Selenium Python Project

PyCharm | POM | PyTest | Allure Reports | Git | Jenkins
from scratch

Python
Pycharm IDE

====== BASIC SETUP =======

Step 1 : Pycharm - create new project with virtual env

Step 2 : Settings > Project > Interpreter > add
selenium

Step 3: Create 6 folders:
tests
pages
drivers
utils
reports
screenshots

Step 4 : Under drivers folder add driver exe for browsers
chrome
firefox...

Step 5 : Under tests folder create a new python file (login_test.py)

Step 6 : Create a simple login test.
run & validate
https://opensource-demo.orangehrmlive.com/

====== PYTEST SETUP =======

Step 7 : Add pytest package -
Settings > Project > Interpreter > add

Step 8 : Create functions (pytest)
*** take care of naming convention as per pytest ***
test_*.py. OR *_test.py
test_setup
test_login
test_logout
Add pytest fixture to test_setup
Run & validate

@pytest.fixture(scope="session")

Step 9 : Create class TestLogin():
add @pytest.fixture(scope="class")
Run & Validate

====== PAGE OBJECT MODEL (POM) SETUP =======

Step 10 : Under pages folder create python file to create classes for pages

---

Step 11 : Add the page objects and actions in respective classes

---

Step 12 : Create objects for pages in test script and use the functions

Run & Validate

====== PAGE OBJECT MODEL (POM) SETUP COMPLETED =======

Step 13 : In utils folder create python file utils and add constants

```
URL       =   "https://opensource-demo.orangehrmlive.com/"
USERNAME    =   "Admin"
PASSWORD    =   "admin123"

In tests
from utils import utils as utils
```

Step 14 : Add package pytest-html and show html reports

--html

--self-contained-html

Step 15 : Add conftest.py

Put the setup teardown here

Remove setup and yield from test class

```
@pytest.fixture(scope="class")
def test_setup(request):
        from selenium import webdriver
        # global driver
        driver = webdriver.Chrome(executable_path="C:/Users/PycharmProjects/

                                AutomationFramework/drivers/chromedriver.exe")
        driver.implicitly_wait(5)
        driver.maximize_window()
        request.cls.driver = driver
```

```
yield
driver.close()
driver.quit()
print("Test Completed")
```

add pytest fixture on login_test class

```
@pytest.mark.usefixtures("test_setup")
```

Run & Validate

References:

https://stackoverflow.com/questions/34466027/in-py-test-what-is-the-use-of-conftest-py-files

---

Step 16 : In conftest.py create function to get browser name from arguments

https://stackoverflow.com/questions/40880259/how-to-pass-arguments-in-pytest-by-command-line

```
def pytest_addoption(parser):
    parser.addoption("--browser", action="store", default="chrome", help="Type in browser name e.g. chrome OR firefox")

browser = request.config.getoption("--browser")
if browser == 'chrome':
    driver = webdriver.Chrome()
```

====== ALLURE REPORTS ======

Step 17 : Add allure reports package

Run with allure reports

Add assertion & validate

```
x = driver.title
```

```
assert x == "abc"
```

Step 18 : Add try except block

```
try:

except AssertionError as error:
    print("Assertion error occurred")
    print(error)
    raise

except:
    print("Some exception occurred")

else:
    print("No exceptions occurred")

finally:
    print("This block will always execute | Close DB")
```

References:
https://realpython.com/python-exceptions/

Step 19 : Add statements to attach screenshot in allure reports

```
import allure

allure.attach(self.driver.get_screenshot_as_png(),
name="screenshot",
    attachment_type=allure.attachment_type.PNG)
```

To get current timestamp

```
import moment
```

test

```
import moment
x = moment.now().strftime("%H-%M-%S_%m-%d-%Y")
print(x)
```

Add function to store screenshot physically

```
driver.get_screenshot_as_file("C:/Users//AutomationFramework/
screenshots/"+screenshotName+".png")
```

Add a function to get test name

To get test name
https://www.stefaanlippens.net/python_inspect/
def whoami():
return inspect.stack()[1][3]

```
currTime = moment.now().strftime("%H-%M-%S_%m-%d-%Y")
testName = utils.whoami()
screenshotName = testName+"_"+currTime
```

Run & Validate with allure reports

====== Git & GitHub =====

Step 20 : Add to GitHub
Connect to GitHub account on pycharm
VCS > Import Into Version Control > Share project on GitHub

A new user can now download or clone this project

https://github.com/Raghav-Pal/PythonAutomationFramework_1.git

---

====== Jenkins =====

Step 21 : Download and Setup Jenkins
Add a new Jenkins Job
In Job:
Pull project from GitHub
Run tests
Publish Allure Reports

Step 22 : Create Jenkins Pipeline

Pipeline - sequence of jobs chained together

BUILD > DEPLOY > TEST > RELEASE

Step 23 : Run the complete project from jenkins

==========

All the best & Keep learning

Raghav Pal

==========