Automate the Web Application

**By Anushka Kamath**

GitHub Link: <https://github.com/anushkakamath/simplilearn_bnp_project4_AutomateWebApp.git>

DESCRIPTION:

This project is developed to automate and test the given website link <http://automationpractice.com/index.php>. It contains various functionalities testing like creating a new user account (Registration), logging into to a user’s account (login), searching for a particular product and adding the product to the cart.

This project is developed using Selenium Chrome web driver and TestNG to automate the testing of the above-mentioned functionalities. The project tests each of the functionalities, which can be observed in the window that opens and also the console displays various outputs to indicate the passing of the tests.

FUNCTIONALITIES OF THE PROJECT:

The project folder titled “BNP\_Selenium\_TestNG” contains a test class (called TestApp.java) that contains the test scripts to automate the website testing using TestNG. Each of these methods are annotated with priority to define the order of execution. This contains 4 automated test methods in a prioritized order to run the tests in order namely:

1. registerTest()

This test method first clicks on the Sign in button on the website homepage. This redirects to the registration / login page. In the registration section the defined email id in that section and clicks on the “Create an account button”. This redirects to the next page that requires other information like gender, first name, last name, password and other details that are entered by finding the input section. It then clicks on the register button and redirects to the My Account page. Here the URL is checked to see if the registration is successful and prints the output on the console. It then logs out of this new account, and redirects to the login page.

1. loginTest()

This test method enters the email id and password in the Sign in section and clicks on the Sign in Button. This redirects to the My Account page. Here Assertion test on the URL is done to check if the Sign in was successful or not and prints the output on the console. Finally, the driver is navigated back to the homepage of the website.

1. searchTest()

In this automated test method, the driver finds the search box and enters the item name “Blouse” and clicks on the search symbol. This then takes the driver to the results of the search page. Here the title of the page is checked to print whether the search was successful on the console.

1. addToCartTest()

In this test, we add the item searched in the previous method to the cart. Thus, from the searched blouse page driver is made to click on the Item blouse to go to the next page that displays all the information of this item. The add to cart button is clicked to add the item into the cart. This gives a pop up that shows the item has been successfully added to the cart, on this pop up the proceed to checkout button is clicked. This shows the summary of the items in cart and again the message of successful test in printed on the console by using Assert.

The test class also contains two other methods namely:

1. initialize()

This method is annotated with @BeforeTest indicating this is the first method to run before testing any of the test cases. Within this block, the initialization steps are defined like initializing the new chrome driver, setting the path to find the chrome driver, specifying the URL for the test, etc.

1. close()

This method is annotated with @AfterTest indicating this block should be run after all the test cases have been tested. In this block, the chrome tab is simply closed after all the tests.

HOW TO RUN THE PROJECT:

This project is created as a maven project in Spring Tool Suite; hence all the necessary dependencies are mentioned in the pom.xml file.

To successfully run the project, the user needs to make sure he has installed the Selenium Chrome web driver exe file and its path needs to be provided within the **TestApp.java** class, to the **path** variable. The user also needs to make sure he has downloaded the TestNG software to be able to run the java class as a TestNG test or to run the testing suite.

1. Open the project folder “BNP\_Selenium\_TestNG” in Spring Tool Suite.
2. Add the TestNG library to the project by right clicking on the project name, clicking build path, then adding the TestNG library within the library section.
3. The path of Selenium Chrome web driver (exe file) should be added to the path variable in the TestApp.java file.

( snippet of code 🡪 public String path = "E:\\Tools\\drivers\\chromedriver.exe"; )

Add the path where chromedriver.exe lies in your machine.

1. Right click on the TestApp.java file and click on Run As > TestNG Test. This automatically runs all the test cases OR open the testing.xml file, Right click and Run As > TestNG Suite.
2. Refresh the project folder and open the index.html file (open with Browser) generated inside test-outputs folder to view report of the tests.