AdvantageDemo

Automation Project

Documentation

AdvantageDEMO

www.advantageonlineshopping.com

Prepared by:

Abu Saeed Mohammad Sayem

This is a project to test AdvantageDemo sample eCommerce website's functionality

Project Requirements:

We should have 5 stories in this project. The project should cover the following things in our test cases and test steps:

- How to open a browser
- How to open URL
- Click and sendkeys
- Synchronization
- Exceptions
- Select class
- Action class
- Testcase with data
- Testcase without data
- In JIRA you should have stories with tasks and points

To fulfill all of our requirements I have created five stories and a few sub-tasks. Let's see those stories and sub-tasks.

1. Validate Membership
    a. Validate Sign Up Process (We have to validate if user can create a new account.)
    b. Validate Sign In Process (We have to validate if user can sign in to his/her account using proper credentials.)
    c. Validate Account Deletion Process (We have to validate if user can delete their account.)
2. Validate Contact Us page (We have to validate if user can use contact us button and form to send a message.)
3. Validate Product Category (We have to validate if user can visit product categories.)
4. Validate Shopping Cart (We have to validate if user can add a product to the shopping cart.)
5. Validate Product Search ability (We have to validate if use can search products from this website.)

For this project I have created just one Page Object Model java file where I have stored all AdvantageDemo Website's UI element's locator. Then I have created 5 feature files to write all of my test cases for 5 different stories, and 5 different StepDefs java files to write test scripts. I have created all these feature files under the Features folder under the AdvantageDemo Maven Project. StepDefs test scripts are stands under StepDefs package of AdvantageDemo Maven Project. And as I said earlier, POM (Page Object Model) java file I have created under the PageObjectModel package of the project. Now I will show you all of my Test Cases, Test Scripts, and POMs along with my code documentation.

Here is my POM file:

```
package PageObjectModel;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class AdvantageDemoPOM {

	WebDriver driver; // Global variable for WebDriver
	/*
	 *  Constructor is not a method but a special type of method
	 *  1. Constructor shouldn't have any return type.
	 *  2. Constructor Name and Class Name should be the same.
	 */

	public AdvantageDemoPOM (WebDriver driver) {
		//This driver is a local driver inside the constructor.
		this.driver = driver; //Global driver = local driver
		PageFactory.initElements(driver, this);
	}

	//Story: Validate Membership\\
	//Task: Validate Signin Process\\

	// Element 1
	@FindBy(xpath="//a[@id=\"hrefUserIcon\"]") // Used custom 'xpath' locator to 
identify User icon
	WebElement click_member_icon;
	public WebElement MemberiCon(){
		return click_member_icon;
	}
	// Element 2
	@FindBy(xpath="//input[@name=\"username\"]") // Used custom 'xpath' locator 
to identify login username text box
	WebElement type_login_username;
	public WebElement LoginUsername(){
		return type_login_username;
	}
	// Element 3
	@FindBy(xpath="//input[@name=\"password\"]") // Used custom 'xpath' locator 
to identify login password text box
	WebElement type_login_password;
	public WebElement LoginPassword(){
		return type_login_password;
	}
	// Element 4
	@FindBy(xpath="//input[@name=\"remember_me\"]") // Used custom 'xpath'
locator to identify Remember Me Check Box
	WebElement click_remember_me;
	public WebElement LoginRememberMe(){
		return click_remember_me;
	}
	// Element 5
	@FindBy(xpath="//button[@id=\"sign_in_btnundefined\"]") // Used custom
'xpath' locator to identify Sign In Button
	WebElement click_signin;
	public WebElement SignIn(){
		return click_signin;
	}
	//Bonus Element Sign Out
	@FindBy(xpath="//*[@id=\"loginMiniTitle\"]/*[3]") // Used custom 'xpath'
locator to identify Sign Out Button
	WebElement click_signout;
	public WebElement AccountSignOut() {
		return click_signout;
	}

	//Story: Validate Membership\\
	//Task: Validate Signup Process\\

	// Element 6
	@FindBy(xpath="//a[@class=\"create-new-account ng-scope\"]") // Used custom
'xpath' locator to identify Create New Account Link
	WebElement click_new_account;
	public WebElement NewAccount(){
		return click_new_account;
	}
	// Element 7
	@FindBy(xpath="//input[@name=\"usernameRegisterPage\"]") // Used custom
'xpath' locator to identify SignUp Username Text Box
	WebElement type_signup_username;
	public WebElement SignUpUsername(){
		return type_signup_username;
	}
	// Element 8
	@FindBy(xpath="//input[@name=\"emailRegisterPage\"]") // Used custom 'xpath'
locator to identify SignUp Email Text Box
	WebElement type_signup_email;
	public WebElement SignUpEmail(){
		return type_signup_email;
	}
	// Element 9
	@FindBy(xpath="//input[@name=\"passwordRegisterPage\"]") // Used custom 
'xpath' locator to identify SignUp Password Box
	WebElement type_signup_password;
	public WebElement SignUpPassword(){
		return type_signup_password;
	}
	// Element 10
	@FindBy(xpath="//input[@name=\"confirm_passwordRegisterPage\"]") // Used 
custom 'xpath' locator to identify SignUp Confirm Password Box
	WebElement type_signup_confirm_password;
	public WebElement SignUpConfirmPassword(){
		return type_signup_confirm_password;
	}
	// Element 11
	@FindBy(xpath="//input[@name=\"first_nameRegisterPage\"]") // Used custom 
'xpath' locator to identify SignUp First Name Box
	WebElement type_first_name;
	public WebElement SignUpFirstName(){
		return type_first_name;
	}
	// Element 12
	@FindBy(xpath="//input[@name=\"last_nameRegisterPage\"]") // Used custom 
'xpath' locator to identify SignUp Last Name Box
	WebElement type_last_name;
	public WebElement SignUpLastName(){
		return type_last_name;
	}
	// Element 13
	@FindBy(xpath="//input[@name=\"phone_numberRegisterPage\"]") // Used custom 
'xpath' locator to identify Phone Number Box
	WebElement type_phone;
	public WebElement SignUpPhone(){
		return type_phone;
	}
	// Element 14
	@FindBy(xpath="//select[@name=\"countryListboxRegisterPage\"]") // Used 
custom 'xpath' locator to identify SignUp Country DropDown
	WebElement select_country;
	public WebElement SignUpCountry(){
		return select_country;
	}
	// Element 15
	@FindBy(xpath="//input[@name=\"cityRegisterPage\"]") // Used custom 'xpath' 
locator to identify SignUp City Box
	WebElement type_city;
	public WebElement SignUpCity(){
		return type_city;
	}
	// Element 16
	@FindBy(xpath="//input[@name=\"addressRegisterPage\"]") // Used custom 
'xpath' locator to identify SignUp Address Box
	WebElement type_signup_address;
	public WebElement SignUpAddress(){
		return type_signup_address;
	}
	// Element 17
	@FindBy(xpath="//input[@name=\"state_/_province_/_regionRegisterPage\"]") // 
Used custom 'xpath' locator to identify SignUp State Box
	WebElement type_state;
	public WebElement SignUpState(){
		return type_state;
	}
	// Element 18
	@FindBy(xpath="//input[@name=\"postal_codeRegisterPage\"]") // Used custom 
'xpath' locator to identify SignUp Postcode Box
	WebElement type_postcode;
	public WebElement SignUpPostCode(){
		return type_postcode;
	}
	// Element 19
	@FindBy(xpath="//input[@name=\"i_agree\"]") // Used custom 'xpath' locator 
to identify 'I agree' Check Box
	WebElement click_agree;
	public WebElement SignUpAgree(){
		return click_agree;
	}
	// Element 20
	@FindBy(xpath="//button[@id=\"register_btnundefined\"]") // Used custom 
'xpath' locator to identify Register Button
	WebElement click_register;
	public WebElement SignUpRegister(){
		return click_register;
	}
	
	//Story: Validate Membership\\
	//Task: Validate Account Deletion Process\\
	
	// Element 21
	@FindBy(xpath="//a[@id=\"menuUserLink\"]") // Used custom 'xpath' locator to 
identify User Account Button
	WebElement click_user_account;
	public WebElement UserAccountPage() {
		return click_user_account;
	}
	// Element 22
	@FindBy(xpath="//*[@id=\"loginMiniTitle\"]/*[1]") // Used 'xpath' locator to 
identify My Account Option
	WebElement click_my_account;
	public WebElement MyAccount() {
		return click_my_account;
	}
	// Element 23
	@FindBy(xpath="//*[@class=\"deleteMainBtnContainer a-button ng-scope\"]") // 
Used custom 'xpath' locator to identify Delete Account Button
	WebElement click_delete_account;
	public WebElement DeleteAccount() {
		return click_delete_account;
	}
	// Element 24
	@FindBy(xpath="//*[@class=\"deleteBtnContainer\"]/*[1]") // Used custom 
'xpath' locator to identify Delete Confirm Button
	WebElement click_yes;
	public WebElement DeleteYes() {
		return click_yes;
	}
	// Element 25
	@FindBy(xpath="//*[@id=\"loginMiniTitle\"]/*[3]") // Used 'xpath' locator to 
identify Sign Out Option
	WebElement click_sign_out;
	public WebElement SignOut() {
		return click_sign_out;
	}

	//Story: Validate Contact Us\\

	// Element 26
	@FindBy(xpath="//*[@ng-click=\"gotoElement('contact_us')\"]") // Used custom 
'xpath' locator to identify Contact Us Button
	WebElement click_contact_us;
	public WebElement ContactUs() {
		return click_contact_us;
	}
	// Element 27
	@FindBy(xpath="//*[@name=\"categoryListboxContactUs\"]") // Used custom 
'xpath' locator to identify Contact Us Product Type
	WebElement select_contact_product;
	public WebElement ContactProductsType() {
		return select_contact_product;
	}
	// Element 28
	@FindBy(xpath="//*[@name=\"productListboxContactUs\"]") // Used custom 
'xpath' locator to identify Contact Us Products
```

```
	WebElement select_contact_products;
	public WebElement ContactProducts() {
		return select_contact_products;
	}
	// Element 29
	@FindBy(xpath="//*[@name=\"emailContactUs\"]") // Used custom 'xpath' locator to identify Contact Us Email
	WebElement type_contact_email;
	public WebElement ContactUsEmail() {
		return type_contact_email;
	}
	// Element 30
	@FindBy(xpath="//*[@name=\"subjectTextareaContactUs\"]") // Used custom 'xpath' locator to identify Contact Us Subject Box
	WebElement type_contact_subject;
	public WebElement ContactUsSubject() {
		return type_contact_subject;
	}
	// Element 31
	@FindBy(xpath="//*[@id=\"send_btnundefined\"]") // Used custom 'xpath' locator to identify Contact Us Send Button
	WebElement click_contact_send;
	public WebElement ContactUsSend() {
		return click_contact_send;
	}

	//Story: Validate Product Search ability\\

	// Element 32
	@FindBy(xpath="//*[@id=\"search\"]") // Used custom 'xpath' locator to identify Search Button
	WebElement click_search;
	public WebElement ClickSearch() {
		return click_search;
	}

	//Element 45
	@FindBy(xpath="//*[@id=\"menuSearch\"]") // Used custom 'xpath' locator to identify search button
	WebElement click_search_second;
	public WebElement SearchClick() {
		return click_search_second;
	}

	// Element 33
	@FindBy(xpath="//*[@id=\"autoComplete\"]") // Used custom 'xpath' locator to identify Search Box
	WebElement type_search;
	public WebElement Search() {
		return type_search;
	}

	// Element 34
	//Following xpath not only define Web UI Element but also defines specific 
text by using selenium text() function.
	@FindBy(xpath="//*[@id=\"output\"]//*[text()='View All']") // Used custom 
'xpath' locator to identify View All Link Text
	WebElement click_view_all;
	public WebElement ClickViewAll() {
		return click_view_all;
	}
	// Element 35
	@FindBy(xpath="//*[@src=\"../../css/images/closeDark.png\"]") // Used custom 
'xpath' locator to identify close button of search box
	WebElement click_close;
	public WebElement CloseSearch() {
		return click_close;
	}

	//Story: Validate Home Page Headphone Category\\

	// Element 36
	@FindBy(xpath="//*[@id=\"headphonesImg\"]") // Used custom 'xpath' locator 
to identify Headphones category Image.
	WebElement hover_headphone;
	public WebElement HoverHeadphone() {
		return hover_headphone;
	}
	// Element 37
	@FindBy(xpath="//*[@id=\"headphonesLink\"]") // Used custom 'xpath' locator 
to identify Headphones category Shop Now Button.
	WebElement click_headphone_shop_now;
	public WebElement HeadphoneShopNow() {
		return click_headphone_shop_now;
	}

	//Story: Validate Shopping Cart\\

	// Element 38
	@FindBy(xpath="//*[text()='HP Chromebook 14 G1(ENERGY STAR)']") // Used 
custom 'xpath' locator with Text() function to identify specific product
	WebElement click_product;
	public WebElement ClickChromebook() {
		return click_product;
	}
	// Element 39
	@FindBy(xpath="//*[@ng-show=\"!firstImageToShow\"]/*[1]") // Used custom 
'xpath' locator to identify BLUE color
	WebElement click_blue;
	public WebElement ColorBlack() {
		return click_blue;
	}
	// Element 40
	@FindBy(xpath="//*[@name=\"quantity\"]") //Used custom 'xpath' locator to 
identify quantity input box
	WebElement type_quantity;
	public WebElement Quantity() {
		return type_quantity;
	}
	// Element 41
	@FindBy(xpath="//*[@name=\"save_to_cart\"]") //Used custom 'xpath' locator 
to identify Add To Cart Button
	WebElement click_addtocart;
	public WebElement AddToCart() {
		return click_addtocart;
	}
	// Element 42
	@FindBy(xpath="//*[@id=\"menuCart\"]") //Used custom 'xpath' locator to 
identify Cart Button from top menu bar
	WebElement click_view_cart;
	public WebElement ViewCart() {
		return click_view_cart;
	}
	// Element 43
	@FindBy(xpath="//*[@class=\"actions\"]//*[text()='REMOVE']") //Used custom 
'xpath' locator with text() function to identify REMOVE button
	WebElement click_remove_cart;
	public WebElement RemoveCart() {
		return click_remove_cart;
	}
	// Element 44
	@FindBy(xpath="//*[@class=\"a-button ng-scope\"]") //Used custom 'xpath' 
locator to identify CONTINUE SHOPPING button
	WebElement click_continue_shopping;
	public WebElement ContinueShopping() {
		return click_continue_shopping;
	}
}
```

Then based on our requirements first I have create a feature file to write my first test cases. This test case I have written to validate AdvantageDemo Membership story. I have used Gherkin language to write this test case. Let's see that feature file below.

First Story. First Test Case

Feature File: AdvantageDemoMembership.feature. It’s a Cucumber feature file written in Gherkin Language with multiple test cases.

```
#Author: Abu Saeed M Sayem AKA Kaspar Moon
#Author Email: kasparmoonva@gmail.com
#Author Website: www.kasparmoon.us
#Story: Membership Validation
#Story number - TW01
#Product Website: www.advantageonlineshopping.com
#Feature: Membership Validation for AdvantageDemo
#Scenario: Positive Test
#Drafted on October 16, 2021
@membership
Feature: Membership Validation for AdvantageDemo
  Goal of this feature is to Membership Validation for AdvantageDemo website.
  To do so, we will create three different test cases for signup,
  sign in, and account deletion.
  User will visit AdvantageDemo website then create an account first then
  sign into that account and at last user will delete that account.

  @createaccount
  Scenario Outline: Create an Account
    Given Visit AdvantageDemo website
    And User will click on the user icon
    When User click on create a new account text and redirect to registration
page
    And User should type "<Username>", "<Email>", "<Password>", and "<Confirm
Password>"
    And User will also type "<First Name>", "<Last Name>", and "<Phone Number>"
    And User also select country and type "<City>", "<Address>", "<State>", and
"<Postal Code>"
    And User will agree with terms conditions
    Then User should click on the register button
    And User will redirect to the homepage

    Examples:
      | Username   | Email                | Password | Confirm Password | First
Name | Last Name | Phone Number | City   | Address        | State    | Postal Code
|
      | kasparmoon | kaspar@kasparmoon.us | Km123456 | Km123456         | Kaspar
| Moon      |   4503402345 | Albany | 5 Central Ave | New York |       12301 |

  @accountsignin
  Scenario Outline: Sign in to the existing account
    Given AdvantageDemo website visit by user
    And Click on user icon by user
    When Type "<Username>" and "<Password>" by user
    And Click on sign in button
    And User can see username beside user icon
    Then User will click on the username
    And Click on the sign out option
    And User will sign out

    Examples:
      | Username   | Password |
      | kasparmoon | Km123456 |

  @deleteaccount
  Scenario Outline: Sign in to the existing account and delete that account
    Given AdvantageDemo visits by user
    And User just click on user icon
    When User just type "<Username>" and "<Password>"
    And User just click on sign in
    And User will just click on username and my account option
    Then User just will redirect to account page and click on delete account
button
    And User can see homepage

    Examples:
      | Username   | Password |
      | kasparmoon | Km123456 |
```

Here I have written test cases based on three different scenarios. So that I can create an account, sign in to that account and then delete that account. By this way I have validate that AdvantageDemo website membership system is properly working.

Based on this feature file's test cases I have created one StepDefs java file for my test script to execute AdvantageDemo Membership validation tests. I have named this StepDefs java file as "AdvantageDemoMembership" under the "StepDefs" package. Here I am going to represent my "AdvantageDemoMembership" java file with my test script with all my documentation.

```
package StepDefs;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;

import PageObjectModel.AdvantageDemoPOM;
import cucumber.api.java.en.And;
import cucumber.api.java.en.Given;
import cucumber.api.java.en.Then;
import cucumber.api.java.en.When;

public class AdvantageDemoMembership {

      WebDriver driver; //Declaring Selenium WebDriver Global veritable.

      //Test Steps for Creating an Account

      @Given("^Visit AdvantageDemo website$")
      public void visit_AdvantageDemo_website() throws Throwable {
            //Using Try/Catch block in case there is any run time error occurs.
            try {
                  System.setProperty("webdriver.chrome.driver","C:\\Program
Files\\Chrome Driver\\chromedriver.exe"); //To open GoogleChrome browser
                  driver = new ChromeDriver(); //Getting all ChromeDriver methods
inside driver.
                  driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
// Implicit wait is a soft wait to fix synchronizing issue.
                  driver.manage().window().maximize(); //To maximize the window
                  driver.get("https://www.advantageonlineshopping.com"); //To open
URL of AdvantageDemo Website
            } catch (Exception e) {
                  System.out.println("Check your Browser's Driver Update");
                  System.out.println("Cause is: " + e.getCause()); //To get cause
of the exception.
                  System.out.println("Message is: " + e.getMessage()); //To get
exception message.
                  e.printStackTrace(); //It helps to trace the exception as I said
at the beginning.
            }
      }

      @And("^User will click on the user icon$")
      public void user_will_click_on_the_user_icon() throws Throwable {
            AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
            CreateAccount.MemberiCon().click(); // To click on the User icon
            Thread.sleep(4000); // Wait to see the result
      }

      @When("^User click on create a new account text and redirect to registration
page$")
      public void
user_click_on_create_a_new_account_text_and_redirect_to_registration_page()
throws Throwable {
```

```
            AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
            CreateAccount.NewAccount().click(); // To click on the create new
account text
            Thread.sleep(5000); // Wait to see the result
      }

      @And("^User should type \"([^\"]*)\", \"([^\"]*)\", \"([^\"]*)\", and
\"([^\"]*)\"$")
      public void user_should_type_and(String arg1, String arg2, String arg3,
String arg4) throws Throwable {
            AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
            CreateAccount.SignUpUsername().sendKeys(arg1); // To type username
            CreateAccount.SignUpEmail().sendKeys(arg2); // To type email
            CreateAccount.SignUpPassword().sendKeys(arg3); // To type password
            CreateAccount.SignUpConfirmPassword().sendKeys(arg4); // To type
confirm password
            WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
      }

      @And("^User will also type \"([^\"]*)\", \"([^\"]*)\", and \"([^\"]*)\"$")
      public void user_will_also_type_and(String arg1, String arg2, String arg3)
throws Throwable {
            AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
            CreateAccount.SignUpFirstName().sendKeys(arg1); // To type first name
            CreateAccount.SignUpLastName().sendKeys(arg2); // To type last name
            CreateAccount.SignUpPhone().sendKeys(arg3); // To type phone number
            WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
      }

      @And("^User also select country and type \"([^\"]*)\", \"([^\"]*)\",
\"([^\"]*)\", and \"([^\"]*)\"$")
      public void user_also_select_country_and_type_and(String arg1, String arg2,
String arg3, String arg4) throws Throwable {
            AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
            Select Country = new Select(CreateAccount.SignUpCountry()); //
Creating Select Object from Selenium for select the country
            Country.selectByVisibleText("United States"); // To select country
name from the drop down list
            CreateAccount.SignUpCity().sendKeys(arg1); // To type the city
            CreateAccount.SignUpAddress().sendKeys(arg2); // To type address
            CreateAccount.SignUpState().sendKeys(arg3); // To type state
            CreateAccount.SignUpPostCode().sendKeys(arg4); // To type post code
```

```
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@And("^User will agree with terms conditions$")
	public void user_will_agree_with_terms_conditions() throws Throwable {
		AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		CreateAccount.SignUpAgree().click(); // To click on 'I agree' option
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@Then("^User should click on the register button$")
	public void user_should_click_on_the_register_button() throws Throwable {
		AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		CreateAccount.SignUpRegister().click(); // To click on the Register
button
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@And("^User will redirect to the homepage$")
	public void user_will_redirect_to_the_homepage() throws Throwable {
		Thread.sleep(5000); // Wait to see the result
		driver.close(); // To close Internet connection with WebDriver
		driver.quit(); // To quit browser
	}

	//Test Steps for Sign in

	@Given("^AdvantageDemo website visit by user$")
	public void advantagedemo_website_visit_by_user() throws Throwable {
		//Using Try/Catch block in case there is any run time error occurs.
		try {
			System.setProperty("webdriver.chrome.driver","C:\\Program
Files\\Chrome Driver\\chromedriver.exe"); //To open GoogleChrome browser
			driver = new ChromeDriver(); //Getting all ChromeDriver methods
inside driver.
			driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
// Implicit wait is a soft wait to fix synchronizing issue.
			driver.manage().window().maximize(); //To maximize the window
			driver.get("https://www.advantageonlineshopping.com"); //To open
URL of AdvantageDemo Website
		} catch (Exception e) {
			System.out.println("Check your Browser's Driver Update");
```

```
                System.out.println("Cause is: " + e.getCause()); //To get cause
of the exception.
                System.out.println("Message is: " + e.getMessage()); //To get
exception message.
                e.printStackTrace(); //It helps to trace the exception as I said
at the beginning.
           }
      }

      @And("^Click on user icon by user$")
      public void click_on_user_icon_by_user() throws Throwable {
           AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
           CreateAccount.MemberiCon().click(); // To click on the User icon
           Thread.sleep(4000); // Wait to see the result
      }

      @When("^Type \"([^\"]*)\" and \"([^\"]*)\" by user$")
      public void type_and_by_user(String arg1, String arg2) throws Throwable {
           AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
           CreateAccount.LoginUsername().sendKeys(arg1); // To type username
           CreateAccount.LoginPassword().sendKeys(arg2); // To type password
           CreateAccount.LoginRememberMe().click(); // To click on the 'I agree'
option
           WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
      }

      @And("^Click on sign in button$")
      public void click_on_sign_in_button() throws Throwable {
           AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
           CreateAccount.SignIn().click(); // To click on the Sign In button
           WebDriverWait wait = new WebDriverWait(driver,40); // Explicit wait is
a soft wait to fix synchronizing issue.
      }

      @And("^User can see username beside user icon$")
      public void user_can_see_username_beside_user_icon() throws Throwable {
           Thread.sleep(4000); // Wait to see the result
      }

      @Then("^User will click on the username$")
      public void user_will_click_on_the_username() throws Throwable {
           AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
```

```
		CreateAccount.UserAccountPage().click(); // To click on the username 
beside user icon
		WebDriverWait wait = new WebDriverWait(driver,40); // Explicit wait is 
a soft wait to fix synchronizing issue.
	}

	@And("^Click on the sign out option$")
	public void click_on_the_sign_out_option() throws Throwable {
		AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver); 
//Object Created to get Web Elements from Page Object Model
		CreateAccount.AccountSignOut().click(); // To click on the sign out 
button
		WebDriverWait wait = new WebDriverWait(driver,40); // Explicit wait is 
a soft wait to fix synchronizing issue.
	}

	@And("^User will sign out$")
	public void user_will_sign_out() throws Throwable {
		Thread.sleep(5000); // Wait to see the result
		driver.close(); // To close Internet connection with WebDriver
		driver.quit(); // To quit browser
	}

	//Test Steps for Delete an Account

	@Given("^AdvantageDemo visits by user$")
	public void advantagedemo_visits_by_user() throws Throwable {
		//Using Try/Catch block in case there is any run time error occurs.
		try {
			System.setProperty("webdriver.chrome.driver","C:\\Program 
Files\\Chrome Driver\\chromedriver.exe"); //To open GoogleChrome browser
			driver = new ChromeDriver(); //Getting all ChromeDriver methods 
inside driver.
			driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS); 
// Implicit wait is a soft wait to fix synchronizing issue.
			driver.manage().window().maximize(); //To maximize the window
			driver.get("https://www.advantageonlineshopping.com"); //To open 
URL of AdvantageDemo Website
		} catch (Exception e) {
			System.out.println("Check your Browser's Driver Update");
			System.out.println("Cause is: " + e.getCause()); //To get cause 
of the exception.
			System.out.println("Message is: " + e.getMessage()); //To get 
exception message.
			e.printStackTrace(); //It helps to trace the exception as I said 
at the beginning.
		}
	}
	@And("^User just click on user icon$")
	public void user_just_click_on_user_icon() throws Throwable {
		AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		CreateAccount.MemberiCon().click(); // To click on the User icon
		Thread.sleep(4000); // Wait to see the result
	}

	@When("^User just type \"([^\"]*)\" and \"([^\"]*)\"$")
	public void user_just_type_and(String arg1, String arg2) throws Throwable {
		AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		CreateAccount.LoginUsername().sendKeys(arg1); // To type username
		CreateAccount.LoginPassword().sendKeys(arg2); // To type password
		CreateAccount.LoginRememberMe().click(); // To click on the 'I agree'
option
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@And("^User just click on sign in$")
	public void user_just_click_on_sign_in() throws Throwable {
		AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		CreateAccount.SignIn().click(); // To click on the Sign In button
		Thread.sleep(4000); // Wait to see the result
	}

	@And("^User will just click on username and my account option$")
	public void user_will_just_click_on_username_and_my_account_option() throws
Throwable {
		AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		CreateAccount.UserAccountPage().click(); // To click on the username
beside user icon
		CreateAccount.MyAccount().click(); // To click on the my account
option
		WebDriverWait wait = new WebDriverWait(driver,40); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@Then("^User just will redirect to account page and click on delete account
button$")
	public void
user_just_will_redirect_to_account_page_and_click_on_delete_account_button()
throws Throwable {
		Thread.sleep(4000); // Wait to see the result
		AdvantageDemoPOM CreateAccount = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		CreateAccount.DeleteAccount().click(); // To click on the Delete
button
		CreateAccount.DeleteYes().click(); // To click on the yes button to
confirm delete
		WebDriverWait wait = new WebDriverWait(driver,40); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@And("^User can see homepage$")
	public void user_can_see_homepage() throws Throwable {
		Thread.sleep(5000); // Wait to see the result
		driver.close(); // To close Internet connection with WebDriver
		driver.quit(); // To quit browser
	}

}
```

When I have executed this script I have found my script can automatically open the browser, visit AdvantageDemo Website and create an account. Then the script can automatically open the browser, visit AdvantageDemo Website and sign in to that new account and sign out automatically. At the end, script can automatically open the browser, visit AdvantageDemo Website and visit “My Account” page after sign in and delete that account. So, finally I can say that I test cases and test script successfully completed the validation process of AdvantageDemo website’s Membership.
For the next story I have created one more feature file to validate AdvantageDemo Contact Us form functionality. This feature file contains two different test cases two validate contact us form’s functionality with two different Scenarios.

Second Story. Second Test Case

Feature File: AdvantageDemoContactUs.feature. It’s a Cucumber feature file written in Gherkin Language.
Here is my feature file’s code below:

```
#Author: Abu Saeed M Sayem AKA Kaspar Moon
#Author Email: kasparmoonva@gmail.com
#Author Website: www.kasparmoon.us
#Story: Validate Contact Us
#Story number - TW02
#Product Website: www.advantageonlineshopping.com
#Feature: Validate Contact Us
#Scenario: Positive Test
#Drafted on October 1, 2021
@contact
```

```
Feature: Validate Contact Us
  To validate contact us form user needs to visit AdvantageDemo Website
  and click on Contact Us button then user will see contact form bottom
  of the page. User needs to fill-up the form and click of the Send button.
  User will see message sent notification.

  @contactusforproduct
  Scenario Outline: Validate Contact Us Form
    Given User will visit AdvantageDemo website
    When User click on contact us button
    And User will select product category
    And User will select a product
    And User type "<email>" address
    And User type "<subject>" message
    Then User click on send button
    And User see the notification

    Examples:
      | email                   | subject
|
      | kasparmoonva@gmail.com | This is a test message subject for AdvantageDemo
website. |

  @contactuswithoutproduct
  Scenario Outline: Validate Contact Us Form
    Given User should open AdvantageDemo website
    When User click contact button
    And User should type "<email>" address
    And User should type "<subject>" message
    Then User should click the send button
    And User will see notification

    Examples:
      | email                   | subject
|
      | kasparmoonva@gmail.com | This is a test message subject for AdvantageDemo
website. |
```

Based on this feature file, I have created “AdvantageDemoContactUs” java file under the “StepDefs” Package to write my test script.

```
package StepDefs;

import java.util.concurrent.TimeUnit;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import PageObjectModel.AdvantageDemoPOM;
import cucumber.api.java.en.Given;
import cucumber.api.java.en.Then;
import cucumber.api.java.en.When;

public class AdvantageDemoContactUs {
	WebDriver driver; //Declaring Selenium WebDriver Global veritable.

	@Given("^User will visit AdvantageDemo website$")
	public void user_will_visit_AdvantageDemo_website() throws Throwable {
		//Using Try/Catch block in case there is any run time error occurs.
				try {

	System.setProperty("webdriver.chrome.driver","C:\\Program Files\\Chrome
Driver\\chromedriver.exe"); //To open GoogleChrome browser
					driver = new ChromeDriver(); //Getting all
ChromeDriver methods inside driver.
					driver.manage().timeouts().implicitlyWait(30,
TimeUnit.SECONDS); // Implicit wait is a soft wait to fix synchronizing issue.
					driver.manage().window().maximize(); //To maximize the
window
					driver.get("https://www.advantageonlineshopping.com");
//To open URL of AdvantageDemo Website
				} catch (Exception e) {
					System.out.println("Check your Browser's Driver
Update");
					System.out.println("Cause is: " + e.getCause()); //To
get cause of the exception.
					System.out.println("Message is: " + e.getMessage());
//To get exception message.
					e.printStackTrace(); //It helps to trace the exception
as I said at the beginning.
				}
	}

	@When("^User click on contact us button$")
	public void user_click_on_contact_us_button() throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		ContactUsForm.ContactUs().click(); //Click on the contact us button
		Thread.sleep(4000); // Wait to see the result
	}

	@When("^User will select product category$")
	public void user_will_select_product_category() throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
```

```
		Select ProductType = new Select (ContactUsForm.ContactProductsType());
//Used Select class for selecting value from a dropdown menu
		ProductType.selectByVisibleText("Speakers"); // Selecting visible text
value from dropdown menu
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@When("^User will select a product$")
	public void user_will_select_a_product() throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		Select ProductType = new Select (ContactUsForm.ContactProducts());
//Used Select class for selecting value from a dropdown menu
		ProductType.selectByVisibleText("HP Roar Plus Wireless Speaker"); //
Selecting index wise value from dropdown menu
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@When("^User type \"([^\"]*)\" address$")
	public void user_type_address(String arg1) throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		ContactUsForm.ContactUsEmail().sendKeys(arg1); //Typing email address
into the text box
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@When("^User type \"([^\"]*)\" message$")
	public void user_type_message(String arg1) throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		ContactUsForm.ContactUsSubject().sendKeys(arg1); //Typing message
subject into the text box
		//WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait
is a soft wait to fix synchronizing issue.
		Thread.sleep(4000); // Wait to see the result
	}

	@Then("^User click on send button$")
	public void user_click_on_send_button() throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		ContactUsForm.ContactUsSend().click(); //Click on the Send button
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
```

```
	}

	@Then("^User see the notification$")
	public void user_see_the_notification() throws Throwable {
		Thread.sleep(5000); // Wait to see the result
		driver.close(); // Close the driver
		driver.quit(); // Quit Browser
	}

	@Given("^User should open AdvantageDemo website$")
	public void user_should_open_AdvantageDemo_website() throws Throwable {
		//Using Try/Catch block in case there is any run time error occurs.
		try {
			System.setProperty("webdriver.chrome.driver","C:\\Program
Files\\Chrome Driver\\chromedriver.exe"); //To open GoogleChrome browser
			driver = new ChromeDriver(); //Getting all ChromeDriver methods
inside driver.
			driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
// Implicit wait is a soft wait to fix synchronizing issue.
			driver.manage().window().maximize(); //To maximize the window
			driver.get("https://www.advantageonlineshopping.com"); //To open
URL of AdvantageDemo Website
		} catch (Exception e) {
			System.out.println("Check your Browser's Driver Update");
			System.out.println("Cause is: " + e.getCause()); //To get cause
of the exception.
			System.out.println("Message is: " + e.getMessage()); //To get
exception message.
			e.printStackTrace(); //It helps to trace the exception as I said
at the beginning.
		}
	}

	@When("^User click contact button$")
	public void user_click_contact_button() throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		ContactUsForm.ContactUs().click(); //Click on the contact us button
		Thread.sleep(5000); // Wait to see the result
	}

	@When("^User should type \"([^\"]*)\" address$")
	public void user_should_type_address(String arg1) throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver);
//Object Created to get Web Elements from Page Object Model
		ContactUsForm.ContactUsEmail().sendKeys(arg1); //Typing email address
into the text box
```

```
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is a soft wait to fix synchronizing issue.
	}

	@When("^User should type \"([^\"]*)\" message$")
	public void user_should_type_message(String arg1) throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver); //Object Created to get Web Elements from Page Object Model
		ContactUsForm.ContactUsSubject().sendKeys(arg1); //Typing message subject into the text box
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is a soft wait to fix synchronizing issue.
	}

	@Then("^User should click the send button$")
	public void user_should_click_the_send_button() throws Throwable {
		AdvantageDemoPOM ContactUsForm = new AdvantageDemoPOM(driver); //Object Created to get Web Elements from Page Object Model
		ContactUsForm.ContactUsSend().click(); //Click on the Send button
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is a soft wait to fix synchronizing issue.
	}

	@Then("^User will see notification$")
	public void user_will_see_notification() throws Throwable {
		Thread.sleep(7000); // Wait to see the result
		driver.close(); // Close the driver
		driver.quit(); // Quit Browser
	}

}
```

After validating two stories now I am going to validate my project's 3rd story which is "Validation of a Product Category". To do so, I have created a Feature file to write my test case by using Gherkin Language. Let's see what I have written in that Feature file.

Third Story. Third Feature File with single test case.

Feature File: AdvantageDemoHeadphone.feature

```
#Author: Abu Saeed M Sayem AKA Kaspar Moon
#Author Email: kasparmoonva@gmail.com
#Author Website: www.kasparmoon.us
#Story: Validate Home Page Headphone Category
#Story number - TW03
#Product Website: www.advantageonlineshopping.com
#Feature: Headphone Category Validation
```

```
#Scenario: Positive Test
#Drafted on October 1, 2021
@headphone
Feature: Headphone Category Validation
  User should visit AdvantageDemo Website then mouse hover on homepage Headphone
Category picture.
  A Shop Now button will appare then user needs to click on Shop Now button and
user will redirect
  to the headphone category page.

  @getheadphonecategory
  Scenario: Headphone Category Validation
    Given User should visit AdvantageDemo website
    And Mouse hover on category picture
    When Click on Shop Now button
    Then User redirect to headphone category page
```

This is my only one test case without any kind of data and this is I think just one feature file where I have just written one test case and one scenario. Based on this test case I have created "AdvantageDemoHeadphone" java file for my test script under the package "StepDefs". Here is what I have written in this file.

```
package StepDefs;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.WebDriverWait;

import PageObjectModel.AdvantageDemoPOM;

import cucumber.api.java.en.And;
import cucumber.api.java.en.Given;
import cucumber.api.java.en.Then;
import cucumber.api.java.en.When;

public class AdvantageDemoHeadphone {

     WebDriver driver; //Declaring Selenium WebDriver Global veritable.

     @Given("^User should visit AdvantageDemo website$")
     public void user_should_visit_AdvantageDemo_website() throws Throwable {
          //Using Try/Catch block in case there is any run time error occurs.
          try {
               System.setProperty("webdriver.chrome.driver","C:\\Program
Files\\Chrome Driver\\chromedriver.exe"); //To open GoogleChrome browser
```

```
                driver = new ChromeDriver(); //Getting all ChromeDriver methods
inside driver.
                driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
// Implicit wait is a soft wait to fix synchronizing issue.
                driver.manage().window().maximize(); //To maximize the window
                driver.get("https://www.advantageonlineshopping.com"); //To open
URL of AdvantageDemo Website
           } catch (Exception e) {
                System.out.println("Check your Browser's Driver Update");
                System.out.println("Cause is: " + e.getCause()); //To get cause
of the exception.
                System.out.println("Message is: " + e.getMessage()); //To get
exception message.
                e.printStackTrace(); //It helps to trace the exception as I said
at the beginning.
           }
     }

     @And("^Mouse hover on category picture$")
     public void mouse_hover_on_category_picture() throws Throwable {
           Actions HeadCategory = new Actions (driver); // Used Actions class and
created an object so that I can work with mouse hover action
           AdvantageDemoPOM CategoryValid = new AdvantageDemoPOM (driver);
//Object Created to get Web Elements from Page Object Model

     HeadCategory.moveToElement(CategoryValid.HoverHeadphone()).build().perform()
; //Performing mouse hover action
           WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
     }

     @When("^Click on Shop Now button$")
     public void click_on_Shop_Now_button() throws Throwable {
           AdvantageDemoPOM CategoryValid = new AdvantageDemoPOM (driver);
//Object Created to get Web Elements from Page Object Model
           CategoryValid.HeadphoneShopNow().click(); //Click on the Shop Now
Button
     }

     @Then("^User redirect to headphone category page$")
     public void user_redirect_to_headphone_category_page() throws Throwable {
           Thread.sleep(7000); // Wait to see the result
           driver.close(); // Close the driver
           driver.quit(); // Quit Browser
     }
}
```

With this test script I have checked that whenever anyone visit AdvantageDemo website and click on the ‘Contact Us’ menu then it will show contact form and automatically fill up the form and submit. User can submit that form based on the product and category and even can just submit normal message without product and category. Then I have created my 4th feature file for my 4th story. And that was, “Shopping Cart Validation”. I have named that feature file “AdvantageDemoShoppingCart” and wrote following test case with Gherkin Language.

Fourth Story. Fourth Feature File with single test case.

Feature File: AdvantageDemoShoppingCart.feature

```
#Author: Abu Saeed M Sayem AKA Kaspar Moon
#Author Email: kasparmoonva@gmail.com
#Author Website: www.kasparmoon.us
#Story: Validate Shopping Cart
#Story number - TW04
#Product Website: www.advantageonlineshopping.com
#Feature: Shopping Cart Validation
#Scenario: Positive Test
#Drafted on October 15, 2021
Feature: Validate shopping cart
  User should visit AdvantageDemo website and search a product
  into the search box and select a product then add that product
  to the shopping cart. User will visit the shopping cart and
  can see that product into the cart.

  @shoppingcart
  Scenario Outline: Shopping cart validation
    Given User will visits AdvantageDemo website
    When User search a "<product>" and select a product with "<quantity>" and add
to cart
    Then User visit shopping cart
    And User should find a recently added product into the cart

    Examples:
      | product    | quantity |
      | Chromebook |        2 |
```

Based on this test case I have written following test script in the java file “AdvantageDemoShoppingCart” under “StepDefs” package. This is very simple test case where I have made the automation system to visit AdvantageDemo website and search a product then add that product into the shopping cart. Let’s see what’s inside the jave file.

```
package StepDefs;

import java.util.concurrent.TimeUnit;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.WebDriverWait;

import PageObjectModel.AdvantageDemoPOM;

import cucumber.api.java.en.And;
import cucumber.api.java.en.Given;
import cucumber.api.java.en.Then;
import cucumber.api.java.en.When;

public class AdvantageDemoShoppingCart {

      WebDriver driver; //Declaring Selenium WebDriver Global veritable.

      @Given("^User will visits AdvantageDemo website$")
      public void user_will_visits_AdvantageDemo_website() throws Throwable {
            //Using Try/Catch block in case there is any run time error occurs.
                      try {

      System.setProperty("webdriver.chrome.driver","C:\\Program Files\\Chrome 
Driver\\chromedriver.exe"); //To open GoogleChrome browser
                            driver = new ChromeDriver(); //Getting all 
ChromeDriver methods inside driver.
                            driver.manage().timeouts().implicitlyWait(30, 
TimeUnit.SECONDS); // Implicit wait is a soft wait to fix synchronizing issue.
                            driver.manage().window().maximize(); //To maximize the 
window
                            driver.get("https://www.advantageonlineshopping.com"); 
//To open URL of AdvantageDemo Website
                      } catch (Exception e) {
                            System.out.println("Check your Browser's Driver 
Update");
                            System.out.println("Cause is: " + e.getCause()); //To 
get cause of the exception.
                            System.out.println("Message is: " + e.getMessage()); 
//To get exception message.
                            e.printStackTrace(); //It helps to trace the exception 
as I said at the beginning.
                      }
      }

      @When("^User search a \"([^\"]*)\" and select a product with \"([^\"]*)\" 
and add to cart$")
      public void user_search_a_and_select_a_product_with_and_add_to_cart(String 
arg1, String arg2) throws Throwable {
		AdvantageDemoPOM Chromebook = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
		Chromebook.ClickSearch().click(); //Click on the search icon
		Thread.sleep(7000); // Wait to see the result
		Chromebook.Search().sendKeys(arg1); // Type into the search box
		WebDriverWait wait = new WebDriverWait(driver,30); // Explicit wait is 
a soft wait to fix synchronizing issue.
		Chromebook.SearchClick().click(); //Click on the search icon
		Thread.sleep(4000); // Wait to see the result
		Chromebook.CloseSearch().click(); // Click to close the search box
		Thread.sleep(4000); // Wait to see the result
		Chromebook.ClickChromebook().click(); // Click to open specific 
product page
		Thread.sleep(4000); // Wait to see the result
		Chromebook.ColorBlack().click(); //To select product color
		Chromebook.Quantity().sendKeys(arg2); //To type product quantity
		Chromebook.AddToCart().click(); // To click on the "Add to Cart" 
button
		WebDriverWait waitt = new WebDriverWait(driver,30); // Explicit wait 
is a soft wait to fix synchronizing issue.
	}

	@Then("^User visit shopping cart$")
	public void user_visit_shopping_cart() throws Throwable {
		AdvantageDemoPOM Chromebook = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
		Chromebook.ViewCart().click(); //To visit shopping cart
		Thread.sleep(5000); // Wait to see the result
		Chromebook.RemoveCart().click(); //To remove the product from the cart
		Chromebook.ContinueShopping().click(); //To click on the "Continue 
Shopping" button
		WebDriverWait wait = new WebDriverWait(driver,30); // Explicit wait is 
a soft wait to fix synchronizing issue.
	}

	@And("^User should find a recently added product into the cart$")
	public void user_should_find_a_recently_added_product_into_the_cart() throws 
Throwable {
		Thread.sleep(5000); // Wait to see the result
		driver.close(); // To close Internet connection with WebDriver
		driver.quit(); // To quit browser
	}

}
```

Right after this test now I am going to show you my 5th and final story to validate Advantage Demo website's product search ability. I have created "AdvantageDemoSearchAbility" feature file under the 'Features' folder, and written following test case.

Fifth Story. Fifth Feature File with multiple test cases written in Gherkin Language.
Feature File: AdvantageDemoSearchAbility.feature

```
#Author: Abu Saeed M Sayem AKA Kaspar Moon
#Author Email: kasparmoonva@gmail.com
#Author Website: www.kasparmoon.us
#Story: Validate Product Search Ability
#Story number - TW05
#Product Website: www.advantageonlineshopping.com
#Feature: Product Search Ability
#Scenario: Positive Test
#Drafted on October 1, 2021
@productsearch
Feature: Product Search Ability
  User should visit AdvantageDemo Website then click on the search icon
  type product name into the box and click on the search icon again then click to
close search box. Next step
  is to click on the search icon again, type another product name, click on view
all text, close search box.

  @searchmouse
  Scenario Outline: Search Mouse
    Given User visits AdvantageDemo website
    When User click on the search icon
    And Type "<mouse>" name
    And Click on search icon
    Then Click on search on close button
    And User will see search result

    Examples:
      | mouse          |
      | wireless mouse |

  @searchlaptop
  Scenario Outline: Search Mouse
    Given User goes to AdvantageDemo website
    When User click search icon
    And Type "<laptop name>" in the search box
    And Click on the view all button
    Then Click to close search box
    And User will see laptop search result
    Examples:
      | laptop name |
      | laptop      |
```

Based on this test case I have created “AdvantageDemoSearchAbility” jave file under the “StepDefs” package to write my test script. Let’s see what I have written there:

```
package StepDefs;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.WebDriverWait;

import PageObjectModel.AdvantageDemoPOM;

import cucumber.api.java.en.And;
import cucumber.api.java.en.Given;
import cucumber.api.java.en.Then;
import cucumber.api.java.en.When;

public class AdvantageDemoSearchAbility {

      WebDriver driver; //Declaring Selenium WebDriver Global veritable.

      @Given("^User visits AdvantageDemo website$")
      public void user_visits_AdvantageDemo_website() throws Throwable {
            //Using Try/Catch block in case there is any run time error occurs.
            try {
                  System.setProperty("webdriver.chrome.driver","C:\\Program 
Files\\Chrome Driver\\chromedriver.exe"); //To open GoogleChrome browser
                  driver = new ChromeDriver(); //Getting all ChromeDriver methods 
inside driver.
                  driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS); 
// Implicit wait is a soft wait to fix synchronizing issue.
                  driver.manage().window().maximize(); //To maximize the window
                  driver.get("https://www.advantageonlineshopping.com"); //To open 
URL of AdvantageDemo Website
            } catch (Exception e) {
                  System.out.println("Check your Browser's Driver Update");
                  System.out.println("Cause is: " + e.getCause()); //To get cause 
of the exception.
                  System.out.println("Message is: " + e.getMessage()); //To get 
exception message.
                  e.printStackTrace(); //It helps to trace the exception as I said 
at the beginning.
```

```
		}
	}

	@When("^User click on the search icon$")
	public void user_click_on_the_search_icon() throws Throwable {
		AdvantageDemoPOM SearchMouse = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
		SearchMouse.ClickSearch().click(); //Click on the search icon
		//WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait 
is a soft wait to fix synchronizing issue.
		Thread.sleep(4000); // Wait to see the result
	}

	@And("^Type \"([^\"]*)\" name$")
	public void type_name(String arg1) throws Throwable {
		AdvantageDemoPOM SearchMouse = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
		SearchMouse.Search().sendKeys(arg1); // Type into the search box
		//WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait 
is a soft wait to fix synchronizing issue.
		Thread.sleep(4000); // Wait to see the result
	}

	@And("^Click on search icon$")
	public void click_on_search_icon() throws Throwable {
		AdvantageDemoPOM SearchMouse = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
		SearchMouse.SearchClick().click(); //Click on the search icon
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is 
a soft wait to fix synchronizing issue.
	}

	@Then("^Click on search on close button$")
	public void click_on_search_on_close_button() throws Throwable {
		AdvantageDemoPOM SearchMouse = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
		SearchMouse.CloseSearch().click(); // Click to close the search box
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is 
a soft wait to fix synchronizing issue.
	}

	@And("^User will see search result$")
	public void user_will_see_search_result() throws Throwable {
		Thread.sleep(7000); // Wait to see the result
		driver.close(); // Close the driver
		driver.quit(); // Quit Browser
	}
	@Given("^User goes to AdvantageDemo website$")
	public void user_goes_to_AdvantageDemo_website() throws Throwable {
		//Using Try/Catch block in case there is any run time error occurs.
		try {
			System.setProperty("webdriver.chrome.driver","C:\\Program 
Files\\Chrome Driver\\chromedriver.exe"); //To open GoogleChrome browser
			driver = new ChromeDriver(); //Getting all ChromeDriver methods 
inside driver.
			driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS); 
// Implicit wait is a soft wait to fix synchronizing issue.
			driver.manage().window().maximize(); //To maximize the window
			driver.get("https://www.advantageonlineshopping.com"); //To open 
URL of AdvantageDemo Website
		} catch (Exception e) {
			System.out.println("Check your Browser's Driver Update");
			System.out.println("Cause is: " + e.getCause()); //To get cause 
of the exception.
			System.out.println("Message is: " + e.getMessage()); //To get 
exception message.
			e.printStackTrace(); //It helps to trace the exception as I said 
at the beginning.
		}
	}

	@When("^User click search icon$")
	public void user_click_search_icon() throws Throwable {
		AdvantageDemoPOM SearchLaptop = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
		SearchLaptop.ClickSearch().click(); //Click on the search icon
		//WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait 
is a soft wait to fix synchronizing issue.
		Thread.sleep(4000); // Wait to see the result
	}

	@And("^Type \"([^\"]*)\" in the search box$")
	public void type_in_the_search_box(String arg1) throws Throwable {
		AdvantageDemoPOM SearchLaptop = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
		SearchLaptop.Search().sendKeys(arg1);
		//WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait 
is a soft wait to fix synchronizing issue.
		Thread.sleep(4000); // Wait to see the result
	}

	@And("^Click on the view all button$")
	public void click_on_the_view_all_button() throws Throwable {
		AdvantageDemoPOM SearchLaptop = new AdvantageDemoPOM(driver); //Object 
Created to get Web Elements from Page Object Model
```

```
		SearchLaptop.ClickViewAll().click();
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@Then("^Click to close search box$")
	public void click_to_close_search_box() throws Throwable {
		AdvantageDemoPOM SearchLaptop = new AdvantageDemoPOM(driver); //Object
Created to get Web Elements from Page Object Model
		SearchLaptop.CloseSearch().click(); // Click to close the search box
		WebDriverWait wait = new WebDriverWait(driver,20); // Explicit wait is
a soft wait to fix synchronizing issue.
	}

	@And("^User will see laptop search result$")
	public void user_will_see_laptop_search_result() throws Throwable {
		Thread.sleep(7000); // Wait to see the result
		driver.close(); // Close the driver
		driver.quit(); // Quit Browser
	}
}
```

I have finished my project work with this step. I have used JIRA software to track my work steps and uploaded to my complete project to my Git Repository.
In this project I have fulfilled following tasks that I have found in our project requirement document.

- In every test scripts I have written codes to open a browser and visit/open URL.
- In those test scripts where I have written test cases with data, I have used sendkeys() method to type data into the text area.
- To synchronize browser and website in all of my test script's I have used implicit wait function. To do so I have used manage() method from selenium. And that method contains implicitlyWait(Long arg, TimeUnit.arg) method under the method timeouts(). Not only that, I have used Thread.Sleep() method for hard wait sometimes. And used WebDriverWait(WebDriver, TimeUnit) method by creating object. That help the system to create a soft wait.
- To handle exception I have used Try/Catch block in every test scripts I have written in this package.
- I have used "Action" class in the story "Validate Home Page Headphone Category".
- I have used "Select" class in the stories "Validate Contact Us" and "Membership Validation".
- Test cases I have written for the stories "Validate Product Search Ability", "Validate Shopping Cart", and "Validate Contact Us" all those are with data test cases.
- Test case I have written for the story "Validate Home Page Headphone Category" that was without data test case.