# 

**Test Automation Framework – Web and Head less browser**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **AUTHOR** | **Description** |
| 06/05/2022 | 1.0 | Bhargav | Propose new process and discussed with stakeholders |
| 06/05/2022 | 1.1 | Bhargav | Updated the document with inputs taken from stakeholders. |
| 06/05/2022 | 1.2 | Bhargav | Updated the document with iOS and Android Installation and test scripts. |
| 06/05/2022 | 1.3 | Bhargav | Updated BDD Framework design in Page Objects, Test cases, Locators and Logs. |

**Table of Contents**

[**1.0 Introduction**](#_76j0d4kdlt57) **2**

[1.1 POM description](#_h7zmzxy4wubb) with Page Factory 3

[**2.0 Technology Stack**](#_kw2kpi55kk9s) **3**

[**3.0 Test Automation Framework**](#_v2uneg85m3h3) **5**

[**3.1 Automation Framework Architecture**](#_ndo4hip6cmxg) **5**

[3.2 Test Suites:](#_57q1gsuiqcfa) 6

[3.3 Continuous Integration](#_mg8u7w3i41uy) 6

[**4.0 Technical Setup for project**](#_e2xpgn2svccu) **7**

[4.1 JDK install](#_7grmv8eiilz1) 7

4.2 IntelliJ Installation preferred Version IDEA CE 2 7

[4.3 Maven](#_isq4ksqjsmqr) 7

[**5.0 Reports Evaluation**](#_3g3puf3su6e5) **9**

[5.1 Cucumber Extent Results](#_2jg7pcm1zz0p) 9

[5.2 Screen Shots](#_gq64zu40y69t) 9

[Go to Screenshots Folder under the project folder.](#_99q0smcdbi2s) 9

[5.3 Log4j Logs](#_qym9r6b862jg) 9

[1.0 Java and Maven Project](#_ln66wr2vkqyu) 9

[1.1 Java Package](#_e9m5ni20quri) 9

[1.2 Class](#_dsz1il4qt39e) 10

[1.3 Object](#_njgmb9yqqokv) 10

[1.4 Page Object](#_3m4iclv9l2rk) 10

[1.6 Create sample s](#_b420xcbei07k)tep definition 11

# **1.0 Introduction**

The document provides information on the current framework and tests under use by the QA team for Automation Testing the scope for the same is currently web and mobile web sites.

The current framework used in Landmark Web Team for automation testing is based on POM design pattern.

## **POM description and Page Factory:**

* The ***Page Factory*** Class in Selenium is an extension to the Page Object design pattern. It is used to initialize the elements of the Page Object or instantiate the Page Objects itself. Annotations for elements can also be created (and recommended) as the describing properties may not always be descriptive enough to tell one object from the other.
* It is used to initialize elements of a Page class without having to use ‘FindElement’ or ‘FindElements’. Annotations can be used to supply descriptive names of target objects to improve code readability. There is however a few differences between C# and Java implementation – Java provide greater flexibility with Page Factory
* **Page Object Model** is a design pattern to create **Object Repository** for Mobile UI elements.
* Under this model, for each Mobile page in the application there should be corresponding page class.
* This Page class will find the Mobile Elements of that Mobile page and also contains Page methods which perform operations on those Mobile Elements.
* Name of these methods should be given as per the task they are performing i.e., if user wants to navigate to CreateTaskPage, POM method name can be ‘navigateToDo ()’. Refer to Appendix section for more details.

# **2.0 Technology Stack**

The current framework is primarily based on Java and Appium along with Selenium Web Driver to extend it to the mobile Applications configured using Maven and uses TestNG for execution and report generation.

**IntelliJ Description**: **IntelliJ** IDEA is a special programming environment or integrated development environment (IDE) largely meant for Java. This environment is used especially for the development of programs. It is developed by a company called JetBrains, which was formally called **IntelliJ**.

**Version**: IntelliJ IDEA CE 2

**Maven**

**Description**: Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

**Version:** 3.0.2

**Selenium Web Driver**

**Description:** Selenium Web Driver is a web application testing framework that allows you to write tests in many programming languages like Java, C#, Groovy, Perl, PHP, Python and Ruby. Currently, we are using Java.

**Version 3.8.1**

**TestNG**

**Description:** TestNG is a testing framework designed to simplify a broad range of testing needs, from unit testing (testing a class in isolation of the others) to integration testing (testing entire systems made of several classes, several packages and even several external frameworks, such as application servers).

**Version: 6.9.8**

**Log4j**

**Description:** Log4j is a reliable, fast and flexible logging framework (APIs) written in Java, which is distributed under the Apache Software License. log4j is a popular logging package written in Java.

**Version: 1.2.70**

**Java/JDK**

**Description:** The programming language under use is Java which is object oriented language.

**Version: 1.8**

**Guice**

**Description:** Google **Guice** (pronounced "juice") is an open source software framework for the Java platform released by Google under the Apache License. It provides support for dependency injection using annotations to configure Java objects

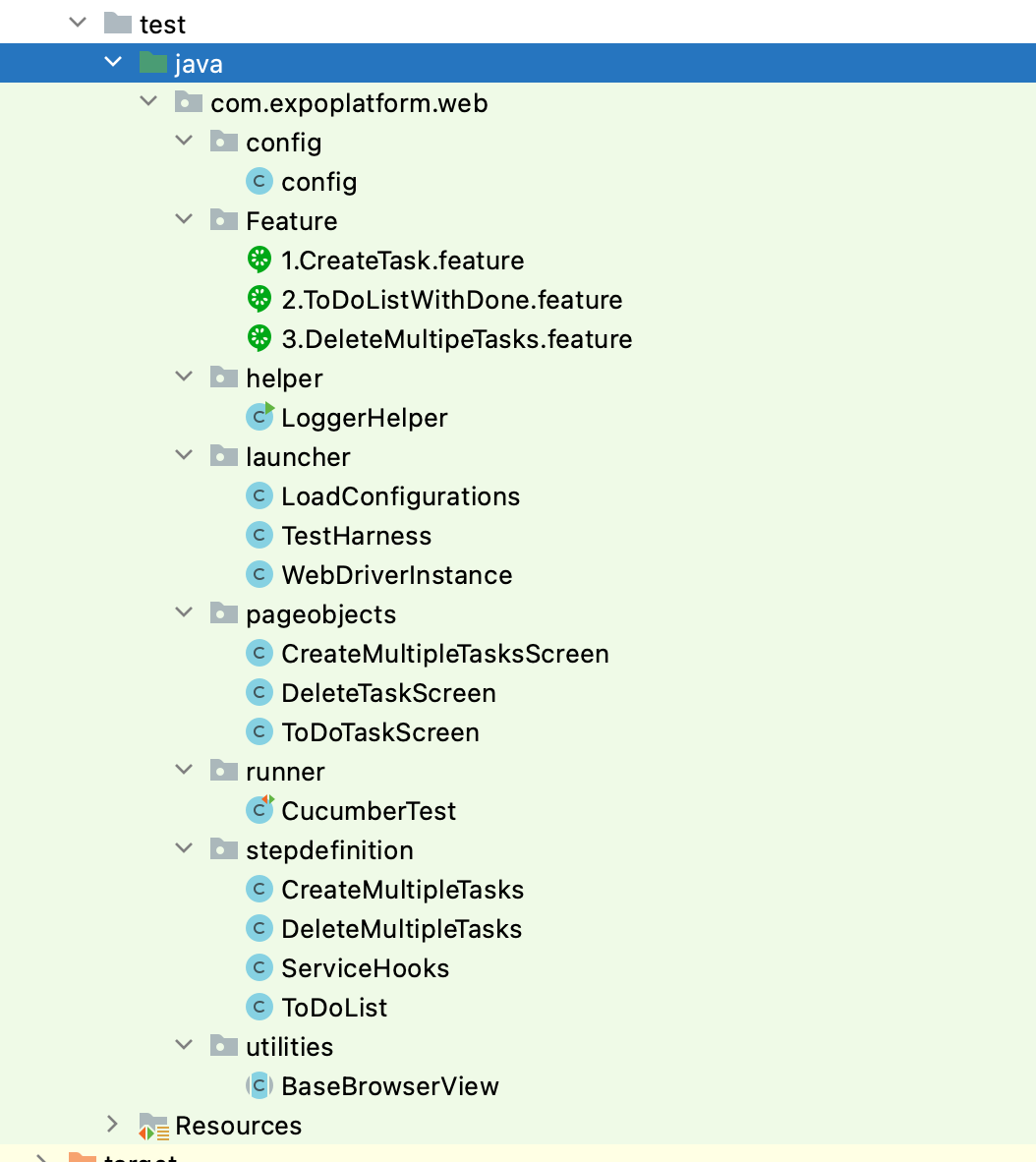
**Version: 3.0**

**3.0 Test Automation Framework**

# **3.1 Automation Framework Architecture3.2 Test Suites:**

## 

## **Package Description:**



## **com.expoplatform.web.Feature**: This package will contain all the features related to application. Ex. Onboarding , Payment, Thank you Etc.

## **com.expoplatform.web.pageobjects**: This package will contain all the Mobile pages irrespective of concept and region.

## **com.expoplatform.web.utilities:** This package will be used to perform following functions.

1. Waits
2. Snapshot
3. Swipe Up/Down
4. Swipe Right/Left
5. Hide Keyboard
6. Alerts
7. Back Button

## **com.expoplatform.web.stepdefinition**: All the test cases will be written under this package based on the scenarios/modules.

## **com. expoplatform.web.runner**: This will contain all the Feature and Step definition file with respect to all pages.

## **com. expoplatform.web.launcher**: This will contain all the config, device setup for iOS and Android with Load Configurations.

## **Java doc**: Add comments to all the methods which creates a java doc.

Currently, there are following test tags:

* Sanity
* Functional
* Regression

In order to run this sanity, functional, and regression test suites, one need to run respective xml file.

For instance:

**tags = {"@Sanity"}, {“@Regression”}**

## **3.3 Continuous Integration**

In order to test seamlessly and find early defects in the product, the job with Jenkins has been set up. Job will trigger once one made any code commit to the project repository.

# **4.0 Technical Setup for project**

## **4.1 JDK install**

URL :<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

Select option : jdk-8u111-macosx-x64.dmg

## **4.2 Intellij Installation preferred version is IDEA CE 2.**

URL:

<https://www.jetbrains.com/idea/download/#section=mac>

## **4.3 Maven**

It is available by default in Eclipse Mars 2.

**5.0 How to Execute**

## **5.1 Web application**

Launch IntelliJ and make sure you have the project in your workspace.

Right Click on Pom.xml and navigate to Run As Configurations Enter the base directory the location where your project is present. Click on Run.

Connect your device with USB.

Right Click on Pom.xml and navigate to Run As-à Run Configurations Enter the base directory the location where your project is present.

Go to environment tab provide variable name as “ANDROID\_HOME” and value “/Users/<>/” à click Run.

# **5.0 Reports Evaluation**

## **5.1 Cucumber Extent Results**

Go to Test Result Folder under the Project folder.

## **5.2 Screen Shots**

## Go to Screenshots Folder under the project folder.

## **5.3 Log4j Logs**

Go to the path setup according to your project.

**Appendix**

## **1.0 Java and Maven Project**

One can create Java/Maven project using IntelliJ tool. For Instance, Open IntelliJ – Browse Package Explorer – Mouse Right Click – Create New Java/Maven project.

## **1.1 Java Package**

One can create Java package under Project Name – src/test/java. The recommended way to create java package is to use organization name such as ‘**com.expoplatform.web.launcher’**. Here com stands for commercial, **expoplatform** stands for organization name, and Launcher stands for usage of that package.

## **1.2 Class**

One can create Java class under any package name define in Section 1.1. For instance, ‘SetUpWebTestEnv.java’. Here the class name refers to setting up Web Test Environment for running test execution. The naming convention of Class name should be first letter uppercase for easy readability.

## **1.3 Object**

One can create Java object to call any methods define in the class. For instance, the class name isOnBoarding.java and one need to call method define in the class via object.

Here is the syntax:

Onboarding = new Onboarding Page(testdriver);

OnBoarding.verify\_Concept\_Logo ();

## **1.4 Page Object**

A page object is methodology to capture UI object in one location. Thereafter, use those objects to perform an action such as click Button, add text in Text Field, click Link, access Text in UI application.

One can create Page Object using same methodology as defined in Section 1.1 and 1.2. The only recommended way to mention while naming Java Package. For instance, **com.expoplatform.web.PageObject** However, the Page Objects contains java source code.

## **1.6 Create sample step definition**

One can create sample test using existing test automation framework. For Mobile tests,‘**com.expoplatform.web.stepdefiniton.**

Currently, step definition with TestNG (Test Next Generation) is being used to add Mobile test. Here is the sample test syntax:

