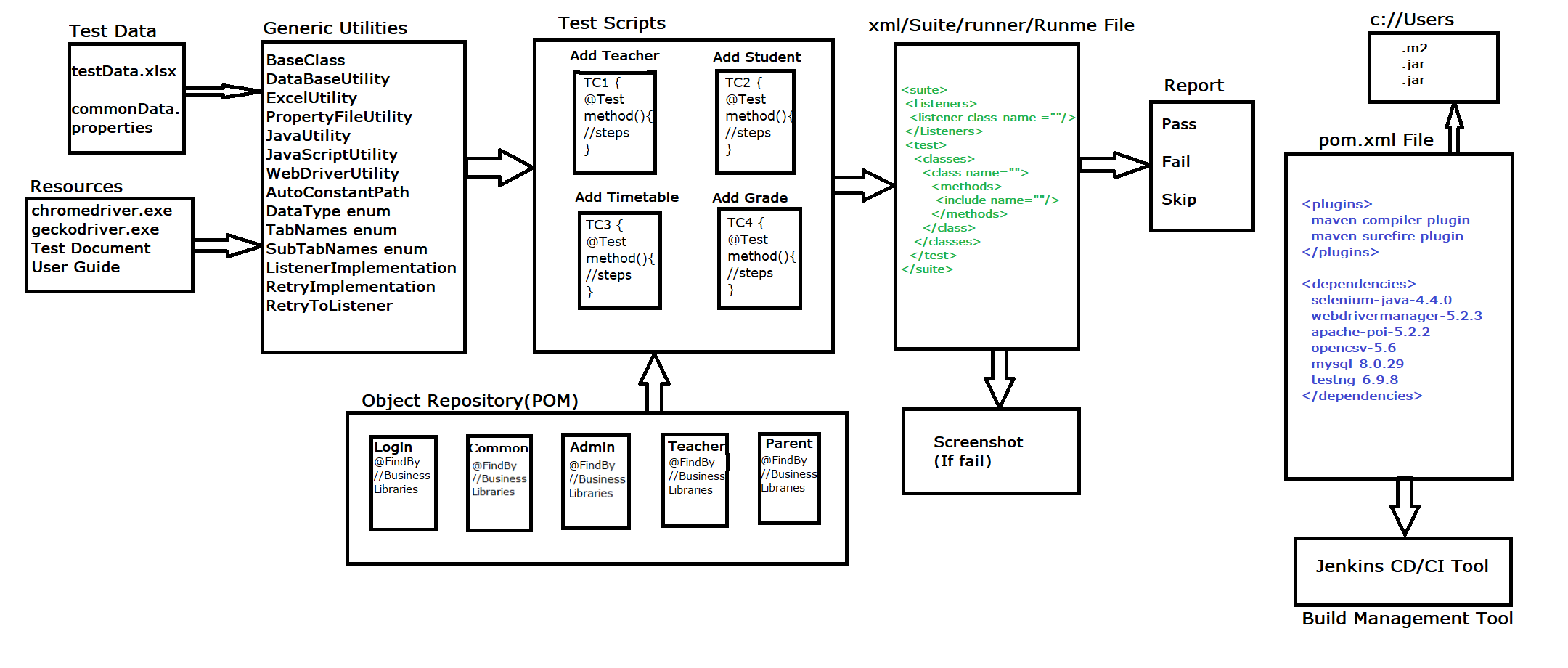
**FRAMEWORK USER GUIDE**

Framework is a collection of tools and processes that integrates various functions like libraries, test data and various reusable modules that facilitates writing the test scripts in a standard format and execute and generate reports of the same with great ease.

We have different types of frameworks – Data Driven, Modular Driven, Keyword Driven, Method Driven and Hybrid frameworks.

My project is a Hybrid framework which is the combination of – Data Driven, Method Driven and Modular Driven frameworks.

**FRAMEWORK ARCHITECTURE :**

****

**List of Components in Framework:**

1. **Generic Utility:**

Generic Components is one of the automation framework components which is common for all the application. Its collection of generic class contains reusable methods / libraries. The methods which can be used to any application is called Generic/common methods.

**Advantages of Generic Components:**

* Reusability of code
* Code Optimization
* Test script development is faster
* Test scripts Code readability
* Generic libraries are common to all automation project
* Avoid duplicate Code
* No need to remember the syntax every time, just create once & use multiple times
* **Base Class:**

Base class consists of all the configuration annotations which is mainly used to perform operations like connect to Data base, precondition, Launching Browser, Login to application, Logout of application, close the browser, Close data base connection. We create this operations using configuration annotations like @BeforeSuite, @BeforeTest, @BeforeClass, @BeforeMethod, @AfterSuite, @AfterTest, @AfterClass @AfterMethod.

* **Database Utility:**

This utility consists of Generic methods and is mainly used to perform all database operations like connect to database, disconnect database, execute query, update query and execute query based on query and it will verify the data.

* **Excel Utility:**

In my project we stored all the test data in Microsoft Excel in sheet, row and cell format to fetch the Test data from excel and run the script we used Excel utility. It consists of generic methods which is used to perform some operations on excel sheet.

* Fetching the data from excel in string format, here we used Data formatter which is used to read the data from excel and give the data in a proper format
* Write the data into excel sheet in a cell
* Get last row count
* **Property File Utility:**

Main purpose of file Utility is to fetch the common data which is common throughout the suite like URL, username, password. In File utility we will have generic methods to get the value from property File and here we give key as parameter and it returns the data in string format to the called method.

* **Auto Constant Path :**

AutoConstantPath is an interface which is store the common variables which is used to run the generic methods, here we store variables like excel path, property file path, JDBC URL, JDBC username, JDBC password, time out consisting of some values. The main purpose of creating a variable is these values or Dynamic and can be updated time to time rather than changing in the script which is tedious job we create an interface and store the values, which will be secure**.**

* **Java Utility :**

Java Utility consists of Java libraries which is used to perform java related operations like to generate random number, generate date according to format.

* **Java script Utility :**

JavaScriptUtility consists of operations performed using JavascriptExecutor interface. We can perform operations like open application, pass text input to the element, click action on element, scroll the web page till the required element.

* **ITestListener :**

ITestListener is an implementation class where we provide implementation for the abstract class present in ITestListener interface. It consists of abstract methods like:

• onTestFailure()

• onFinish()

• onStart()

• onTestFailedButWithinSuccessPercentage()

• onTestFailesWithTimeout()

• onTestSkipped()

• onTestStart()

• onTestSuccess()

Here we use @Listeners annotation and give path of ITestListener generic class path as parameter in Test script in testNG.xml.

**<Listeners>**

**<listener class-name = “path of ITestListener Implementation class”/>**

**</Listeners>**

* **Retry Analyser Implementation class :**

Retry Analyzer is a implementation class which is used to implement declaration for Abstract method of IRetryAnalyzer Interface. Sometimes the script might be failure due to poor server connection during that time we should run the script for defined number of times to pass the script so we use IRetryAnalyzer. Here we make use of counter and retry limit variables. Counter is a initialization value, retry limit value is limit value given by the developer.

* **Web Driver Utility :**

WebdriverUtility is a generic class, which contains web driver specific reusable actions like

* launchBrowser(): launches the specified browser like chrome, firefox, IE etc
* waitTillLoad () : waits implicitly till the page is loaded
* maximizeBrowser() : maximizes the web page
* navigateToApplication() : opens the application based on the entered URL
* moveToElement() : mouse hovers to particular element
* getScreenshot() : takes the screenshot of the mentioned page, generally used for failed test cases
* waitTillElementVisible() : it’s an explicit wait. Waits till the element is visible on web page
* switchToFrame() : used to Switch to Frame Window based on index
* selectByIndex() : used to select the value from the drop-down based on index
* selectByVisbleText() : used to select the value from the drop-down based on visible text
* switchToWindow () : used to Switch to Any Window based on Window Title
* **Data Type enum :**

Data type enum consists of all data type constants which are used when String data is to be converted to any other data type.

* **TabNames enum :**

It consists of all the tab names as constants and we declare constructor to the same which accepts String data as argument. This enum is used when we have common dynamic xpaths and we should replace particular data with this enum.

1. **Object Repository :**

It is the collection of elements, locators & business libraries in one place. It is developed using POM design pattern.

As per the rule of the automation, we should not hardcode elements within test Scripts instead, we should get elements from Object Repository, because in Agile process due to frequent requirement changes, modification & maintenance of elements is tedious job.

In my project we had 10 user stories and for each user story we created a class and stored all the elements of that particular user story in that class using @FindBy. Inside this class we keep methods to perform common operation/Business operations like create, update, login, logout of that particular user story. We can reuse the same method many number of times.

**Advantages of POM:**

* Well organized structure
* Handle stale element exception
* Maintenance & modification of element is easy
* We can directly store Web Elements in java class
* Better fit for Agile processes
* Supports Auto healing feature

1. **Test Data :**

**Common data :** Common data is used to store common values which is used throughout the framework and can be modified easily whenever build is moved from one server to another server. We use property files to store common data here the data will be stored in the form of key and value. By creating reference variable for FileProperties class with getProperty(key) method we can fetch the value of that key from property file.

**Test data :** This is the data which is used to run the test script. This data will be stored in excel. We fetch the data using reference variable of excel Library file class in test script. We pass sheet name, row count and cell count in getExcelData() method the fetched data will be in the form of String.

1. **Resources :**

**Geckodriver.exe:** Geckodriver is an executable file which is used to launch the firefox browser. It acts like a translator which is used to establish connection between selenium webdriver and Mozilla firefox browser. It is used to control firefox browser.

**Chromedriver.exe:** Chromedriver is a executable file which is used to launch the chrome browser. It acts like a translator which is used to establish connection between selenium webdriver and Chrome browser.

1. **Test Script :**

Test script consists of @Test once after executing @BeforeSuite, @BeforeTest, @BeforeMethod @Test will get executed that is it will perform action on the application with the help of POM class methods, General utilities and Test data.

1. **TestNG.xml :**

TestNG.xml file is a configuration file that helps in organizing tests. It allows testers to create and handle multiple test classes, define test suites and tests. It makes a tester's job easier by controlling the execution of tests by putting all the test cases together and run it under one XML file. It consists of series of xml code which starts from suit tag, test tag, class tag.

**Steps to create TestNG.xml file :**

Select test script -> Right click ->TestNG -> convert to TestNG ->Give suite name and test name ->Finish In my project

We had different TestNG.xml files for different operations depending on the execution type like Batch execution, smoke execution, Regression execution, Regional Regression execution, Parallel execution, Distributed parallel execution. Once after executing the testNG.xml it will generate the testNG reports in test-output folder, it consists of passed test case, failed test case, skipped test case, emailable report, testing-report, testing-results.

1. **Reports :**

TestNG, by default, generates multiple reports as part of its test execution. These reports mainly include TestNG HTML report, TestNG email-able report, TestNG report XML, and JUnit report XML files. These files can be found under the test-output report folder.

1. **Screenshot :**

In real time if the test case gets failed then we have to capture the screenshot of it to capture the screenshot we use TakesScreenshot interface. TakesScreenshot is an interface and it consist of one abstract method called getScreenshotAs(). Since we need this operation only when test case fails we use ITestListener class where we provide screenshot implementation for onTestFailure() to take screenshot if test case fails.

1. **pom.xml :**

pom.xml is an xml file which is used to add dependencies from maven repository. Using this we can easily add plugins to the project.

In my project we mainly used pom.xml to add dependencies and whenever the plugin gets a new version, we used to just change the version name and save the project the plugin used to get updated. Whenever we create a maven project by default pom.xml will be present, here we will store all plugins. Once after downloading all the plugins will be added in maven dependencies folder in .jar format.

1. **Jenkins :**

Jenkins is a continuous integration/ continuous development tool which is used to run the frame work.

**Tools Used in Framework :**

|  |  |
| --- | --- |
| **IDE** | **Eclipse** |
| **Framework Type** | **Hybrid Framework** |
| **Build Management Tool** | **Maven** |
| **Unit Testing Tool** | **TestNG** |
| **Automation Tool** | **Selenium** |
| **Language Binding** | **Java** |
| **Project Management Tool** | **Jira** |
| **Methodology** | **Agile – Scrum** |
| **Excel Tool** | **Apache POI** |
| **Reporting Tool** | **Extent Reports** |

**Advantages of Framework :**

1. **Faster test script development :**

In framework we design several reusable generic codes and store them in libraries. This feature helps testers to develop the test scripts faster as they need not write thousands of line of test scripts. They can rely on these libraries.

1. **Code Optimization :**

Using generic utilities and POM business libraries test scripts can be optimized. We store all the reusable codes in generic utilities and elements and business libraries in POM pages and invoke these libraries in the test scripts whenever needed reducing the number of lines of code.

1. **Code Reusability :**

Generic utilities contains all the reusable codes for several web based actions, codes to read data from Excel and property files which facilitates reusability of these codes to several test scripts.

1. **Maintenance and Modification of Test Data is easy :**

Since all the test data is stored separately in Excel and property file, we can maintain and modify this data easily without disturbing the actual test scripts.

1. **Maintenance and Modification of web elements is easy :**

POM provides us a well organized way to store all the web elements and respective business libraries which facilitates easy maintenance and modification of the addresses of the elements. It also helps to handle StaleElementReferenceException.

1. **Easy Debugging :**

Frameworks focuses primarily on improving code quality and readability. In addition to providing the most efficient way to code and develop test scripts, frameworks facilitate easy debugging and software maintenance.Selenium has built-in support for debugging.

1. **Improved Code Efficiency :**

Framework provides predefined template which makes it easy to develop the test scripts and code efficiency.

1. **Screenshots for failed test scripts can be generated :**

We have a generic method in web driver utility which enables us to take the screenshot of the failed test scripts. This method can be associated with the onTestFailure() method in ITestListener implementation class and generate the screenshot of failed test script when testNG.xml file is executed.

1. **Rerun the failed test scripts :**

We can rerun the failed test scripts using RetryAnalyser, the implementation class of IRetryAnalyzer interface. We provide implementation to the only method retry() in this class and rerun the failed test scripts as many times as we wish.

1. **Generate Reports :**

TestNG has a built-in feature which helps in generation of html reports automatically when xml file is executed. Apart from this we can also collaborate Extent Reports to generate both high level and low level reports using which we can analyse test reports of several test scripts easily.

1. **Browser compatibility testing :**

TestNG enables us to execute the test scripts on different browsers in parallel. This feature makes it easy to perform browser compatibility testing.

1. **Batch Execution :**

Using testNG tool we can perform batch execution. This tool generates .xml file automatically and we can customize this file to execute the complete test suite.

**EXPLAIN YOUR FRAMEWORK.**

Framework is a well-organized structural collection of all the reusable components which acts like a template to develop and execute the test scripts with great ease.

My framework is a Hybrid framework which is the combination of Data Driven, Method Driven and Modular Driven frameworks. We used Selenium-Java Client Binding. Our project is Page Object Model design pattern. As per the Page Object Model, we have maintained a class for every web page. Each web page has a separate class and that class holds the functionality and members of that web page. Separate classes for every individual test.  We have separate packages for Pages and Tests. All the web page related classes come under the **Pages** package and all the tests related classes come under **Tests** package.

For example, Home Page and Login Page have separate classes to store element locators. For the login test, there would be a separate class which calls the methods from the Home Page class and Login Page class.

Base class (BaseClass.java) deals with all the common functions used by all the pages. This class is responsible for loading the configurations from properties files, Initializing the WebDriver, Implicit Waits, Extent Reports, and also to create the object of FileInputStream which is responsible for pointing towards the file from which the data should be read.

Utility classes stores and handles the functions which can be commonly used across the entire framework. The reason behind creating a utility class is to achieve reusability.

Property file stores the information that remains static throughout the framework such as browser-specific information, application URL, etc.

All the details which change as per the environment and authorization such as URL, Login Credentials are kept in this file. Keeping these details in a separate file makes it easy to maintain.

Screenshots will be captured and stored in a separate folder and also the screenshots of failed test cases will be added to the extent reports.

 All the test data will be kept in an excel sheet (testData.xlsx). By using ‘testData.xlsx’, we pass test data and handle data-driven testing. We use [Apache POI](https://www.softwaretestingmaterial.com/handling-excel-files-using-apache-poi/) to handle excel sheets.

We used TestNG for Assertions, Grouping, and Parallel execution.

We used Maven for build, execution, and dependency purpose. Integrating the TestNG dependency in the pom.xml file and running this pom.xml file using Jenkins.

By using Jenkins CI (Continuous Integration)/ CD (Continuous Development) Tool, we execute test cases on a daily basis.

For the reporting purpose, we are using Extent Reports. It generates beautiful HTML reports. We use the extent reports for maintaining logs and also to include the screenshots of failed test cases in the Extent Report.