

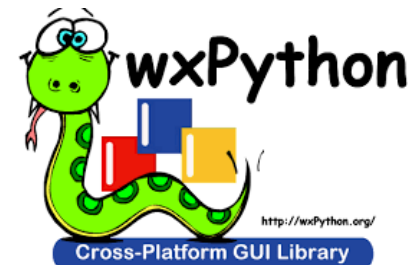
ROBOT FRAMEWORK

Building Automation Testing Using
Robot Framework



PRE-REQUISITES

You need to complete these before



ROBOT FRAMEWORK

Introduction robot
framework



Introduction Robot Framework



is a Python-based



extensible keyword-driven automation framework for acceptance testing



acceptance test driven development (ATDD),



behavior driven development (BDD)



robotic process automation (RPA)

Introduction Robot Framework



free to use without licensing costs



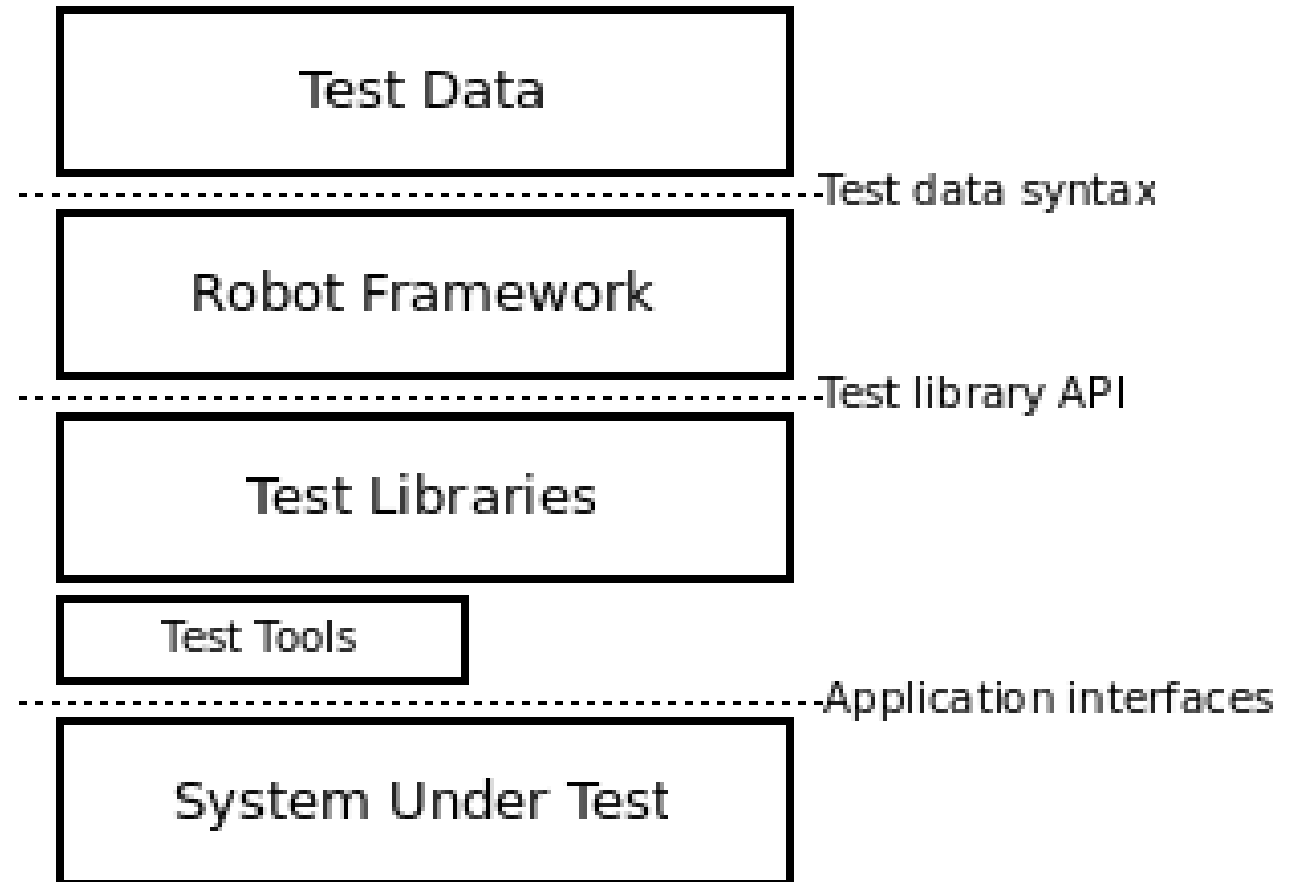
easy syntax, utilizing human-readable keywords



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

High-level architecture

- Robot Framework is a generic, application and technology independent framework
- It has a highly modular architecture illustrated in the diagram



Features of Robot Framework



Keywords



Variables



Libraries



Resources



Data driven test cases



Test Case Tagging



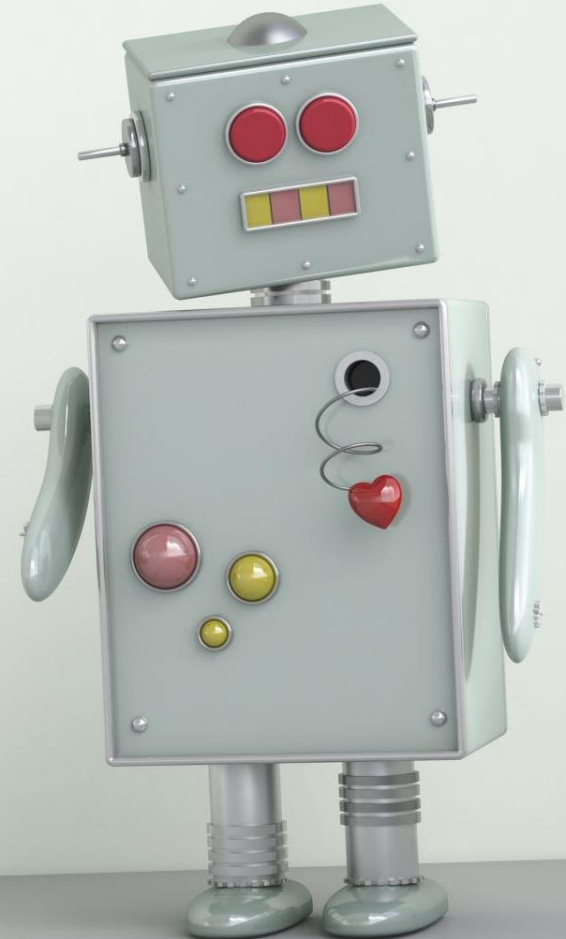
Reports and Logs

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

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



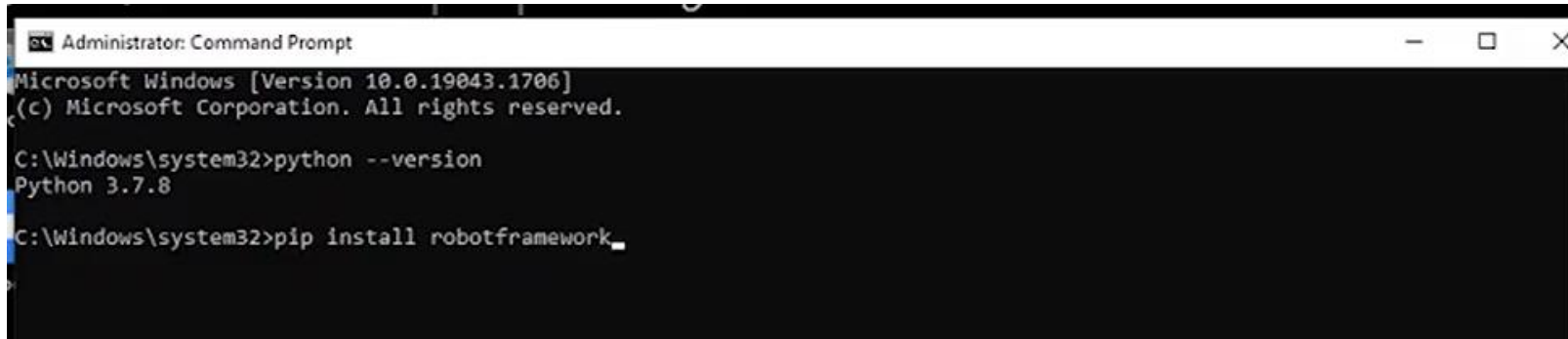
```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19043.1706]
(c) Microsoft Corporation. All rights reserved.
C:\Windows\system32>
```



```
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>
```

Python --version

Install robot framework



```
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_
```

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

```

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.
```

Type `pip install wxpython==4.0.7`

Install ride

```
C:\Windows\system32>pip install robotframework-ride
Processing c:\users\syazw\appdata\local\pip\cache\wheels\fc\7e\ad\62316f036476f8a3eba3830dcb12649d85d209e5a42416c348\rob
otframework_ride-1.7.4.2-py3-none-any.whl
Collecting PyPubSub
  Using cached Pypubsub-4.0.3-py3-none-any.whl (61 kB)
Collecting Pygments
  Using cached Pygments-2.12.0-py3-none-any.whl (1.1 MB)
Collecting Pywin32
  Using cached pywin32-304-cp37-cp37m-win_amd64.whl (12.2 MB)
Requirement already satisfied: wxPython<=4.0.7.post2 in c:\program files\python37\lib\site-packages (from robotframework-ride) (4.0.7)
Requirement already satisfied: pillow in c:\program files\python37\lib\site-packages (from wxPython<=4.0.7.post2->robotframework-ride) (9.1.1)
Requirement already satisfied: six in c:\program files\python37\lib\site-packages (from wxPython<=4.0.7.post2->robotframework-ride) (1.16.0)
Requirement already satisfied: numpy; python_version >= "3.0" in c:\program files\python37\lib\site-packages (from wxPython<=4.0.7.post2->robotframework-ride) (1.21.6)
Installing collected packages: PyPubSub, Pygments, Pywin32, robotframework-ride
```

Type `pip install robotframework-ride`

Install Selenium Library

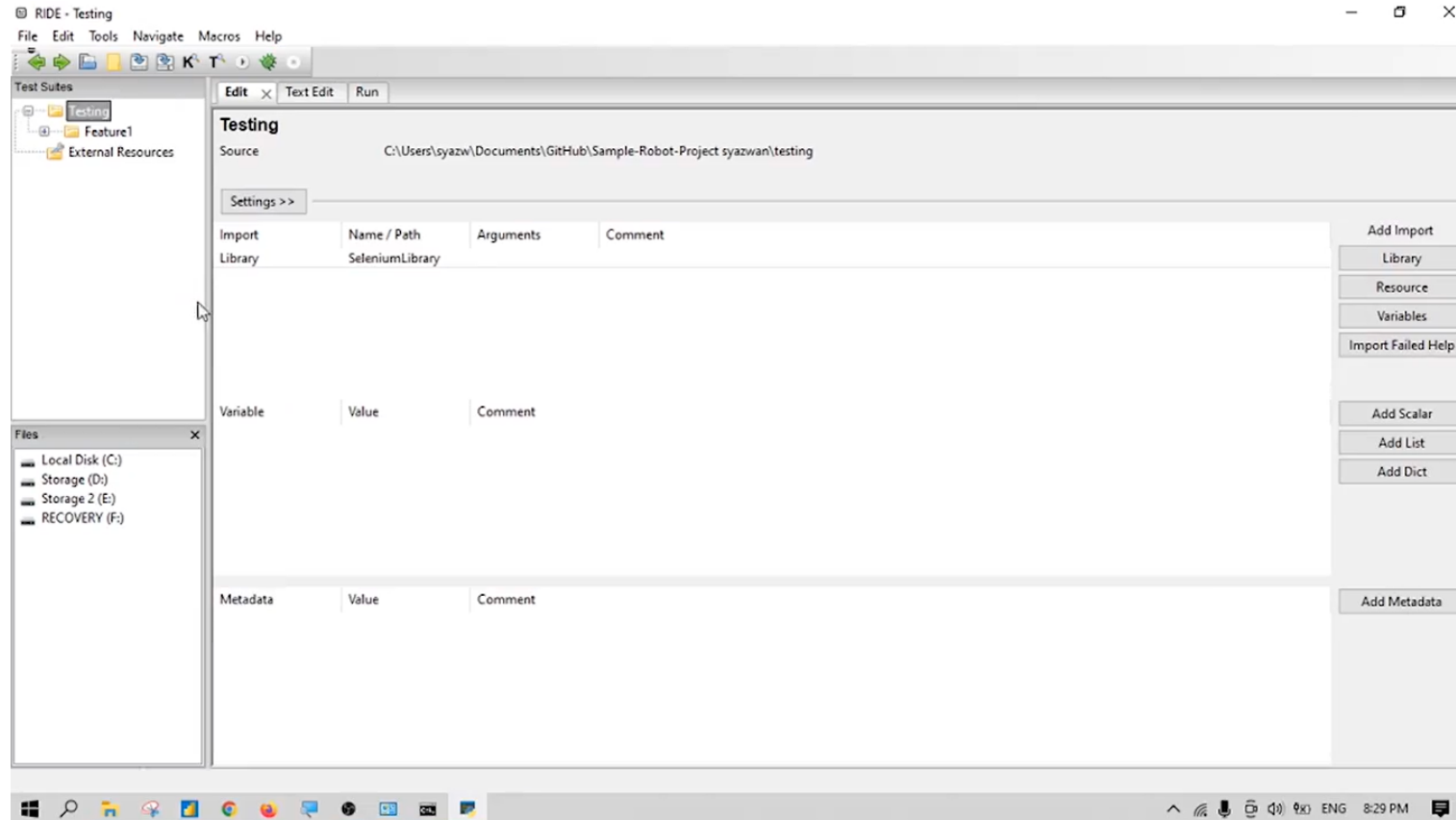
```
C:\Windows\system32>pip install --upgrade robotframework-seleniumlibrary
Collecting robotframework-seleniumlibrary
  Using cached robotframework_seleniumlibrary-6.0.0-py2.py3-none-any.whl (95 kB)
Collecting selenium>=4.0.0
  Using cached selenium-4.1.5-py3-none-any.whl (979 kB)
Requirement already satisfied, skipping upgrade: robotframework>=3.2.2 in c:\program files\python37\lib\site-packages (from robotframework-seleniumlibrary) (5.0.1)
Collecting robotframework-pythonlibcore>=2.2.1
  Using cached robotframework_pythonlibcore-3.0.0-py2.py3-none-any.whl (9.9 kB)
Collecting urllib3[secure,socks]>=1.26
  Using cached urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
```

Type `pip install --upgrade robotframework-seleniumlibrary`

Open ride robot framework

```
you should consider upgrading via the 'C:\Program Files\Python37\python.exe -m pip install --upgrade pip' command.  
C:\Windows\system32>ride.py
```

Type ride.py



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

Syntax 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

← → ↻ 🔒 opensource-demo.orangehrmlive.com/index.php/auth/login



LOGIN Panel

LOGIN

[Forgot your password?](#)

Emojis	Win+Period
Undo	Ctrl+Z
Redo	Ctrl+Shift+Z
Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Paste as plain text	Ctrl+Shift+V
Select all	Ctrl+A
Spell check	▶
Writing Direction	▶
Block element...	
Inspect	

password : admin

Elements Console Sources

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>...</head>
  <body data-new-gr-c-s-check-loaded="14.1062.0" data-gr-ext-installed"> == $0
    <div id="wrapper">...</div>
    <!-- wrapper -->
    <script type="text/javascript">...</script>
    <div id="tiptip_holder" style="max-width:200px;">...</div>
    </body>
    <grammarly-desktop-integration data-grammarly-shadow-root="true">...</grammarly-desktop-integration>
  </html>
```

html body

Styles Computed Layout Event Listeners DOM Breakpoints

Filter :hov .cls +

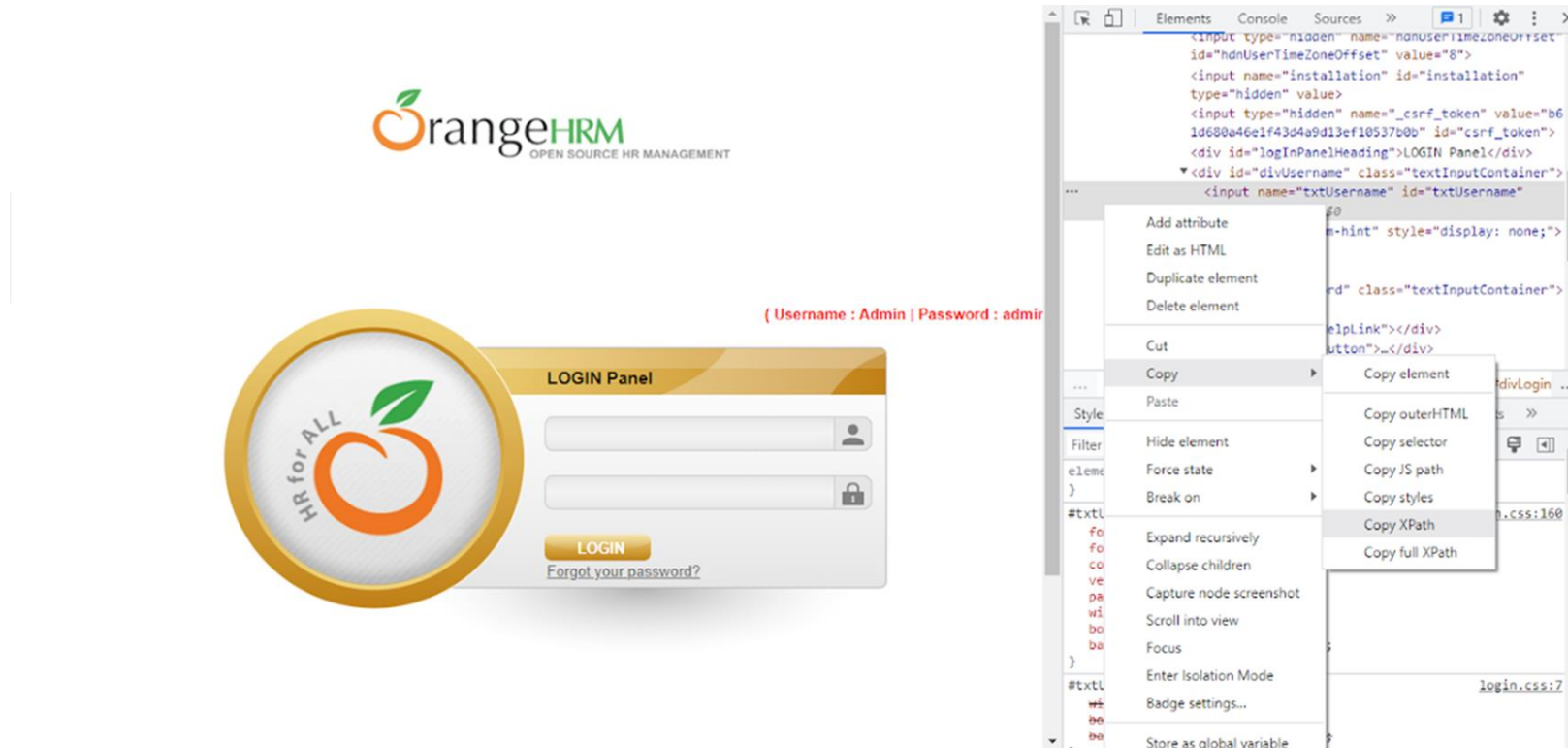
```
element.style {
}

body {
  background-color: #FFFFFF;
  height: 700px;
  font-family: Roboto, Arial, Helvetica, sans-serif;
}

body {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 13px;
  color: #5d5d5d;
}

body {
  line-height: 1;
}
```

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



First testing

Success Adding Candidate

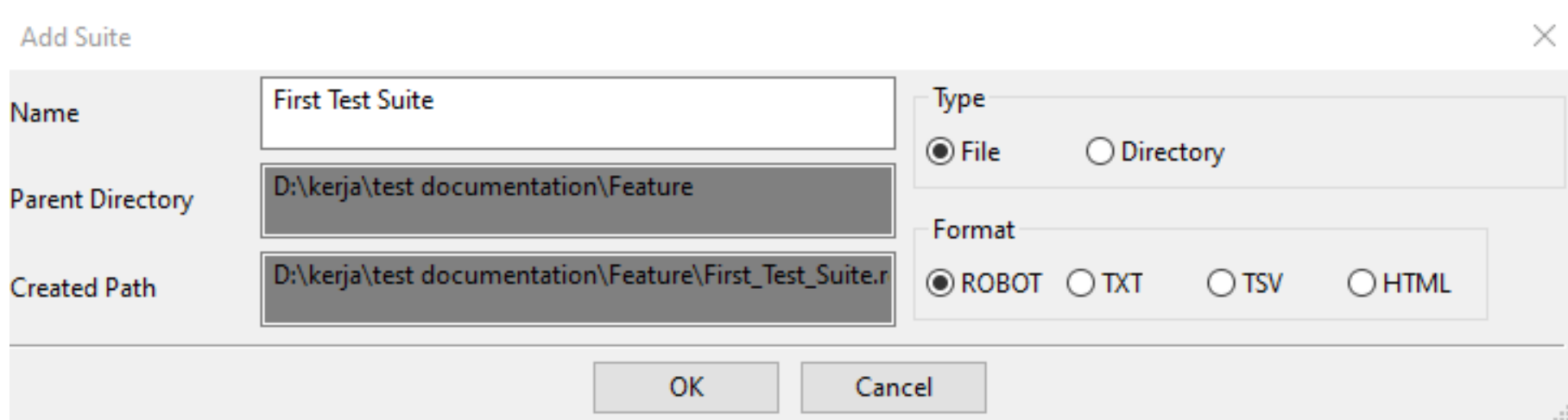
Step 1: Create Directory/folder

- 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

The screenshot shows the 'Add Suite' dialog box. The 'Name' field is set to 'Feature'. The 'Parent Directory' field is set to 'D:\kerja\test documentation'. The 'Created Path' field is set to 'D:\kerja\test documentation\Feature__init__.i'. The 'Type' section has 'Directory' selected. The 'Format' section has 'ROBOT' selected. The 'OK' and 'Cancel' buttons are at the bottom.

Step 2: Create Test Suite

- i. Right click on Feature Directory and choose 'New Suite'.
- ii. In Add Suite window, named it as 'First Test Suite' and then click OK. This test suite can be found under Feature.



The screenshot shows a dialog box titled "Add Suite" with a close button (X) in the top right corner. The dialog contains the following fields and options:

- Name:** A text box containing "First Test Suite".
- Parent Directory:** A text box containing "D:\kerja\test documentation\Feature".
- Created Path:** A text box containing "D:\kerja\test documentation\Feature\First_Test_Suite.r".
- Type:** Two radio buttons: "File" (selected) and "Directory".
- Format:** Four radio buttons: "ROBOT" (selected), "TXT", "TSV", and "HTML".

At the bottom of the dialog are two buttons: "OK" and "Cancel".

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

Library

Name: SeleniumLibrary Browse

Args:

Alias:

Comment:

Give name, optional arguments and optional alias of the library to import.
Separate multiple arguments with a pipe character like 'arg 1 | arg 2'.
Alias can be used to import same library multiple times with different names.

OK Cancel

Edit × Text Edit Run Parser Log

First Test Suite

Source: D:\kerja\test documentation\Feature\First_Test_Suite.robot

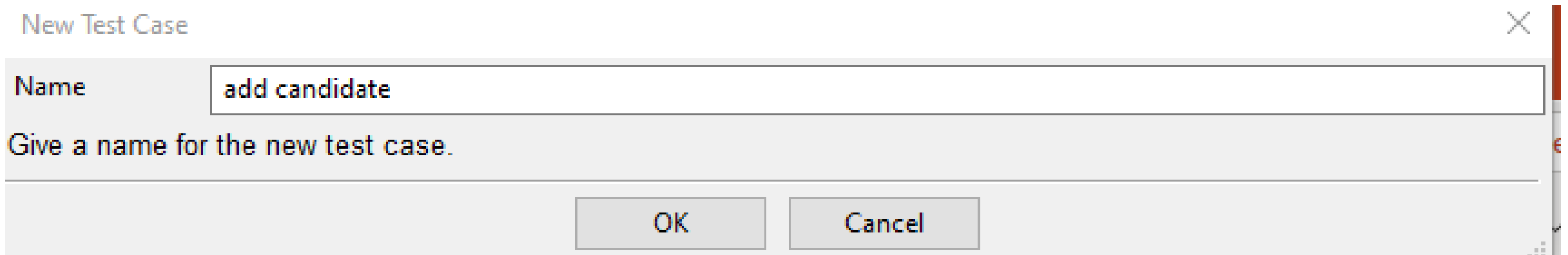
Settings >>

Import	Name / Path	Arguments	Comment
Library	SeleniumLibrary		

Add Import
Library
Resource
Variables
Import Failed Help

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 'add candidate' and then click OK. This testcase can be found under Test Suite.



New Test Case

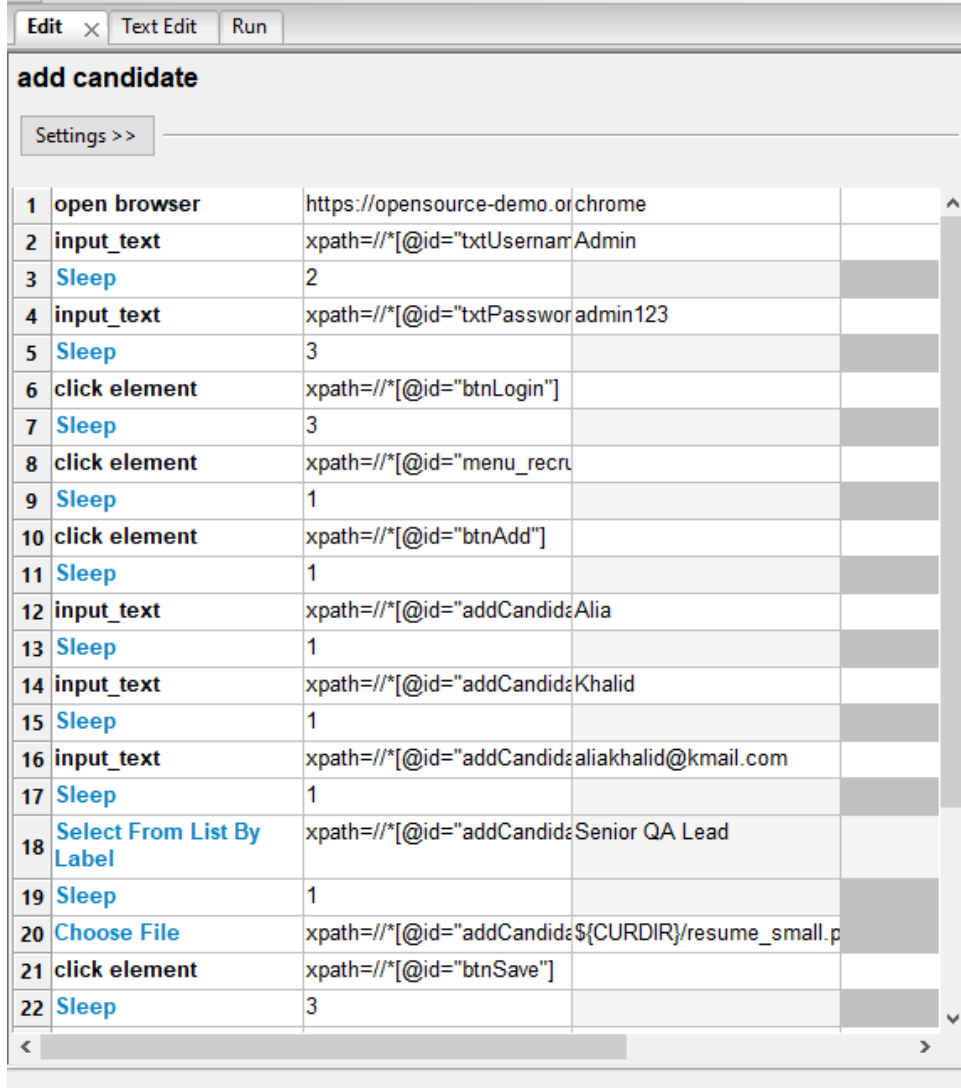
Name add candidate

Give a name for the new test case.

OK Cancel

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

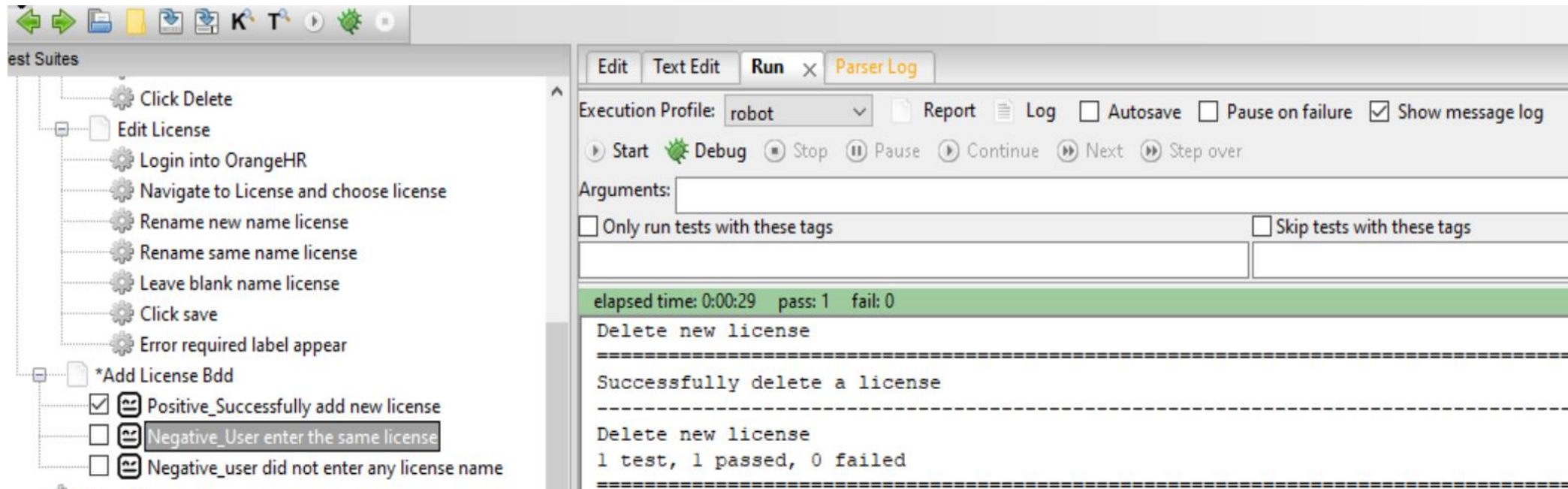


The screenshot shows a software interface for editing a test case titled "add candidate". It features a table with 22 rows, each representing a test step. The table has four columns: a step number, a keyword, an XPath expression, and a data value. The keywords include "open browser", "input_text", "Sleep", "click element", "Select From List By Label", and "Choose File". The XPath expressions are used to locate various elements on a webpage, such as login fields, a menu, and a file upload button. The data values include a URL, usernames, passwords, and a file path.

Step	Keyword	Value 1 (XPath)	Value 2 (Data)
1	open browser	https://opensource-demo.orchrome	
2	input_text	xpath=//*[@id="txtUsername"]	Admin
3	Sleep	2	
4	input_text	xpath=//*[@id="txtPassword"]	admin123
5	Sleep	3	
6	click element	xpath=//*[@id="btnLogin"]	
7	Sleep	3	
8	click element	xpath=//*[@id="menu_recru"]	
9	Sleep	1	
10	click element	xpath=//*[@id="btnAdd"]	
11	Sleep	1	
12	input_text	xpath=//*[@id="addCandidateAlia"]	
13	Sleep	1	
14	input_text	xpath=//*[@id="addCandidateKhalid"]	
15	Sleep	1	
16	input_text	xpath=//*[@id="addCandidatealiakhalid@kmail.com"]	
17	Sleep	1	
18	Select From List By Label	xpath=//*[@id="addCandidateSenior QA Lead"]	
19	Sleep	1	
20	Choose File	xpath=//*[@id="addCandidate\${CURDIR}/resume_small.p"]	
21	click element	xpath=//*[@id="btnSave"]	
22	Sleep	3	

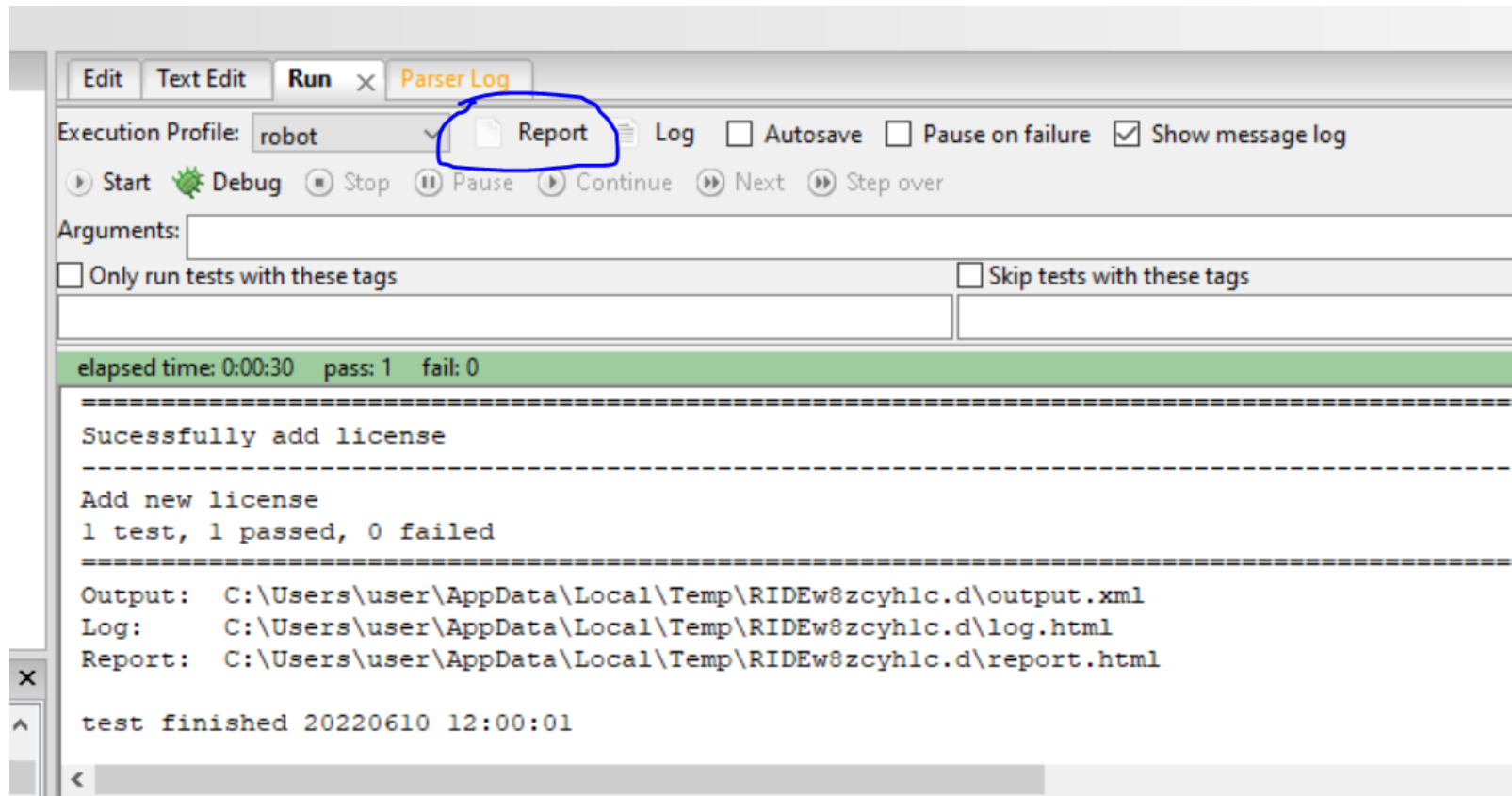
STEP 6: Run test case

- 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

Add new license Report

Generated
20220610 12:00:01 UTC+08:00
7 minutes 31 seconds ago

Summary Information

Status:

All tests passed

Start Time:

20220610 11:59:31.170

End Time:

20220610 12:00:01.325

Elapsed Time:

00:00:30.155

Log File:

[log.html](#)

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	1	1	0	0	00:00:30	<div></div>

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags						<div></div>

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Add new license	1	1	0	0	00:00:30	<div></div>


Test Details

AllTagsSuitesSearch


Suite:

Test:

Include:



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

0 Add And Delete Candidate Nonbdd

Source D:\kerja\pakwanoranghr\Candidate Feature Test Suites\0_add_and_delete_candida

Settings <<

Documentation

Suite Setup

Suite Teardown

Test Setup

Test Teardown

Test Template

Test Timeout

Force Tags

Default Tags

Import	Name / Path	Arguments	Comment	Add Import
Library	SeleniumLibrary			Library
				Resource
				Variables

- 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

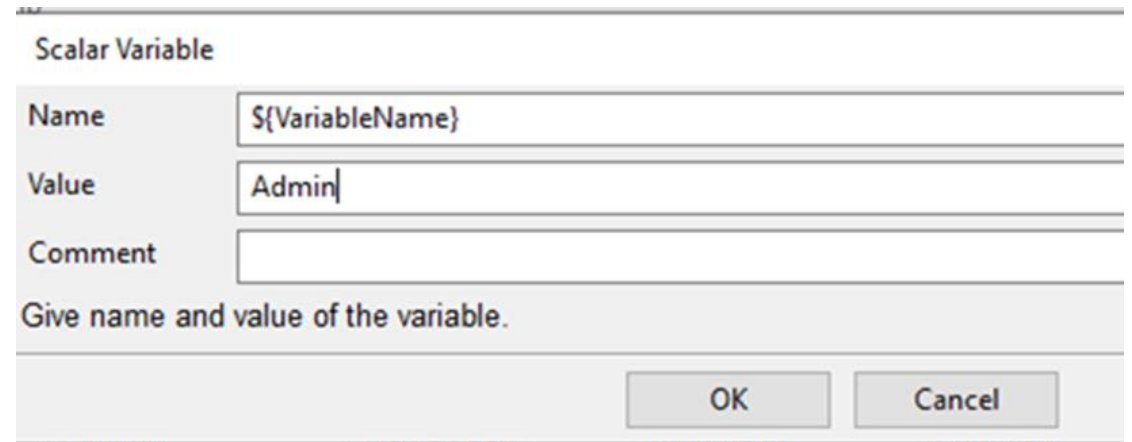
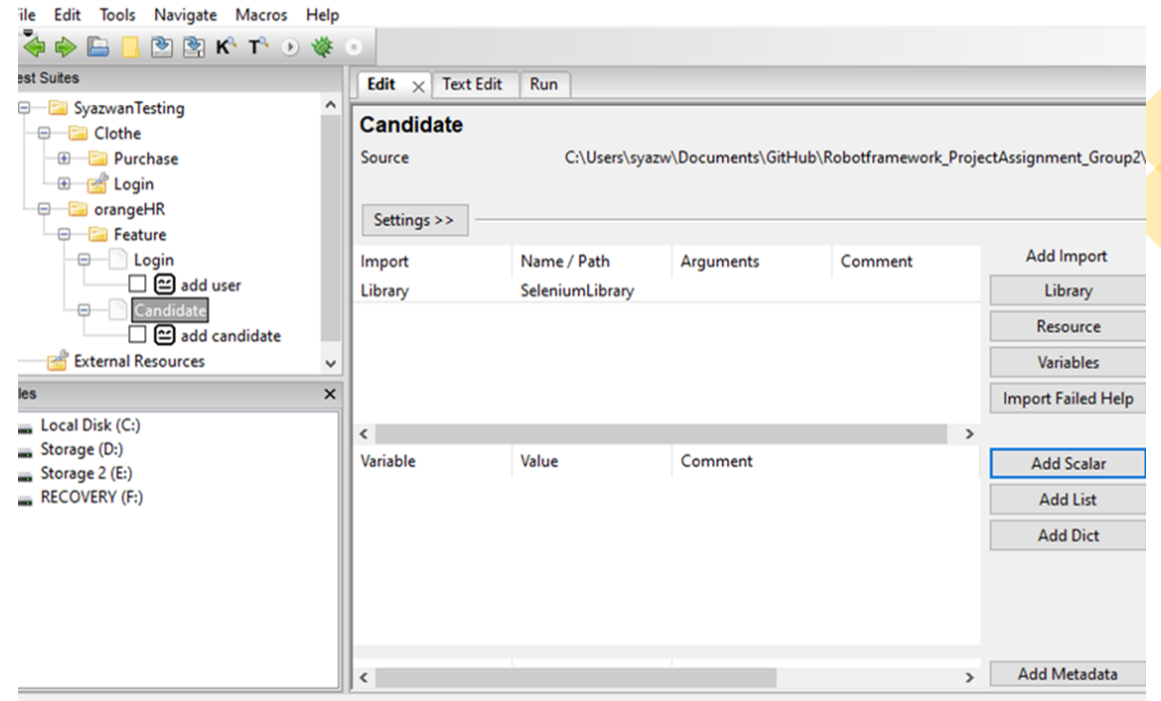
Variable, Keywords File and Resource File

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="txtUsername"]	<u><code>\${VariableName}</code></u>
---	------------	------------------------------	--------------------------------------

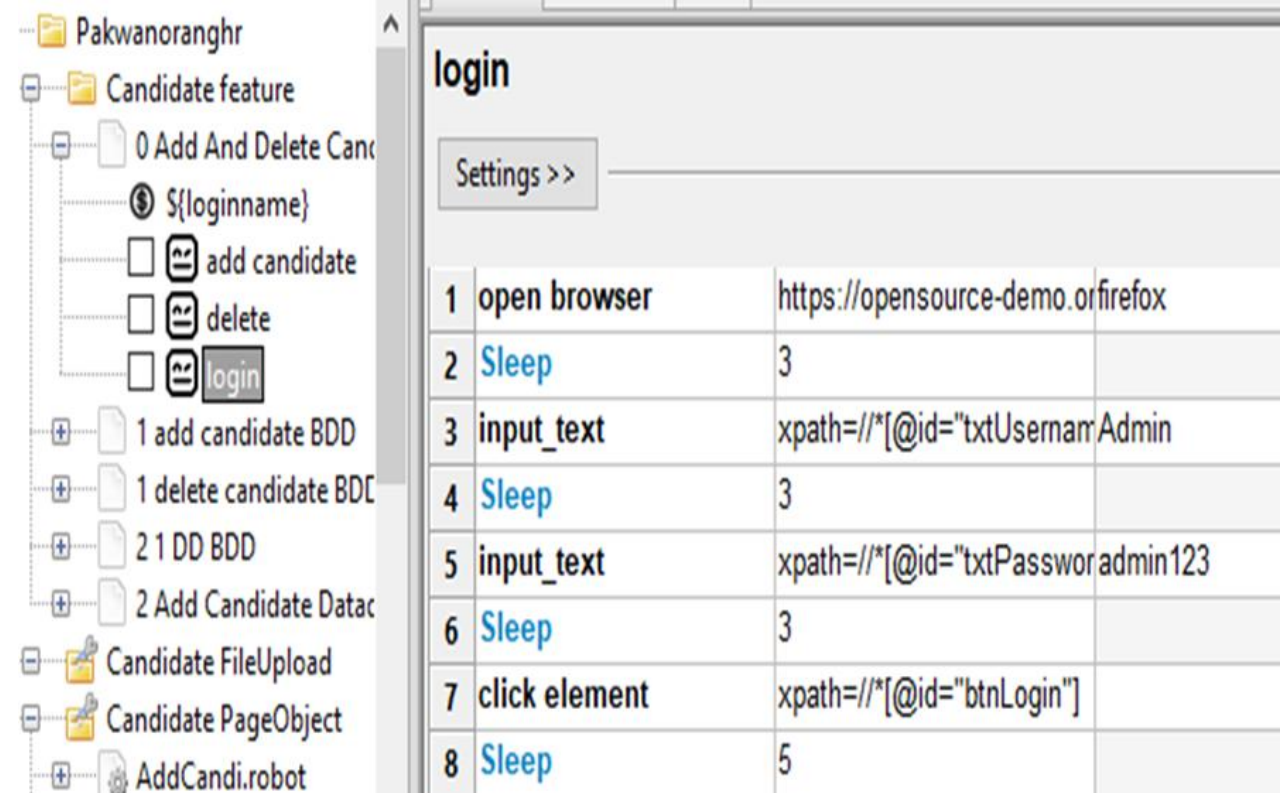
*If colour of Variable is green, it means this variable is existing. But if its colour is light purple/pink, it means this variable is not existing.

Keywords File

Combining existing keywords/test steps together

Steps:

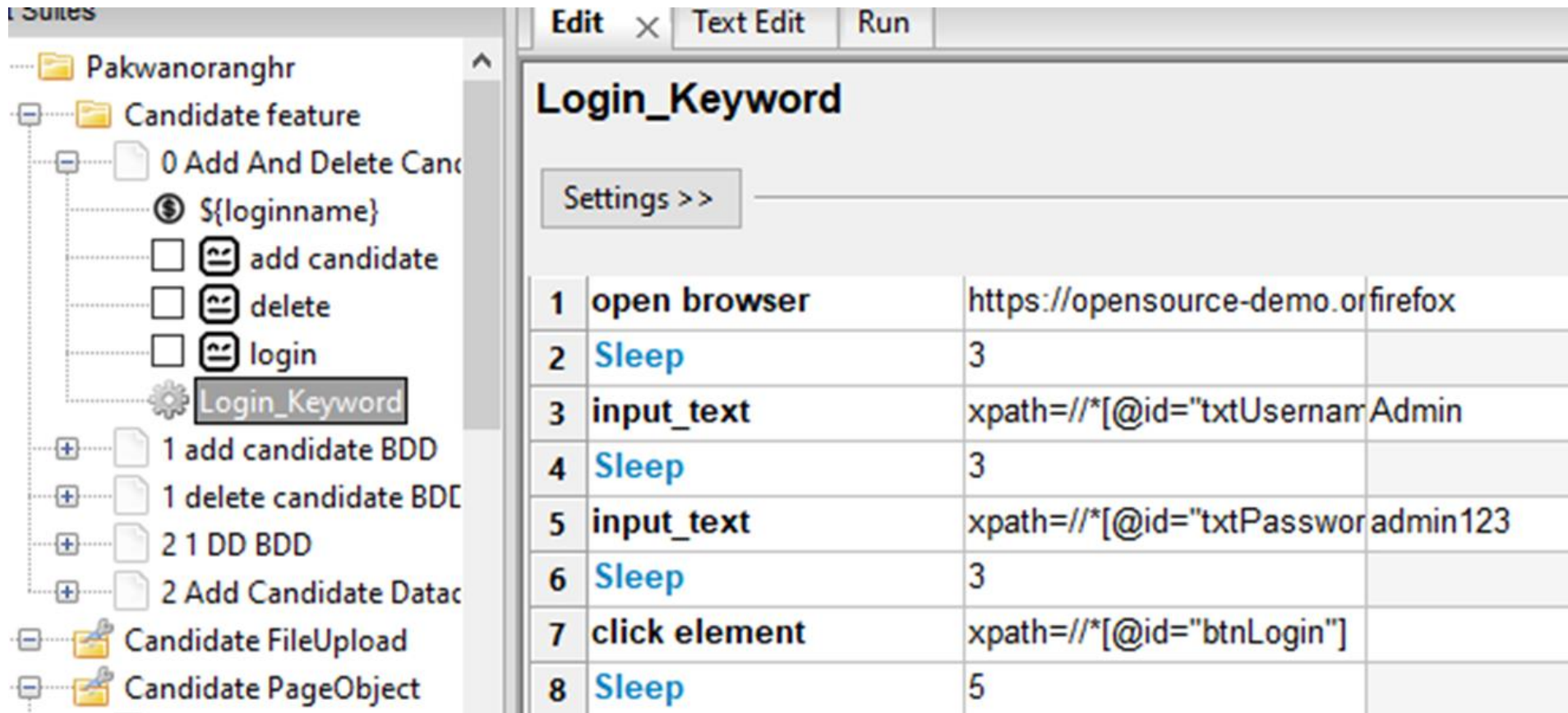
- i. Select any test case inside test suite. Inside test case, there are many keywords/test steps.
- ii. Main objective for this is creating keywords file so it can be set on Test Setup.
- iii. Select on test suite and then right click-> New User Keywords to create keywords file



The screenshot displays the TestNG IDE interface. On the left, a tree view shows a project named 'Pakwanoranghr' containing a 'Candidate feature' suite. Inside this suite, there are several test cases, including '0 Add And Delete Canc', '1 add candidate BDD', '1 delete candidate BDD', '2 1 DD BDD', and '2 Add Candidate Datac'. The 'login' test case is highlighted. On the right, the 'login' test case is expanded, showing a table of steps. The table has three columns: Step Number, Keyword, and Value. The steps are as follows:

Step Number	Keyword	Value
1	open browser	https://opensource-demo.orfirefox
2	Sleep	3
3	input_text	xpath=//*[@id="txtUsernameAdmin
4	Sleep	3
5	input_text	xpath=//*[@id="txtPasswordadmin123
6	Sleep	3
7	click element	xpath=//*[@id="btnLogin"]
8	Sleep	5

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

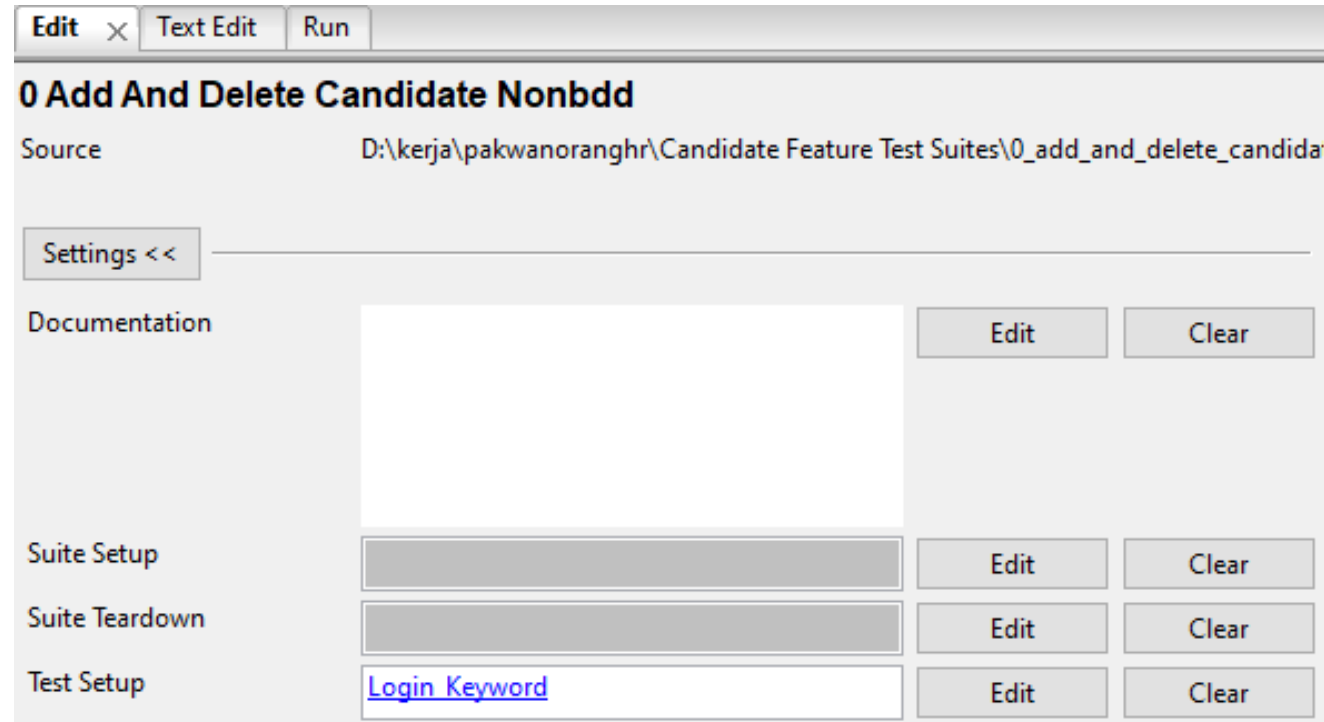


The screenshot displays the Selenium IDE interface. On the left, a tree view shows the project structure under 'Suites'. The 'Candidate feature' folder is expanded, showing several keywords: '0 Add And Delete Canc', '\$ {loginname}', 'add candidate', 'delete', 'login', and 'Login_Keyword' (which is highlighted). Below these are '1 add candidate BDD', '1 delete candidate BDD', '2 1 DD BDD', and '2 Add Candidate Data'. Further down are 'Candidate FileUpload' and 'Candidate PageObject'.

The main area on the right is titled 'Login_Keyword' and contains a 'Settings >>' button. Below the button is a table with 8 rows, each representing a keyword and its arguments.

1	open browser	https://opensource-demo.orfirefox
2	Sleep	3
3	input_text	xpath=//*[@id="txtUsernamAdmin
4	Sleep	3
5	input_text	xpath=//*[@id="txtPassworadmin123
6	Sleep	3
7	click element	xpath=//*[@id="btnLogin"]
8	Sleep	5

- 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: Add candidate \${firstname} \${lastname}

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

The screenshot displays the Test Suites interface on the left and the Keyword Editor on the right. The Test Suites pane shows a tree structure with 'Pakwanoranghr' as the root, followed by 'Candidate Feature Test Suites', and then '0 Add And Delete Candidate Nonbdd'. Under this suite, several keywords are listed: \${fname}, \${lname}, \${email}, \${resume}, add candidate (checked), delete, Login_Keyword, and 'Add candidate \${firstname} and \${lastname}' (highlighted). The Keyword Editor on the right shows the title 'Add candidate \${firstname} and \${lastname}' and a 'Find Usages' button. Below the title, there is a 'Settings >>' button and a table with 3 columns: Step Number, Keyword, and XPath. The table contains the following data:

Step Number	Keyword	XPath
1	input_text	xpath=//*[@id="addCandidate\${firstname}]
2	Sleep	1
3	input_text	xpath=//*[@id="addCandidate\${lastname}]
4		
5		
6		

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

Add candidate \${fname} \${lname}

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.

1	Sleep	1	
2	click element	xpath=//*[@id="menu_recr	
3	Sleep	1	
4	click element	xpath=//*[@id="btnAdd"]	
5	Sleep	1	
6	Add candidate \${fname} and \${lname}		
7	Sleep	1	
8	input_text	xpath=//*[@id="addCandidate\${email}	
9	Sleep	1	
10	Select From List By Label	xpath=//*[@id="addCandidate	Senior QA Lead
11	Sleep	1	
12	click element	xpath=//*[@id="btnSave"]	
13	Sleep	3	
14	click element	xpath=//*[@id="btnBack"]	
15	Sleep	2	

Edit × Text Edit Run

0 Add And Delete Candidate Nonbdd

SourceD:\kerja\pakwanoranghr\Candidate Feature Test Suites\0_

Settings >>

ImportLibrary

Name / PathSeleniumLibrary

Arguments

Add Import

Library

Resource

Variables

Import Failed Help

Add Scalar

Add List

Add Dict


< >

VariableValue


\${fname}Ali

\${lname}Khalid

< >

A large, solid orange circle is positioned on the left side of the slide, partially cut off by the edge.

For more detail by looking test suite that used
setup, teardown, variable and keyword file,
can refer to '1 Adding Candidate More Setting'
test suite

A yellow dashed line is located in the bottom right corner of the slide, consisting of several short, curved segments.

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

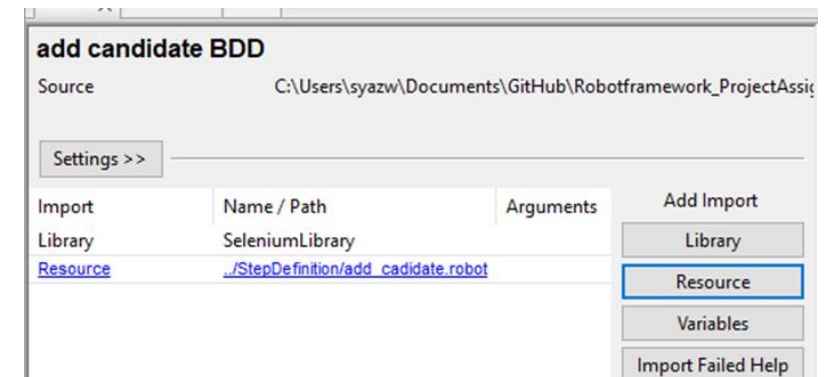
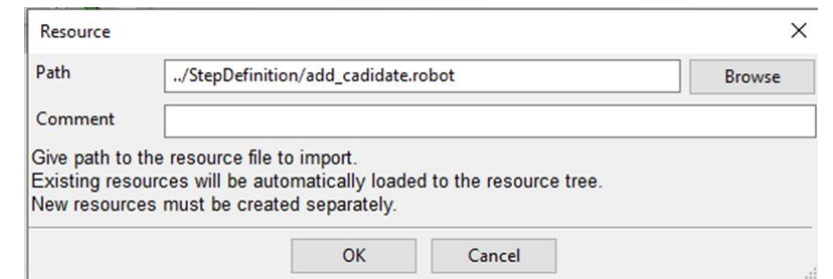
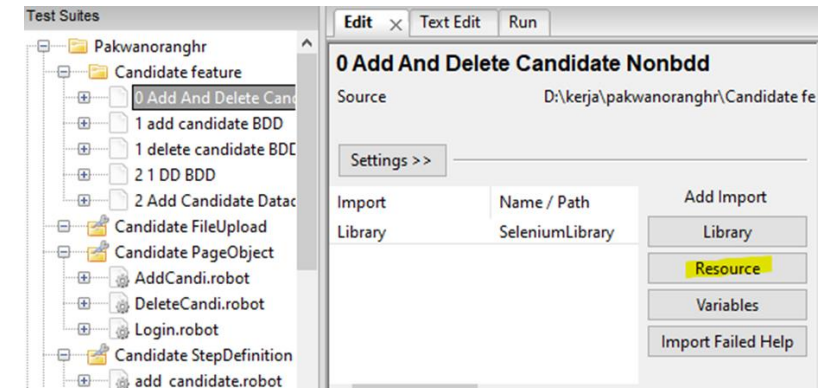
Steps:

- i. Create scenarios resource 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

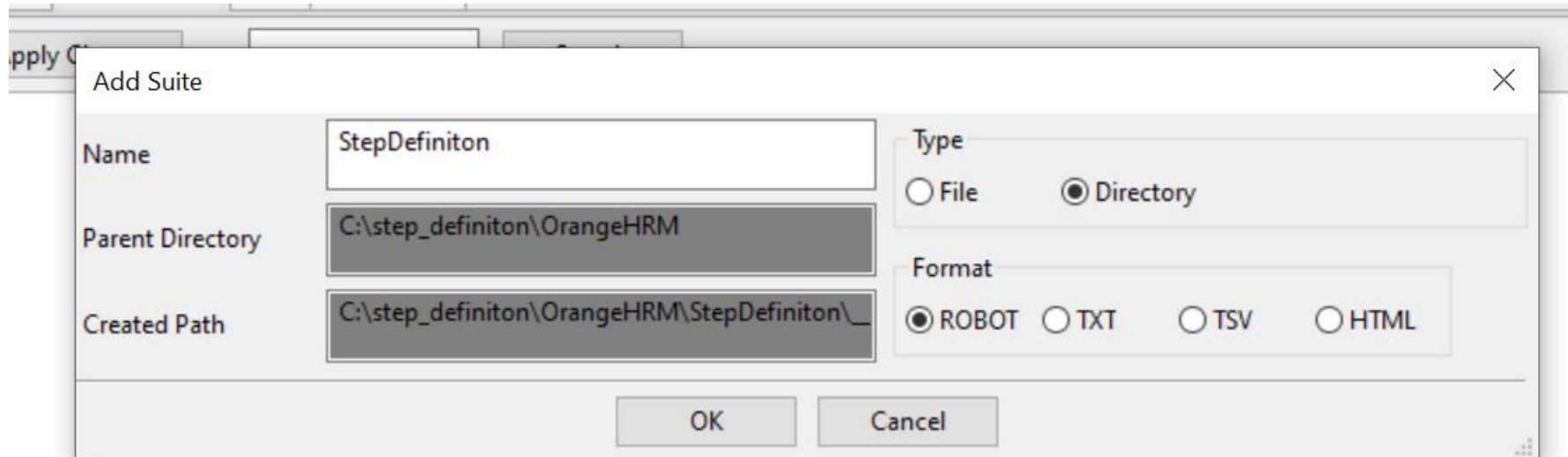


Behavior-driven development

- 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.

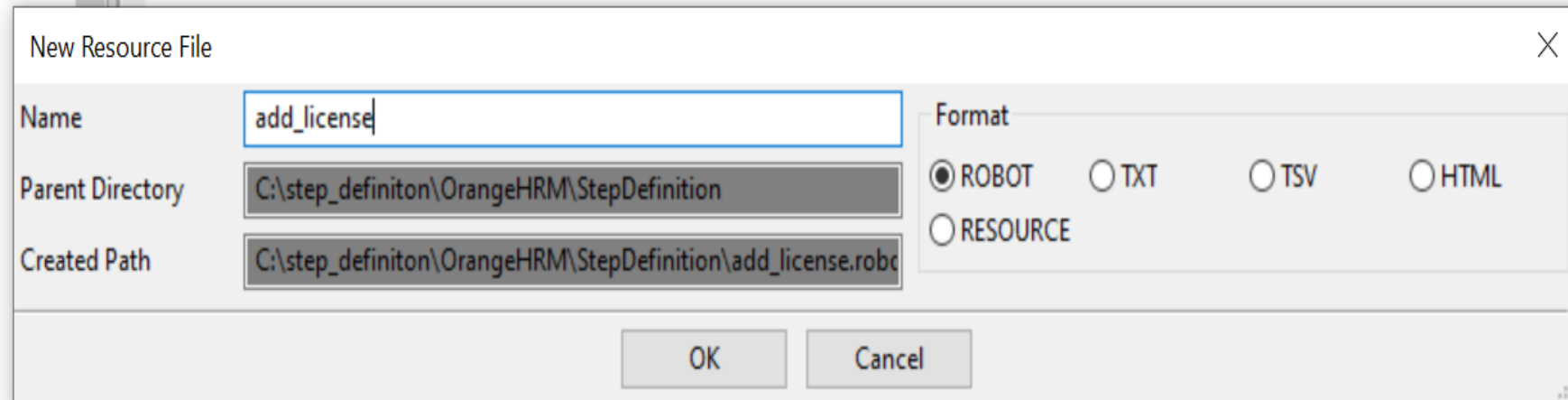
STEP 1 : Create StepDefinition directory folder

- i. Create new project
- ii. Right-click on the project and choose new suite. Select the type as 'Directory' and format 'ROBOT'.
- iii. Name the file as 'StepDefinition' then click OK.



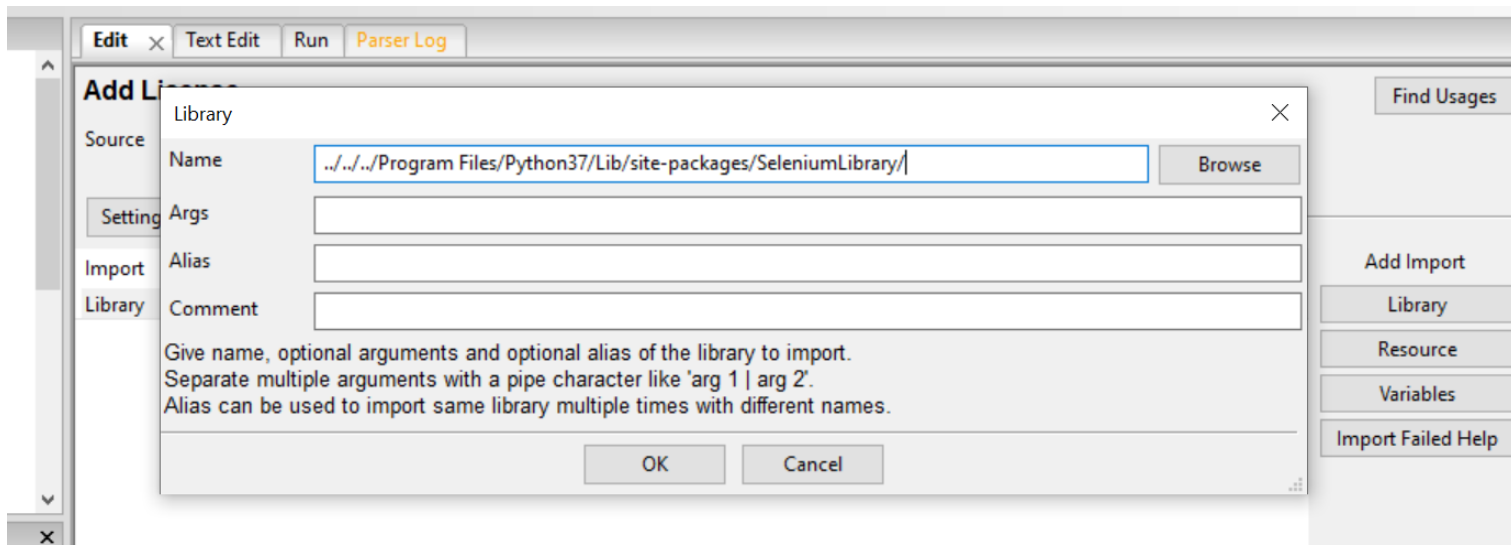
STEP 2: Create Step Definiton directory folder

- i. Right-click on the StepDefiniton file and choose new resource.
Name the file. Eg: Add_license.robot
- ii. Choose format 'ROBOT'. Then click 'OK'



STEP 3: Import library & resources

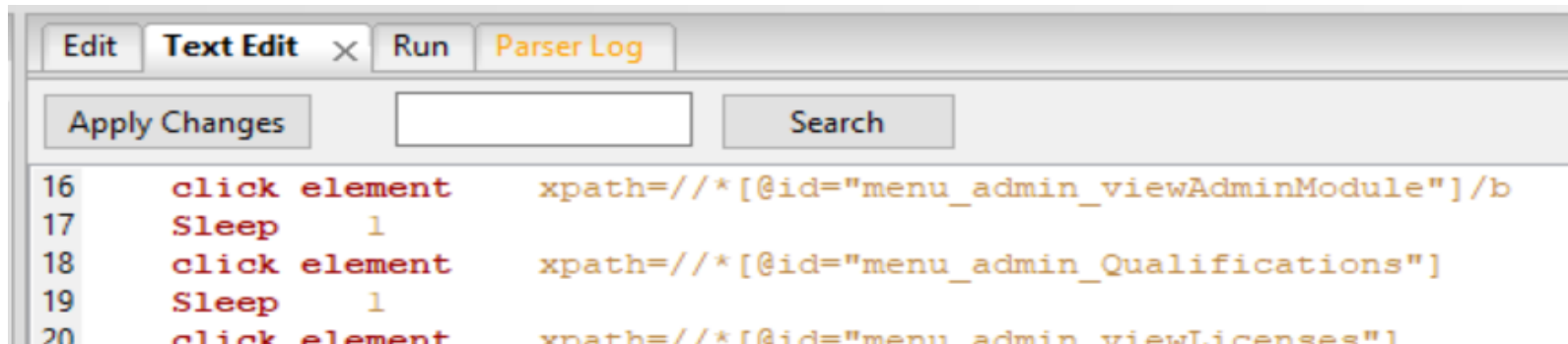
- i. Click on the 'add_license.robot@ resources file
- ii. Click edit. Then click on 'Library' on the right-side.
- iii. Browse and choose your Selenium library folder.
- iv. Click 'OK'
- v. Click 'Resources' to add other resources file



STEP 4 :Create user keyword

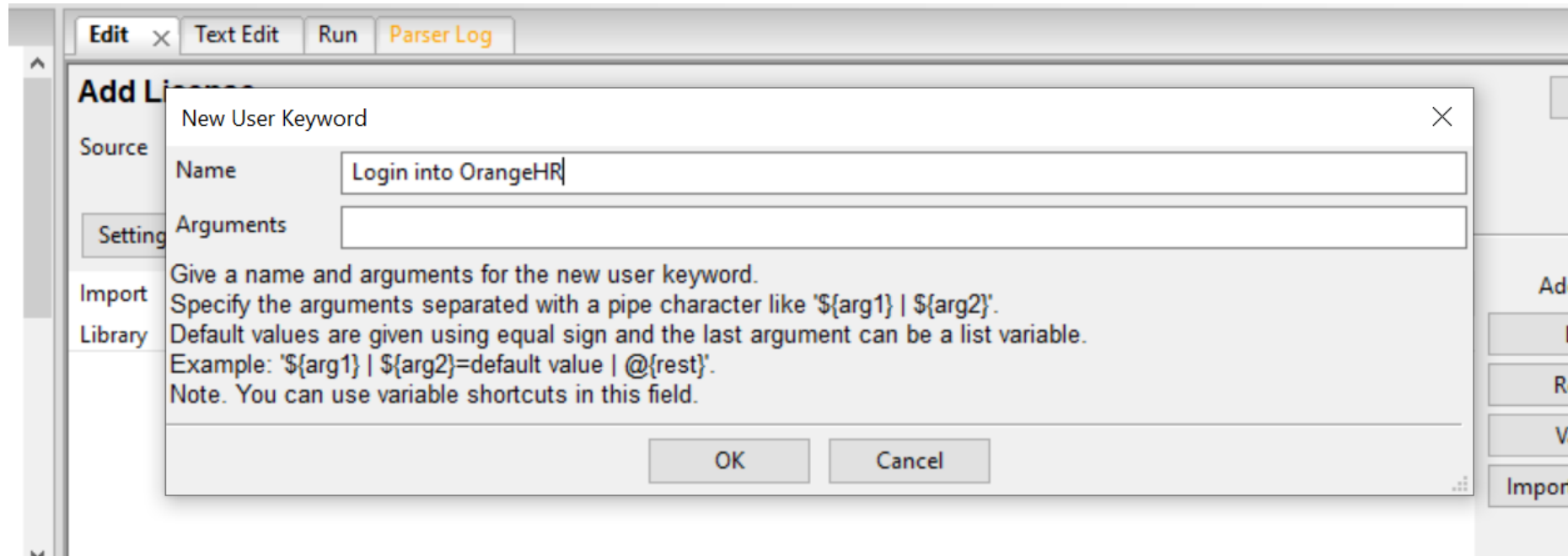
There is two ways two create user-keyword

- i. Tab 'Edit'
- ii. Tab 'Text Edit'

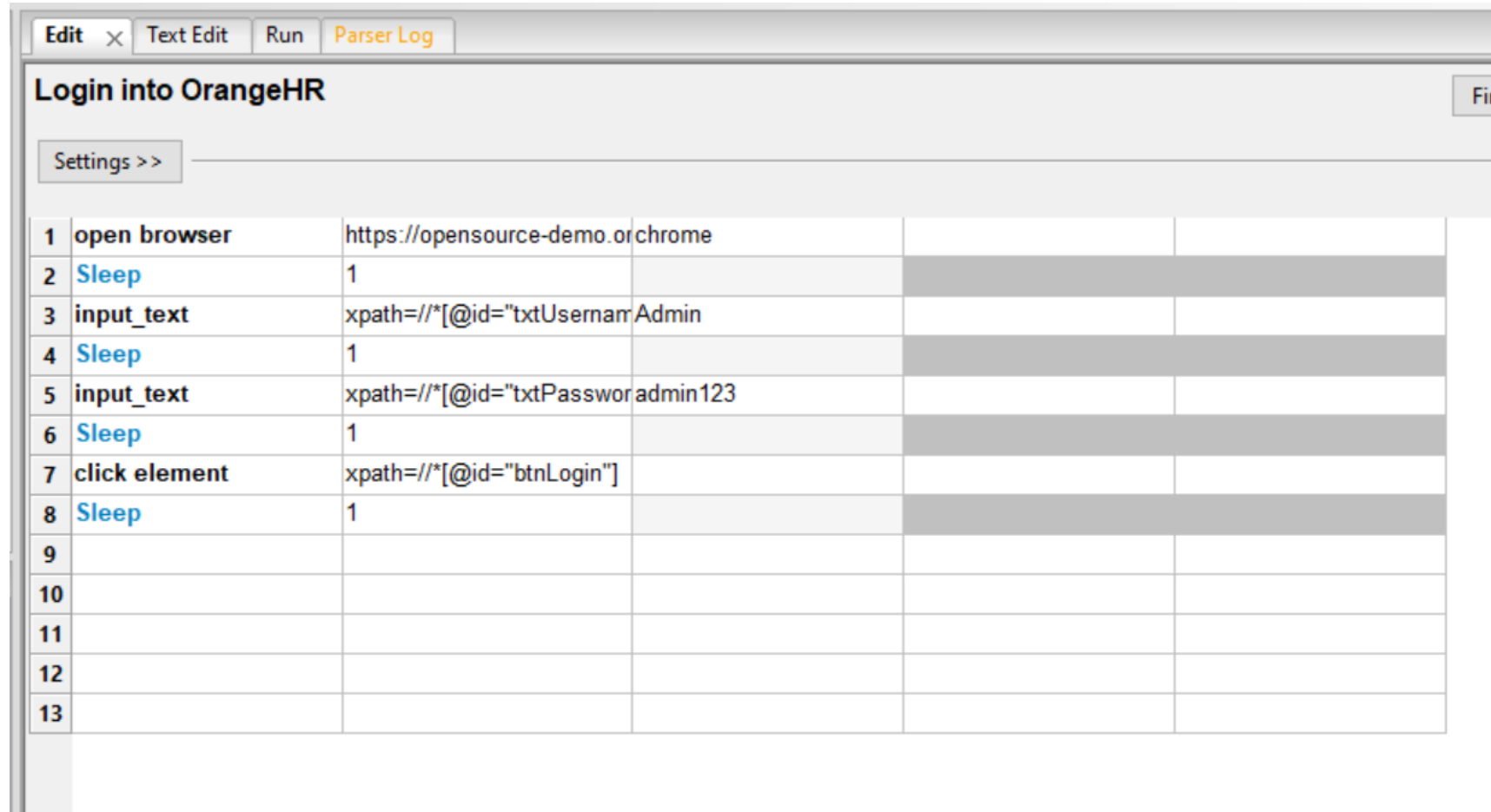


STEP 4.1: Create user keyword (Tab 'edit')

- i. Right-click on resource file eg: add_license.robot
- ii. Select new user keyword
- iii. Enter new user keyword and click 'OK'



Write steps, xpath and value into scenario file as shown in below:
Then, save the file.



The screenshot shows the Orange3 Scenario Editor interface. At the top, there are tabs for 'Edit', 'Text Edit', 'Run', and 'Parser Log'. The main title is 'Login into OrangeHR'. Below the title is a 'Settings >>' button. The scenario is represented as a table with 5 columns: Step Number, Action, XPath, Value, and an empty column. The steps are as follows:

Step Number	Action	XPath	Value	
1	open browser	https://opensource-demo.or	chrome	
2	Sleep	1		
3	input_text	xpath=//*[@id="txtUsernam	Admin	
4	Sleep	1		
5	input_text	xpath=//*[@id="txtPasswor	admin123	
6	Sleep	1		
7	click element	xpath=//*[@id="btnLogin"]		
8	Sleep	1		
9				
10				
11				
12				
13				

STEP 4.2 : Create user keyword (Tab 'Text edit')

- i. Directly click on 'Text Edit' Tab
- ii. Write steps, xpath and value into scenario file as shown in below:
- iii. Then save the file

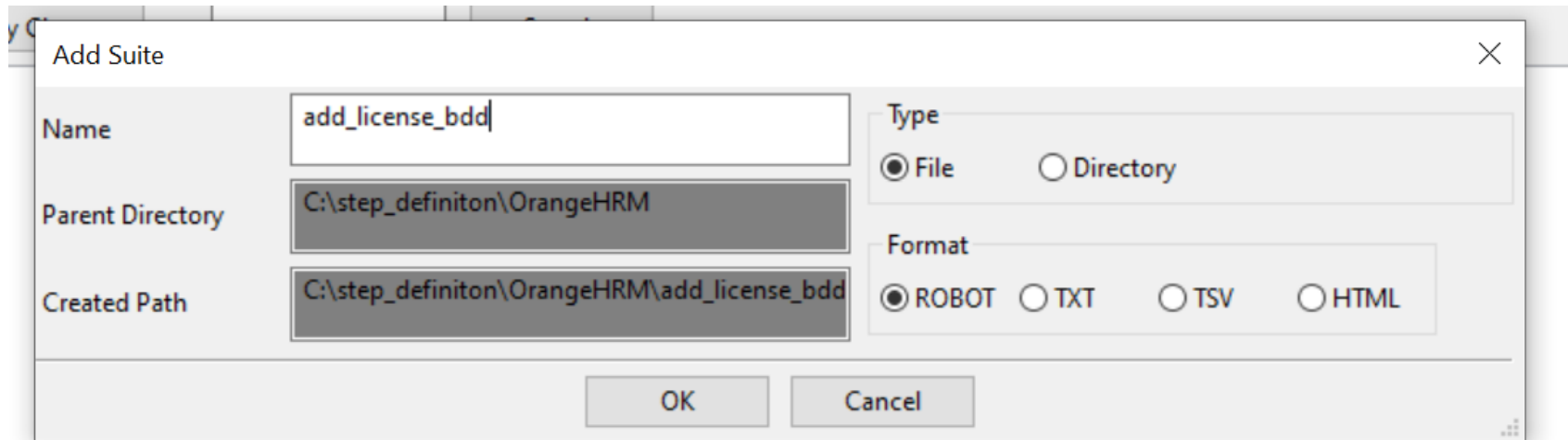


The screenshot shows the 'Text Edit' tab in the OrangePyTest application. The interface includes a toolbar with 'Edit', 'Text Edit', 'Run', and 'Parser Log' buttons. Below the toolbar is a search bar with 'Apply Changes' and 'Search' buttons. The main area displays a scenario file with the following content:

```
1 *** Settings ***
2 Library           SeleniumLibrary
3
4 *** Keyword ***
5 Login into OrangeHR
6   open browser    https://opensource-demo.orangehrmlive.com/    chrome
7   Sleep           1
8   input_text      xpath=//*[@id="txtUsername"]    Admin
9   Sleep           1
10  input_text      xpath=//*[@id="txtPassword"]    admin123
11  Sleep           1
12  click element   xpath=//*[@id="btnLogin"]
13  Sleep           1
14
```

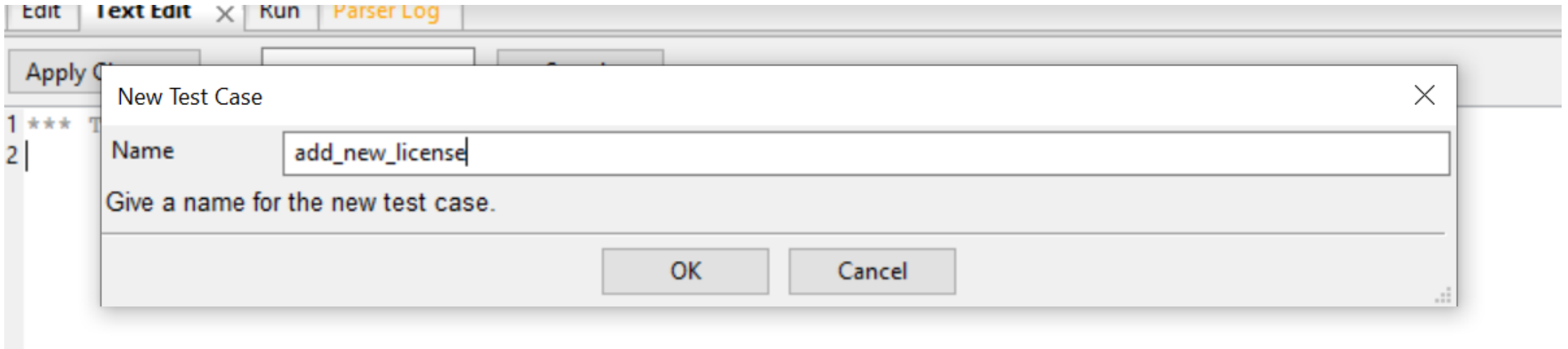
STEP 5 : Create new suite for bdd

- i. Right-click on feature folder and choose new suite
- ii. Name the bdd file eg: add_license_bdd
- iii. Choose type 'File' and format 'ROBOT' then click 'OK'



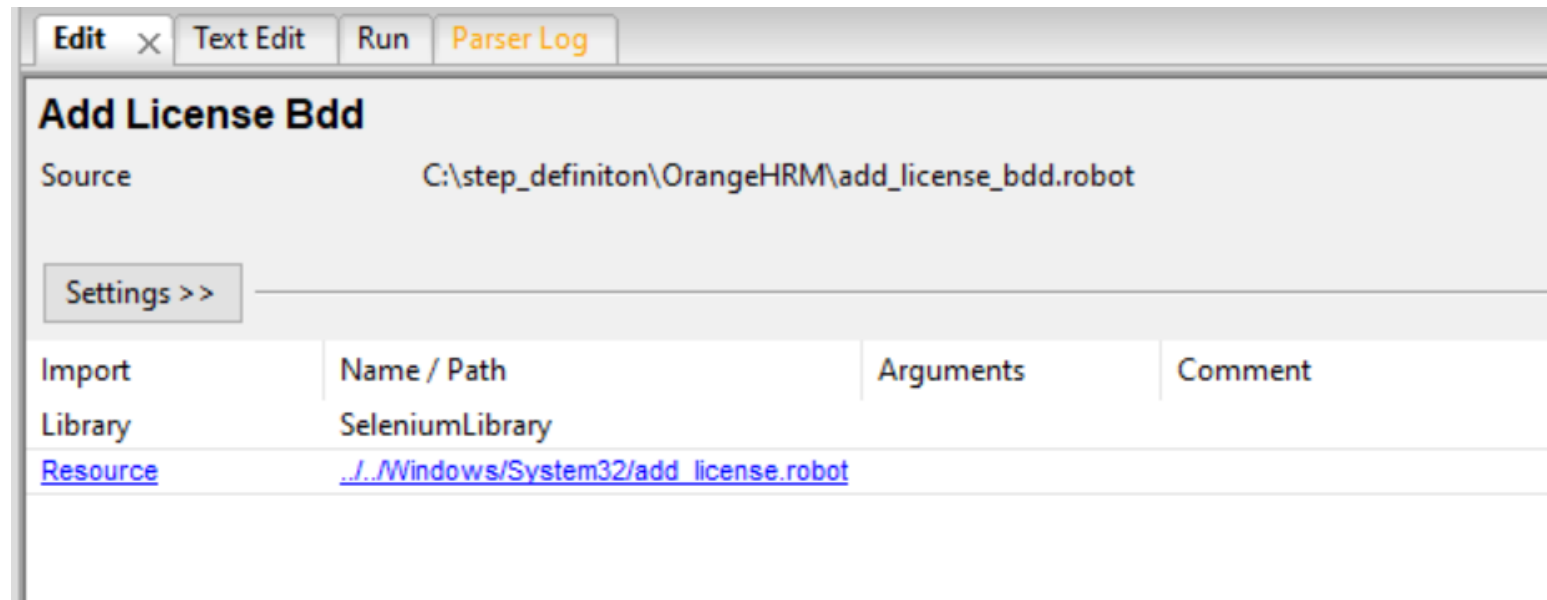
STEP 6 : Create new test case

- i. Right-click on bdd file and choose new test case.
- ii. Name the tase case. Eg: add_new_license
- iii. Click 'OK'



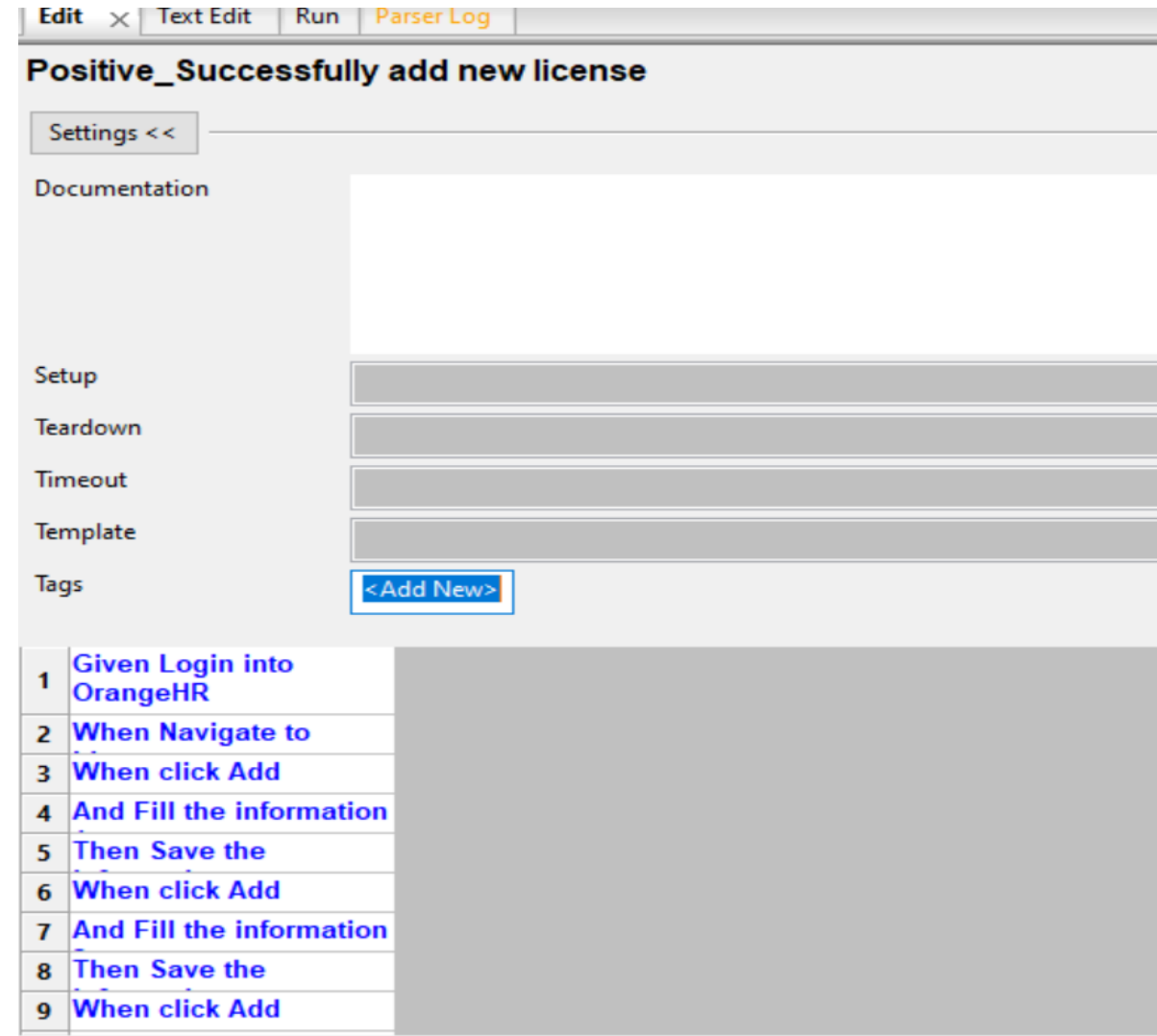
STEP 7 : Import library and resources

- i. Click on library and import Selenium Library
- ii. Click on resources and import your resources file
- iii. Then, click 'OK'



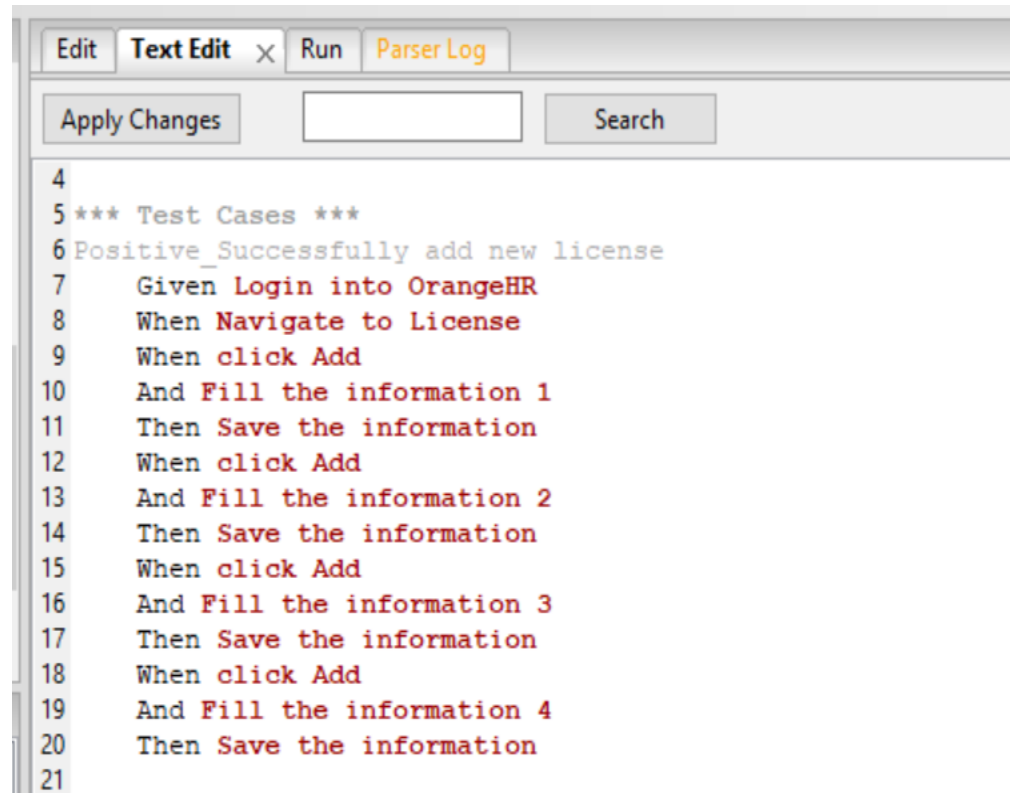
STEP 8 : Write scenarios

- i. You can write and edit your scenarios through 'Edit' tab
- ii. If the colour turn out 'blue' means the scenarios written are valid from your resources file.



STEP 9 : Write scenarios

- i. You can also write and edit your scenarios through 'Text Edit' tab



The screenshot shows a software interface with a 'Text Edit' tab selected. The tab bar includes 'Edit', 'Text Edit' (with a close button), 'Run', and 'Parser Log'. Below the tab bar is a toolbar with an 'Apply Changes' button, a text input field, and a 'Search' button. The main text area contains a Gherkin scenario for adding a new license, with line numbers 4 through 21 on the left margin. The scenario text is as follows:

```
4
5 *** Test Cases ***
6 Positive_Successfully add new license
7     Given Login into OrangeHR
8     When Navigate to License
9     When click Add
10    And Fill the information 1
11    Then Save the information
12    When click Add
13    And Fill the information 2
14    Then Save the information
15    When click Add
16    And Fill the information 3
17    Then Save the information
18    When click Add
19    And Fill the information 4
20    Then Save the information
21
```

STEP 10: Run test case

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

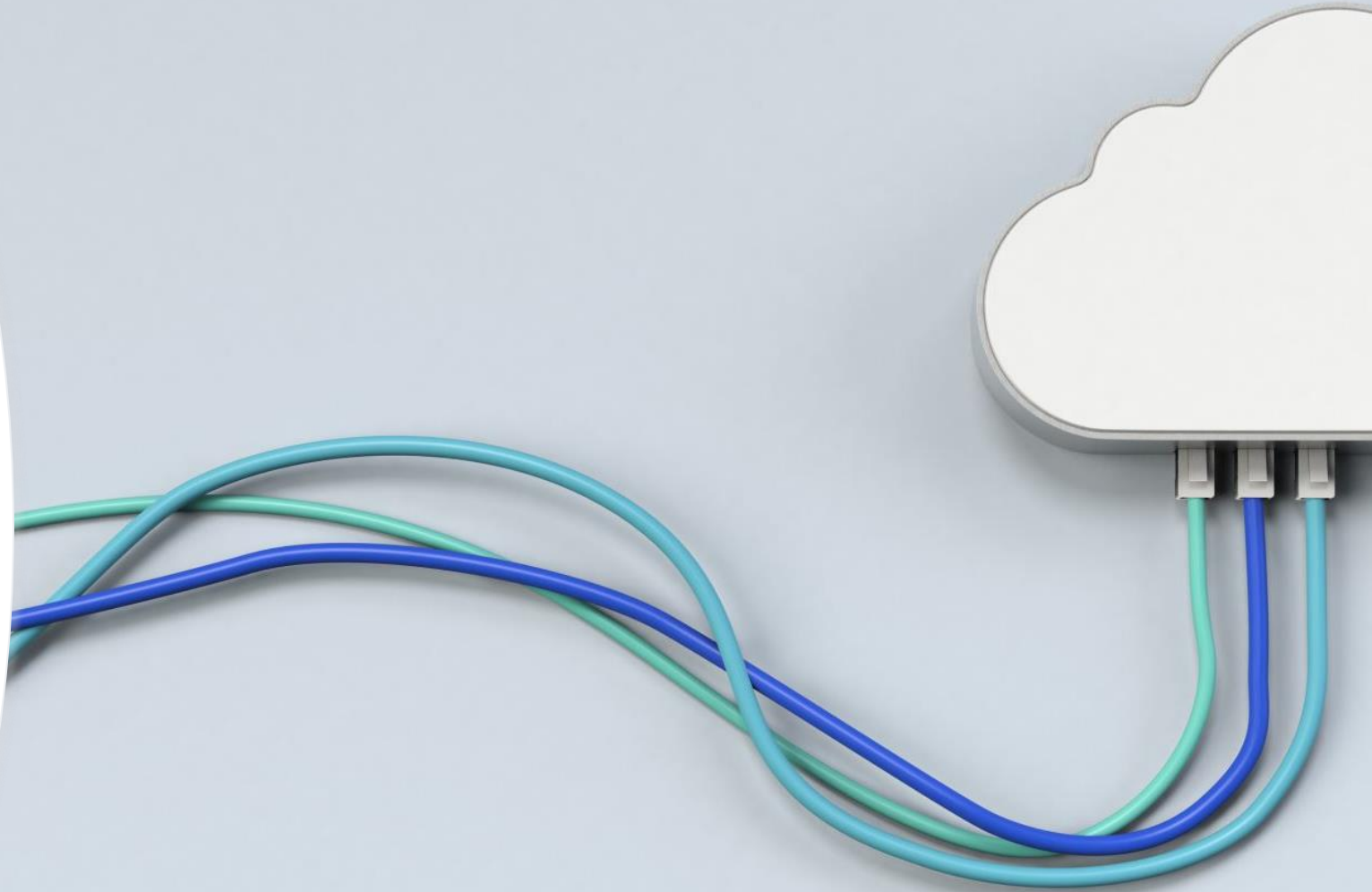
STEP 11: Test case report

- i. Click report button on 'Run' tab to check the report.



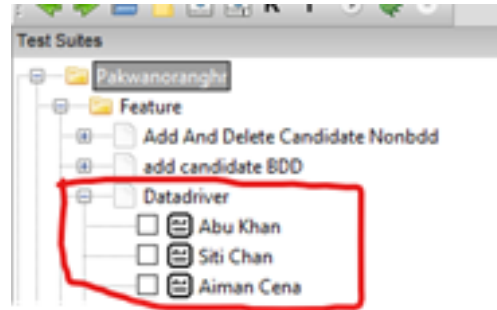
Data Driven

*Learn creating keywords and variable first

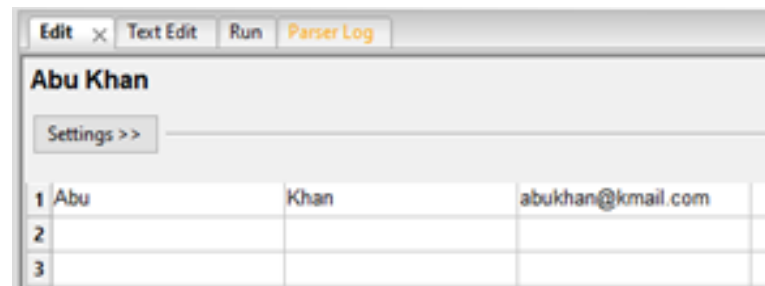


Steps:

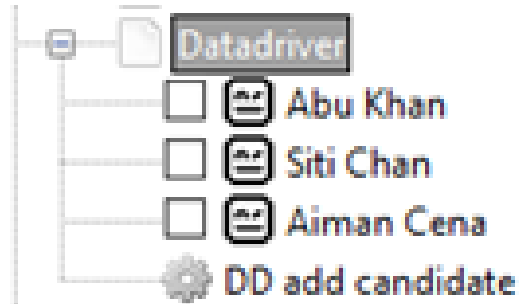
1. Create test suite for data driven testing (exp: named as Datadrivers)
2. Create test cases that represent data. For this example, create three test cases



3. Click on test case and click Edit tab. Inside test cases, put data value for testing. For example, each test cases have three data value such as first name, last name and email. Then, put data value to another test cases



4. On Datadriven test suite, create Keywords (exp: named as DD add candidate)



5. Click on Datadriven and then click on Text Edit tab.

6. Under ****Setting****, type *Test Template DD add candidate* (name for keyword)

```
1 *** Settings ***
2 Test Template      DD add candidate
3 Library            SeleniumLibrary
4
5 *** Test Cases ***
6 Abu Khan
7     Abu    Khan    abukhan@kmail.com
8
9 Siti Chan
10    Siti    Chan    sitichan@kmail.com
11
12 Aiman Cena
13    Aiman   Cena    aimancena@kmail.com
14
15 *** Keywords ***
16 DD add candidate
17
```

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

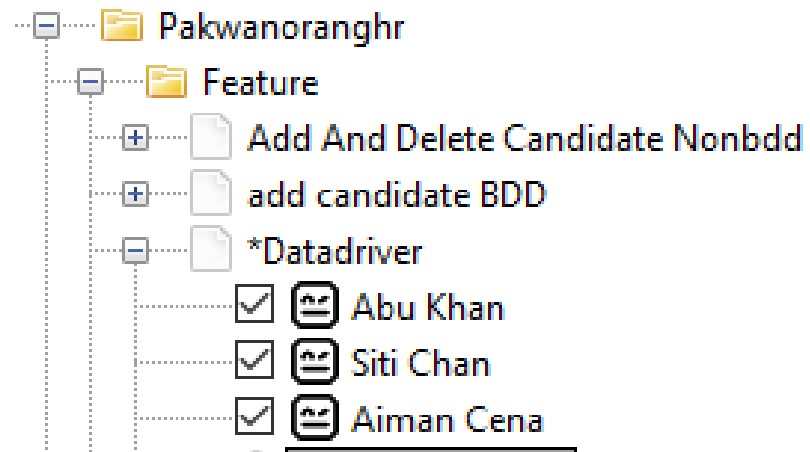
```
1 *** Settings ***
2 Test Template      DD add candidate
3 Library            SeleniumLibrary
4
5 *** Test Cases ***
6 Abu Khan
7     Abu    Khan    abukhan@kmail.com
8
9 Siti Chan
10    Siti    Chan    sitichan@kmail.com
11
12 Aiman Cena
13    Aiman    Cena    aimancena@kmail.com
14
15 *** Keywords ***
16 DD add candidate
17     [Arguments]    ${dd_fname}    ${dd_lname}    ${dd_email} ←
18
19
```

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

Edit x Text Edit Run ParserLog		
DD add candidate		
Settings >>		
1	open browser	https://opensource-demo.orchrome
2	Sleep	1
3	input_text	xpath=//*[@id="txtUsername"]Admin
4	Sleep	1
5	input_text	xpath=//*[@id="txtPassword"]admin123
6	Sleep	1
7	click element	xpath=//*[@id="btnLogin"]
8	Sleep	1
9	click element	xpath=//*[@id="menu_recru"]
10	Sleep	1
11	click element	xpath=//*[@id="btnAdd"]
12	Sleep	1
13	input_text	xpath=//*[@id="addCandidate_firstname"]\${dd_fname}
14	Sleep	1
15	input_text	xpath=//*[@id="addCandidate_lastname"]\${dd_lname}
16	Sleep	1
17	input_text	xpath=//*[@id="addCandidate_email"]\${dd_email}
18	Sleep	1
19	Select From List By Label	xpath=//*[@id="addCandidate_vacancy"]Senior QA Lead
20	Sleep	1
21	click element	xpath=//*[@id="btnSave"]
22	Sleep	3
23	click element	xpath=//*[@id="btnBack"]
24	Sleep	2

```
DD add candidate
[Arguments]    ${dd_fname}    ${dd_lname}    ${dd_email}
Sleep          1
input_text     xpath=//*[@id="addCandidate_firstname"]    ${dd_fname}
Sleep          1
input_text     xpath=//*[@id="addCandidate_lastname"]    ${dd_lname}
Sleep          1
input_text     xpath=//*[@id="addCandidate_email"]    ${dd_email}
Sleep          1
Select From List By Label    xpath=//*[@id="addCandidate_vacancy"]    Senior QA Lead
Sleep          1
click element    xpath=//*[@id="btnSave"]
Sleep          3
click element    xpath=//*[@id="btnBack"]
Sleep          2
```

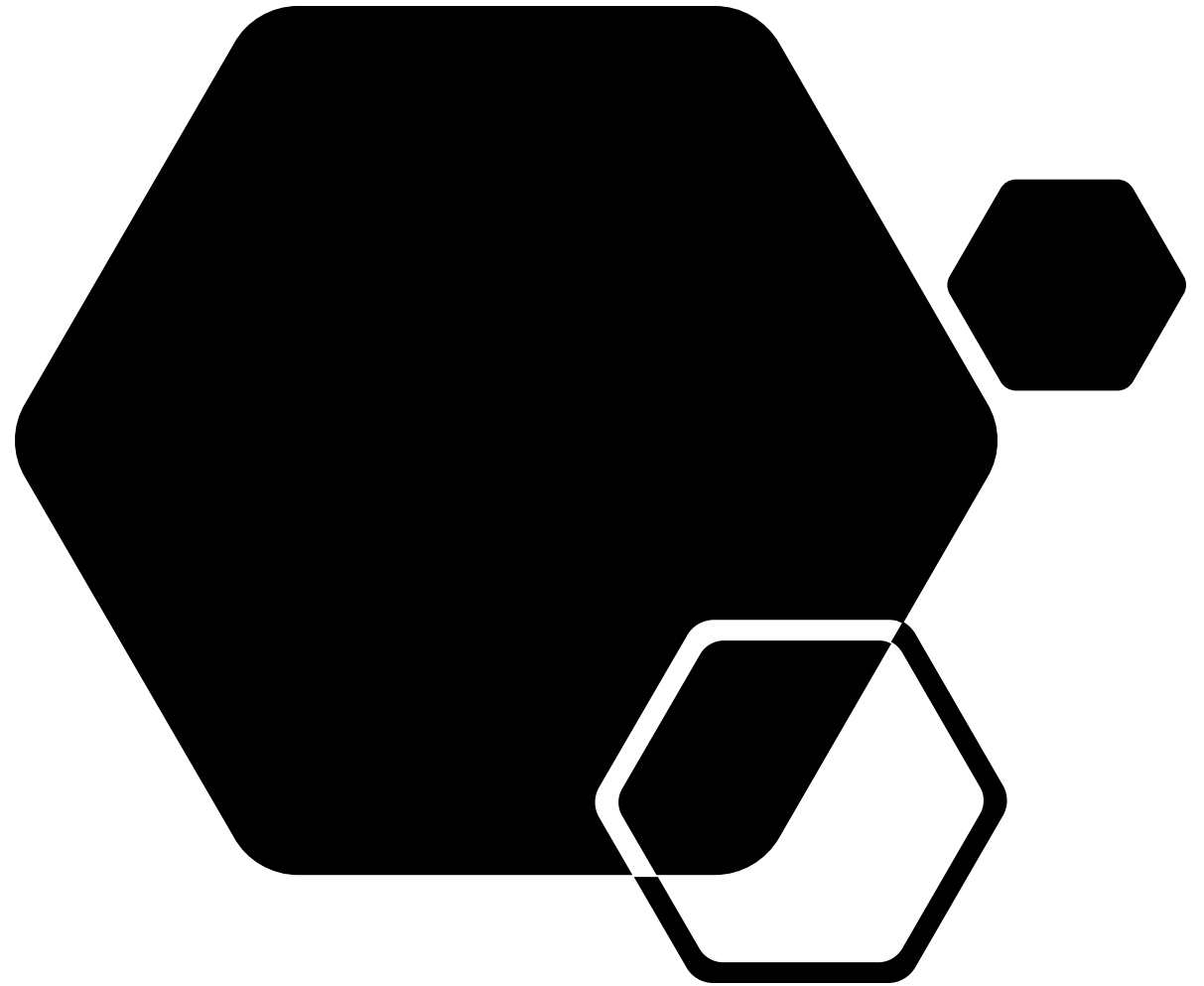
9. Tick test cases.



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

Data Driven + BDD

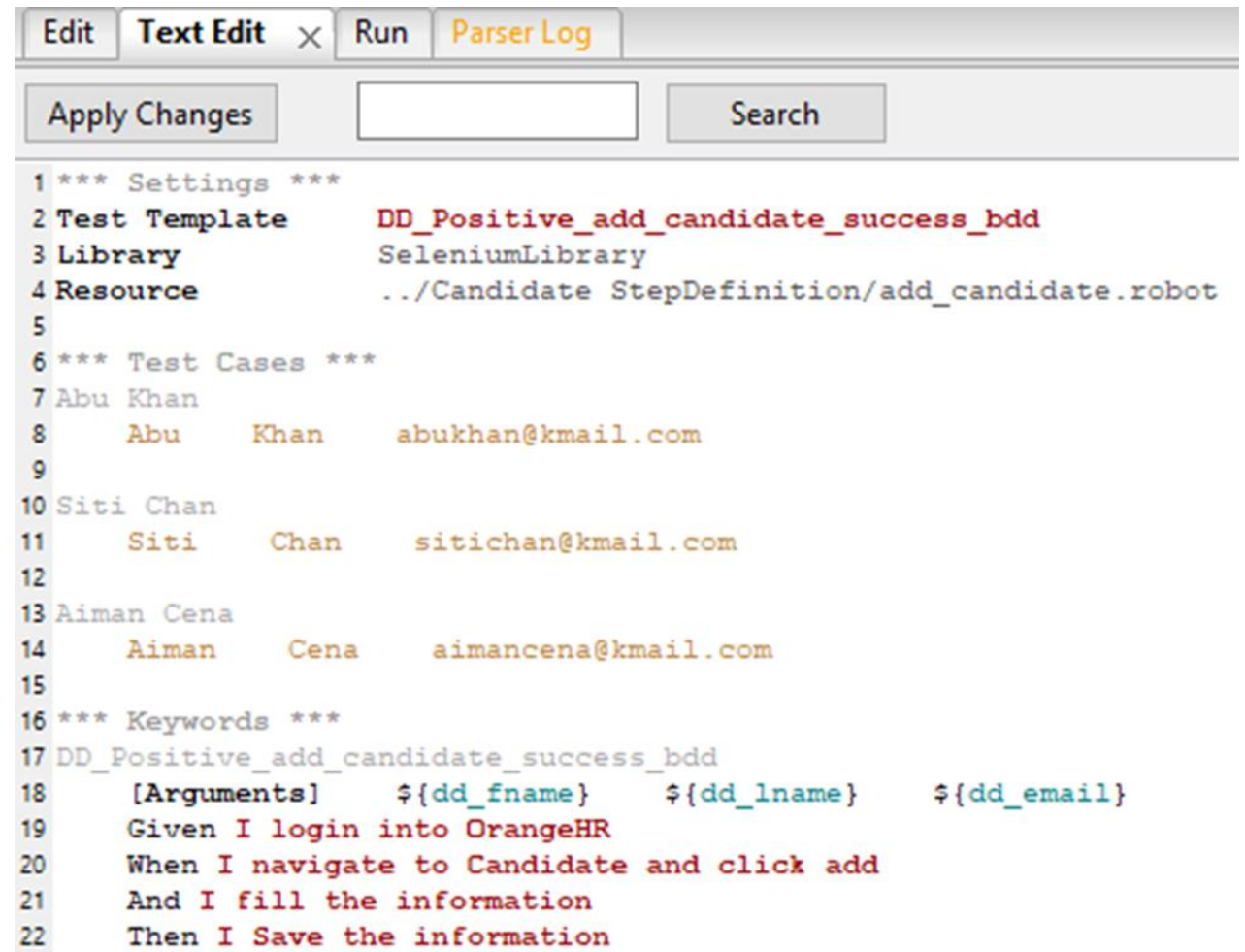
*Learn creating keywords and variable
first



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



The screenshot shows a text editor window with a toolbar at the top containing 'Edit', 'Text Edit', 'Run', and 'Parser Log'. Below the toolbar are buttons for 'Apply Changes', a search input field, and a 'Search' button. The main text area contains BDD test suite code with line numbers 1 through 22. The code defines settings, test cases for three candidates (Abu Khan, Siti Chan, and Aiman Cena), and a keyword for adding a candidate with arguments for first name, last name, and email.

```
1 *** Settings ***
2 Test Template      DD_Positive_add_candidate_success_bdd
3 Library            SeleniumLibrary
4 Resource           ../Candidate StepDefinition/add_candidate.robot
5
6 *** Test Cases ***
7 Abu Khan
8     Abu            Khan            abukhan@kmail.com
9
10 Siti Chan
11     Siti           Chan            sitichan@kmail.com
12
13 Aiman Cena
14     Aiman          Cena            aimancena@kmail.com
15
16 *** Keywords ***
17 DD_Positive_add_candidate_success_bdd
18     [Arguments]    ${dd_fname}    ${dd_lname}    ${dd_email}
19     Given I login into OrangeHR
20     When I navigate to Candidate and click add
21     And I fill the information
22     Then I Save the information
```

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

'I fill information \${firstname},
\${lastname} and \${email}'

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

The screenshot displays the Robot Framework IDE interface. On the left, a test suite named 'delete_candidate.robot' is visible, containing several keywords. The main area shows a keyword table with 10 rows. A red rectangle highlights the third column of the first five rows, which contain variable names: \${ac_lname}, \${ac_fname}, \${ac_email}, and \${email}.

I fill the information for \${firstname}, \${lastname} and \${email}		
Settings >>		
1	input_text	\${ac_lname}
2	Sleep	1
3	input_text	\${ac_fname}
4	Sleep	1
5	input_text	\${ac_email}
6	Sleep	1
7	Select From List By Label	Senior QA Lead
8	Sleep	1
9	Choose File	D://kerja/pakwanoranghr/Fil
10	Sleep	1

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

'I fill information `${dd_fname}`, `${dd_lname}` and `${dd_email}`'

Note: name of variables must be same with argument variable

```
16 *** Keywords ***
17 DD_Positive_add_candidate_success_bdd
18     [Arguments]    ${dd_fname}    ${dd_lname}    ${dd_email}
19     Given I login into OrangeHR
20     When I navigate to Candidate and click add
21     And I fill the information for ${dd_fname}, ${dd_lname} and ${dd_email}
22     Then I Save the information
23
```

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-L9l2l8K8dEhw6HXy-Z-33w3>