FRAMEWORK

Framework Definition:-

It is well organised structured collection of reusable components .Where one driver(.xml) will take care of the execution without any manual intervention, that makes automation development execution and modification is easier and faster.

🡪I have worked oh Vtiger application and it is belongs to CRM domain and my project type is Hybrid.

Hybrid Framework:-

1.I have created hybrid kind of software.

2.It is a combination of two or more Framework.

3.It is a combination of data driven framework and modular driven Framework.

Data Driven Framework:-

1.whenever we need to test the application with huge set of the data, then we go for data driven framework.

2.In data driven framework every test will have dedicated @dataprovider annotation and will have separate excel sheet for every test.

3.whenever need to execute same test with different set of data we go for @dataProvider annotation.

Modular Driven Framework:-

1.In this Framework the tester can create testScript module wise by breaking down the whole application into the samller modules as per the client requirements and create test script individually.

2.whenever application are huge/contains lots of module we prefer modular framework.

* Framework Architecture and Explanation

Framework Components:-

1.Generic Utility

2.Object repository

3.Test data

4.Resources

5.Test-Script

6.Test Driven File(Test-suite)

7.HTML-Reports.

8.Screenshot

9.Maven Build File(POM.xml)

Generic Utility:-

Generic means common for all projects all the utilities available in generic utility which can be used to any application.

Base Test:-

It is implemented using testing , it contains configuration annotation.

Java utility:-

It is one of the subcomponent in generic utility which contains several methods, capture system date, generate random number. Some testcase does not allow unique data.

Database utility:-

In my test script developer their some test scripts related to database need to execute select query, their some test scripts required some pre conditions. I want to insert some customer into database.

Web driver utility:-

During testscript developer required few actions on web browser let say switching to windows, implicity wait, explicitly wait, select the value from dropdown, mouse hover operation these all are kept inside the utility .

* Advantages:-during testscript developer no need to remember syntax ,just create an object by taking help of the utility.

File Utility:-.

As per the rule of automation data should not be hardcoded within a script instead of hard coding. Im going to keep data in property file and Xls file.In order to read the data from properties file this utility is being used (excel utility, fileutility, json utility, json simple class).

Listener utility:-

🡪Listener utility is developed by using testNG, during my test script execution if any test script got failed ,I want to take a screenshot.

🡪Sometimes after my Script failure, I want to re-execute the script multiple times. During execution I want to capture the logs. Every logs in such cases we can go for listener Utility.

Object Repository:-

Basically, Object Repository is one of the component in my framework. where we are going to store business library and then web Elements are stored in page wise. This repository developed using pom design pattern also it contains business libraries are building block of the testcase.

Test-Data:-

1.As per the rule of automation we should not hard code the data within a test script.Their are two types of data

* Env-Config data
* Test script
* Test data

2.URL is a environment data, all these suite I want to execute in a different environment, u just change the url in environment-configuration.If u want to do Cross-browser/ Non-functional/ Compatability testing I want to run in a different browser.

3.During test script development we required some test script related data. such kind of data we can keep in excel. By taking help of data we can read data from Excel all test script data is paste in excels, data is segregated based on module(modification and maintenance of data easier and faster).

Resources:-

🡪It is one of the component in my framework during my test script development. I want to extend my base class internally.

i want to execute my base class, because my base class internally take care of the launching the browser. Base class required some resources to launch the browser.

🡪In resource folder not only .exe file we can also store some automation related documents like user write documents,the test scripts available in Excels.

Test Script:-

Test Script component is a collection of all testNG classes. All these testNG classes developed using @Test annotation during test Script development ,iam going take a help of all these Utility.

TestNG-Driver File(Suite File):-

Basically this suite file collection of suites.For example-

When I get a new build I want to execute smoke test cases.i just take a smoke suite, inside smoke suite I just collect all the testcases and put it inside xml file.

Basically it is going to execute the smoke suite it is going to check all smoke scenarios and then check the basic functionality of application. Then we are going to execute regression suite. then we are going to execute parallel cross-browser suite and parallel-Distribute suite.

HTML Reports:-

🡪It will execute all the test cases and produce the result in HTML repot.it contains pass/fail/skipped test scripts.

🡪This will produce the Stability of Application(Quality of application).

ScreenShot:-

If any test script got failed we can screenshot in screenshot folder. basically this screenshot taken by listener.

Maven Build File(POMxml):-

1.During development of framework we have used maven project.POM.xml is a project configuration file.

2.Inside the POM.xml we are going to keep all dependencies which is required to develop this framework.

🡪Selenium:-For web interaction.

🡪Apache poi:-to read data from Excel.

🡪Json Simple:-To read data from json file.

🡪Apache common io:-To take a screenshot.

🡪mysql-connector:-To connect to database.

.M2[Local repository]:-

1.Inside the POM.xml file we keep all the dependency which is required to develop this framework. If we run the project via POM.xml file instead of testNG xml file.

2.If any jar is missing ,it will automatically connect to internet and download the jar into .m2 folder in our local system and then attach to Eclipse.

3.It will automatically create one folder called C://user/.m1 and then attach to eclipse than trigger the testNG suite file.

Execution Flow of Framework :

When a new build is available, Jenkins initiates the execution of the ‘pom.xml’ file. This ‘pom.xml’ file in turn, runs the TestNG XML file, which triggers the test scripts. During the execution of a test script, the base class helps launch the browser and log into the application. The script utilizes the object repository to find the elements and test data is fetched from properties and excel file to execute the test steps. Results are then stored in the results folder. If any test scripts fails, a screenshot is saved in the screenshot folder.

This process continues until all test scripts have been executed. Once all test scripts are complete, a final HTML report is generated.

OOPs concept in my Framework

INHERITANCE :

“ One class acquiring the properties of another class is called as Inheritance”. Here properties is defined as the methods and variables.

Inheritance can be achieved by ‘extends’ Keyword.

Usage in Selenium Framework :

* Base Test class: It provides implementation for all the Configurion annotation and base class can also be used to initialize web driver interface , web Driver waits , property files , excels etc…
* We can create an Object for utility classes inside base class and which can be inherited by test classes.

Example:

Public class BaseClass{

@BeforeSuite()

Public void configBS(){

System.out.Println(“===connect to db, Report config===”);

}

@BeforeClass()

Public void configBC(){

System.out.Println(“===Launch the Browser===”);

}

@BeforeMethod()

Public void configBM(){

System.out.Println(“===Login to app===”);

}

@AfterMethod()

Public void configBS(){

System.out.Println(“===Logout an app===”);

}

@ AfterClass ()

Public void configBS(){

System.out.Println(“===Close the browser===”);

}

@AfterSuite()

Public void configBS(){

System.out.Println(“===close to db, Report Backup===”);

}

Public class CreateContactTest extends BaseClassTest{

@Test

Public void createContact(){

System.out.println(“execute createcontact and verify”);

}

@Test

Public void createContactWithDate(){

System.out.println(“execute createcontact with date and verify”);

}

Multiple Inheritance :

In my framework multiple inheritance can achieved in ListImpClass

Public class ListImpClass implements ITestListener,ISuiteListener{

}0

ENCAPSULATION :

The process of wrapping up of Data members and member functions together as a single unit is called as Encapsulation.

Rules to achieve Fully Encapsulated class :

* Class must be declared with public keyword.
* There must be at least one constructor in the class and must be declared with public keyword.
* All the class members must be declared with private keyword.
* We need to use getters() and setter() to access and initialize value in another class.

Encapsulation in my Framework :

I have used the Encapsulation in my framework in the Page Object Model class.

In the POM class at first we will find the location of the virtual object by @findBy annotation and after finding the location virtual object becomes web element and we declare the web elements initially with private keyword.

In the POM class we use the constructor, and we create the object of the POM class of particular webpage in our main test Script then the constructor of that particular POM class will be executed in which we have a PageFactory.initElement(driver,this) from which ‘this’

Keyword will hold the object address of current class and initialize to the driver.

Advantages of using POM class and Advantages of using Encapsulation rules :

* Maintenance and Modification of all the Web Elements with respect to each and every webpage will be easy.
* We can avoid stale Element Referance Exception(whenever we execute the constructor of POM class , the driver focuses exactly on the current webpage by this we can avoid the exception).
* We can restrict or limit the access scope of the web Elements of each and every web page by using private keyword to the users or to a team.
* We can achieve the code Optimization by using business logic in POM class with respect to Scripts.
* Debugging is easy.

Polymorphism :

Objects behaving differently based on the input and with same behaviour provides multiple outputs and it can happen during runtime or compile time is called as polymorphism.

a.Compile Time Polymorphism : The method declaration bind with the method implementation during the compile time by compiler based on the arguments is called as compile time polymorphism.

Example : Method overloading and constructor overloading.

* Method Overloading : declaring the multiple methods with same name but with different number of arguments in the same class are called as Method Overloading.

I have used Method overloading in my framework for hye dropdown handling to select the required dropdown with help of Select class.

Public void handlingDropdown(WebElement ele, string vis text){

Select s = new Select(ele);

s.selectByVisibleText(vistext);

}

Public void handlingDropdown(WebElement ele, int indexNo){

Select s = new Select(ele);

s.selectByIndex(indexNo);

}

b.Run Time Polymorphism : The method declaration bind with the method implementation during the execution time by the jvm based on the object creation is called as Run Time Polymorphism.

Example : Method Overriding.

Mehod Overriding : Giving the new implementation from subclass to super class method is called as method overriding.

* To achieve Method Overriding we have a criteria to follow-

🡪We should follow the inheritance relationship.

🡪Method name, parameters and return type of both the super class and sub class method should be same.

🡪Visibility of subclass method should be equal or higher than the super class method.

🡪To achieve the method overriding , super class method should be declared as final.

I have method overriding in my framework where the all the abstract methods present in webDriver(interface) are goven implementation in the RemoteWebdriver(implementation class) and further for their usage we have inherited all the implemented methods to classes like ChromeDriver(class),FirefoxDriver(class)….etc…

I have used method overriding in my framework for launching the browser where driver object behaves differently with respect to particular class.

Example : ChromeDriver driver = new ChromeDriver();

WeDriver driver = new ChromeDriver();

I have used method overriding in my framework to get the screenshot of failed test script by providing the implementation to abstract method present in the ITestListener interface in my implementation class.

I have used method overriding in my framework to re-run the failed test scripts multiple times with RetryAnalyzer.

Abstraction :

The process of hiding the internal implementation of the application and providing only the functionality to the end user is called as abstraction.

To achieve abstraction we need

1. Abstract class(which can not be 100% abstract but 0 to 100%).
2. Interface(partial and it consist of only abstract methods).

Disadvantages of Abstract class :

* We can not achieve 100% abstraction but 0 to 100%.

Multiple inheritance is not possible.

* We can create a constructor of abstract class but we cannot create the object as abstract keyword present in abstract class restrict the object creation.
* We can not make abstract class as final or private but it must be always public.
* We cannot create a object of abstract class so no one can access data by creating the object.
* Tight coupling is possible but where as loose coupling is not possible in abstract class so we go for interface.

I have used abstraction in my framework where all the abstract methods present in WebDriver interface were new implementation in remotewebDriver class without creating object of RemotWebdriver but we only create ChromeDriver object of ChromeDriver class. We do upcasting.

Example:

ChromeDriver driver = new ChromeDriver();

WebDriver driver = new ChromeDriver();

So, now all the abstract methods present in driver. we are getting only the functionality to use in main test scripts but we exactly don’t know where implementation is happening.