Project	DemoAutomationProject
Document Title	Project Document

DemoAutomationProject

POM Based Data Driven Selenium Automation Framework using Java, TestNG, Maven, Log4j & ExtentReport

Selenium Automates Browsers. That's it !
What you do with that power is entirely up to you.

https://www.selenium.dev/

INDEX

Introduction to Automation testing:

Testing is an essential part of a software development process. While testing intermediate versions of products/projects being developed, testing team needs to execute a number of test cases. In addition, prior to release every new version, it is mandatory that the version is passed through a set of “regression” and “smoke” tests. Most of all such tests are standard for every new version of product/project, and therefore can be automated in order to save human resources and time for executing them.

Benefits of Automation Testing:

- Reduction of test’s time execution and human resources required.
- Complete control over the tests’ results (“actual results” vs “expected results”)
- Possibility to quickly change test’s preconditions and input data, and re-run the tests dynamically with multiple sets of data

Automation workflow for the application can be presented as follows:

- First of all, it is required to identify tasks that an application has to accomplish.
- Second, a set of necessary input data has to be created.
- Third, expected results have to be defined in order one can judge that an application (a requested feature) works correspondingly.
- Fourth, Executes a test.
- Finally, compares expected results with actual results, and decides whether the test has been passed successfully.

Environment Specifications:

1. Selenium Webdriver (Supports all major browsers, we use Mozilla, chrome and IE)
2. Eclipse IDE
3. Java
4. TestNG
5. Maven
6. Log4j
7. Extent Report
8. Apache POI

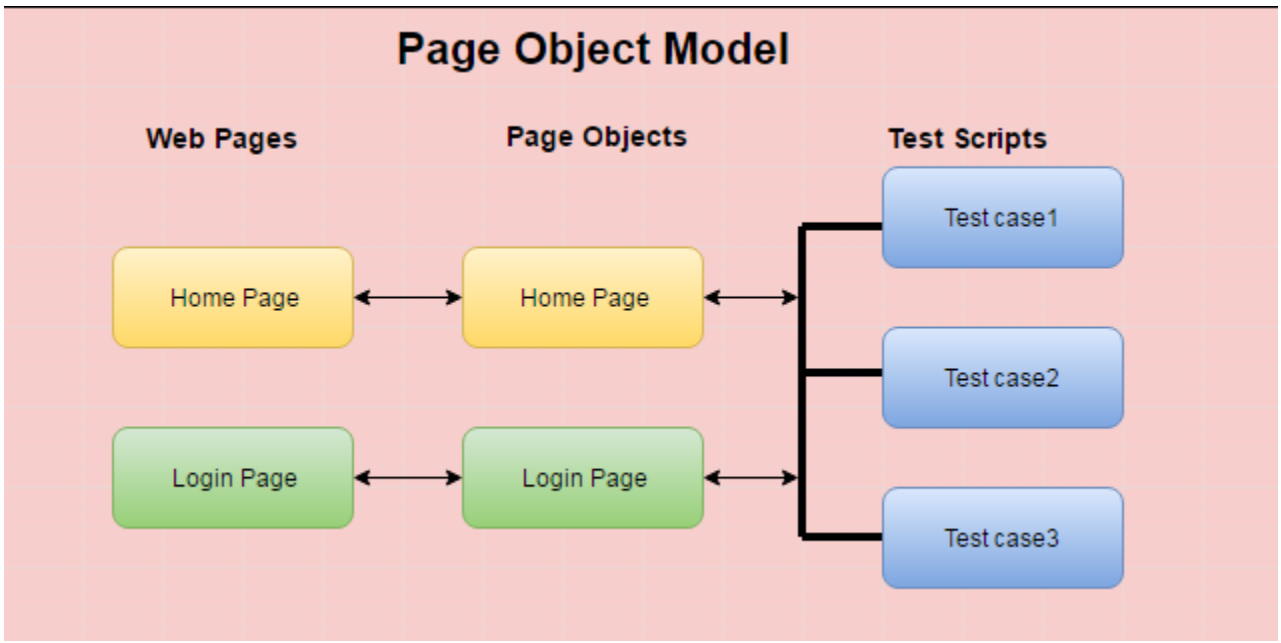
Framework & Design used in this Project:

1. **Page Object Model:**
   Page Object Model, also known as POM, is a design pattern in Selenium that creates an object repository for storing all web elements. It is useful in reducing code duplication and improves test case maintenance.
   In Page Object Model, consider each web page of an application as a class file. Each class file will contain only corresponding web page elements. Using these elements, testers can perform operations on the website under test.

   Advantages of Page Object Model:

   - Helps with easy maintenance.
   - Helps with reusing code.
   - Readability and Reliability of scripts.

   Workflow Diagram of Page Object Model:



2. **Data Driven Framework:**

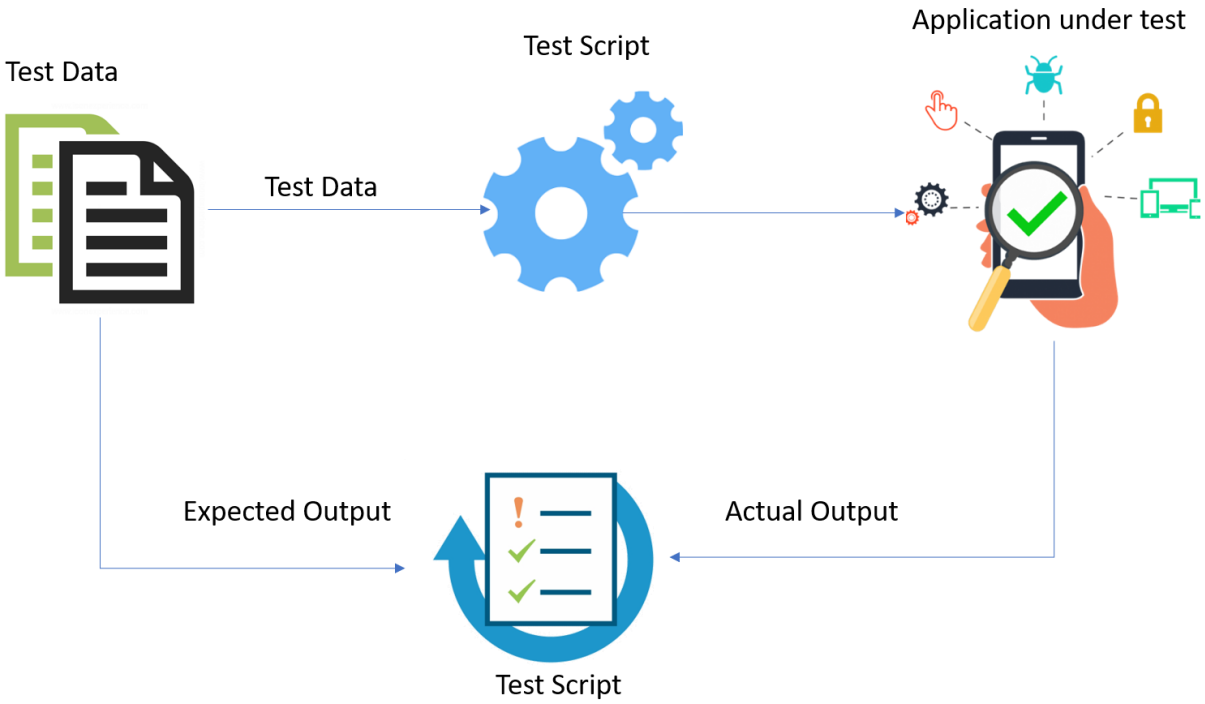
Data Driven framework is used to drive test cases and suites from an external data feed. The data feed can be data sheets like xls, xlsx, and csv files.

A Data Driven Framework in Selenium is a technique of separating the “data set” from the actual “test case” (code). Since the test case is separated from the data set, one can easily modify the test case of a particular functionality without making changes to the code.

Advantages of DataDriven Framework:

- Test cases can be modified without much changes to code.
- It allows testing the application with multiple sets of data values, especially during regression testing.
- It helps us to separate the logic of the test cases/scripts from the test data.

Workflow Diagram of Data Driven Approach:



Tools Used in this Framework:

1. **Selenium** :
   Selenium is a well know open source testing framework, which is widely used for testing Web-based applications.
   Selenium Webdriver supports most of all browsers to run your test cases and many programming languages like C#, Java, Python, Ruby, .Net, Perl, PHP, etc... to create and modify your test scripts.

2. **Eclipse IDE:**
   Eclipse is an integrated development environment (IDE) for Java. The Eclipse IDE is the most known product of the Eclipse Open Source project.

3. **Java:**
   Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation.

4. **TestNG:**
   TestNG is an automation testing framework in which NG stands for “Next Generation”.
   The design goal of TestNG is to cover a wider range of test categories: unit, functional, end-to-end, integration, etc., with more powerful and easy-to-use functionalities.

5. **Maven:**
   Maven is a build automation tool used primarily for Java projects. Maven can also be used to build and manage projects written in C#, Ruby, Scala, and other languages. The Maven project is hosted by the Apache Software Foundation, where it was formerly part of the Jakarta Project.

6. **Log4j:**
   Log4j is a fast, reliable and flexible logging framework which is written in java. It is an open-source logging API for java. Simply the logging means some way to indicate the state of the system at runtime. Logs are used to capture and persists the important data and make it available for analysis at any point in time.

7. **ExtentReports:**
   ExtentReports is an open-source reporting library useful for test automation. It can be easily integrated with major testing frameworks like JUnit, NUnit, TestNG, etc. These reports are HTML documents that depict results as pie charts.

8. **ApachePOI:**
   Apache POI is an open source popular API that allows programmers to create, modify, and display MS Office files using Java programs.

File Formats Used in the Framework:

- ***Properties file*** – We use properties file to store and retrieve the UI elements of an application or a website and data set file paths. It contains id of the elements, name, xpath or Css selector etc.

- ***Excel files*** — Excel files are used to pass multiple sets of data to the application.

- ***Xml file*** **–** Is used to execute the test scripts. Based on the package or classes or Tests mentioned in the xml file scripts will be executed

The DemoAutomationProject Explanation in detail:

1. Object Repository:

UIMap is a concept for defining, storing, and serving UI elements of an application or a website. The UIMap properties file contains a set of 'key-value' pairs, where key is an alias of the UI element, and a value is the locator.

2. Test Data:

Data set stores the data files, Script reads test data from external data sources and executes test based on it. Data sets increases test coverage by performing testing with various inputs and reduce the number of overall test scripts needed to implement all the test cases.

3. Test Automation Scripts:

A test is considered as a single action or a sequence of actions, that defines whether a specific feature meets functional requirements. It has multiple test files / packages / class files which will be executed based on the configurations defined in testng.xml.

4. Reports / Executed Results:

Test report/results is a document which contains summary of test activities. After execution is completed, it is very important to communicate the test results and findings to the project manager along with the screenshots for failed tests and with that decisions can be made for the release.

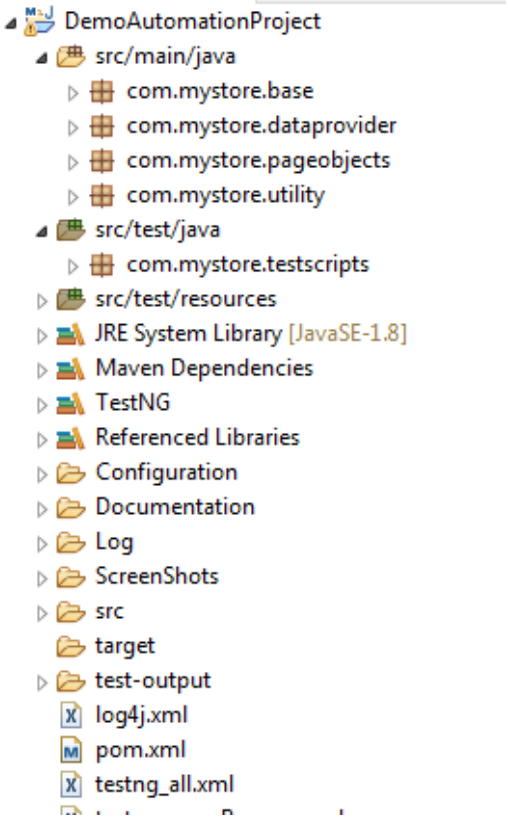
5. TestNG xml file:

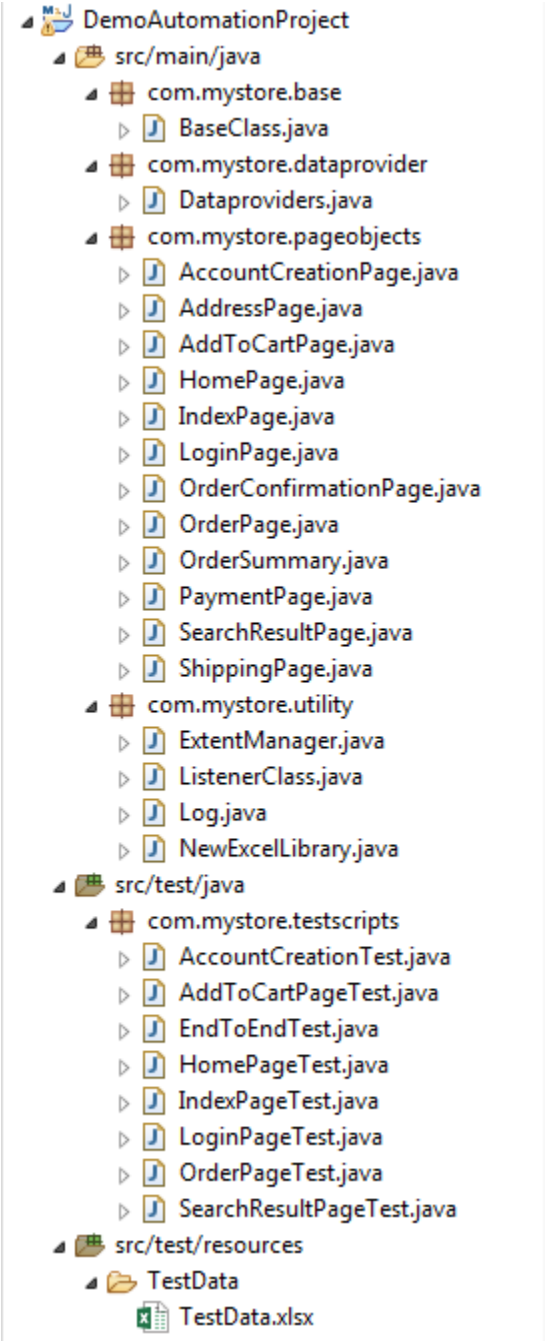
In order to create a test suite and run separate test cases, we need framework which drives the automation. Here testng.xml can be called as "driver" which drives several test cases automated using selenium code. Advantage of using TestNG with Selenium is of running multiple test cases from multiple classes using xml configuration file.

6. POM.xml file:

In order to maintain the dependencies of the tools used in this framework POM.xml is created.

Design Diagram of the DemoAutomationProject:





Installation & Configuration Steps of the tools used in this project:

1. Java:

Step 1) Go to link. Click on JDK Download for Java download JDK 8.

Step 2) Next,

1. Accept License Agreement
2. Download Java 8 JDK for your version 32 bit or JDK download 64 bit.

Step 3) When you click on the Installation link the popup will be open. Click on I reviewed and accept the Oracle Technology Network License Agreement for Oracle Java SE development kit and you will be redirected to the login page. If you don't have an oracle account, you can easily sign up by adding basics details of yours.

NOTE: You will be required to create an Oracle Account to start Java 8 download of the file.

Step 4) Once the Java JDK 8 download is complete, run the exe for install JDK. Click Next.

Step 5) Select the PATH to install Java in Windows… You can leave it Default. Click next.

Step 6) Once you install Java in windows, click Close.

How to set Environment Variables in Java: Path & Classpath:

Step 1) Right Click on the My Computer and Select the properties.

Step 2) Click on advanced system settings.

Step 3) Click on Environment Variables to set Java runtime environment.

Step 4) Click on new button for user variable.

Step 5) Type PATH in the Variable name.

Step 6) Copy the path of bin folder which is installed in JDK folder.

Step 7) Paste Path of bin folder in Variable value. Click on OK Button.

Step 8) You can follow a similar process to set CLASSPATH.

Step 9) Click on OK button.

Refer below link for detailed description of Java Installation & Configuration steps:

https://www.guru99.com/install-java.html

2. Eclipse IDE:

Step 1) Installing Eclipse -> Open your browser and type https://www.eclipse.org/
Step 2) Click on “Download” button.
Step 3) Click on “Download 64 bit” button.
Step 4) Click on “Download” button.
Step 5) Install Eclipse.
Step 6) Open Installed file and click Run button.
Step 7) Click on “Eclipse IDE for Java Developers”.
Step 8) Click on “INSTALL” button.
Step 9) Click on “Launch” button.
Step 10) Click on “Launch” button.

Refer below link for detailed description of Java Installation & Configuration steps:
https://www.guru99.com/install-eclipse-java.html

Creation & Configuration Steps of this project:

Step 1) Open Eclipse.
Step 2) Click on File menu.
Step 3) Click on New.
Step 4) Click on Project.
Step 5) Select Maven Project.
Step 6) Click Next.
Step 7) Choose your Workspace location.
Step 8) Click on Next.
Step 9) Enter DemoAutomationProject in GroupID.
Step 10) Enter DemoAutomationProject in Artifact ID.
Step 11) Click on Finish.
Step 12) Open pom.xml
Step 13) Add Dependencies of Selenium, TestNG,
WebdriverManager,ApachePOI,Log4j,ExtentReports.
Refer this link for dependencies => **https://mvnrepository.com/**

Step 14) Save pom.xml file to install and build the dependencies.
Step 15) Create Configuration folder in this project.
Step 16) Create properties file in configuration folder**.**
Step 17) Create Documentation folder in this project to keep project related docs in this.
Step 18) Create com.mystore.base package in src/main/java.
Step 19) Create base class in com.mystore.base package.
Step 20) Create com.mystore.dataprovider package in src/main/java.
Step 21) Create Dataproviders class in com.mystore.dataprovider.
Step 22) Create com.mystore.pageobjects package in src/main/java.
Step 23) Create Page objects class (as per project requirement) in com.mystore.pageobjects.
Step 24) Create com.mystore.utility package in src/main/java.
Step 25) Create utility classes in (as per project requirement) in com.mystore.utility.
Step 26) Create Test data folder in src/test/resources package to keep test data in this.
Step 27) Create com.mystore.testscripts package in src/test/java.
Step 28) Create your test scripts (as per project requirement) classes in com.mystore.testscripts.
Step 29) Generate testng xml for the test scripts class.

Steps for Log4j Installation & Configuration:

Step 1) Add/Create Log4j.xml in project directory

Step 2) Add/Create Log Class in utility Package

Step 3) Configure @BeforeSuite at BaseClass to configure log4j.xml
DOMConfigurator.configure("log4j.xml");

Step 4) Need to just Call in methods in testCase from Log class.

Steps for DataDriven Testing and DataProvider:

Step 1) Add/Create ExcelLibrary in utility package.

Step 2) Create a Folder and add TestData.xlxs in that.

Step 3) Create a package for DataProvider and add DataProvider class there and create the object of ExcelLibrary Class

Step 4) Add the DataProvider methods.

Step 5) Call the DataProvider methods from testcases.

Steps for Extent Report Configuration

Step 1) Add/Create ExtentManager Class in utility Package-- to create the object of ExtentHtmlReporter and load extent-config.xml

Step 2) Create a folder or Save Extent Report under test-output

Step 3) Create Screenshots folder in project directory.

Step 4) Configure ExtentManager.setExtent() in @BeforeSuite method in BaseClass

Step 5) Configure ExtentManager.endReport() in @AfterSuite method in BaseClass

Step 6) Add/Create screenShot method in Action/BaseClass

Step 7) To attach the screenshot in extent report
Add/Create a Listener Class – ListenerClass.

Step 8) To call the listener Add the below listener (inside suite tag) setting in testng.xml

```
<listeners>
        <listener class-name="com.Project.util.ListenerClass"></listener>
</listeners>
```

How to load the project and Run test scripts:

Prerequisites :-

You should have Java and Eclipse installed in your machine and Internet Connectivity.

Step 1) Download the DemoAutomationProject.
Step 2) Open Eclipse.
Step 3) Click on the File menu.
Step 4) Select Open Projects from file System.
Step 5) Click on directory.
Step 6) Locate the project.
Step 7) Click on finish.
Step 8) Wait for few seconds for building the dependencies.
Step 9) Go through the project file.
Step 10) Right click on the particular testng xml file which you want to execute.
Step 11) Click on Run as and then click on Testng suite.
Step 12) Observe the execution of test scripts.
Step 13) Navigate to the test output folder.
Step 14) Navigate to Extent report folder.
Step 15) Open the MyReport.html file for test out report.

Thank you!