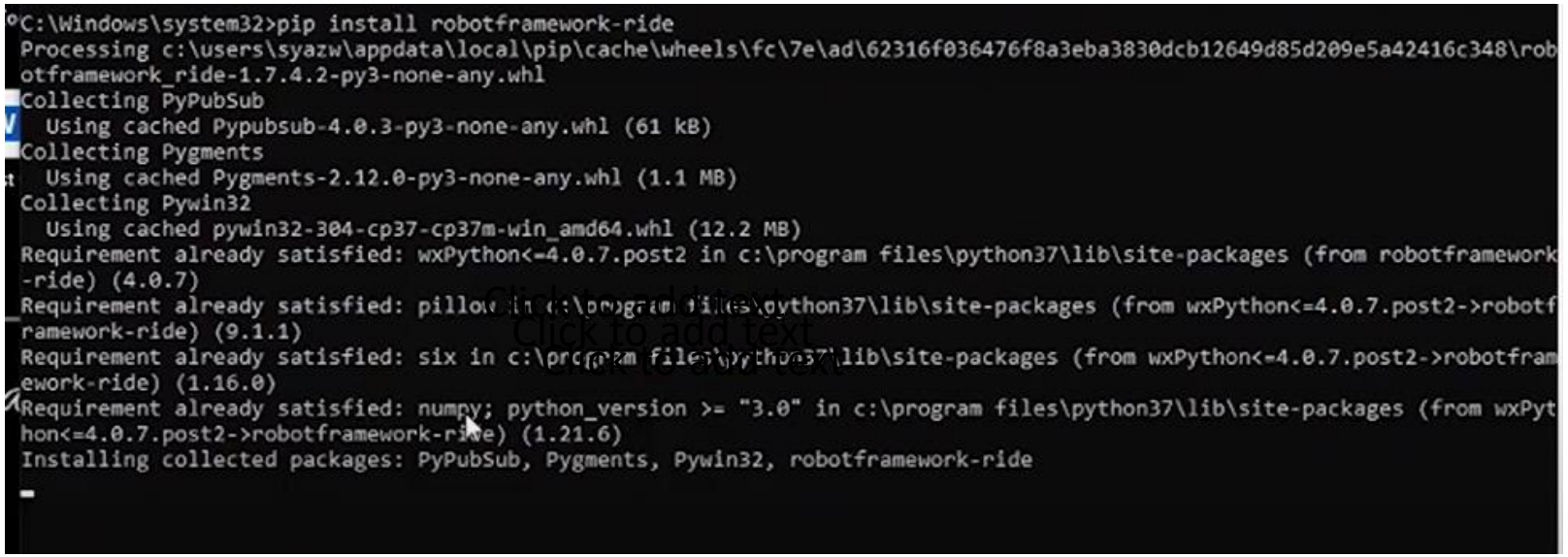
- wxPython is a wrapper for the cross-platform GUI API wxWidgets for the Python programming language
- It is implemented as a Python extension module



Type pip install wxpython==4.0.7

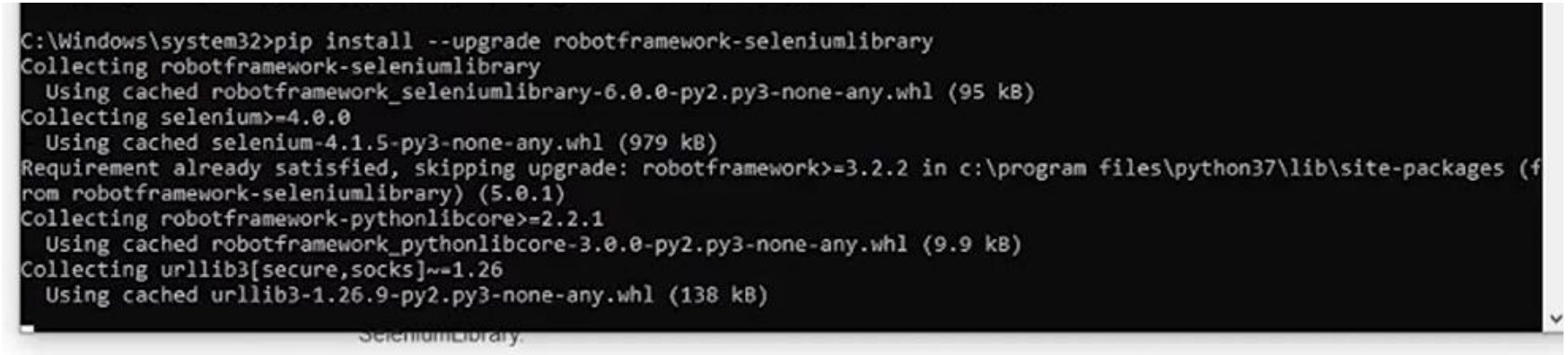
Install ride

- Ride is **a testing editor for Robot Framework**.
- We will write test cases in Ride



Type pip install robotframework-ride

Install Selenium Library

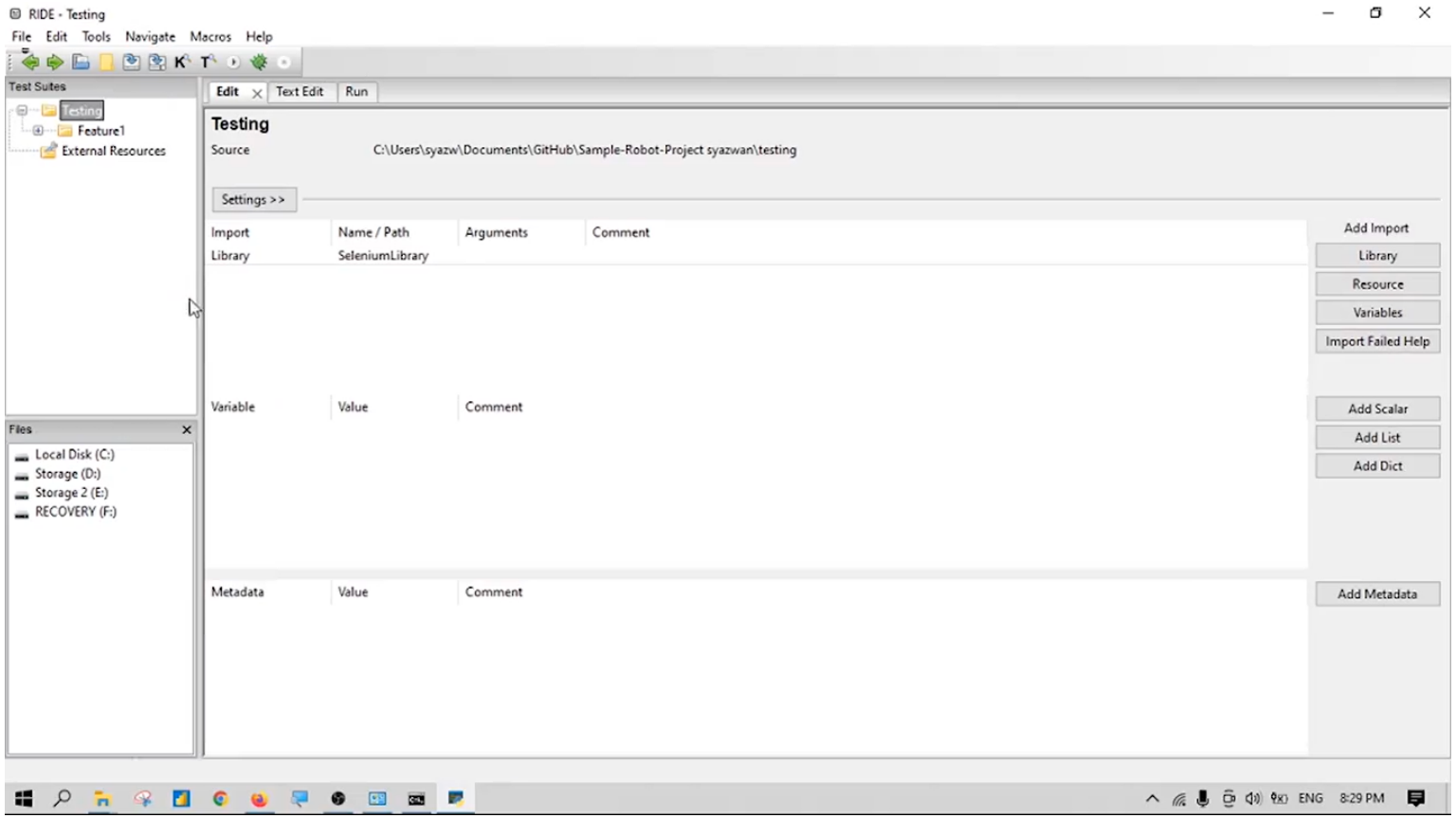


Type pip install –upgrade robotframework-seleniumlibrary==5.1.1

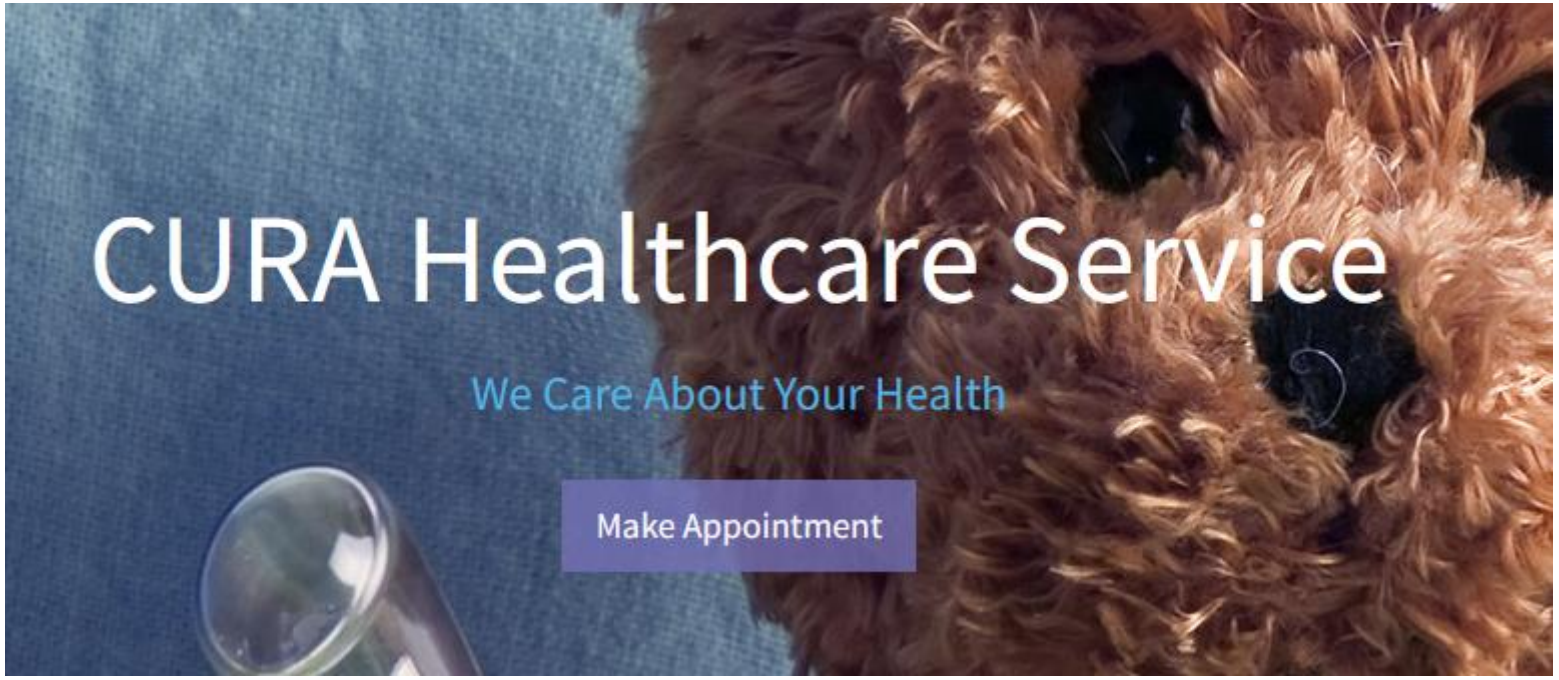
Open ride robot framework

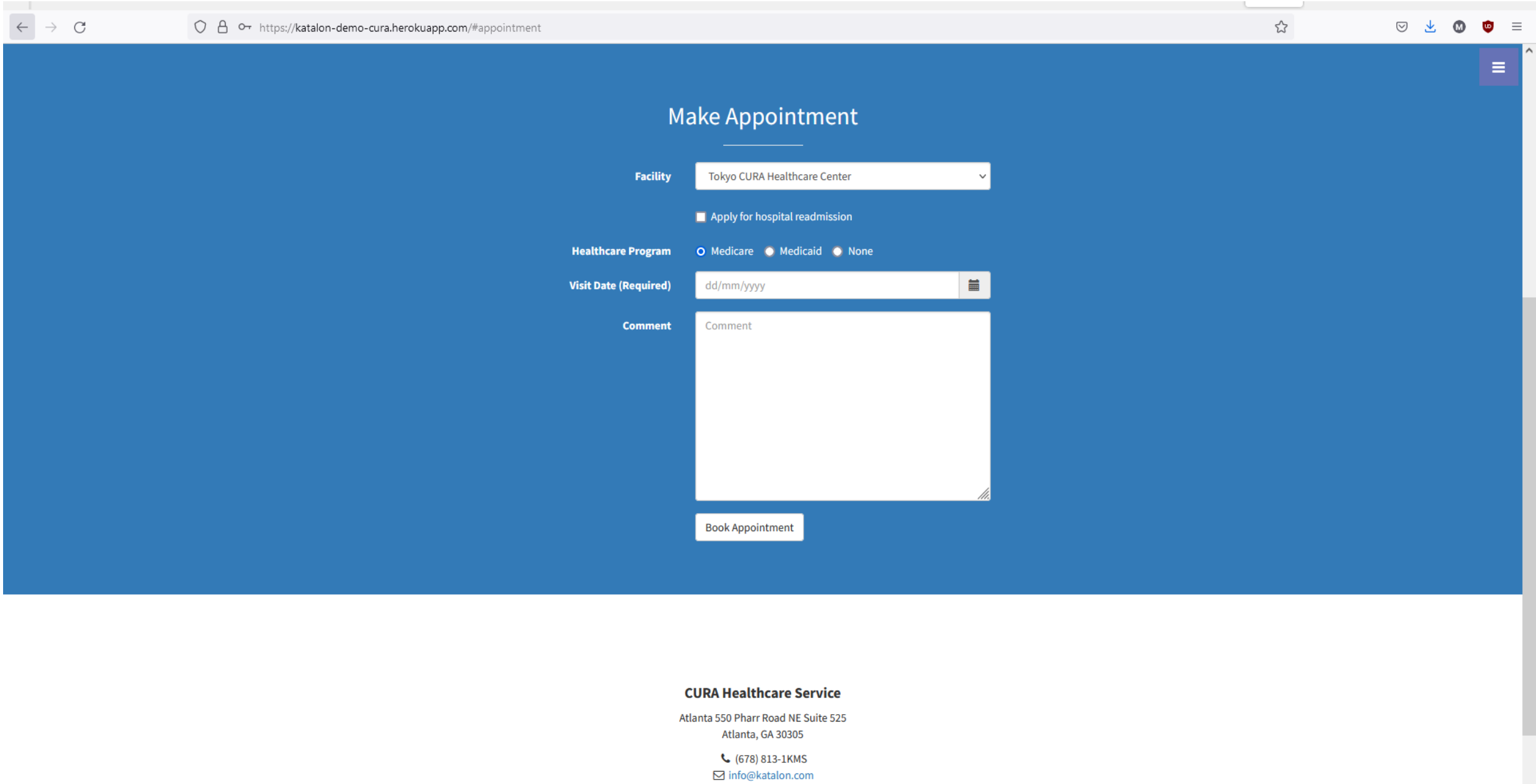
```
C:\Windows\system32>ride.py
```

Type ride.py



Webpage sample:




https://katalon-demo-cura.herokuapp.com/#appointment
Make Appointment
Facility
Tokyo CURA Healthcare Center
Apply for hospital readmission
Healthcare Program
Medicare
Medicaid
None
Visit Date (Required)
dd/mm/yyyy
Comment
Comment
Book Appointment
CURA Healthcare Service
Atlanta 550 Pharr Road NE Suite 525
Atlanta, GA 30305
(678) 813-1KMS
info@katalon.com

List Feature and Test Cases in Project

1. (+) Make Appointment
2. (+) Make Appointment with **Behavior-driven development**
3. (+) Make Appointment with Data Driven
4. (+) Make Appointment with Data Driven integrate with Excel file

Test data sections

Different sections in data	
Section	Used for
Settings	1) Importing test libraries resource files and variable files 2) Defining metadata for test suites and test cases
Variables	Defining variables that can be used elsewhere in the test data.
Test Cases	Creating test cases from available keywords.
Tasks	Creating tasks using available keywords. Single file can only contain either tests or tasks.
Keywords	Creating user keywords from existing lower-level keywords
Comments	Additional comments or data. Ignored by Robot Framework.

Keywords in Selenium Library

List of keywords that used in this project

Keywords	Deceptions
Open browser	Opens a given browser instance to the given url address
Close Browser	Closes browser window/tab
Click Element	Click the element identified by locator(element)
Input Text	Types the given text into the text field identified by locator
Select From List by Label	Selects options from selection list locator by given value
Choose File	Inputs the file_path into the file input field locator.
Element Text Should Be	Verifies that element locator contains exact the text expected

Sytax of Keywords in Table in Edit tab.

1st Column	2nd column	3rd column
Open browser	[link of website]	[webdriver] : chrome
Close Browser		
Click Element	Path of element (xpath,html, etc)	
Input Text	Path of element	Test data
Select From List by Label	Path of element	Value
Choose File	Path of element	Directory of test data file
Element Text Should Be	Path of element	text that should be

- For Choose File, recommend to save test data file into folder that save test suite. Then on third column, type ' ${CURDIR}/file name '

If want to find more library keywords, can refer to:

https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html#library-documentation-top

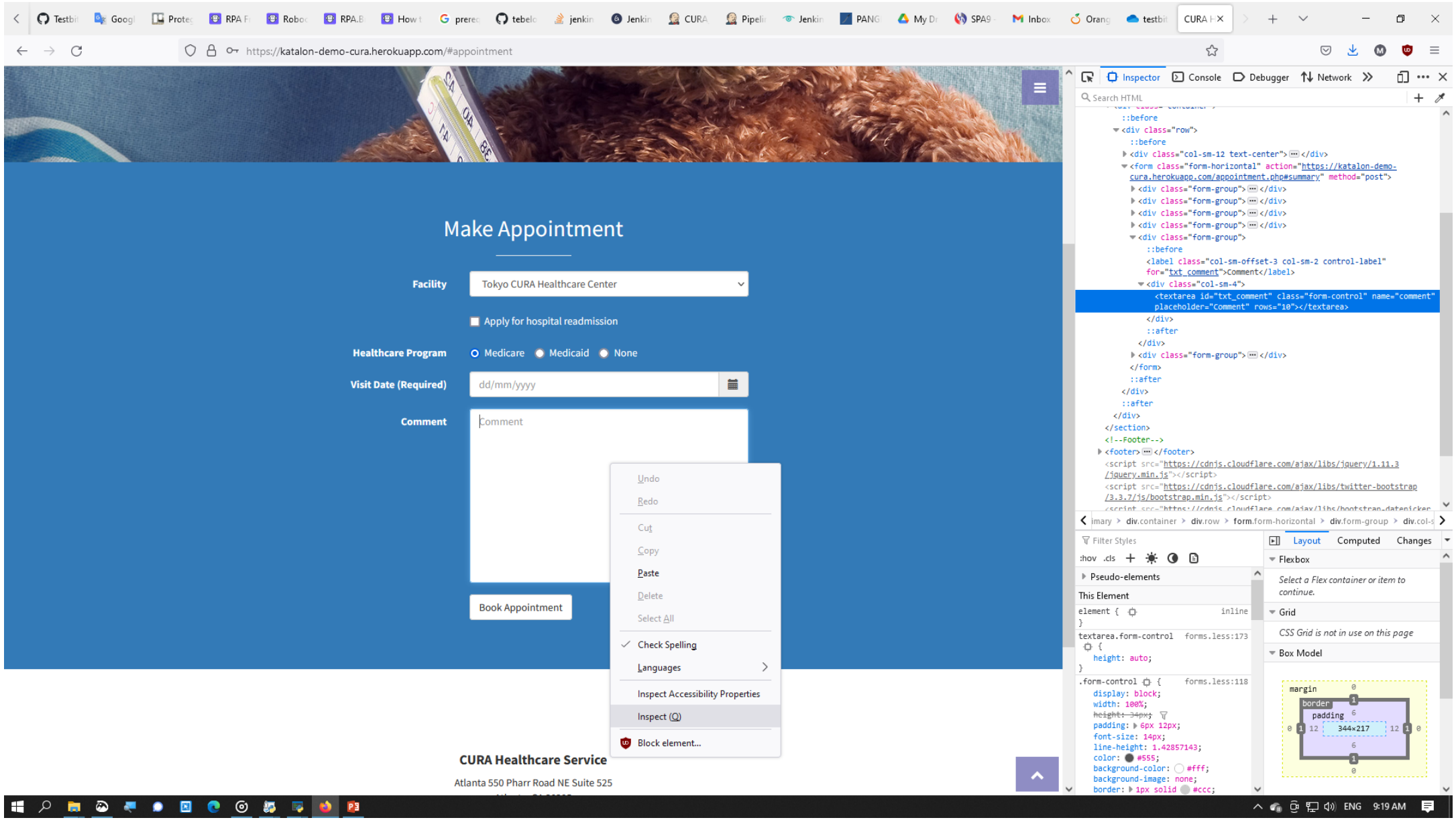
Copy Element (Xpath) in webpage

Steps:

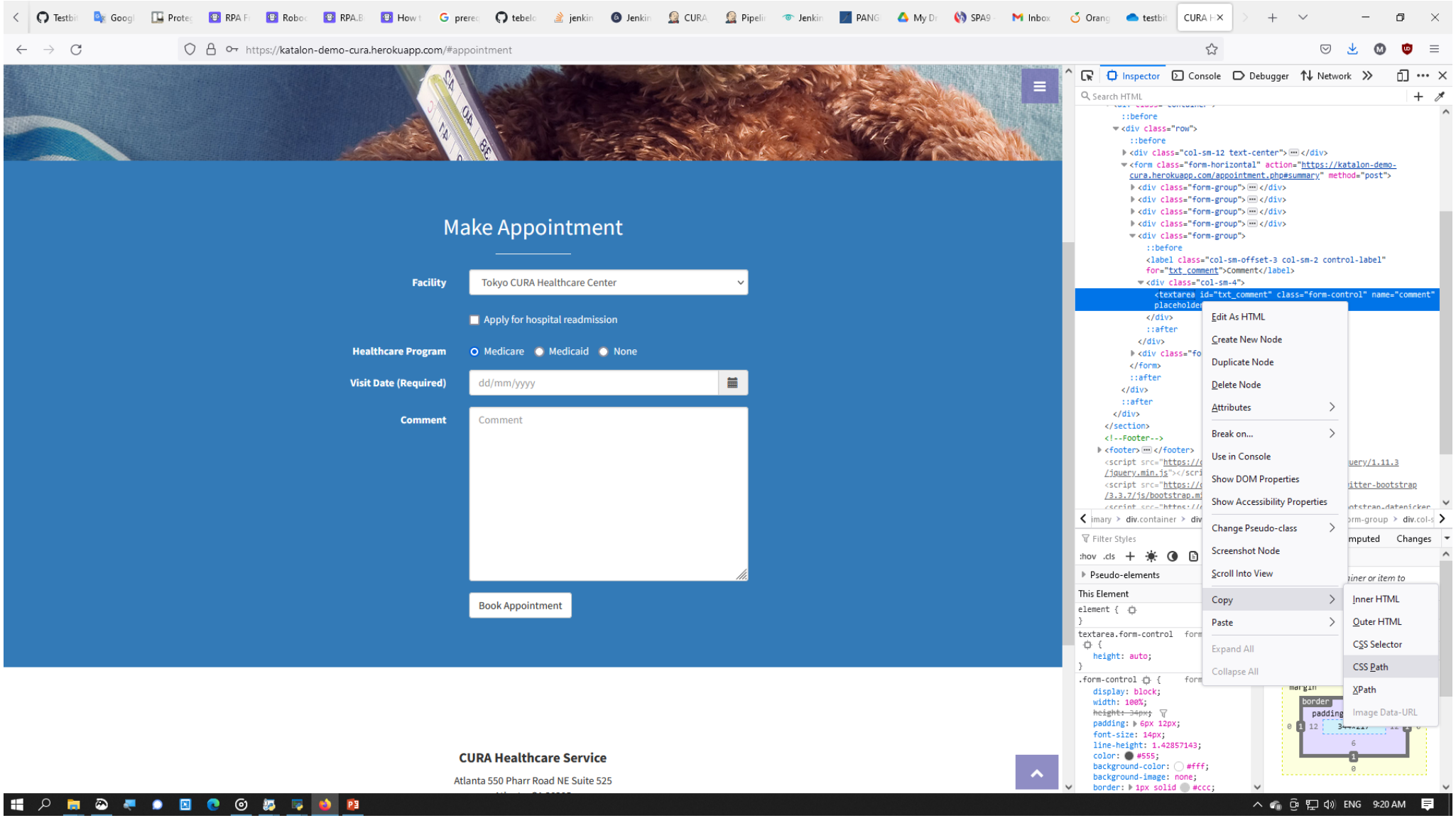
i. Choose element that want to get code

ii. Right click and then click on Inspect

iii. Inspect coding tab is shown on right (or left/below) and it will highlight code that represent


https://katalon-demo-cura.herokuapp.com/#appointment
Make Appointment
Facility
Tokyo CURA Healthcare Center
Apply for hospital readmission
Healthcare Program
Medicare
Medicaid
None
Visit Date (Required)
dd/mm/yyyy
Comment
Book Appointment
Undo
Redo
Cut
Copy
Paste
Delete
Select All
Check Spelling
Languages
Inspect Accessibility Properties
Inspect (Q)
Block element...
CURA Healthcare Service
Atlanta 550 Pharr Road NE Suite 525
Inspector
Console
Debugger
Network
Search HTML
<textarea id="txt_comment" class="form-control" name="comment" placeholder="Comment" rows="10"></textarea>
Layout
Computed
Changes
Flexbox
Select a Flex container or item to continue.
Grid
CSS Grid is not in use on this page
Box Model
margin
border
padding
344×217
ENG
9:19 AM

iv. Right click on the highlight code and click Copy -> Copy XPath



First testing

Simple Make Appointment

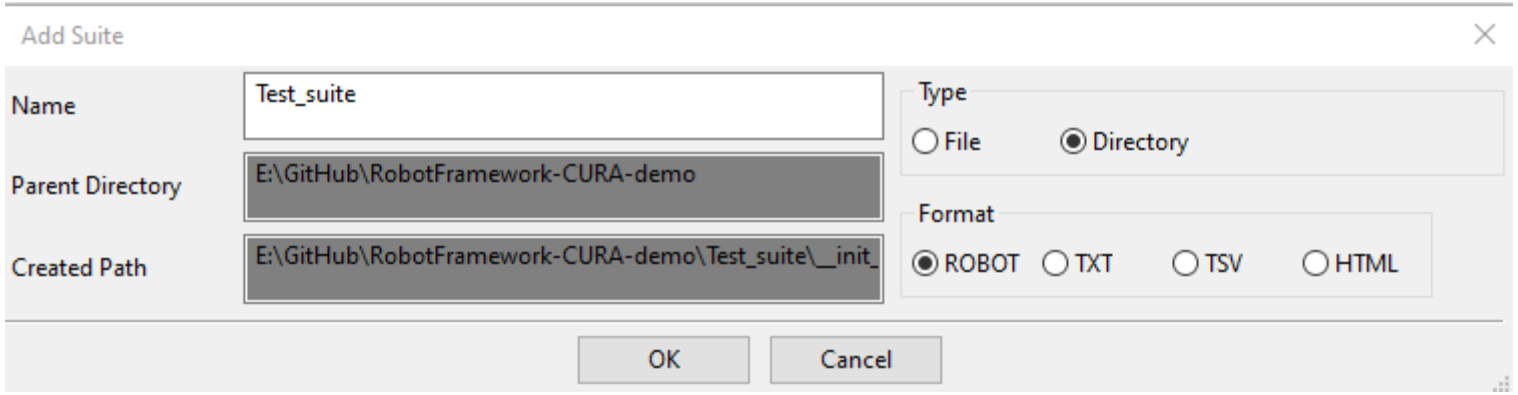
Test Steps for Make Appointment test case:

1. Open browser
2. Click ‘Make Appointment’ button
3. Log in using given username and password
4. Choose Facility
5. Tick ‘Apply for hospital readmission
6. Choose ‘Healthcare program
7. Put comment
8. Click ‘Book Appointment’ button
9. Validate ‘Appointment Confirmation’

Step 1: Create Directory/folder

i. Create new directory/folder (optional) for saving step definitions: Right click on main directory (on left side of RIDE) and choose ‘New Suite’.

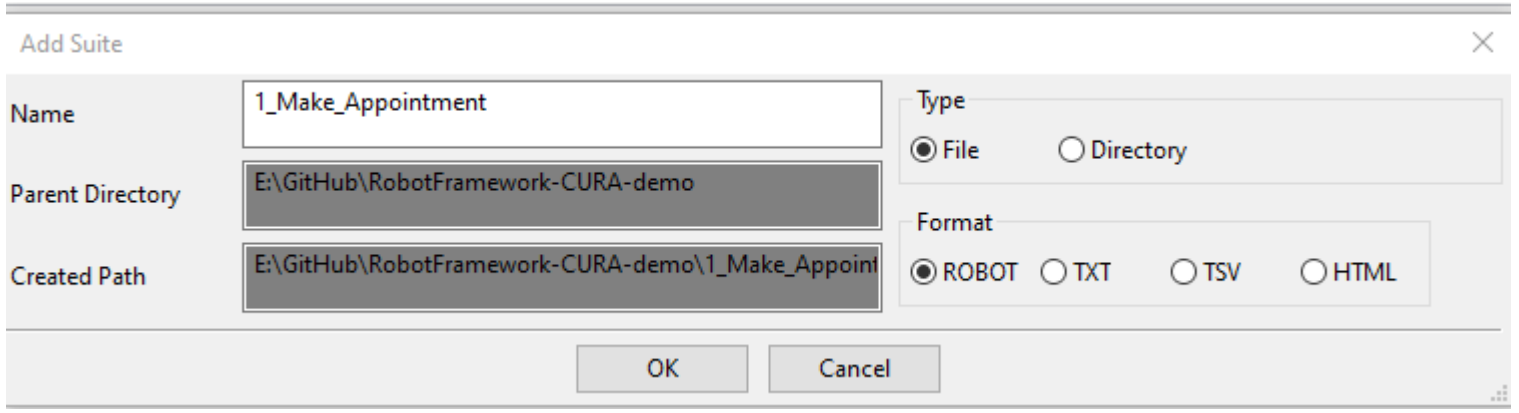
ii. In Add Suite window, change Type from ‘File’ into ‘Directory’. Then click ‘OK’. ‘Feature’ directory can be found under main directory



Step 2: Create Test Suite

i. Right click on Feature Directory and choose ‘New Suite’.

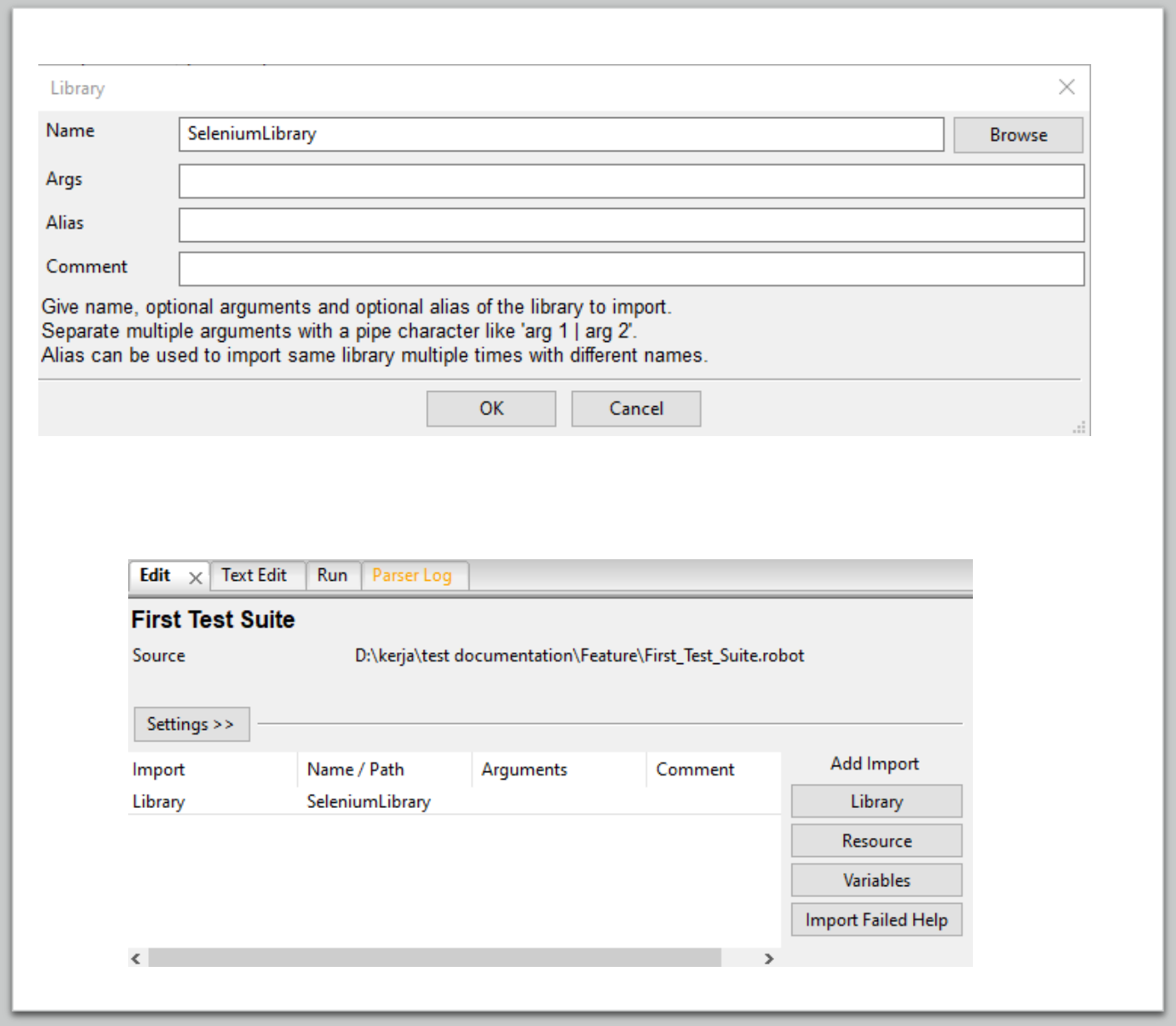
ii. In Add Suite window, named it as ‘1_Make_Appointment’ and then click OK. This test suite can be found under Feature.



Step 3: Import Selenium Library

i. Click on library under 'import' and import Selenium Library by typing: 'SeleniumLibrary'. Then click OK.

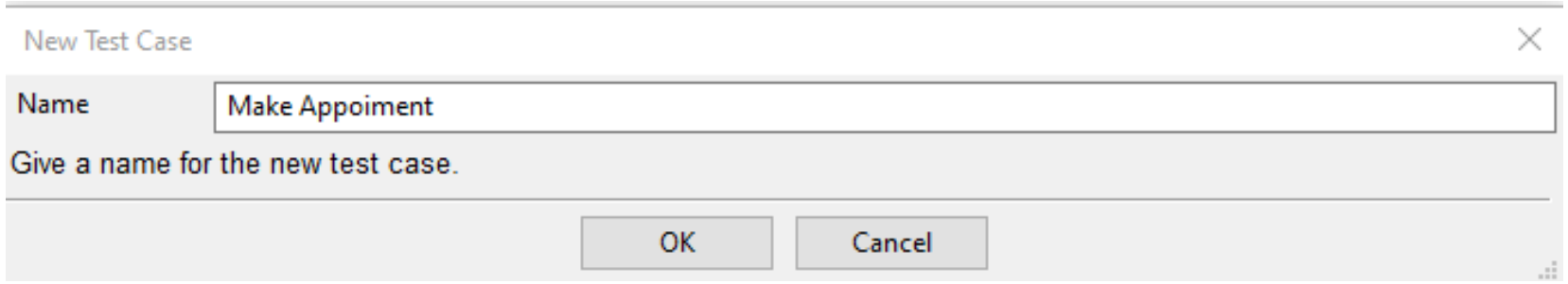
ii. On Edit Section (2nd picture), if colour of SeleniumLibrary is black, it means this library is valid. If colour is red, it means this library is invalid



Step 4: Create Test Case

i. Right click on main directory (on left side of RIDE) and choose ‘New Test Case’.

ii. In Add Suite window, named it as ‘Make Appointment’ and then click OK. This test case can be found under Test Suite.



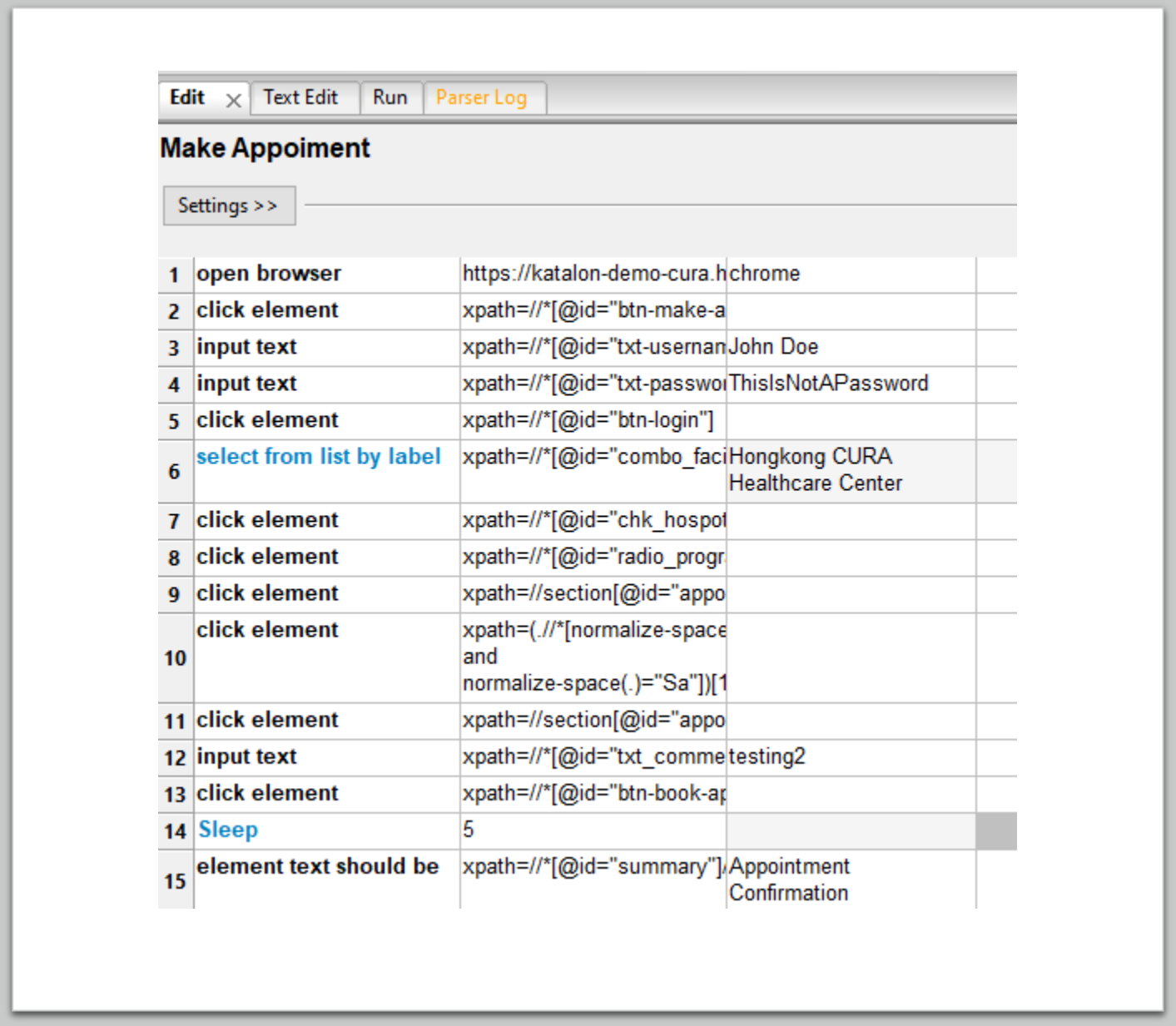
Step 5: Put test steps into Test Case

i. Click on test case and click Edit tab.

ii. In First column, put library keywords based on steps that have planned for this testing

iii. Copy element (XPath) of object inside webpage. Then put elements into second column

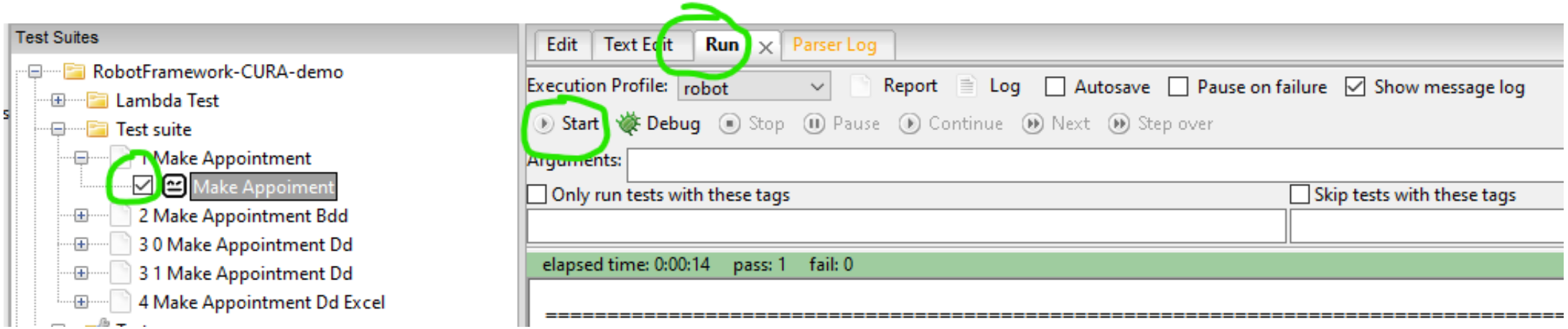
iv. Put value (only for steps that involve test data) into third column



1	open browser	https://katalon-demo-cura.h	chrome
2	click element	xpath=//*[@id="btn-make-a	
3	input text	xpath=//*[@id="txt-usernan	John Doe
4	input text	xpath=//*[@id="txt-passwo	ThisIsNotAPassword
5	click element	xpath=//*[@id="btn-login"]	
6	select from list by label	xpath=//*[@id="combo_faci	Hongkong CURA Healthcare Center
7	click element	xpath=//*[@id="chk_hospot	
8	click element	xpath=//*[@id="radio_progr	
9	click element	xpath=//section[@id="appo	
10	click element	xpath=(.//*[normalize-space and normalize-space(.)="Sa"])[1	
11	click element	xpath=//section[@id="appo	
12	input text	xpath=//*[@id="txt_comme	testing2
13	click element	xpath=//*[@id="btn-book-ap	
14	Sleep	5	
15	element text should be	xpath=//*[@id="summary"]	Appointment Confirmation

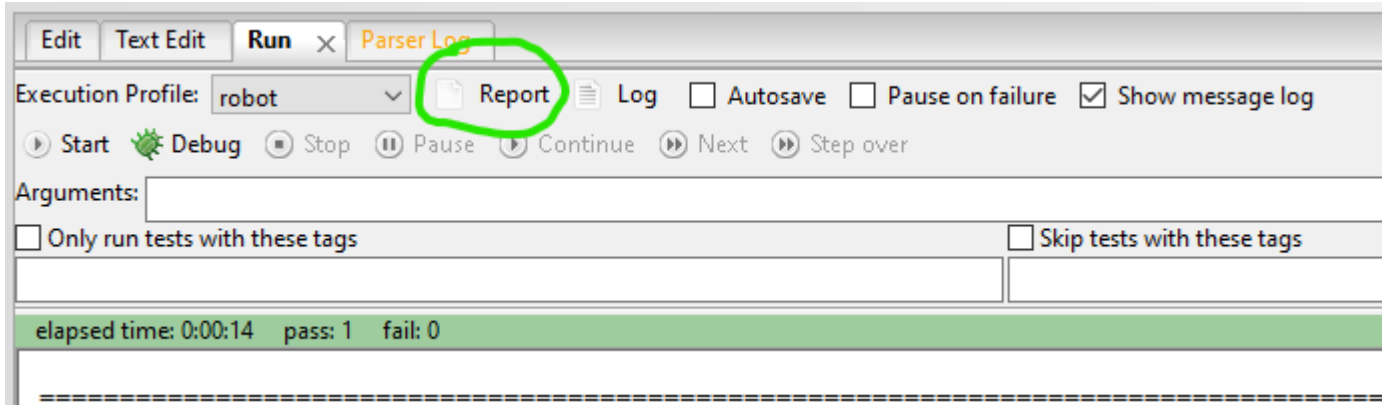
STEP 6: Run test case

i. Tick the test case you want to run.
ii. Go to ‘Run’ tab
iii. Click ‘Start’ robot button.

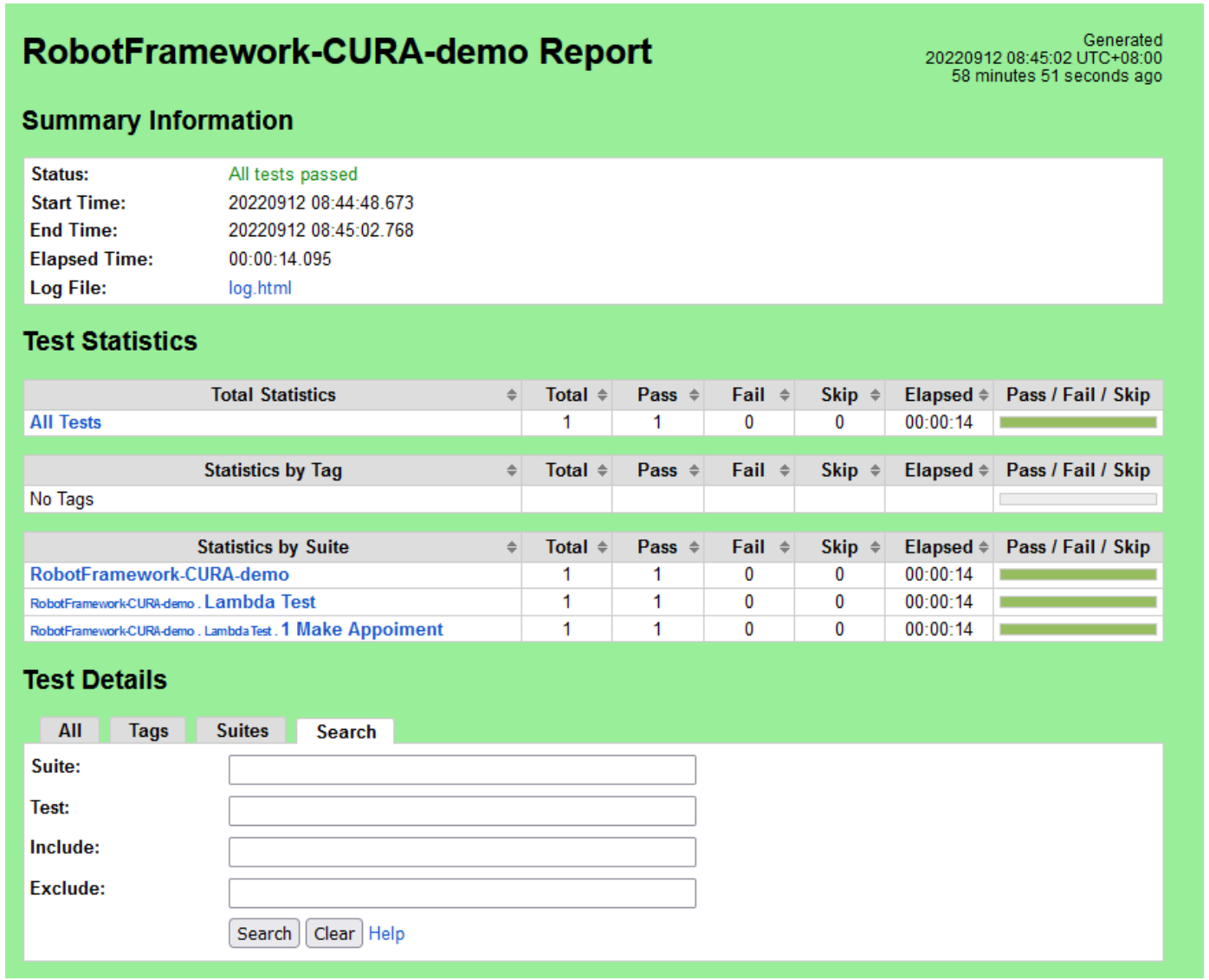


STEP 7: Test case report

- Click ***Report*** button on ‘Run’ tab to check the report.



Example of test case report



RobotFramework-CURA-demo Report

Generated
20220912 08:45:02 UTC+08:00
58 minutes 51 seconds ago

Summary Information

Status:	All tests passed
Start Time:	20220912 08:44:48.673
End Time:	20220912 08:45:02.768
Elapsed Time:	00:00:14.095
Log File:	log.html

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	1	1	0	0	00:00:14	

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags						

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
RobotFramework-CURA-demo	1	1	0	0	00:00:14	
RobotFramework-CURA-demo . Lambda Test	1	1	0	0	00:00:14	
RobotFramework-CURA-demo . Lambda Test . 1 Make Appoiment	1	1	0	0	00:00:14	

Test Details

All | Tags | Suites | Search

Suite:
Test:
Include:
Exclude:
Search Clear Help

REPORT

RobotFramework-CURA-demo Log

Generated
20220912 08:45:02 UTC+08:00
59 minutes 25 seconds ago

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	1	1	0	0	00:00:14	

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags						

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
RobotFramework-CURA-demo	1	1	0	0	00:00:14	
RobotFramework-CURA-demo.Lambda Test	1	1	0	0	00:00:14	
RobotFramework-CURA-demo.Lambda Test.1 Make Appoiment	1	1	0	0	00:00:14	

Test Execution Log

SUITE **RobotFramework-CURA-demo** 00:00:14.095

Full Name: RobotFramework-CURA-demo
Source: E:\GitHub\RobotFramework-CURA-demo
Start / End / Elapsed: 20220912 08:44:48.673 / 20220912 08:45:02.768 / 00:00:14.095
Status: 1 test total, 1 passed, 0 failed, 0 skipped

SUITE **Lambda Test** 00:00:14.065

Full Name: RobotFramework-CURA-demo.Lambda Test
Source: E:\GitHub\RobotFramework-CURA-demo\Lambda_Test
Start / End / Elapsed: 20220912 08:44:48.703 / 20220912 08:45:02.768 / 00:00:14.065
Status: 1 test total, 1 passed, 0 failed, 0 skipped

SUITE **1 Make Appoiment** 00:00:14.063

Full Name: RobotFramework-CURA-demo.Lambda Test.1 Make Appoiment
Source: E:\GitHub\RobotFramework-CURA-demo\Lambda_Test\1_make_appoiment.robot
Start / End / Elapsed: 20220912 08:44:48.705 / 20220912 08:45:02.768 / 00:00:14.063
Status: 1 test total, 1 passed, 0 failed, 0 skipped

TEST **Make Appoiment** 00:00:13.854

Full Name: RobotFramework-CURA-demo.Lambda Test.1 Make Appoiment.Make Appoiment
Start / End / Elapsed: 20220912 08:44:48.914 / 20220912 08:45:02.768 / 00:00:13.854
Status: PASS

SETUP	open_n_login.**Open_n_login**	00:00:05.877
KEYWORD	SeleniumLibrary.**Select From List By Label** xpath=//*[@id="combo_facility"], Hongkong CURA Healthcare Center	00:00:00.102
KEYWORD	SeleniumLibrary.**Click Element** xpath=//*[@id="chk_hospotal_readmission"]	00:00:00.056
KEYWORD	SeleniumLibrary.**Click Element** xpath=//*[@id="radio_program_medicaid"]	00:00:00.043
KEYWORD	SeleniumLibrary.**Click Element** xpath=//section[@id="appointment"]/div/div/form/div[4]/div/div/div/span	00:00:00.062
KEYWORD	SeleniumLibrary.**Click Element** xpath=(.//*[normalize-space(text()) and normalize-space(.)="Sa"])[1]/following::td[25]	00:00:00.062
KEYWORD	SeleniumLibrary.**Click Element** xpath=//section[@id="appointment"]/div/div/form/div[4]/div/div/div/span	00:00:00.049
KEYWORD	SeleniumLibrary.**Input Text** xpath=//*[@id="txt_comment"], testing2	00:00:00.091
KEYWORD	SeleniumLibrary.**Click Element** xpath=//*[@id="btn-book-appointment"]	00:00:00.340
KEYWORD	BuiltIn.**Sleep** 5	00:00:05.001
KEYWORD	SeleniumLibrary.**Element Text Should Be** xpath=//*[@id="summary"]/div/div/div[1]/h2, Appointment Confirmation	00:00:00.027
TEARDOWN	SeleniumLibrary.**Close Browser**	00:00:02.135

To make testing process become proper, advance and easily, there are several features/setting that we will learn:

- Test Setup
- Test Teardown
- Variable
- Keywords File
- Resource File

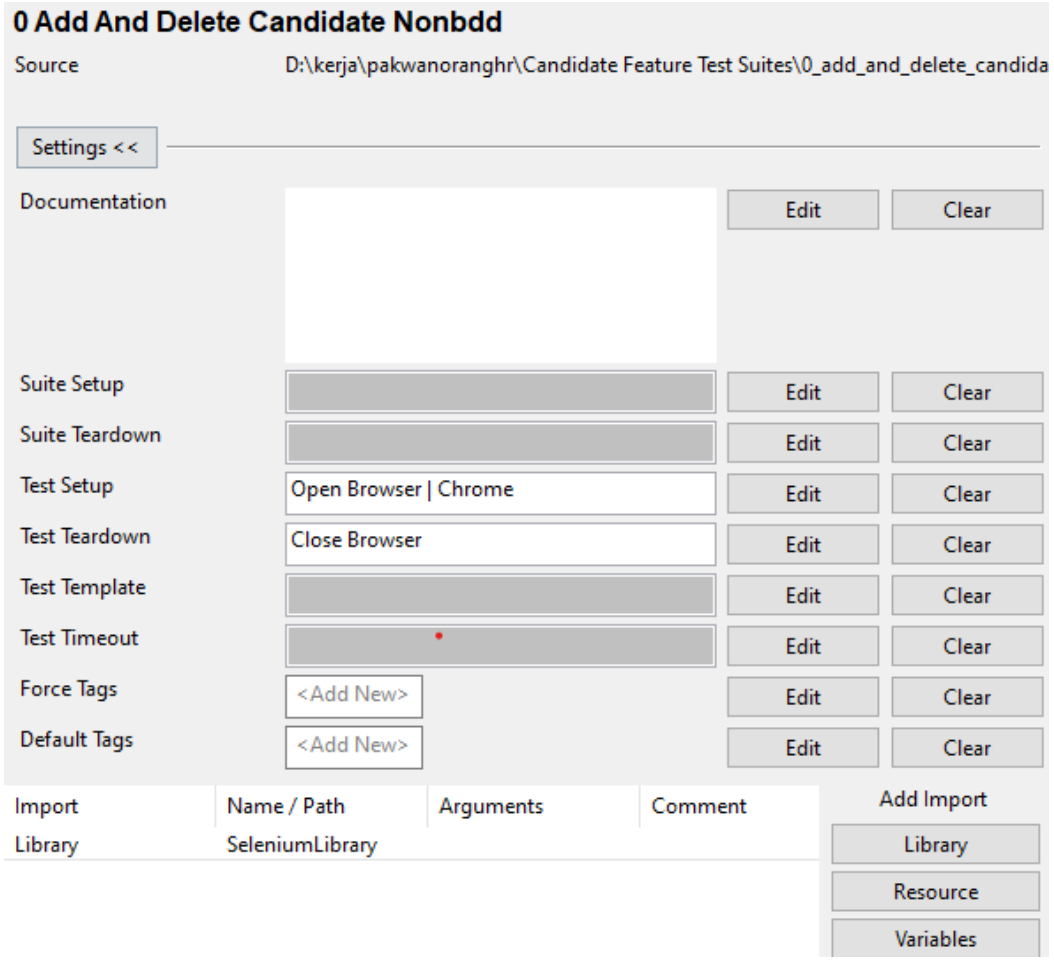
Test Setup and Teardown

Test Setup and Test Teardown

Test Setup: keyword/step that is executed before a test case(s)

Test Teardown: Keyword/step that is executed after a test case (s)

- Both can be set in Test Suite.
- Steps
  - Go to test suite in Edit tab
  - Click Setting
  - Find Setup and Teardown and click 'Edit'
  - Put keyword/step



- But only one keyword/step can be set for both.
- To have keywords/steps in setup and teardown, keywords file should be created and set to both

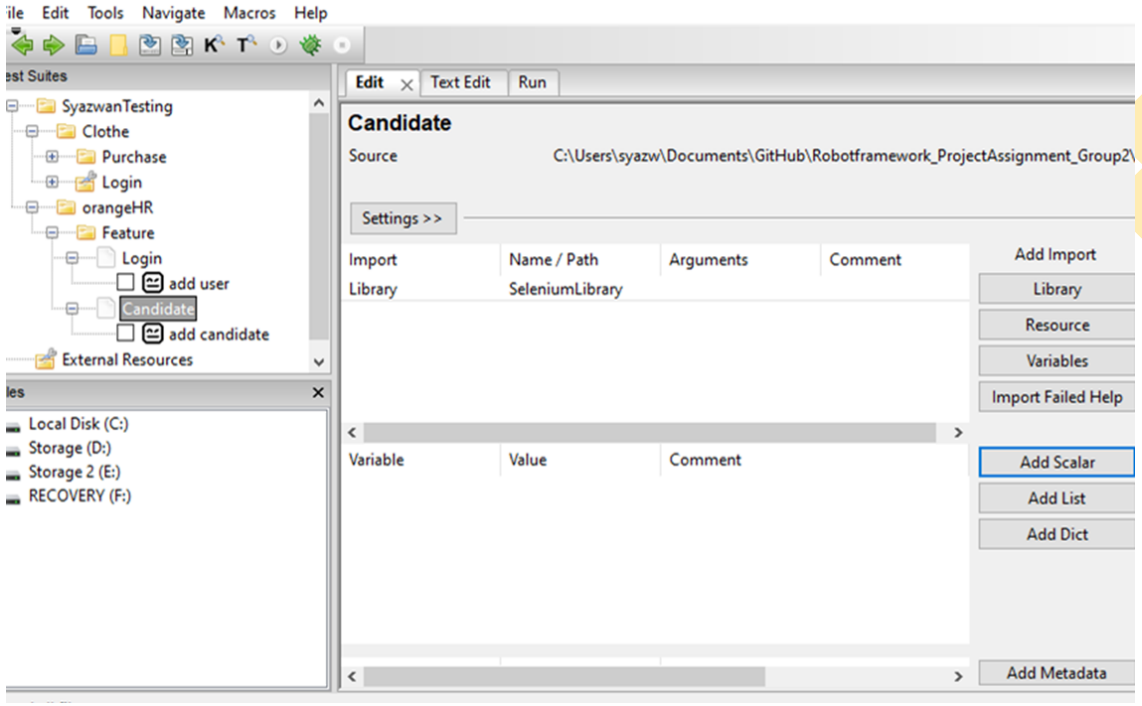
Variable, Keywords File and Resource File

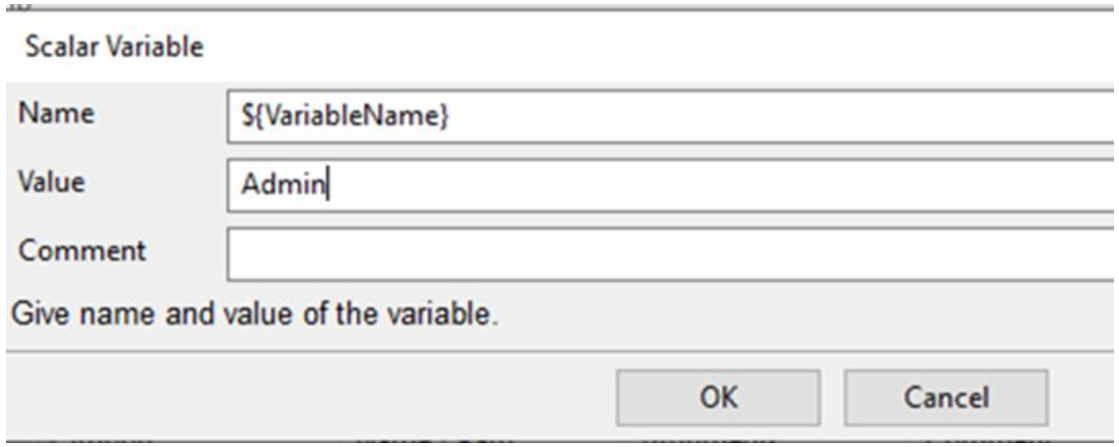
Steps (using edit tab)

i. Navigate to Test Suit

ii. Click on ‘Add Scalar’ on right bottom
(1st picture)

iii. Add Scalar window is popup, write variable name inside {} and put value that you want
(2nd picture)





iv. Now go to test case

v. To add variable, write ${Variablename} on third column as shown below:

3	input_text	xpath=//*[@id="txtUsernam	${VariableName}

*If colour of Variable is green, it means this variable is existing. But if its colour is light purple/pink, it means this variable is not existing.

Keywords File

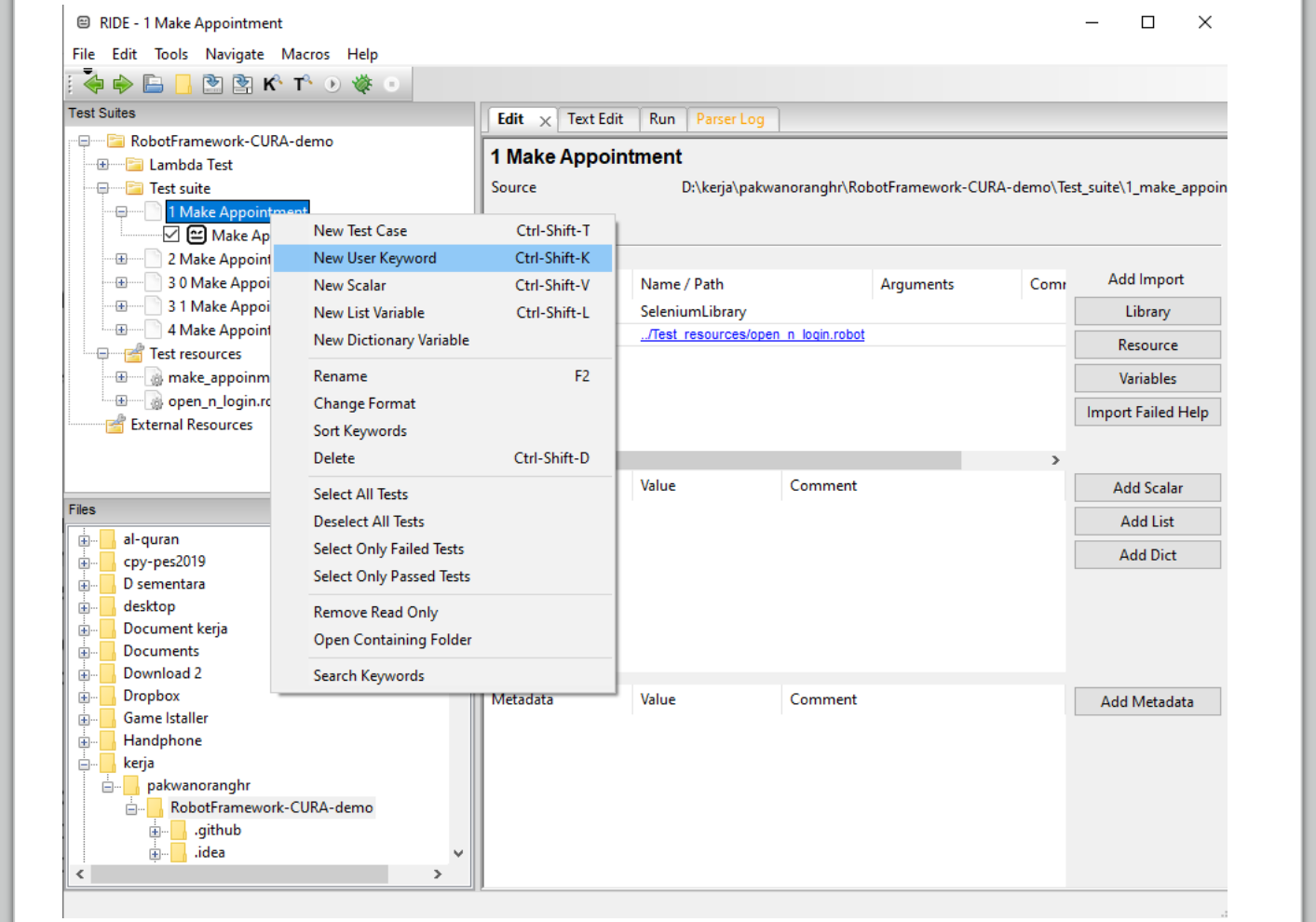
Combining existing keywords/test steps together

Steps:

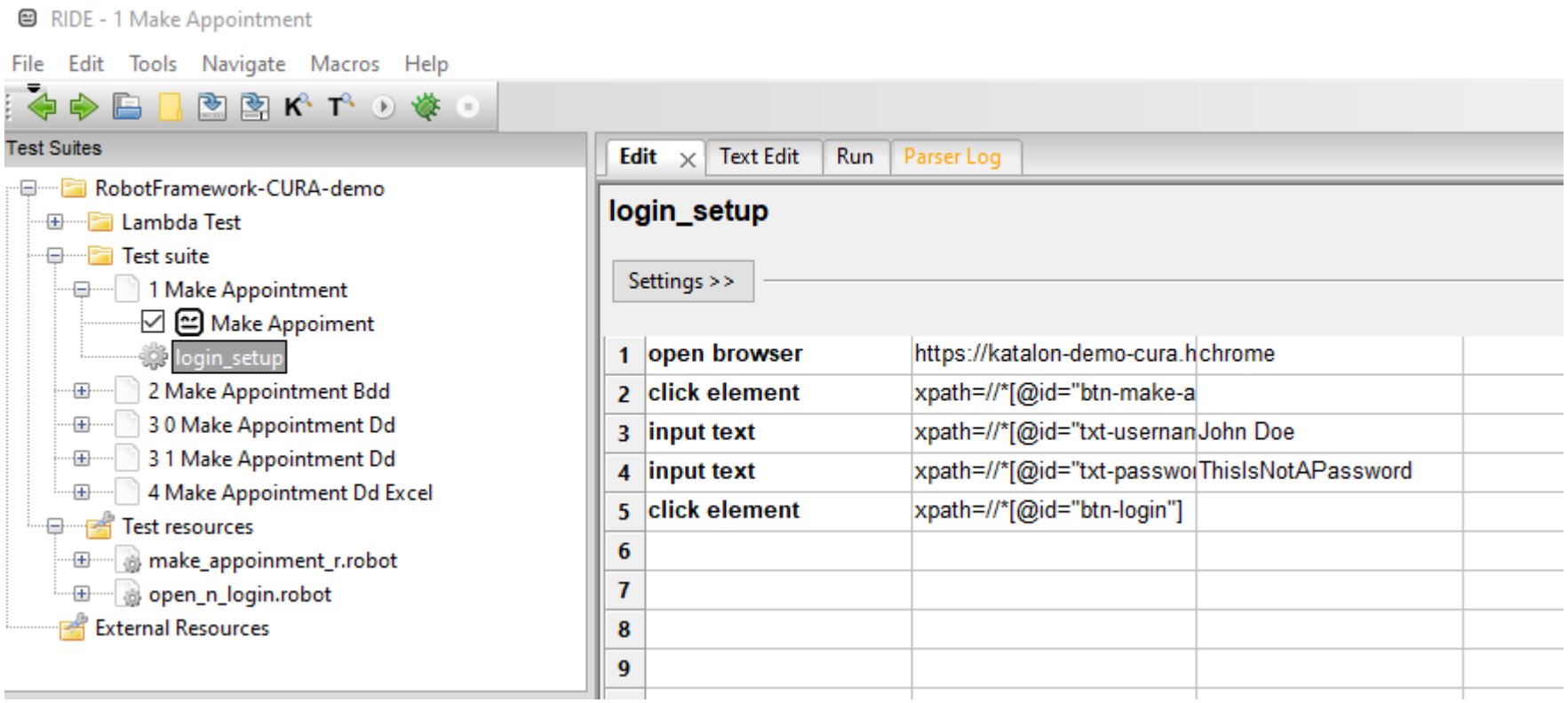
i. Select any test case inside test suite. Inside test case, there are many keywords/test steps.

ii. Main objective for this is creating keywords file so it can be set on Test Setup.

iii. Select on test suite and then right click-> New User Keywords to create keywords file

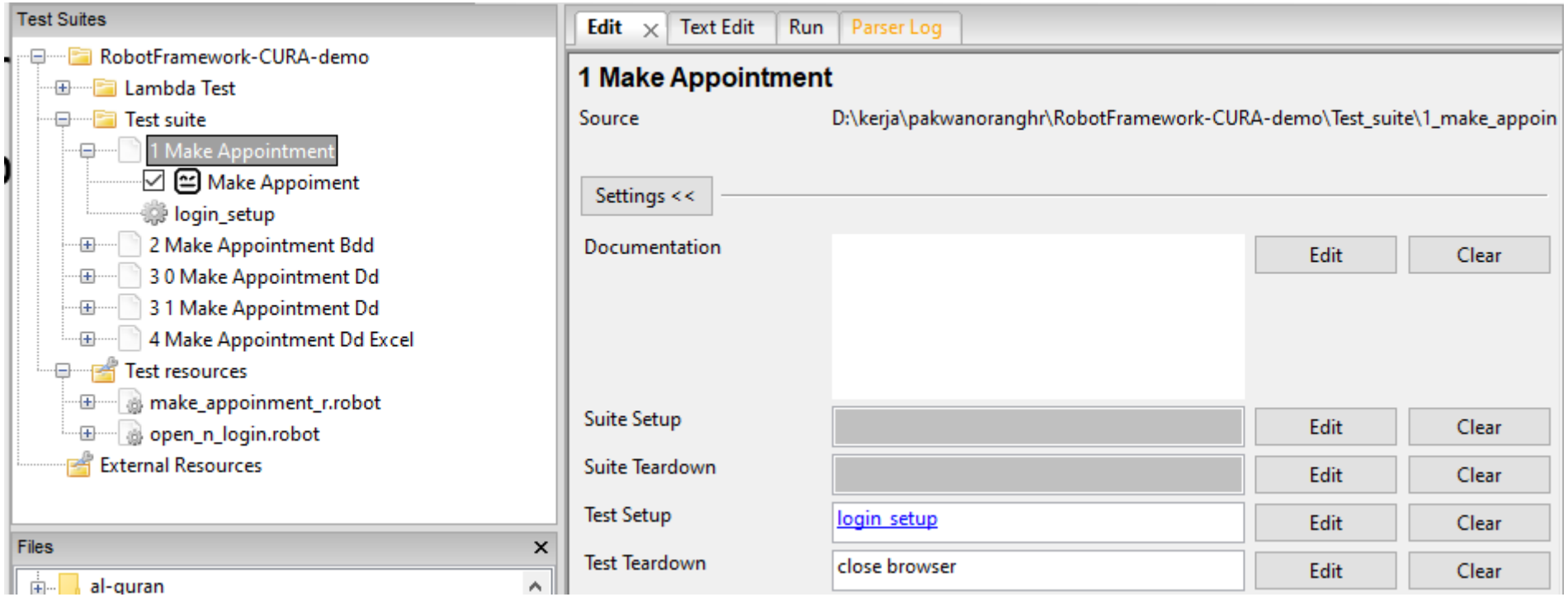


iv. Inside keywords file, copy/write keywords/test steps



v. After that, go to login test cases and delete (if you want).

vi. Go to Test Suite -> Setting. And then put keywords file name on Test Setup



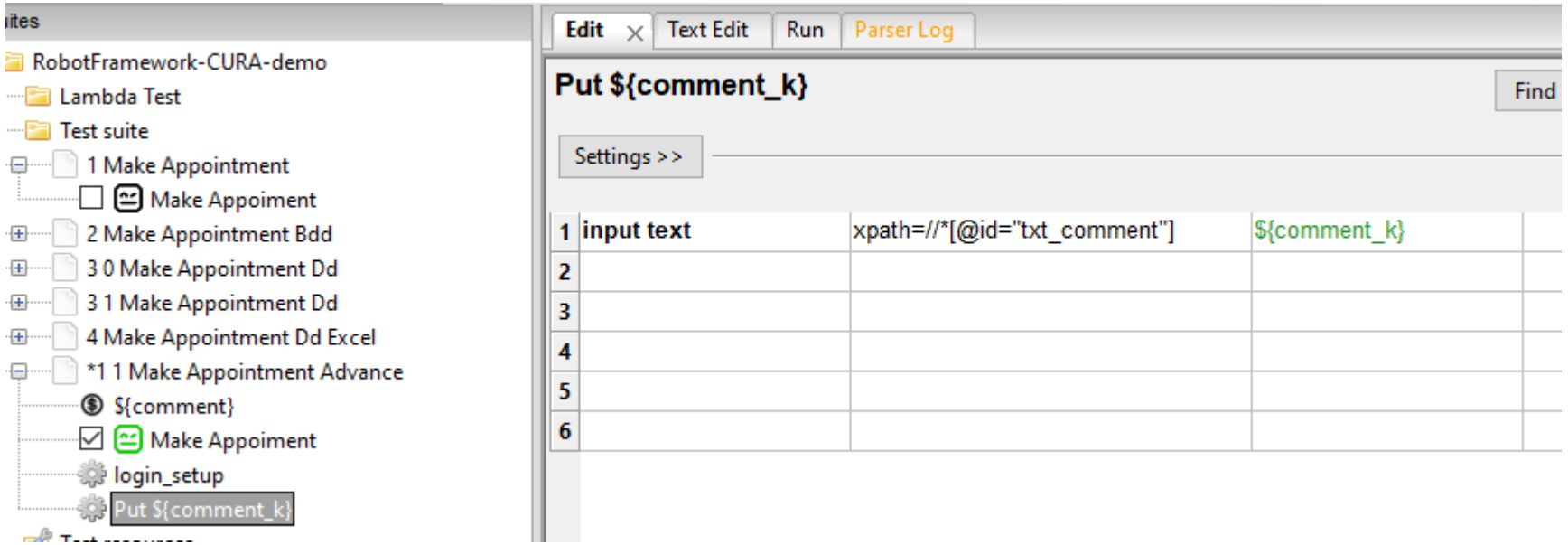
vii. This test case can be run

There is Keywords file that can link with variables.

i. Create keywords file and name it with any name include the variable

Example: Put comment

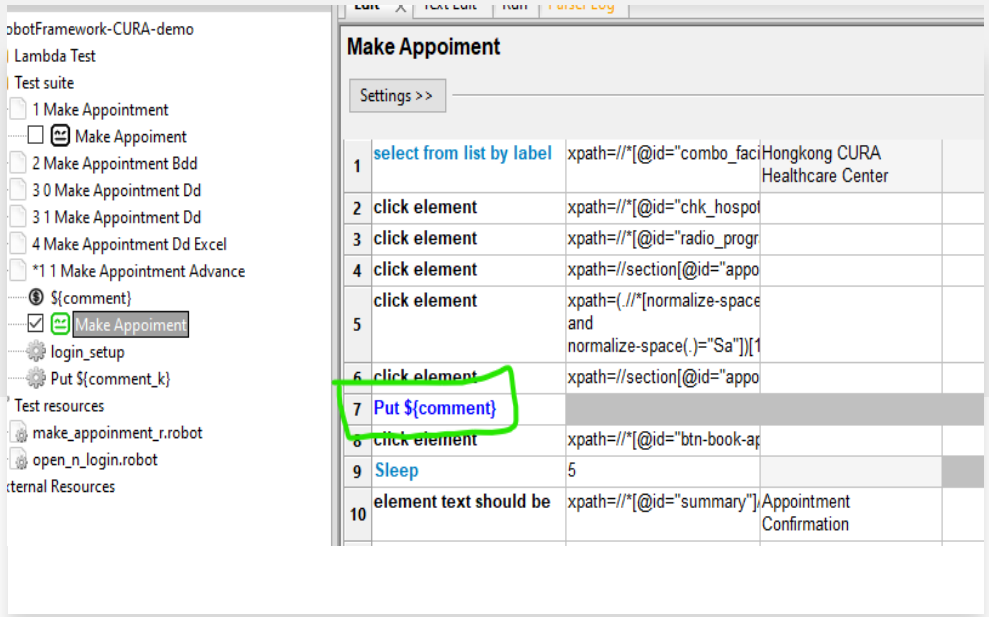
ii. Put keywords/steps that related to those variable. After that, put variable names on third column and row that related



iii. Go to test case Add Candidate, replace those two input steps with:

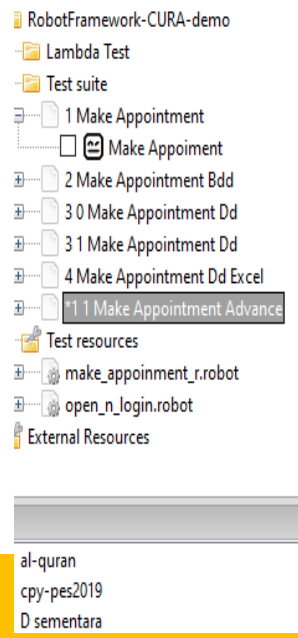
Put ${comment}

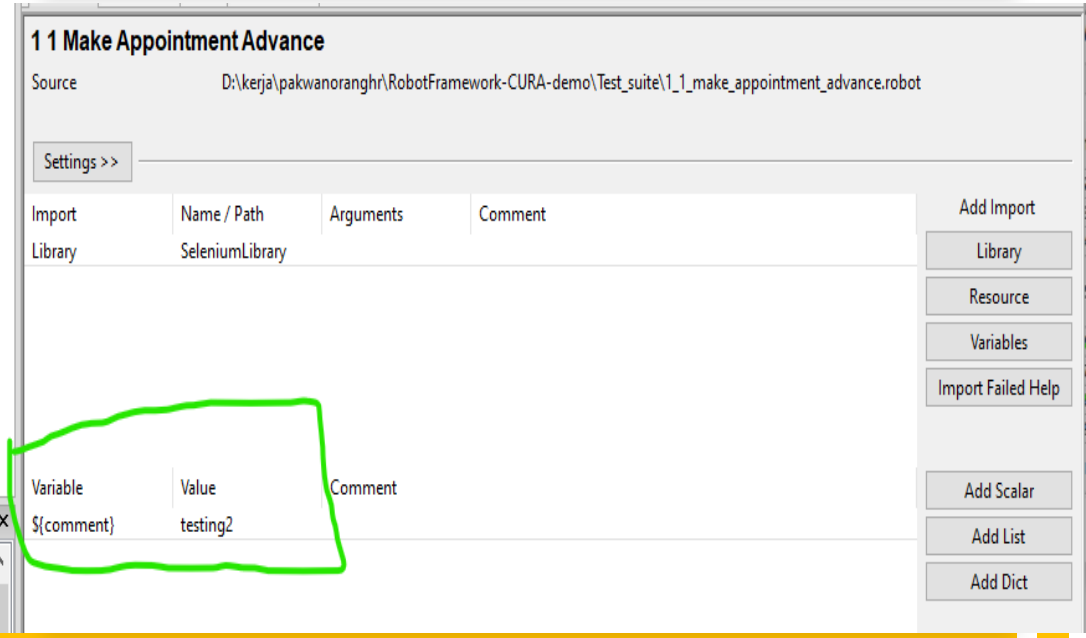
Refer 1st picture



Note: Make sure name of variables are same with variable in test suite, not with keywords file name (refer 2nd picture)

iv. Test case can be run.





For more detail by looking test suite that used setup, teardown, variable and keyword file, can refer to '1 1 Make Appointment' test suite

Resource File

- Keywords and variables from test suite A cannot be used on another test suits.
- Resource files provide a mechanism (keywords and variable) for sharing them

Steps:

i. Create resource file by right click on main folder (or other folder/directory) and click ‘New Resource’.

ii. ‘New resource file’ window will be popped up. Give a name and click ‘OK’.

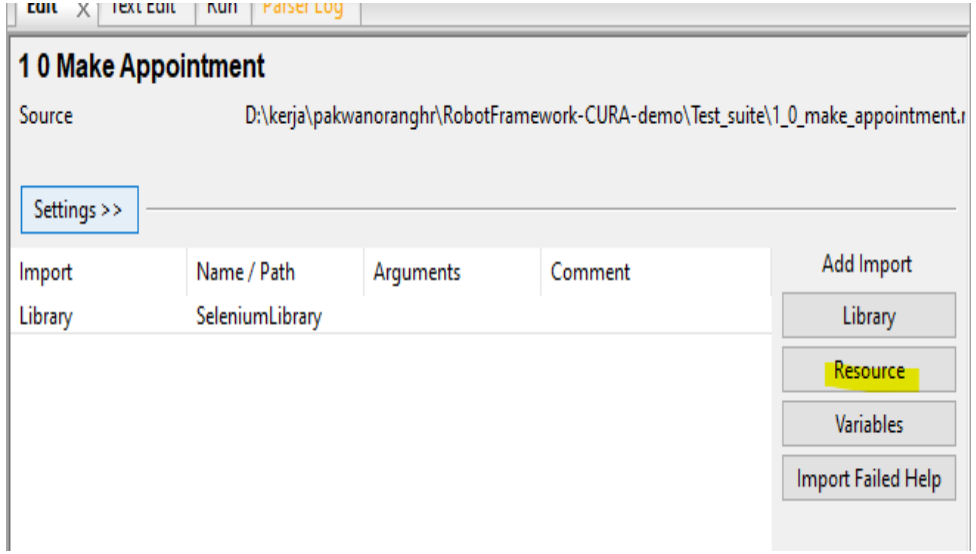
iii. Click on that scenarios resource file

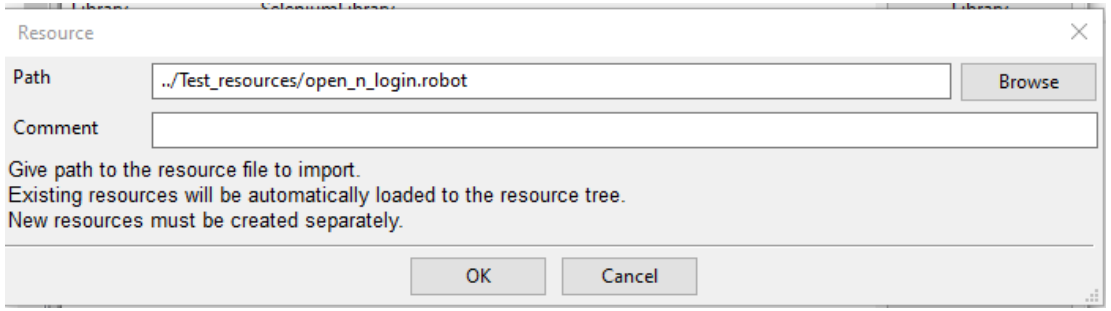
iv. You can create variables or keywords inside resource file.

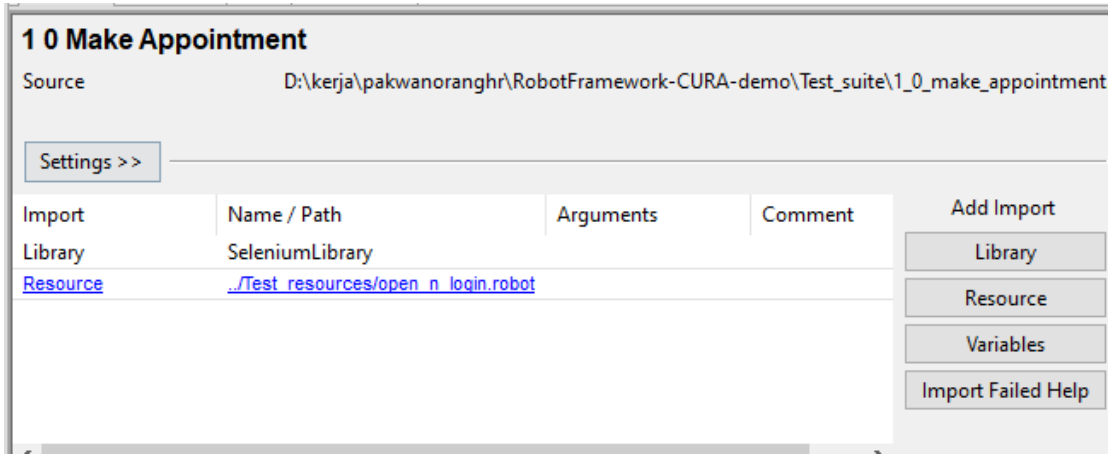
v. Go to any test suite. Then, click ‘Resource’ under Add Import

vi. Resource’ window will be popped up. Browse scenario resource file as shown in below and click OK.

vii. The list of added library and resource is shown. If library and resource are invalid, colour of library and resource word become red.







Behavior-driven development

Behavior-driven development

- BDD is method when an application is documented and designed around the behavior a user expects to experience when interacting with it.
- BDD helps to avoid bloat, excessive code, unnecessary features or lack of focus from developer and others.

Step 1: Create ‘Open n login’ and ‘Make_appointment’ resources files

i. Create folder that can store resources file (refer page 34)

ii. Create ‘Open n login’ and ‘make appointment’ resources files under this folder (refer page 58)

iii. In ‘Open n login’ resources file, create new user keyword: open_login

iv. Write steps in open_login user keyword as shown in below:

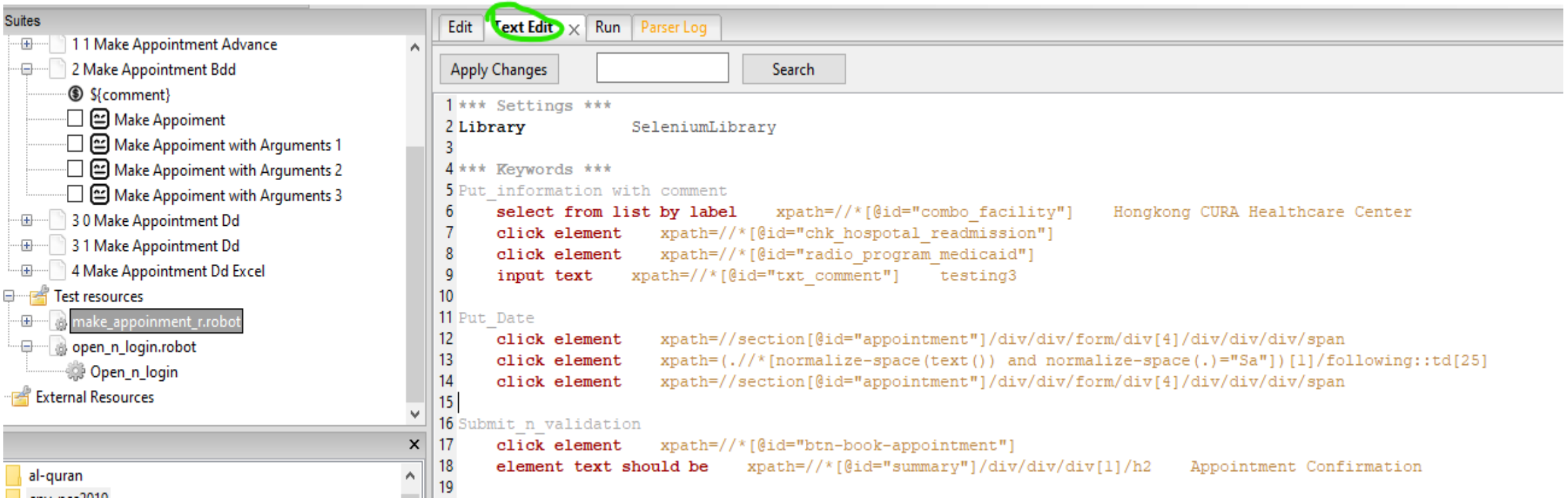
Open_n_login

Settings >>

1	**open browser**	https://katalon-demo-cura.herokuapp.com	chrome
2	**click element**	xpath=//*[@id="btn-make-appointment"]	
3	**input text**	xpath=//*[@id="txt-username"]	John Doe
4	**input text**	xpath=//*[@id="txt-password"]	ThisIsNotAPassword
5	**click element**	xpath=//*[@id="btn-login"]	
6			
7			
8			
9			
10			

v. In ‘make appointment’ resources file, create new user keyword: Put_information with comment, Put_Date and Submit_n_validation

vi. There are other method to write test steps. Click on In ‘make appointment’ resources file, then click ‘Test Edit’ tab. You can write it by using robot framework syntax. You can write test steps as shown in figure below:



Step 2: Create BDD Test Suite

i. Create new test suite under ‘Test_suite’ folder and naming it as ‘2 make appointment bdd’. (refer page 35 and 36)

ii. In test suite, import Open n login’ and ‘make appointment’ resources files. (Refer page 59)

iii. Then create test case (refer page 37)

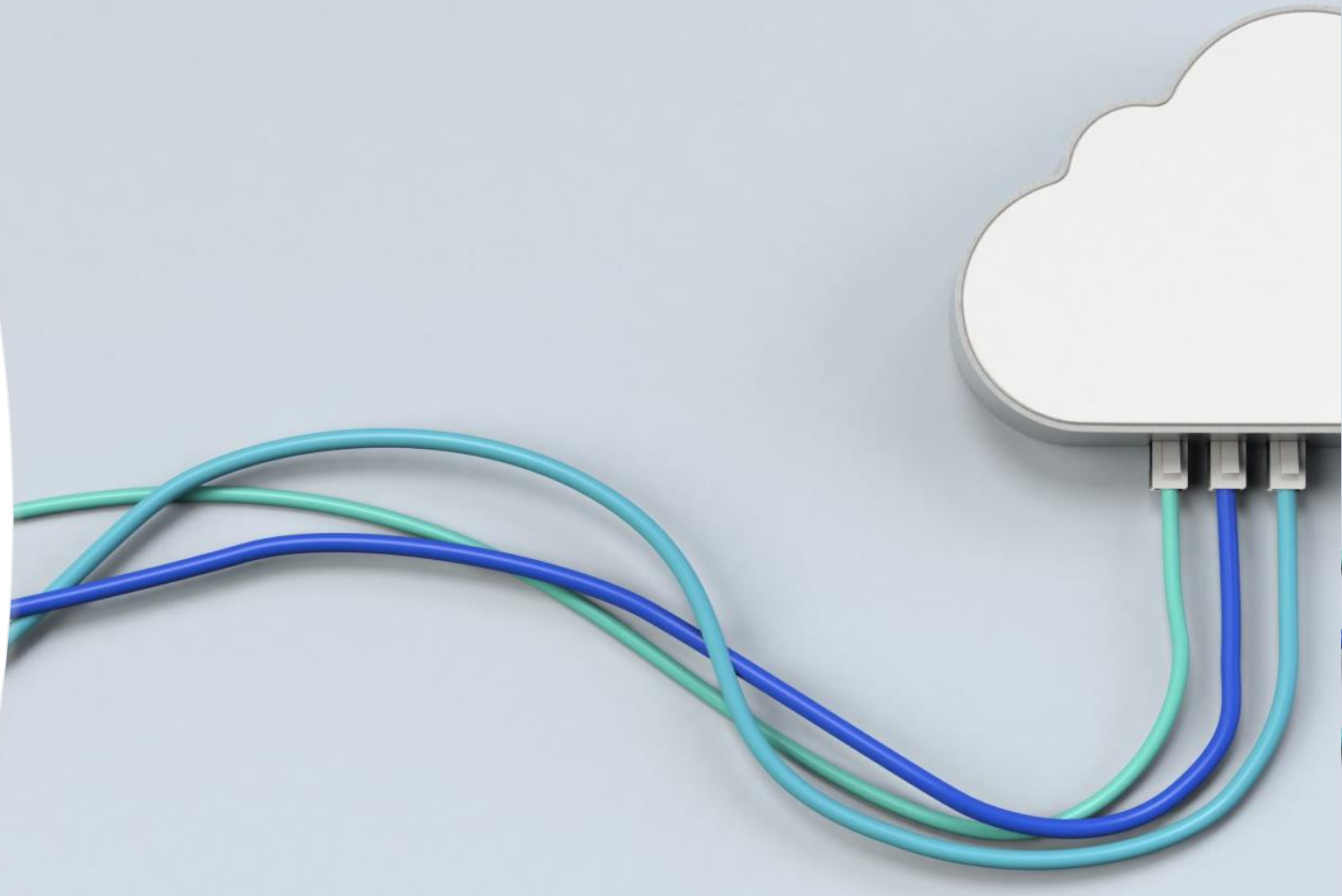
iv. Inside test case, you can put name of keywords as shown in figure below:

```
1  *** Settings ***
2  Test Setup         Open_n_login
3  Test Teardown      close browser
4  Library            SeleniumLibrary
5  Resource           ../Test_resources/open_n_login.robot
6  Resource           ../Test_resources/make_appoinment_r.r
7
8  *** Variables ***
9  ${comment}         comment_aggl
10
11 *** Test Cases ***
12 Make Appoiment
13     When Put_information with comment
14     And Put_Date
15     Then Submit_n_validation
16
```

v. After than, go to ‘Run’ tab and click ‘start’ to execute test case

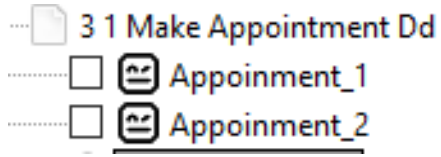
Data Driven

*Learn creating keywords and variable first



Steps:

1. Create test suite for data driven testing (e.g.: named as make appointment dd)
2. Create test cases that represent data. For this example, create three test cases



3. Click on test case and click Edit tab. Inside test cases, put data value for testing. For example, each test cases have two data value such as facility and comment. Then, put data value to another test cases

Appoinment_1

Settings >>

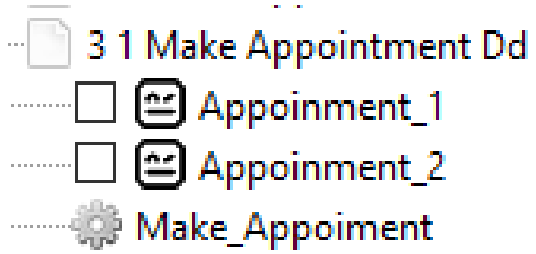
1	Hongkong CURA Healthcare Center	comment 1
2		
3		
4		
5		
6		

Appoinment_2

Settings >>

1	Tokyo CURA Healthcare Center	comment 2
2		
3		

4. On test suite, create Keywords (e.g.: named as make_appoiment)



5. Click on this test suite and then click on Text Edit tab.

6. Under **Setting**, type *make_appoiment*(name for keyword)

```
*** Settings ***
Test Setup        Open_n_login
Test Teardown     close browser
Test Template     Make_Appoiment
Library           SeleniumLibrary
Resource          ../Test_resources/open_n_login.robot
Resource          ../Test_resources/make_appoinment_r.robot

*** Test Cases ***
Appoinment_1
    Hongkong CURA Healthcare Center    comment 1

Appoinment_2
    Tokyo CURA Healthcare Center    comment 2

```

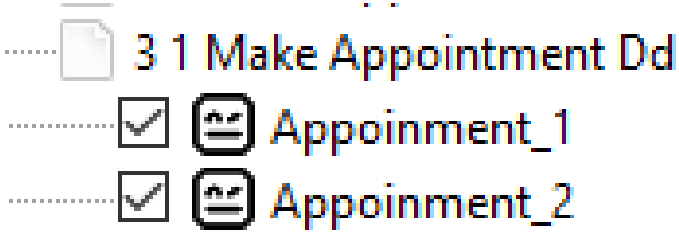
7. Under Keyword, type *[Arguments] name of variables* that will be used as following:

```
*** Settings ***
Test Setup        Open_n_login
Test Teardown     close browser
Test Template     Make_Appoiment
Library           SeleniumLibrary
Resource          ../Test_resources/open_n_login.robot
Resource          ../Test_resources/make_appoinment_r.robot

*** Test Cases ***
Appoinment_1
    Hongkong CURA Healthcare Center    comment 1

Appoinment_2
    Tokyo CURA Healthcare Center    comment 2

*** Keywords ***
Make_Appoiment
    [Arguments]    ${facility}    ${comment}
```

8. Then, type steps (either on Edit or Test Edit tab) in keyword. For value, put name of variables on third columns for some steps as shown below:

```
*** Keywords ***
Make_Appoiment
    [Arguments]    ${facility}    ${comment}
    select from list by label    xpath=//*[@id="combo_facility"]    ${facility}
    click element    xpath=//*[@id="chk_hospotal_readmission"]
    click element    xpath=//*[@id="radio_program_medicaid"]
    click element    xpath=//section[@id="appointment"]/div/div/form/div[4]/div/div/div/span
    click element    xpath=(.//*[normalize-space(text()) and normalize-space(.)="Sa"])[1]/following::td[25]
    input text    xpath=//*[@id="txt_comment"]    ${comment}
    sleep    3
    click element    xpath=//*[@id="btn-book-appointment"]
    element text should be    xpath=//*[@id="summary"]/div/div/div[1]/h2    Appointment Confirmation
```

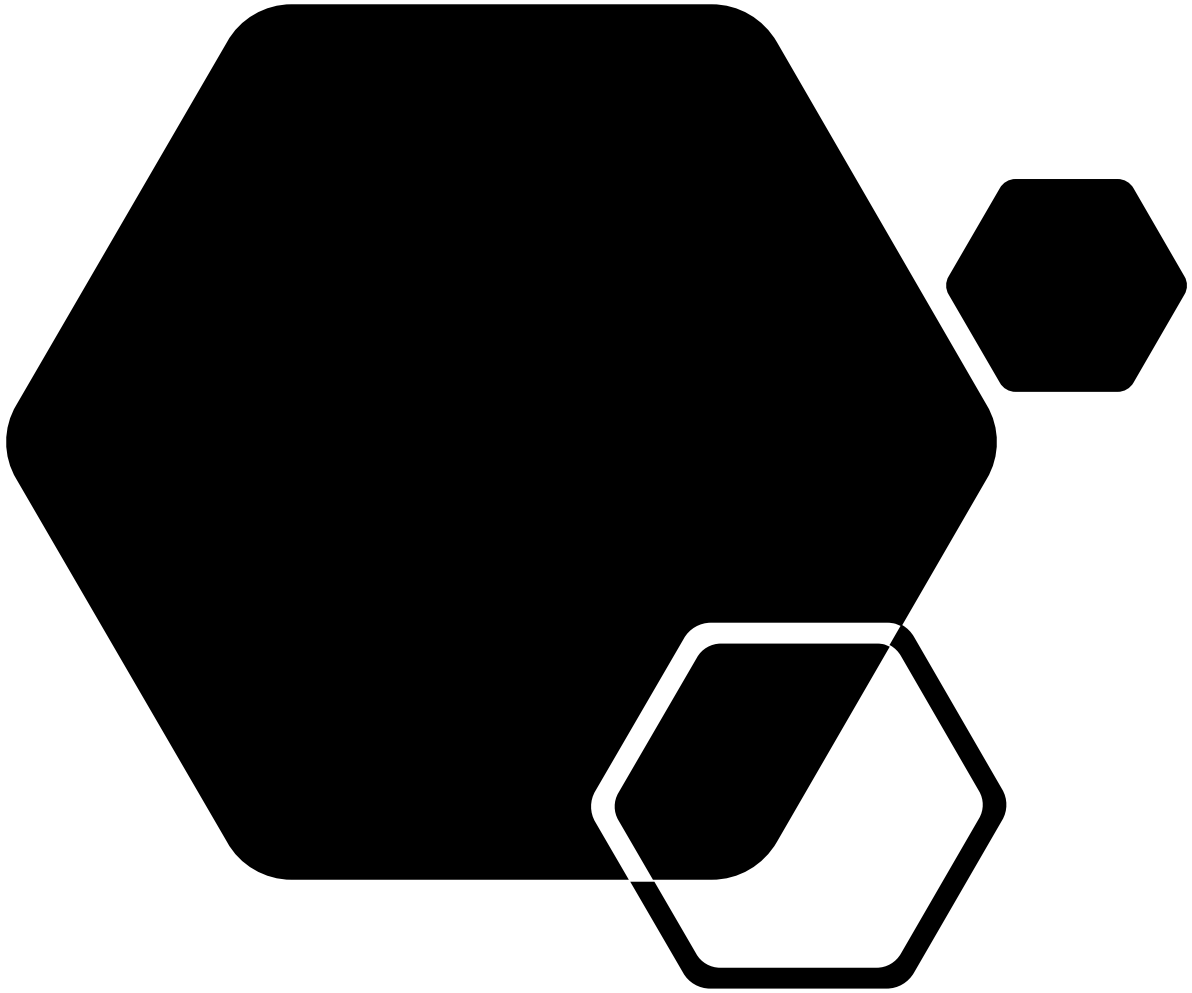
9. Tick test cases.



10. Go to *Run* tab, and then click *Start* . It runs ticked test cases.

Data Driven + BDD

*Learn creating keywords and variable first



Steps:

Take Success Add Candidate BDD for example

1. Create new Test Suite which import StepDefinition resources file.

2. Create test cases that represent data (refer to page 67).

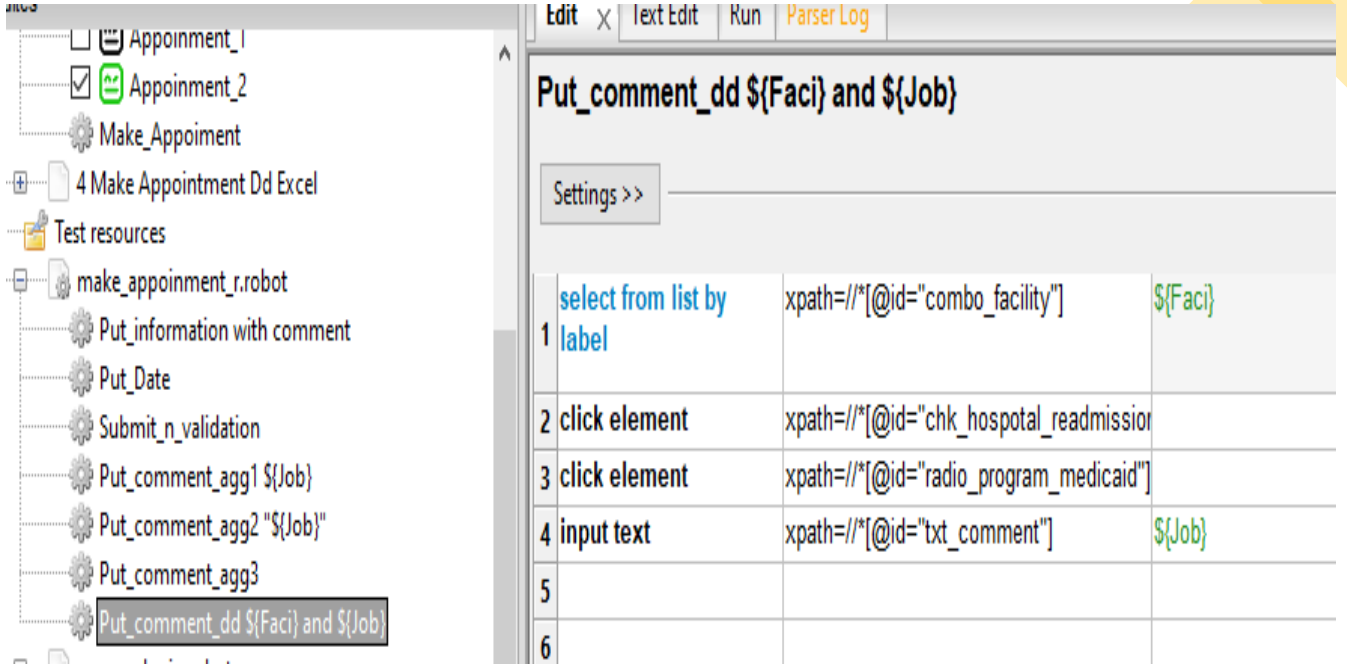
3. Create Keywords and put argument and BDD step (from previous BDD test case)

```
*** Settings ***
Test Setup         Open_n_login
Test Teardown      close browser
Test Template      Make_Appoiment
Library            SeleniumLibrary
Resource           ../Test_resources/open_n_login.robot
Resource           ../Test_resources/make_appoinment_r.robot

*** Test Cases ***
Appoinment_1
    Hongkong CURA Healthcare Center    comment 1

Appoinment_2
    Tokyo CURA Healthcare Center    comment 2

*** Keywords ***
Make_Appoiment
    [Arguments]    ${facility}    ${comment}
    When Put_information_dd ${facility} and ${comment}
    And Put_Date
    Then Submit_n_validation

```

4. We want step ‘Put information' be used for data driven. So, recall create keywords file will variable and create keywords file and name it as

Put_information_dd ${Faci} and ${Job}

5. Inside Keywords file, change value (in third column and row that involve only) into name of variable



7. Go back to Test Suite. Change 'I fill information' into

'Put_information_dd ${facility} and ${comment} '

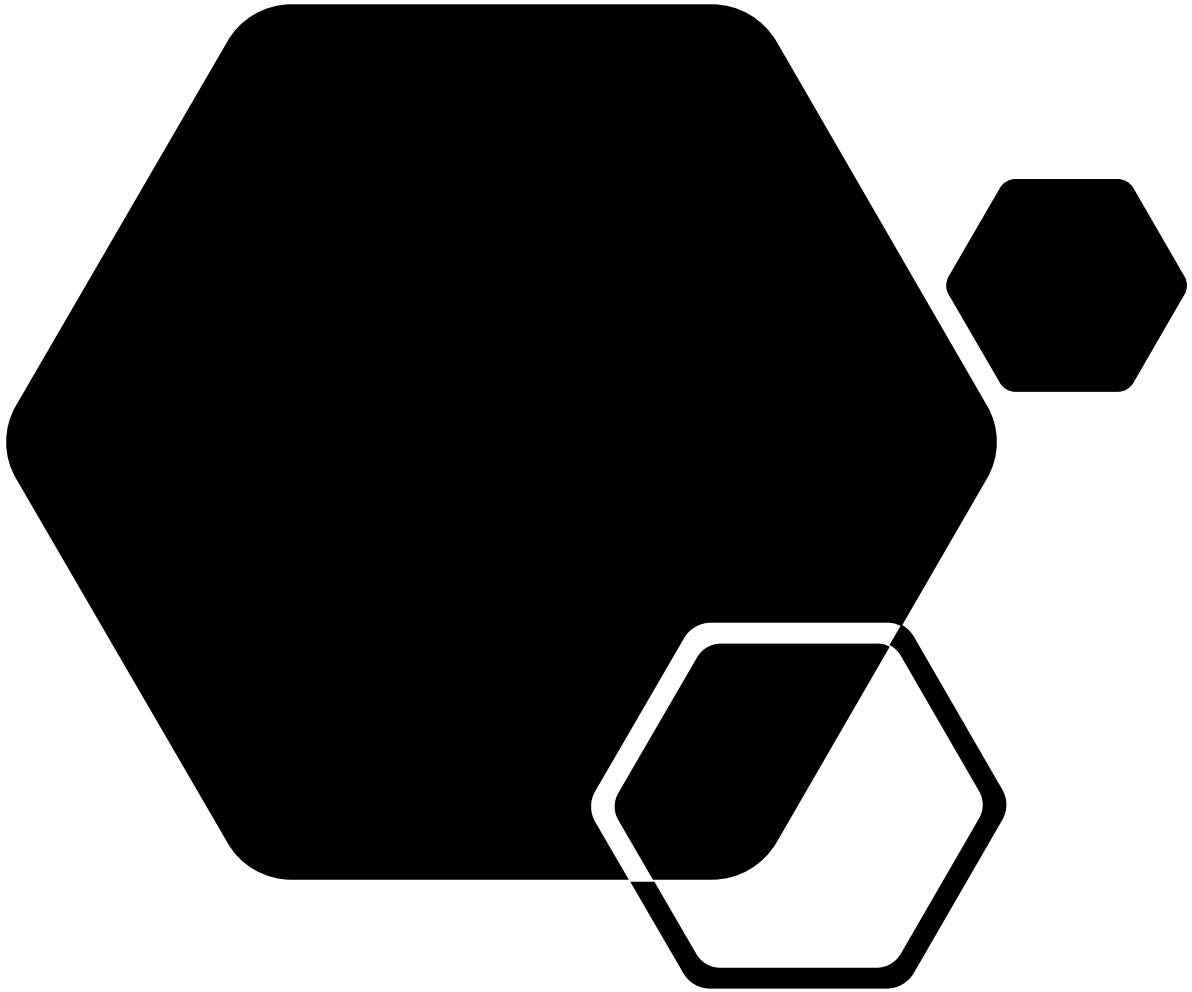
Note: name of variables must be same with argument variable

```
16 *** Keywords ***
17 Make_Appoiment
18     [Arguments]    ${facility}    ${comment}
19     When Put_information_dd ${facility} and ${comment}
20     And Put_Date
21     Then Submit_n_validation
22
```

8. Go to *Run* tab, and then click *Start* . It runs ticked test cases.

Data Driven Using External Test Data

Example: Excel, csv



NOTE:

- Install DataDriver library on command prompt by typing

  pip install --upgrade robotframework-datadriver[XLS]

Steps BDD + DD+ External Excel file:

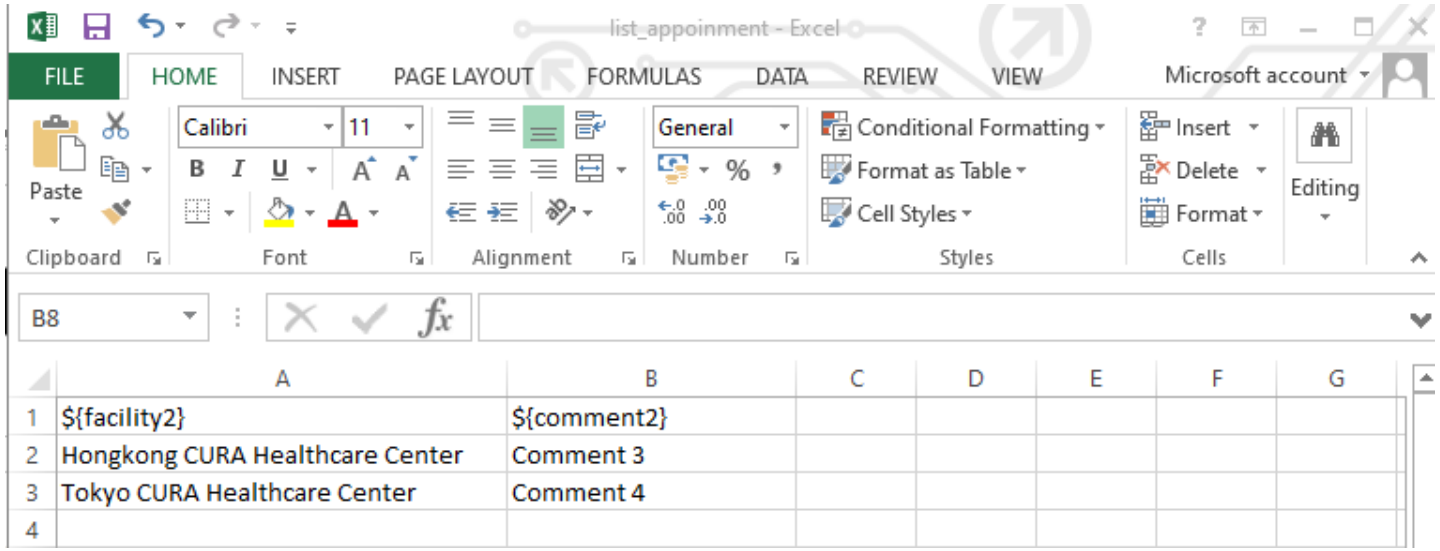
1. Before that, create empty excel file in the same folder of Test Suite in project file
2. Go back to RIDE. Create new test suite for
3. Just using/copy previous BDD+DD script.
4. Go to ‘Test Edit’ tab, write this script:

   Library DataDriver list_appoinment.xlsx sheet_name=Sheet1

5. After that, write this script under Test Case session:

```
*** Test Cases ***
appoinment
    ${facility2}    ${comment2}
```

6. On excel file, write name of variable (with Robot framework format) on first row:



	A	B	C	D	E	F	G
1	${facility2}	${comment2}					
2	Hongkong CURA Healthcare Center	Comment 3					
3	Tokyo CURA Healthcare Center	Comment 4					
4							

7. On test case and keywords, change variable name into name that same as name in Excel table:

```
14 *** Keywords ***
15 Make_Appoiment
16     [Arguments]    ${facility2}    ${comment2}
17     When Put_information_dd ${facility2} and ${comment2}
18     And Put_Date
19     Then Submit_n_validation
20
```

8. Go to *Run* tab, and then click *Start* . It runs ticked test cases.

Reference

- https://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html
- https://www.tutorialspoint.com/robot_framework/index.htm
- https://youtu.be/ErTN5rE6t8s
- https://youtube.com/playlist?list=PLhW3qG5bs-L9l2I8K8dEhw6HXy-Z-33w3