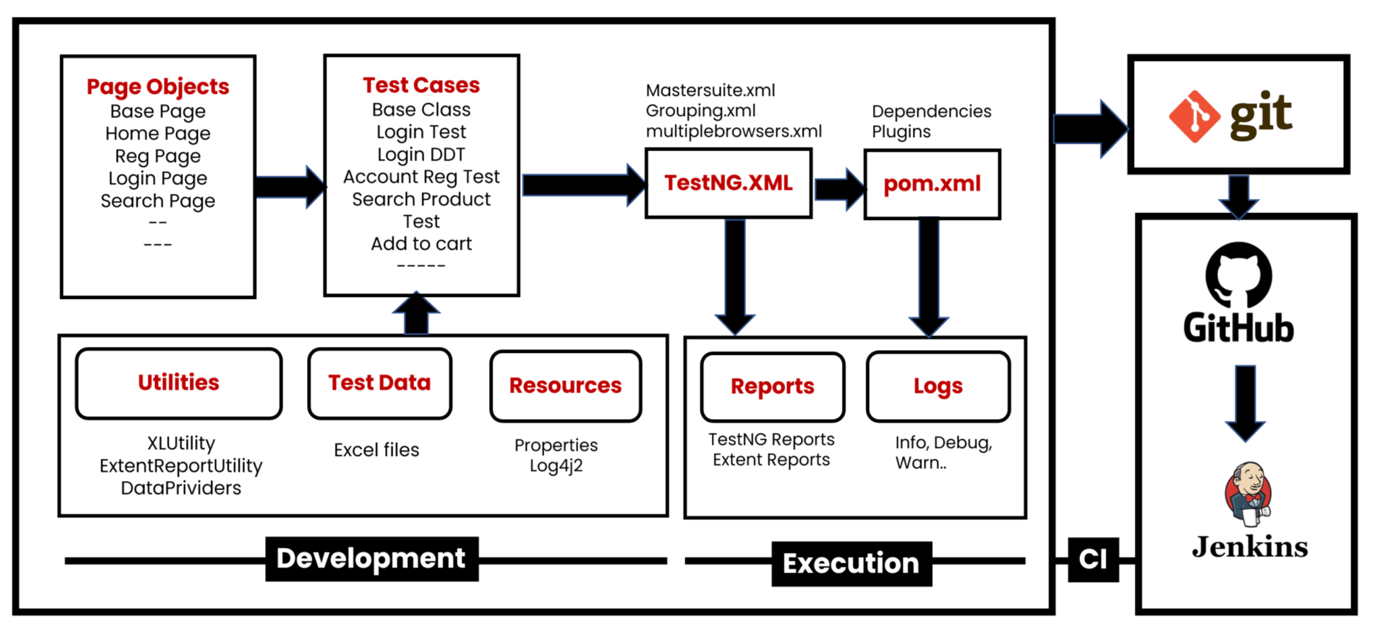
**Hybrid Automation Framework**

**(Java, Selenium, TestNG, Maven & Page Object Model)**

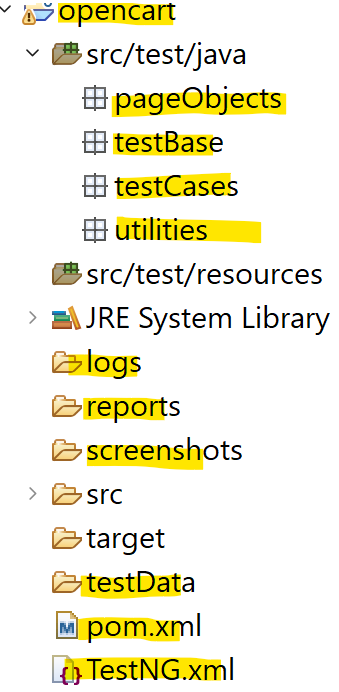


**Create a new Maven Project**

**Add required dependencies in pom.xml (Please check links below)**

https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java https://mvnrepository.com/artifact/org.apache.poi/poi https://mvnrepository.com/artifact/org.apache.poi/poi-ooxml https://mvnrepository.com/artifact/org.apache.logging.log4j/log4j-core https://mvnrepository.com/artifact/org.apache.logging.log4j/log4j-api https://mvnrepository.com/artifact/commons-io/commons-io https://mvnrepository.com/artifact/org.apache.commons/commons-lang3 https://mvnrepository.com/artifact/org.testng/testng https://mvnrepository.com/artifact/com.aventstack/extentreports

**Create Folder Structure**



**Development of Hybrid Driven Framework**

**1) Test case: Account Registration**1.1: Create BasePage under "**pageObjects**" which includes only constructor. This will be invoked by every Page Object Class constructor (Re-usability).  
1.2: Create Page Object Classes for HomePage, RegistrationPage under **pageObjects** package. (These classes extends from BasePage).  
1.3: Create AccountRegistrationTest under "**testCases**"  
1.4: Create BaseClass under **testBase** package and copy re-usable methods.  
1.5: Create re-usable methods to generate random numbers and strings in BaseClass.

**2) Adding logs to test case (log4j2)**2.1: Add **log4j2.xml** file under **src/test/resourses**.  
2.2: Update BaseClass.  
2.3: Add log statements to AccountRegistrationTest.

**3) Run Tests on Desired Browser/Cross Browser/Parallel**

3.1: Create testng.xml file to Run Test Cases and parameterize browser name and OS to BaseClass →setup() method.

3.2: Update BaseClass →setup() method, launch browser based on conditions.

3.3: Maintain separate xml to run tests multiple browsers parallelly.

**4) Read Common values from config.properties file.**4.1: Add config.properties file under **src/test/resoures**.  
4.2: Update BaseClass →setup() method, add script to load config.properties file.

4.3: Replace hard coded values in Test Cases like url, username, password etc...

**5) Login Test Case**

5.1: Create and update page object classes. LoginPage, MyAccountPage – new classes HomePage – update by adding login link element

5.2: Create LoginTest

5.3: Add entry testng.xml

**6) Data Driven Login Test**

6.1: Prepare test data in Excel, place the excel file inside the **testData** folder.

6.2: Create ExcelUtility class under **utilities** package.

6.3: Update Page Object class MyAccountPage , add logout link element)

6.4 : Create DataProviders class in **utilities** package to maintain data providers for data driven tests.

6.5: Create LoginDataDrivenTest under **testCases** package.

6.6: Add an Entry in testng.xml file

**7) Grouping Tests.**

7.1: Add all test cases into specific group ( sanity, regression , master etc.).

7.2: Also add BaseClass methods setup() & teardown() to all groups.  
7.3: Create separate TestNG xml file(grouping.xml) to run groups and include groups which we

want to execute.

**8) Add Extent Reports to Project**

8.1: Create ExtentReportUtility utility class under **utilities** package.

8.2: Add captureScreen() method in BaseClass

8.3: Add ExtentReportUtility (Listener class) entry in testng.xml file.

8.4: Make sure WebDriver is *static* in BaseClass, we refer same driver instance in ExtentReportUtility.

**9) Run Failed Tests.**

test-output→testng-failed.xml

**10) Run Tests on Selenium Grid**

**Pre-requisite:** Grid Standalone/Hub & Node setup (Refer the Grid Setup document)

10.1: Add **execution\_env=local/remote** in config.properties file under resources folder.

10.2: Update setup() method in the BaseClass (capture execution environment from config.properties file then add required capabilities of OS & Browser in conditions).

10.3: Run the tests from testing.xml

1. **Run Tests on Docker with Selenium Grid Environment.**

(note : Host os & then virtual machine os (different os). Buying virtual machine is costly that’s why we use docker

Here instaed of virtual machine we will have containers)

\*\* Refer the DockerSetup Document.

**DockerHub** (https://hub.docker.com/)will have multiple images(for os, browsers) which you can download & create container of it & use that container to test your test cases.

Download docker & to check the version of it use below cmd on cmd prompt :

docker version (if it is showing some configuration error then open window services, search for docker desktop service, & click on stop & then start )

**12) Run Tests using Maven pom.xml, Command Prompt & run.bat file.**

**13) Push the Code to Git & GitHub Repository**

**14) Run Tests using Jenkins.**