**Selenium Hybrid Framework**

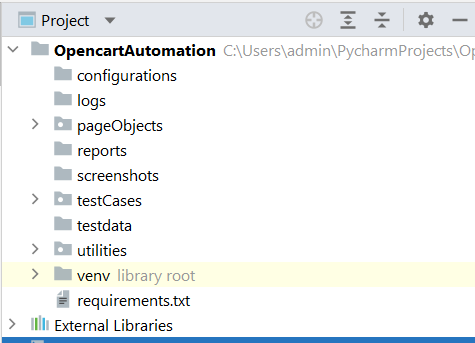
**(Python, Selenium, PyTest, Page Object Model, HTML Reports)**

**Step 1: Create new Project & Install Required Packages/plugins**

* **Pytest – Python built-in unit test framework (5.4.3 Preferred)**
* **selenium - selenium libraries**
* **webdriver-manager – for drivers**
* **pytest-html – reports**
* **pytest-xdist - parallel testing**
* **openpyxl – xl file**
* **allure-pytest - report**

**Create requirements.txt file contains all packages and install it.**

**Step 2: Create Folder Structure**



**Step 3: Automating Register account test case**

**3.1: Create page object classes for HomePage & AccountRegistration page under "pageObjects"**

**3.2: Create conftest.py under "testCases" with driver manager.**

**3.3: Create AccountRegistration testcase under "testCases"**

**3.4 Write a utility file to generate random string for email.**

**Step 4: capture screenshot on failures**

**4.1. Update AccountRegistration Test case with capture Screenshot under "testCases"**

**Step 5: Read common values from ini file.**

**5.1: Add "*config.ini*" file in "configurations" folder.**

**5.2: Create "*readProperties.py*" utility file under *utilities* package to read common data.**

**5.3: Replace hard coded values in AccountRegistration testcase.**

**Step 6: Adding logs to test case**

**6.1: Add *customLogger.py* under *utilities* package.**

**6.2: Add logs to AccountRegistration test case.**

**Step 7: Run Tests on Desired Browser(Cross Browser Testing)/Parallel**

**7.1: update *contest.py* with required fixtures which will accept command line argument (browser).**

**7.2: Pass browser name as argument in command line**

**To Run tests on desired browser**

pytest -s -v .\testCases\test\_001\_AccountRegistration.py --browser edge

**To Run tests parallel**

pytest -s -v **-n=3** .\testCases\test\_001\_AccountRegistration.py --browser edge

**Step 8: Generate pytest HTML Reports**

8.1: Update conftest.py with pytest hooks

***pytest -s -v --html=reports\report.html --capture=tee-sys .\testCases\test\_001\_A***

***ccountRegistration.py --browser chrome***

**Step 9: Automate Login Test case**

**9.1: Create page object class LoginPage under "pageObjects"**

**9.2: Create Login test under "testCases"**

**Step 10: Automating Data Driven Test Case**

**10.1: Prepare test data in Excel sheet, place the excel file inside the “testData” folder.**

**10.2: Create "*ExcelUtils.py*" utility class under utilities package.**

**10.3 : Create MyAccount PageObject class**

**10.3: Create LoginDataDrivenTest under testCases**

**10.4: Run the test case**

**Step 11: Grouping Tests**

11.1: Grouping markers( Add markers to every test method)

***@pytest.mark.sanity***

***@pytest.mark.regression***

11.2: Add Marker entries in pytest.ini file

**pytest.ini**

--------

***[pytest]***

***markers =***

***sanity***

***regression***

11.3: Select groups at run time

***-m "sanity"***

***-m "regression"***

***-m "sanity and regression"***

***-m "sanity or regression"***

Run Command:

***pytest -s -v -m "sanity or regression" .\testCases***

**Step 12: Run Tests in Command Prompt & run.bat file.**

**12.1: Create requirements.bat file contains packages**

pip install pytest==5.4.3

pip install selenium

pip install webdriver-manager

pip install pytest-html

pip install pytest-xdist

pip install pytest-ordering

pip install openpyxl

pip install allure-pytest

**12.2 : Create run.bat file**

**pytest -s -v -m "sanity".\testCases**

**12.2 Open command prompt as Administrator and then run *run.bat file***

**Step 13: Push the Code to Git & GitHub Repository**

**Git workflow**

1) git init --> Create an Empty git repository(Local repository)

2) git config --global user.name "your name"

got config --global user.email "your email"

3) git status ---> to know the status of the files

4) git add -A ---->add all the files into staging/indexing area

git add filename.java --> add specific file into staging/indexing area

git add \*.java

5) git commit -m "user comment"

Github

--------------

1) Create new account (Sign up)

2) Login to github ---> create a new empty remote repository

Remote Repo url:

https://github.com/pavanoltraining/OpencartV11.git

3) Create a token

Reference link: https://docs.github.com/en/github/authenticating-to-github/keeping-your-account-and-data-secure/creating-a-personal-access-token

Create Personal Access Token on Github

---------------

From your Github account, go to Settings => Developer Settings => Personal Access Token => Generate New Token (Give your password) => Fillup the form => click Generate token => Copy the generated Token.

4) push your code into Remote repository

git remote add origin https://github.com/pavanoltraining/OpencartV11.git ---> only once

git push -u origin master

Pull files from remote to local

-------------------------------

git pull origin master

**2nd round**

---------

add

commit

push

pull

status

**Step 14: Run Tests using Jenkins**