README

The problem chosen for the assignment is automating an end-to-end user e-commerce transaction flow using any open source tool for Walmart.com with an existing customer on chrome or safari

Technical Choice:

I chose Selenium Webdriver to automate the scenarios as I have tried automating web pages using selenium on my own interest in the past, to learn the framework and that is why I wanted to do the assignment with it.

Development tool used was eclipse IDE

Instructions to run:

1) There are totally four files as part of the solution: WalmarPages.Java, FunctionLibrary.java, objects.properties, application.properties.
2) Copy the folder Walmart_Automation from github in C:\
3) Hence above mentioned files will have the following folder path
   C:\Walmart_Automation\Walmart\bin\functions\FunctionLibrary.class
   C:\Walmart_Automation\Walmart\bin\functions\WalmartPages.class
   C:\Walmart_Automation\Walmart\bin\properties\application.properties
   C:\Walmart_Automation\Walmart\bin\properties\objects.properties
   Same goes with the files in the src folder.
4) Put the chromedriver.exe in the C folder → C:\
5) Copy the folder Selenium from github in C:\ , and hence the path of the library files will be like C:\Selenium\selenium-2.48.2\libs\
6) Import the project with eclipse as existing projects into workspace with root directory as
   C:\Walmart_Automation\Walmart
7) Configure Java build path to pull the required libraries from the c:\Selenium folder path
8) Run WalmartPages.java to start the execution
9) The console will have detailed result on what product was added to the cart and assert condition's output. If the Asserts fail, the console output will have that result as well.

Program execution flow:

1) open up walmart.com and click on sign in link to enter login credentials
2) after login , enter one of the search term and click enter
3) from the product list choose any one of the item and add it to the cart
4) verify the product title and number of items in the cart to match the item that was chosen to be added
5) remove the item from the cart and loop through other search terms repeating steps 3 and 4
6) close the browser once all search terms are covered
7) the console window gives the output of the execution

Assumptions/ Trade-Offs in the automating workflows:

1) The object.properties file already has valid username and password details of an existing customer.
2) The browser would automatically allow the location to be aware for the website and hence avoid problem of finding the product availability at a specific location.
3) From the result after searching for a specific search term, a random product is selected not restricting to a particular item from the list.
4) On automation involving 'iphone' as the search term, the program chooses the first product in the list and will only be able to choose any one of the color. The program ignores the ability to choose a specific color and availability at a specific location. If the phone chosen out of the list of items contains a color that is available, the program will have no problem selecting the first available color and adding it to the cart.
5) The title text of the product selected will have full name of the product and not truncated information with […] at the end. On few runs I saw product title selected has truncated text displayed due to limitTo method on the DOM element.
6) Testing on chrome driver, on some runs, loading the url 'http://www.walmart.com' and refreshing the page to avoid the modal dialog, does not give back the scope to the testing script. Another work around is to directly open up login page url by changing the url parameter to 'www.walmart.com/account/login' and enter credentials, skipping the popup page load.

Program Future Scope:

1) Create a testNG class and have this validation as part of a test method and having the login and other initial steps as part of before class. This is could further increase the scope for adding more tests and having a separate test output result.
2) Work on dealing with StaleObjectReferenceException in a much efficient way, which is now dealt with simple for loop to repeat object identification.
3) Work on better ways to deal with automating the iphone work flow taking into consideration of non-availability of a specific color
4) Improve code for robustness and add more object validation before making a click event.