**Hybrid Framework Explanation**

->Ours is a Hybrid framework which is combination of TestNg , Data Driven, POM (object repository) and Keyword Driven framework.

->Our entire framework was built upon **maven** project. Maven contains **pom.xml** file which includes maven sure fire plugin and maven compiler plugin and all the dependencies (testNg, Apache POI (apache.poi is necessary for data driven from excel file), Selenium-java, Extent Report, htmlunit-driver (headless) , WebDriver Manager , ojdbc6 and API (groupId , artifactId and version for each dependency are included inside dependencies tag by connecting to maven repository)

**Modules** / **Structure** :

src/test/java -> All java files (testNg and Java) are placed in respective packages(module names)

->utiltiy package contain :

1.**libray business functions** : All common functions that are used across the project like ReadingPropertyFile, LaunchBrowser, StartExtentReports, WaitForPageToLoad, ScreenShot, SelectValueFromDropDown, DoubleClick, ContextClick (Right click), SwitchToFrame, ScrollDown, ScrollIntoView, findElement which accept locator type and return web Element , findElements which will accept locator type and return list of WebElemets, ExplicitWait , WriteToExcelFile , validatingLinks , PressEnterKey..etc

(functions/methods are declared as public static).

2.**Constants** : All the constants that we are using in our project (Quantity, locators, ..etc)

3.**Object Repository** (**POM**): All the locators are placed in the object repository .(declared as public static final string).

src/test/resources ->contain the resources of our project like extent reports, testdata.xlsx file, .property file, queries, payloads (body of API).

screenshots and downloads folders.

->Our Test Execution is going to start from the selected testNg.xml file. We are having separate xml files for sanity smoke and B2B or E2E or Regression. Respective xml files will be picking up the respective classes and methods and start the execution. (refer to the testNg PPT and xml files)

->Our Test cases class is going to extends the library class (**Inheritance**)

First **@BeforeSuite** is the Annotation that will be executing : It contains the common library method called "ReadingPropertyFIle" . It is going to read the information provided in property file.

**@BeforeTest** : It is going to launch the browser(launchBrowser) and application URL based on the information provided in property file.

**@BeforeClass** : it is going to start the extent reports, (**we are using extent report version 4**). Inside every test case (@Test) an entry has been created to capture the status of the test cases.

**@BeforeMethod :** Our test data is stored in excel file. We are reading the excel file and store in HashMap and passing to script whenever necessary.

@**Test** cases are going to execute according to the priority that we have given and depends on methods, groups specified and parameters specified...etc

@**AfterMethod** : using ITestResult Interface we are going to get the status of the testcase (Success or failure or skipped).

capture the screen shot (on pass or fail or skipped) and attach to the extent report.

**@AfterClass :** Close the files that are opened like excel file, workbook and sheet, close the database connections… etc

**@AfterTest :** extent\_Reports.flush();

**@AfterSuite :** Close the driver instance's(window/windows) using close or quit.

**Reporting :**

We are having testNg generated emailable report which is an html file and ExtentReportV4.html

Do u have any experience in CI/CD :

**Jenkins** : creating a pipe line and configuring the master and slave.

**------------------------------------**

**TestNg** -> Refer PPT

**DataDrven** -> If we want to execute a test cases with huge amount/quantity of test data then we can use data driven approach . Apache POI library/jar files are going to support Data Driven.

**Page Object Model** (Object Repository)-> for every page in an application , locators (id, name, xpath , cssselector, tagName, ClassName, LinkeText, PartialLintText) are identified and maintained separately which is called as object repository . This is going to separate/segregate the test scripts and locators.

**Keyword Driven** (property file): Based on the Key that we have provided we will retrieve the value . Storing the configuration details like which browser to launch , which environment app URL need to launched. Data Base connection Details, API URI / URL in form of .properties file, which browser version we need to launch ..etc