To automate the Amazon application using Selenium WebDriver with Java

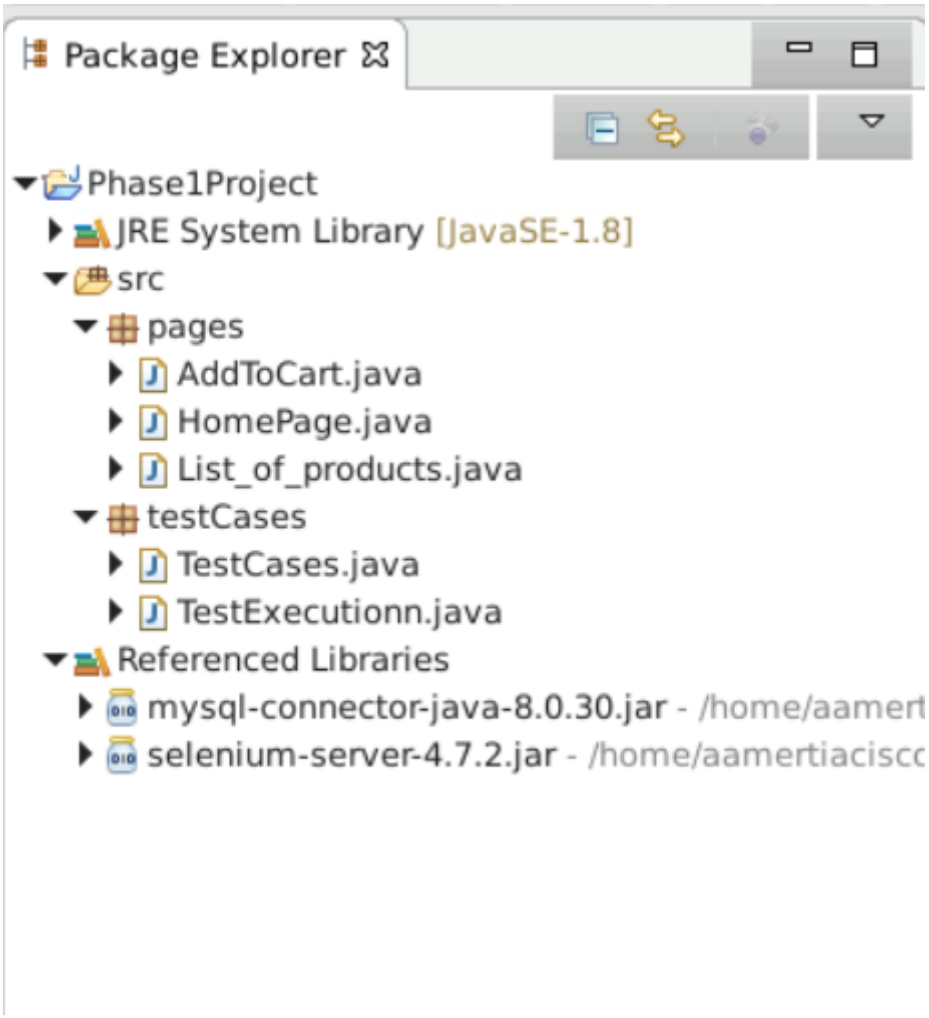
NAME -Aadarsh Singh Mertia

Git-
github(https://github.com/aadarsh76/Automating_Amazon_Using_Selenium_Webdriver.git)

Description: Automating the Amazon application to get the list of types of a particular product.

Functionality: Two packages are being created.

1. pages.
2. testCases.



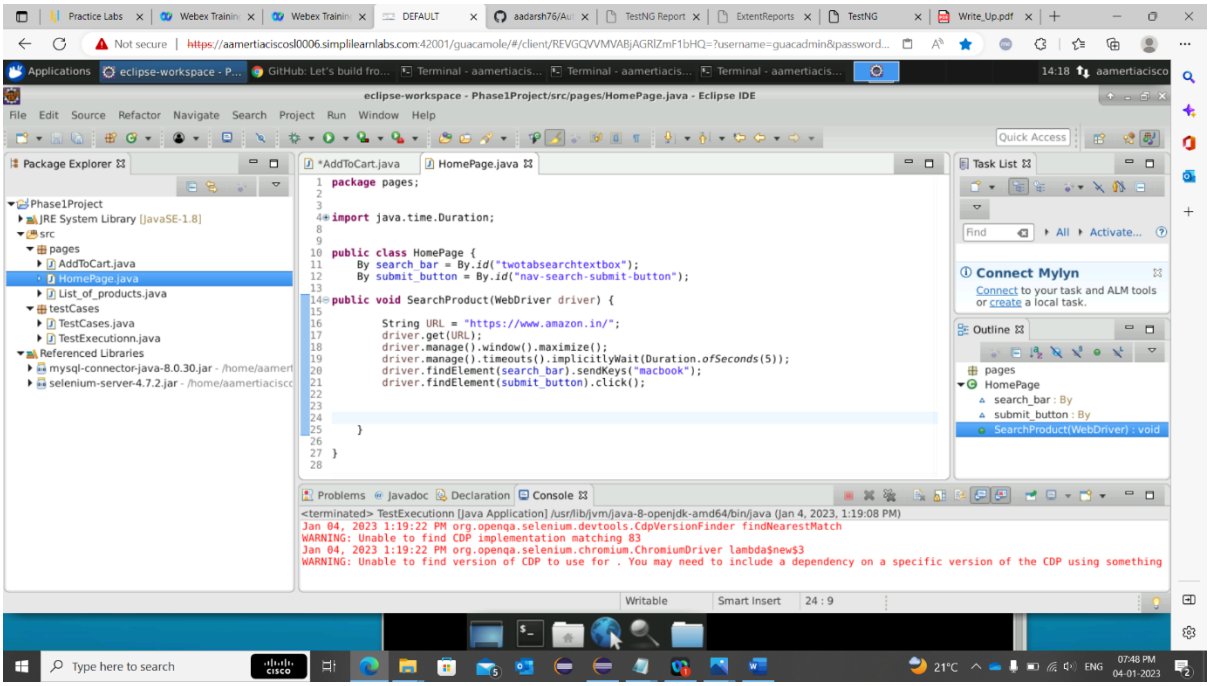
Pages: We have created 3 classes inside package pages

1. AddToCart
2. HomePage
3. List_of_products

AddToCart: Inside this there is one function called addtocart. Which will find the xpath of add to cart button and click on it.

```java
public class AddToCart {

    public void addintocart(WebDriver driver) {
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));
        driver.findElement(By.id("add-to-cart-button")).click();



    }


}
```

1.Homepage: Inside this there is two global variable names as search bar and submit button. There is one function name as Search Product which will open Amazon website then it will maximize window, also sometime our program execution happens faster but amazon website takes times to appear for that we will wait for max of 5 sec so that amazon website can appear, and then we will type MacBook into search box and we search for it.



2.List_of_products: Inside this we are doing our main work. There are 4 functions inside it, we will discuss about every function in detail here.

i) PrintFirst4Products_Name: Here we are taking an array of web element type and basically we are finding first 4 product appear first after searching for MacBook. Also as name of some product are too big so we are only considering first 20 character of the entire string name. We will return string array from this function.

```
15 public class List_of_products {
16
17     public String[] PrintFirst4Products_Name(WebDriver driver) {
18             List<WebElement> macbookList = driver.findElements(By.xpath("//*[@class=\"a-size-mec
19             System.out.println(macbookList.size());
20             String arrPro_name[] = new String[macbookList.size()];
21             int cnt = 0;
22             for(WebElement rb : macbookList) {
23
24
25
26                 String strname = rb.getText();
27                 String strname1 = strname.substring(0,20);
28                 arrPro_name[cnt] = strname1;
29                 cnt++;
30         }
31             for (int indx = 0; indx < 4; indx++) {
32                 System.out.println(arrPro_name[indx]);
33             }
34
35          return arrPro_name;
```

ii) PrintFirst4Products_Price: Here we are doing same instead of showing product name we are showing product price by same method as of previous function. Sometime prices shown will be 87,900 so it is difficult to compare, so for that we are removing, from the price string. We will return string array from this function.

```
41         public String[] PrintFirst4Products_Price(WebDriver driver) {
42
43             List<WebElement> list_of_price = driver.findElements(By.xpath("//*[@class=\"a-price-whole\"]"));
44             System.out.println(list_of_price.size());
45             String arrPro_price[] = new String[list_of_price.size()];
46             int cont=0;
47             for(WebElement rb : list_of_price) {
48                 String strname = rb.getText();
49                 strname = strname.replace(",", "");
50                 arrPro_price[cont] = strname;
51                 cont++;
52         }   System.out.println("priting price ");
53             for (int indx = 0; indx < 4; indx++) {
54             System.out.println(arrPro_price[indx]);
55         }
56
57             return arrPro_price;
58
59         }
60
```

iii) ValidateListofProducts: Here we are doing validation of the product after comparing product name from amazon website and from our database as well as we also comparing product price from amazon website and from databases. If they match, then we will show output this product matches else not matches. Here we are considering 4 string arrays 1. Acquiring string array from PrintFirst4Products_Name function. 2. Acquiring string array from PrintFirst4Products_Price function. 3. Getting product name from database. 4. Getting product price from database.
Prerequisites: We must create a database and table inside it and also put product name and product price manually so that we can validate. Also here we are connecting our application to the database and getting product name and product price inside a list. Then after having 4 string array, we are ready to make comparison.

```
public void ValidateListofProducts(WebDriver driver, String [] arrPro_name , String [] arrPro_price ) {

    database connection
    String connstring = "jdbc:mysql://localhost:3306/project";
    String strquery = "select * from macbook ";

    System.out.println("..................");
    String arrPro_namefromDB[] = new String[4];
    String arrPro_pricefromDB[] = new String[4];

    try {
        Connection conn = DriverManager.getConnection(connstring,"root","root");
        Statement statement = conn.createStatement();
        ResultSet rs = statement.executeQuery(strquery);
        System.out.println(rs.next());
        int x=0;
        while(rs.next()) {
        System.out.println(rs.getString(2)+" "+rs.getString(3)+" "+rs.getString(1));
        arrPro_namefromDB[x] = rs.getString(2);
        arrPro_pricefromDB[x] = rs.getString(3);
        x++;

        }
```

```
            } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            }
            System.out.println("printing arr_price list");


            for (int indx = 0; indx < 4; indx++) {
                Boolean flag=false;
                for(int j=0; j<4; j++) {

                    if (arrPro_name[indx].equals(arrPro_namefromDB[j])  &&  (arrPro_price[indx].equals(arrPro_pricefromDB[j]))) {

                        flag=true;
                        break;
                    }


                }

                if (flag==true) {
                    System.out.println(arrPro_name[indx]+ " "+"product matches");
//                  ClickOnProduct(driver);
                }else {
                    System.out.println(arrPro_name[indx]+" "+"product doesnt matches");
                }
            }

   System.out.println("everything working and fine");
}
```

iii) ClickOnProduct: Inside this we are passing xpath of any product and click on it . then it will show that product details inside another window. So first I close first window then I get into second window using switch function.

```
public void ClickOnProduct(WebDriver driver) {
    Actions action = new Actions(driver);
    ArrayList<String>Win_List   = new ArrayList<String>(driver.getWindowHandles());
    action.keyDown(Keys.CONTROL).build().perform();
    driver.findElement(By.xpath("//*[contains(text(),\"2020 Apple MacBook A\")]")).click();
    action.keyUp(Keys.CONTROL).build().perform();
    ArrayList<String>Win_List1   = new ArrayList<String>(driver.getWindowHandles());
    driver.switchTo().window(Win_List1.get(0));
    driver.close();
    driver.switchTo().window(Win_List1.get(1));


}
```

Inside testCases: We have created two classes: -

1. Testcases: Inside this we have 3 classes.

i) TestPrintListOfProducts: It will print first 4 product name and product price and then we close this driver. We are just calling desired function to do so.

```
public class TestCases {
    public void TestPrintListOfProducts() {


//       driver
        List_of_products listPage = new List_of_products();
        HomePage home = new HomePage();
        WebDriver driver = new ChromeDriver();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));

        home.SearchProduct(driver);
        listPage.PrintFirst4Products_Name(driver);
        listPage.PrintFirst4Products_Price(driver);

        driver.close();

}
```

ii) ValidateListOfProducts: Here we are calling desired function so that we can validate product, connect to database and show product details.

```
    public void ValidateListOfProducts() {

/       driver
        List_of_products listPage = new List_of_products();
      HomePage home = new HomePage();
      WebDriver driver = new ChromeDriver();
      driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));

      home.SearchProduct(driver);
      listPage.ValidateListofProducts(driver, listPage.PrintFirst4Products_Name(driver), listPage.PrintFirst4Products_Price(dr
/       listPage.ClickOnProduct(driver);
/       AddtoCart

      driver.close();


  }
```

iii) clickonproducts: Here we are clicking on the product and then adding into cart the cart.

```
public void clickonproducts() {
    WebDriver driver = new ChromeDriver();
    driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));

    HomePage home =new HomePage();
    List_of_products listpage = new List_of_products();
    AddToCart cart = new AddToCart();

    home.SearchProduct(driver);
    listpage.ClickOnProduct(driver);
    cart.addintocart(driver);

    try {
        Thread.sleep(5000);
    } catch (InterruptedException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

//  cart.viewCart(driver);
    driver.close();



}

}
```

2.TestExecutionn: There is only one function inside it i.e., main function. we don’t have main function until in any previous classes, here only we have main function, so that we can execute program from here only. So, we are calling desired testcase function here and letting program to run and show the output.

```java
package testCases;



public class TestExecutionn {
	public static void main(String[] args) {
		TestCases testCases = new TestCases();
		testCases.TestPrintListOfProducts();
		testCases.ValidateListOfProducts();
		testCases.clickonproducts();
	}
}

```