**Automation Exercise**

**Design pattern concept**

The main idea behind the Page Object Model (POM) design pattern, is to create an Object Repository for the pages in our application/website, which we will then used in the tests. In other words, instead of including the page elements and the test code together within the test, we separate them into 2 different entities: the Objects and the TestCase. Using this concept, each page in the application will have a corresponding page class. The page classes will identify the elements in the page and contain methods that perform operations on these page elements

**Benefits**

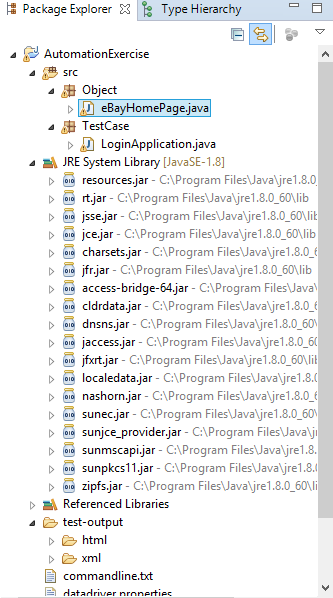
* Maintainable and Reliable
  + POM design pattern uses class hierarchy for common components
  + Each element in the entire application/website is mapped exactly once
  + Reduces code and code duplication
  + UI change fixes done in one place
* Readable, Modular and clear
  + Strict separation between test flows and app views, allows to create any test flow with the existing repository
  + Any change in the app causes a change only in one place (repository)
  + POM repository is not framework dependent

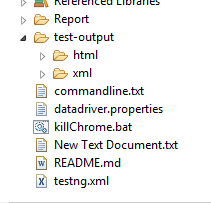
**Prerequisites**

* Eclipse
* Selenium version 2 with all required jar
* Java 1.8

The complete source code is available in GitHub: https://github.com/bhushanbhavsar17/AutomationExercise

**Snapshot:**





**Framework Details:**

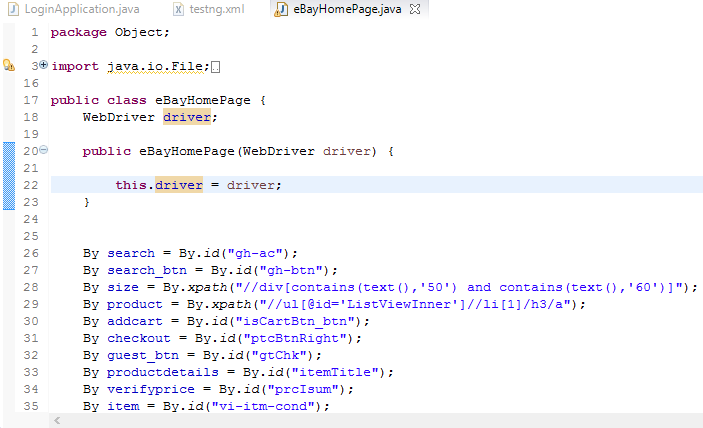
**Step 1:** Create a package for the Objects by adding a package to project name **AutomationExercise**.

**Step 2:** Add the landing page class to the package, named **eBayHomePage.Java**

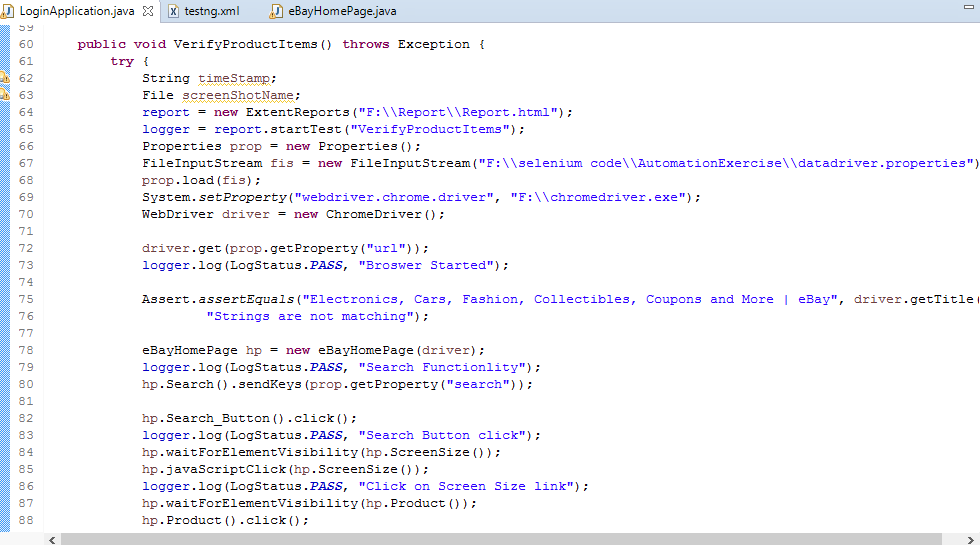
This class will contain all the objects locators and methods of the WebPage.

**Step 3:** Add a constructor to the **eBayHomePage**class in order to initialize the driver (i.e. assign the webDriver).

**Step 4:** Use XPath expressions to address the elements on the landing page.  
The Object Spy can help identify the elements on the page, and tailored XPath expressions can be used to address them.



**Step 5:** Create the **LoginApplication** class to the Testcase package.

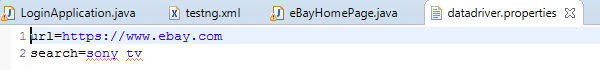


1) This class will have script for navigating the webpage. Script has Logger and assertion added to each step of validation.

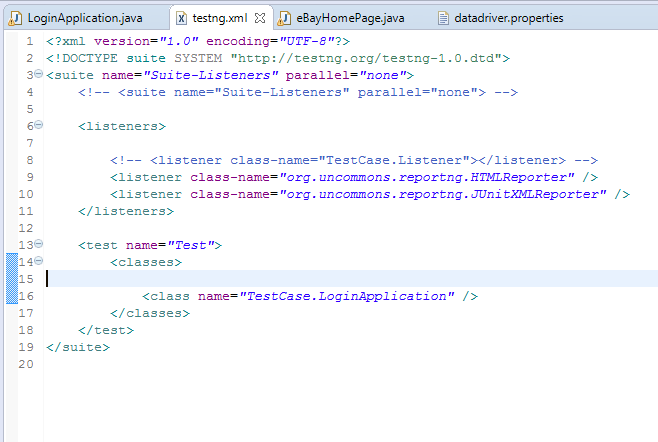
2) This class also implemented the ExtendReports for reporting the results.

3) Class have try catch block implemented. In try block script have been added and catch block contains if the script fails then it will throw exception with appropriate message.

**Step 6:** Adding external file named **DataDriver.Properties** which hasURL and search data.

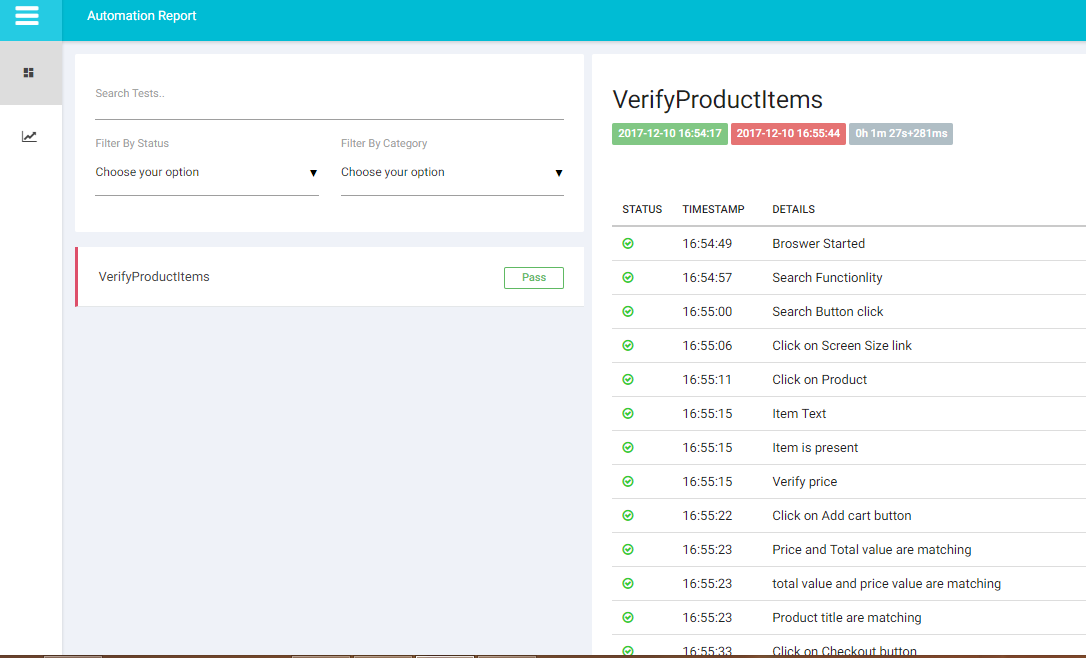


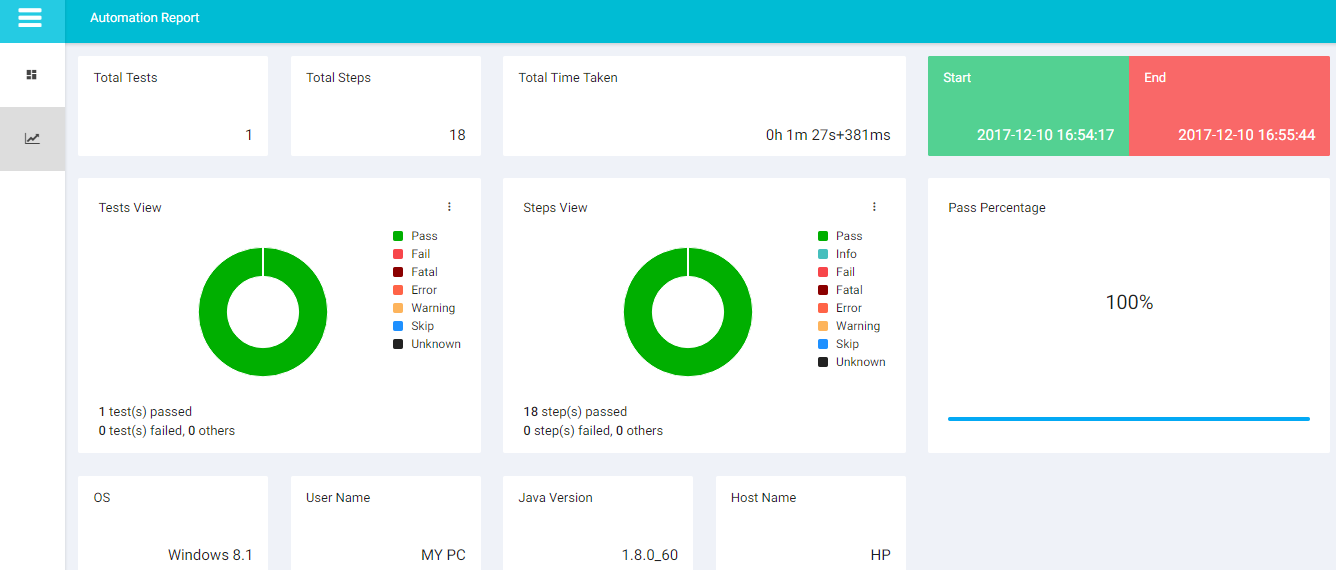
**Step 7:** Adding testng suite file named **testing.xml** having class name which need to be run.

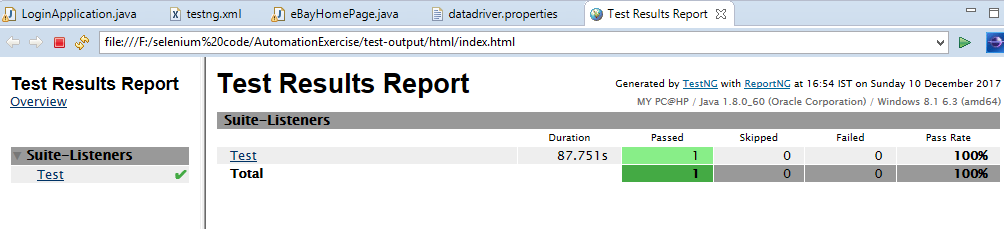


**Generating a report**

1) Report will be generated in Report folder and test-output folder under html folder.







**Run Project :**  Right click on testing.xml file and run as TestNG suite.

