Day 11 Test case Programs:

BaseTest:

```
package com.ibm.test;

import org.testng.annotations.Test;
import org.testng.AssertJUnit;
import java.io.File;
import java.io.IOException;
import java.sql.Driver;
import java.sql.SQLException;
import java.util.Date;
import java.util.HashMap;
import java.util.concurrent.TimeUnit;

import org.apache.commons.io.FileUtils;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.Assert;
import org.testng.AssertJUnit;
import org.testng.Reporter;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.BeforeSuite;
import org.testng.annotations.DataProvider;
import org.testng.annotations.Test;

import com.ibm.pages.AdminPage;
import com.ibm.pages.BannersPage;
import com.ibm.pages.LoginPage;
import com.ibm.pages.MarketingPage;
import com.ibm.pages.SystemPage;
import com.ibm.pages.UserPage;
import com.ibm.utilities.DBUtil;
import com.ibm.utilities.DbNew;
import com.ibm.utilities.ExcelUtil;
import com.ibm.utilities.PropertiesFileHandler;

public class BaseTest {

        WebDriver driver;
        WebDriverWait wait;
        PropertiesFileHandler propFileHandler;
        HashMap<String, String> data;

        /*ChromeOptions options = new ChromeOptions();
         options.addArguments("--disable-notifications");
  WebDriver driver1 = new ChromeDriver(options);*/


        @BeforeSuite
        public void preSetForTest() throws IOException {
                String file = "./TestData/data.properties";
                propFileHandler = new PropertiesFileHandler();
                data = propFileHandler.getPropertiesAsMap(file);
        }

        @BeforeMethod
        public void Initialization() {
                System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
                driver = new ChromeDriver();
                wait = new WebDriverWait(driver, 60);
                driver.manage().window().maximize();
                driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

        }



         @AfterMethod public void closeBrowser()
         { driver.quit(); }




         @Test(priority = 1, testName = "BannerListPageValidation")
         public void TestCase1() throws IOException, InterruptedException {

         String url = data.get("url"); String userName = data.get("user"); String
         passWord = data.get("password");

         driver.get(url);

         LoginPage login = new LoginPage(driver, wait); // Login to Admin Page
         login.enterEmailId(userName); login.enterPassWord(passWord);
         login.clickOnLogin();
```

```
        AdminPage admin = new AdminPage(driver, wait); admin.clickOnCatalog(); // ToSelect
Catalog

        BannersPage page = new BannersPage(driver, wait); page.clickOnBanners(); // To Select
Banners
        Thread.sleep(10000);

        int noOfEntries= page.noOfEntriesInBannerList();
        System.out.println("Number of Entries: " + noOfEntries);

        page.clickOnPage1(); // To Check Page2 Thread.sleep(10000);

        page.clickOnPage2(); // To Check Page1 Thread.sleep(10000);

        page.clickOnNext(); // To Check Next Link

        page.clickOnPrevious(); // To Check Previous link

        }


        @Test(priority = 2, testName = "EditCoupnsInMarketingTab")
        public void TestCase2() throws IOException, InterruptedException {

        try {

        String url = data.get("url");
        String userName = data.get("user");
        String passWord = data.get("password");
        String couponName= data.get("couponname");
        String couponCode= data.get("code");
String expectedmsg= data.get("successmsg");

driver.get(url);


LoginPage login = new LoginPage(driver, wait); // Login to Admin Page
login.enterEmailId(userName); login.enterPassWord(passWord);
login.clickOnLogin();



MarketingPage marketing = new MarketingPage(driver, wait); // To Select Marketing
marketing.clickOnMarketing(); marketing.clickOnCoupons(); // To Select Marketing
marketing.clickOnAction(); //To select Action button under Coupon List
marketing.clickOnEdit(); //To Select Edit under Action
Thread.sleep(10000); marketing.enterCouponName(couponName); //To Edit Coupon Name
marketing.enterCode(couponCode); marketing.clickOnSave(); //To save Coupon

Thread.sleep(10000);
String actualmsg= marketing.getTextToVerifySuccessMsg();
System.out.println(actualmsg);

Assert.assertTrue(actualmsg.contains(expectedmsg),"Assertion on Updated coupon code");

} catch (Exception e) { // TODO: handle exception
System.out.println(e.getMessage());
Reporter.log("Negataive Credentials Failed"); //screenshot
TakesScreenshot ts=(TakesScreenshot) driver;
File file= ts.getScreenshotAs(OutputType.FILE);
Date date =new Date(); String currentDate=date.toString().replace(":", "-");
FileUtils.copyFile(file, new
File("./screenshots/Error_"+currentDate+".png"));
```

```
Assert.fail(); }

}


@Test(priority = 3, testName = "AddNewMailInMarketingTab")
public void TestCase3() throws IOException, InterruptedException {

try {

String url = data.get("url"); String userName = data.get("user");
String passWord = data.get("password");
String customermail= data.get("customermailid");
String subject= data.get("Subject");
String expectedmsg1= data.get("Errormsg");


driver.get(url);


LoginPage login = new LoginPage(driver, wait); // Login to Admin Page
login.enterEmailId(userName); login.enterPassWord(passWord);
login.clickOnLogin();


MarketingPage marketing = new MarketingPage(driver, wait); // To Select Marketing
marketing.clickOnMarketing(); marketing.clickOnMail(); //To select Mail
marketing.clickOnAddMail(); //To Click on new mail +
marketing.fillToDetail(); //Fill To detail
marketing.fillCustomerMailId(customermail); //Fill Customermail id
```

```
 marketing.fillSubject(subject); //Fill Subject

 marketing.clickOnSaveMail();


 //To get the Error message text to verify it is on the same page


 String saveMailErrormsg= marketing.getTextToVerifyErrorMsg();

 System.out.println(saveMailErrormsg);


 Assert.assertTrue(saveMailErrormsg.contains(expectedmsg1),

 "Assertion on Saving a New Mail");


 } catch (Exception e) { // TODO: handle exception

 System.out.println(e.getMessage());

 Reporter.log("Negataive Credentials Failed"); //screenshot

 TakesScreenshot ts=(TakesScreenshot) driver;

 File file= ts.getScreenshotAs(OutputType.FILE);

 Date date =new Date(); String currentDate=date.toString().replace(":", "-");

 FileUtils.copyFile(file, new

 File("./screenshots/Error_"+currentDate+".png"));


 Assert.fail(); } }




@Test(priority = 4, testName = "AddNewReturnsReasonInSystemTab")

public void TestCase4() throws IOException, InterruptedException {

        try {


        String url = data.get("url");

        String userName = data.get("user");

        String passWord = data.get("password");

        String returnReason = data.get("returnreason");
```

```
		String expectedmsg2 = data.get("successRetutnReasonMsg");

		driver.get(url);

		LoginPage login = new LoginPage(driver, wait); // Login to Admin Page
		login.enterEmailId(userName);
		login.enterPassWord(passWord);
		login.clickOnLogin();

		SystemPage system = new SystemPage(driver, wait); // To Select System Tab
		system.clickOnSystem(); // To select System
		Thread.sleep(5000);
		system.clickOnReturns(); // To click on Returns
		system.clickOnReturnReasons(); // To click on Return Reasons
		system.clickOnAddNewReturnReason(); // To click on Add New Return reasons
		system.enterReturnReasons(returnReason); // To Enter Return Reason to Add
		system.clickOnReturnReasonSave(); // To Save Return Reason

		// To get the Success message after Adding New Return Reason

		String saveReturnReasonMsg = system.getTextToVerifyReturnreasonMsg();
		System.out.println(saveReturnReasonMsg);

		Assert.assertTrue(saveReturnReasonMsg.contains(expectedmsg2), "Assertion on Adding a New Return Reason");

		// To get Row count of Return Reasons

		int noOfReturnReason = system.noOfReturnReasonsList();
		System.out.println("Number of Return Reasons: " + noOfReturnReason);
```

```
        // To get the text for Return Reason name


        String searchentry = system.getTextForReturnReason();

        System.out.println("Return Reason Name: " + searchentry);


        } catch (Exception e) { // TODO: handle exception

                System.out.println(e.getMessage());

                Reporter.log("Negataive Credentials Failed"); //screenshot

                TakesScreenshot ts=(TakesScreenshot) driver;

                File file= ts.getScreenshotAs(OutputType.FILE);

                Date date =new Date(); String currentDate=date.toString().replace(":", "-");

                FileUtils.copyFile(file, new

                File("./screenshots/Error_"+currentDate+".png"));


                Assert.fail(); }


}


@Test(priority = 5, testName = "EditShippingLocationInSystemTab")

 public void TestCase5() throws IOException, InterruptedException {

        try {

        String url = data.get("url");

        String userName = data.get("user");

        String passWord = data.get("password");

        String negativeMessage = data.get("negativemsg");



        driver.get(url);


        LoginPage login = new LoginPage(driver, wait); // Login to Admin Page

        login.enterEmailId(userName);
```

```
            login.enterPassWord(passWord);
            login.clickOnLogin();


            SystemPage system = new SystemPage(driver, wait); // To Select System Tab
            system.clickOnSystem(); // To select System
            Thread.sleep(5000);


            system.clickOnShipping(); //To click on Shipping
            system.clickOnShippingLocation();//To click on Shipping Location
            system.clickOnShippingLocationAction(); //To click on Action button under Shipping
Locations
            system.clickOnShippingLocationEdit(); //TO Edit Shipping Locations
            system.clearCityNameText(); //To clear city name
            //system.clearSortOrderNumber(); //To clear SortOrderNumber
            system.clickOnShippingLocationsSave(); //To click on save


            String tootilpmessage= system.getValidationForNoCityName(); //To get Tooltip
Message
            System.out.println("tooltip message: " + tootilpmessage);


            Assert.assertTrue(tootilpmessage.contains(negativeMessage), "Assertion on
Negative Test case Validation upon Saving Shipping Locations");
            Reporter.log("Assertion on Negative Test case Validation upon Saving Shipping
Locations");


            } catch (Exception e) { // TODO: handle exception
                  System.out.println(e.getMessage());
                  Reporter.log("Negataive Credentials Failed"); //screenshot
                  TakesScreenshot ts=(TakesScreenshot) driver;
                  File file= ts.getScreenshotAs(OutputType.FILE);
                  Date date =new Date(); String currentDate=date.toString().replace(":", "-");
                  FileUtils.copyFile(file, new
                File("./screenshots/Error_"+currentDate+".png"));

                Assert.fail(); }


        }


@Test(priority = 6, testName = "ToDeleteTabInCatalogTabList")
public void TestCase6() throws IOException, InterruptedException {
        try {
        String url = data.get("url");
        String userName = data.get("user");
        String passWord = data.get("password");
        String expectedDeletemsg = data.get("deletedtabmsg");
        String deletedTabName= data.get("deletedTabName");


        driver.get(url);

        LoginPage login = new LoginPage(driver, wait); // Login to Admin Page
        login.enterEmailId(userName);
        login.enterPassWord(passWord);
        login.clickOnLogin();

        AdminPage admin = new AdminPage(driver, wait); // To Select System Tab
        admin.clickOnCatalog(); //To select Catalog
        admin.clickOnTabs(); //To select Tabs
        admin.clickOnTabAction(); //To click on Action button
        admin.clickOnTabDelete(); //To click on Delete
        Thread.sleep(90000);
```

```
            admin.clickOnDeleteIt(); //To delete a selected Tab


            String deletedmsg= admin.getTextToVerifyDeletedTabMsg();

            System.out.println(" message: " + deletedmsg);


             Assert.assertTrue(deletedmsg.contains(expectedDeletemsg),"Assertion on Deleted
Tab");

             Assert.assertTrue(deletedmsg.contains(expectedDeletemsg),"Assertion on Deleted
Tab");
```

```
             Reporter.log("Assertion on Deleting a Tab");




            admin.clickOnLogout(); //Logout from Admin Page


            String url1 = data.get("userpage");

            driver.get(url1);

            UserPage user= new UserPage(driver, wait);

             String deletedTab = user.getTabNotPresentTextMsg();


             Assert.assertFalse(deletedTab.contains(deletedTabName), "Assertion on Deleted
Tab Not present in Userpage");

             //System.out.println(" to verify Tab not present: " + deletedTab);

             Reporter.log("Deleted Tab not present in User Page");


            } catch (Exception e) { // TODO: handle exception

                   System.out.println(e.getMessage());

                   Reporter.log("Negataive Credentials Failed"); //screenshot

                   TakesScreenshot ts=(TakesScreenshot) driver;

                   File file= ts.getScreenshotAs(OutputType.FILE);

                   Date date =new Date(); String currentDate=date.toString().replace(":", "-");

                   FileUtils.copyFile(file, new

                   File("./screenshots/Error_"+currentDate+".png"));


                   Assert.fail();
```

```
                }



        }


@Test(priority = 7, testName = "ToEditUsersCredentials")
public void TestCase7() throws IOException, InterruptedException {
        try {
        String url = data.get("url");
        String userName = data.get("user");
        String passWord = data.get("password");
        String usersName = data.get("username");
        String userPassword= data.get("userpassword");
        String confirmUserPassword= data.get("confirmuserpassword");
        String userSuccesspmessage= data.get("usersuccesspmessage");
        String userMailId= data.get("usermailid");

        driver.get(url);

        LoginPage login = new LoginPage(driver, wait); // Login to Admin Page
        login.enterEmailId(userName);
        login.enterPassWord(passWord);
        login.clickOnLogin();

        SystemPage system = new SystemPage(driver, wait); // To Select System Tab
        Thread.sleep(10000);
        system.clickOnSystem();
        system.clickOnUsers(); //To click on Users
        system.clickOnUsersAction(); //To click on Users Action
        system.clickOnEditUsers(); //To click on Users Edit
        system.enteUsersName(usersName); //To click on Users Username
```

```
            system.enteUsersPassword(userPassword); //To click on Users Password

            system.enteUsersConfirmPassword(confirmUserPassword); //To click on Users
confirm password

            system.clickOnUsersSave();  //To click on Users Save


            //To validate the presence of the Edited Users


            String editedUsersMsg= system.getValidationToEditUsername();

            Assert.assertTrue(editedUsersMsg.contains(userSuccesspmessage), "Assertion on
Updating  a user ");

            Reporter.log("Assertion on Updating  a user");


            //To Logout from Admin page


            AdminPage admin = new AdminPage(driver, wait);

            admin.clickOnLogout();


            // Login to admin portal  using Edited user credential to verify the edited user have
the access to admin portal

            login.enterUserEmail(userMailId);

            login.enterUserPassword(userPassword);

            login.clickOnLogin();


            } catch (Exception e) { // TODO: handle exception

                System.out.println(e.getMessage());

                Reporter.log("Negataive Credentials Failed"); //screenshot

                TakesScreenshot ts=(TakesScreenshot) driver;

                File file= ts.getScreenshotAs(OutputType.FILE);

                Date date =new Date(); String currentDate=date.toString().replace(":", "-");

                FileUtils.copyFile(file, new

                File("./screenshots/Error_"+currentDate+".png"));
                Assert.fail();
                }



        }

@Test(priority = 8, testName = "ToAddProductsToCart")
public void TestCase8() throws IOException, InterruptedException {
        try {
        String url = data.get("userpage");
        String userFullName = data.get("fullname");
        String phoneNumber = data.get("phonenumber");
        String pwd = data.get("pwd");
        String cpwd= data.get("cpwd");
        String searchText= data.get("searchtext");
        String product1AvailInCart = data.get("product1Name");
        String product2AvailInCart = data.get("product2Name");

        driver.get(url);

        UserPage user = new UserPage(driver, wait); // Login to Admin Page
        user.clickOnSignUp(); //To select SignUp
        user.enterUserFullName(userFullName); //To enter Fullname
        user.enterUserPhoneNumber(phoneNumber); //To enter Phonenumber
        user.enterUserSignUpPassword(pwd); //To enter pwd
        user.enterUserConfirmPassword(cpwd);  //To enter confirm Pwd
        user.clickCheckBox(); //Click on I agree to terms and condition
        user.clickOncompSignUp(); //To complete SIgnUP

        Thread.sleep(50000);
```

```
            Alert alert= driver.switchTo().alert();
            alert.accept();

            Thread.sleep(50000);

            driver.findElement(By.xpath("(//*[@placeholder='Search for
products...'])[1]")).click();

            Thread.sleep(5000);

            driver.findElement(By.xpath("(//*[@placeholder='Search for
products...'])[2]")).sendKeys("rice");

            Thread.sleep(5000);

            driver.findElement(By.xpath("//*[text()='ORANGE IDLY FINE RICE
25kg']/ancestor::a")).click();
            user.cliickOnAddToCart();

            driver.findElement(By.xpath("(//*[@placeholder='Search for
products...'])[1]")).click();

            Thread.sleep(5000);
            driver.findElement(By.xpath("(//*[@placeholder='Search for
products...'])[2]")).sendKeys("rice");

            Thread.sleep(5000);
            user.clickOnProduct2();
            user.cliickOnAddToCart();

            user.clickOnCart();

  //To validate the presence of the Products added
```

```
        String product1Incart= user.getValidationAddedProduct1InCart();

        Assert.assertTrue(product1Incart.contains(product1AvailInCart), "Assertion on
Product1 available in cart ");

        Reporter.log("Assertion on Product1 available in cart");


        String product2Incart= user.getValidationAddedProduct2InCart();

        Assert.assertTrue(product2Incart.contains(product2AvailInCart), "Assertion on
Product2 available in cart ");

        Reporter.log("Assertion on Product2 available in cart");


        } catch (Exception e) { // TODO: handle exception

            System.out.println(e.getMessage());

            Reporter.log("Negataive Credentials Failed"); //screenshot

            TakesScreenshot ts=(TakesScreenshot) driver;

            File file= ts.getScreenshotAs(OutputType.FILE);

            Date date =new Date(); String currentDate=date.toString().replace(":", "-");

            FileUtils.copyFile(file, new

            File("./screenshots/Error_"+currentDate+".png"));


            Assert.fail();

            }



        }


    @Test(priority = 9, testName = "DbDeleteValidationInCatalogTab")

    public void TestCase9() throws IOException, InterruptedException, SQLException {

     try {


        String url = data.get("url");

        String userName = data.get("user");
```

```
                String passWord = data.get("password");
                //String expectedDeletemsg = data.get("deletedtabmsg");
                String pageSource = driver.getPageSource();
                String deltabName= data.get("deletedTabName");
                String noOfRecords= data.get("noofrecordsfound");


                driver.get(url);

                LoginPage login = new LoginPage(driver, wait); // Login to Admin Page
                login.enterEmailId(userName);
                login.enterPassWord(passWord);
                login.clickOnLogin();

                AdminPage admin = new AdminPage(driver, wait); // To Select System Tab
                admin.clickOnCatalog(); //To select Catalog
                admin.clickOnTabs(); //To select Tabs
                //Thread.sleep(90000);
                admin.clickOnTabAction(); //To click on Action button
                admin.clickOnTabDelete(); //To click on Delete
                Thread.sleep(90000); // without Thread, its not clicking DleteIt button
                admin.clickOnDeleteIt(); //To delete a selected Tab
                 Thread.sleep(90000);

                //To Verify the deleted tab under the tab list
                Assert.assertTrue(driver.getPageSource().contains("You have successfully deleted
data!"));
                Reporter.log("Assertion on Deleted Tab not present in Tab List");
                System.out.println("message: You have successfully deleted data!");
                admin.searchForTabName(deltabName);

                String expectedTabName= admin.getTextToVerifyDeletedTab();

                System.out.println(expectedTabName);

                Assert.assertEquals(noOfRecords, expectedTabName);

                Reporter.log("Assertion on Deleted Tab");

                //To validate Deleted Tab is not present on Database


            int actualTabName =  DBUtil.countQuery("SELECT count(*) FROM `as_tabs` where
name='Tab_To_delete' "); //use count query

            System.out.println(actualTabName);

            Assert.assertEquals(0,actualTabName);

            //Assert.assertTrue(actualTabName.contains(expectedTabName));

            //Assert.assertTrue(expectedTabName.getText().contains(actualTabName), "Data Tables
are Empty" + actualTabName);

            Reporter.log("Assertion on Deleted Tab not present in Database");

            System.out.println("Tab Name = Tab_To_delete is not available");


           } catch (Exception e) { // TODO: handle exception

                 System.out.println(e.getMessage());

                 Reporter.log("Negataive Credentials Failed"); //screenshot

                 TakesScreenshot ts=(TakesScreenshot) driver;

                 File file= ts.getScreenshotAs(OutputType.FILE);

                 Date date =new Date(); String currentDate=date.toString().replace(":", "-");

                 FileUtils.copyFile(file, new

                 File("./screenshots/Test_case9_Error_"+currentDate+".png"));


                 Assert.fail();

                 }




                }
```

```
   @Test(priority=10)
   public void Testcase10() throws IOException, InterruptedException, SQLException {
         try {


          String url = data.get("url");
                    String userName = data.get("user");
                    String passWord = data.get("password");
                    String customerName= data.get("customername");
                    String noOfRecords= data.get("noofrecordsfound");



                    driver.get(url);


                    LoginPage login = new LoginPage(driver, wait); // Login to Admin Page
                    login.enterEmailId(userName);
                    login.enterPassWord(passWord);
                    login.clickOnLogin();


                    AdminPage admin = new AdminPage(driver, wait); // To Select System Tab
          admin.clickOnCustomers();
          admin.customerNameSearch(customerName);
                    String expCustomername =admin.getTextToVerifyCustomerName();
           System.out.println(expCustomername);


                    //To validate Presence of a Customer Record on Database


             String actualCustomerName =  DBUtil.singleDataQuery("SELECT name FROM as_customer
WHERE name='shobha' ");
             System.out.println(actualCustomerName);
             Assert.assertEquals(actualCustomerName.trim(),expCustomername);
             Reporter.log("Assertion on customer record  present in Database");
          System.out.println("Customer Name = shobha is  available");


        // Navigate to admin panel and delete a customer record
         admin.clickOnCustomers();
         admin.customerNameSearch(customerName);
         admin.clickOnCustomersAction();
         admin.clickOnCustomerDelete();
        Thread.sleep(90000); // without Thread, its not clicking DleteIt button
        admin.clickOnDeleteIt();


        //To Verify the deleted customer under the Customer  list
                Assert.assertTrue(driver.getPageSource().contains("You have successfully deleted
data!"));
                Reporter.log("Assertion on Deleted Customer not present in Tab List");
                System.out.println("message: You have successfully deleted data!");




                admin.customerNameSearch(customerName);
                String expDeletedCustomer = admin.getTextToVerifyDeletedCustomer();
                System.out.println(expDeletedCustomer);
                Assert.assertEquals(noOfRecords, expDeletedCustomer);
                Reporter.log("Assertion on Deleted Customer");


                //To validate Deleted Customer is not present on Database


          int countOfActualCustomerName =  DBUtil.countQuery("SELECT count(*) FROM
as_customer WHERE name='shobha' "); //use count query
          System.out.println(countOfActualCustomerName);
          Assert.assertEquals(0,countOfActualCustomerName);
          Reporter.log("Assertion on Deleted Customer name not present in Database");
          System.out.println("Customer Name = shobha is not available");
```

```
            //To validate Customer Record count is Decreased by 1


            int countOFTotalCustomerRecord= DBUtil.countQuery("SELECT count(*) FROM
as_customer");

            System.out.println(countOFTotalCustomerRecord);

            Assert.assertEquals(31,countOFTotalCustomerRecord);

            Reporter.log("Assertion on Customer Record count in Database");

            System.out.println("Customer name= shobha not present in Database");




        } catch (Exception e) { // TODO: handle exception

                        System.out.println(e.getMessage());

                        Reporter.log("Negataive Credentials Failed"); //screenshot

                        TakesScreenshot ts=(TakesScreenshot) driver;

                        File file= ts.getScreenshotAs(OutputType.FILE);

                        Date date =new Date(); String currentDate=date.toString().replace(":", "-");

                        FileUtils.copyFile(file, new

                        File("./screenshots/Day_10_Test_case_Error_"+currentDate+".jpg"));

                        System.out.println(e.getMessage());

                        Assert.fail();

                        }


    }


    @Test(priority=11)

    public void Testcase11() throws IOException, InterruptedException, SQLException {

        try {


         String url = data.get("userpage");

                String userFullName = data.get("fullname");
                String phoneNumber = data.get("phonenumber");
                String pwd = data.get("pwd");
                String cpwd= data.get("cpwd");
                String customerPhoneNumber= data.get("customerphonenumber");
                String customerPassword= data.get("customerpassword");
                String invalidCustomerLoginExp= data.get("invalidloginerrmsg");
                String customerName= data.get("customername");

                driver.get(url);
                //Validation of homepage with Title and URL
                String URL=driver.getCurrentUrl();
                System.out.println(URL);
                Assert.assertEquals("https://atozgroceries.com/", URL);
                Reporter.log("Assertion on Home page URL");

                String title = driver.getTitle();
                System.out.println(title);
                Assert.assertEquals("AtoZ Grocery | Wholesaler in Bangalore", title);
                Reporter.log("Assertion on Home page Title");

                UserPage user = new UserPage(driver, wait);

                //Validate the invalid login credentails error message
                user.clickOnLogin(); //click on Login
                user.enterCustomerPhoneNmunber(customerPhoneNumber);
                user.enterCustomerLoginPwd(customerPassword);
                user.clickOnCustomerLogin();
                //To Verify the login credentails error message
                Thread.sleep(50000);
                String invalidCustomerLoginActual= user.getTextForCustomerIvalidLogin();
                System.out.println(invalidCustomerLoginActual);
```

```
			Assert.assertTrue(invalidCustomerLoginActual.contains(invalidCustomerLoginExp),
"Assertion on Invalid Customer Login error message ");
			Reporter.log("Assertion on Ivalid Phone Number or Password");
			System.out.println("message: Invalid Phone Number or Password");
			Thread.sleep(50000);
			user.clickOnCustomerLoginClose(); //To close the Login window

			//User Sign up

			user.clickOnSignUp(); //To select SignUp
			user.enterUserFullName(userFullName); //To enter Fullname
			user.enterUserPhoneNumber(phoneNumber); //To enter Phonenumber
			user.enterUserSignUpPassword(pwd); //To enter pwd
			user.enterUserConfirmPassword(cpwd);  //To enter confirm Pwd
			user.clickCheckBox(); //Click on I agree to terms and condition
			user.clickOncompSignUp(); //To complete SignUP

			Thread.sleep(50000);

			Alert alert= driver.switchTo().alert();
			alert.accept();

			//Defect1 (explained in Day 11 test case_Solution_SS document)
			//To Validate the presence of MyAccount and absence of Login Link.
			Thread.sleep(50000);

			Assert.assertTrue(driver.getPageSource().contains("My Account"));
			Reporter.log("My account Link which is present");
			System.out.println("My Account Link is Present");
```

```
            Thread.sleep(50000);

            //Assert.assertFalse(driver.getPageSource().contains("Login")); //Defect 2(Explained in Day 11 test case_Solution_SS)

            Assert.assertTrue(driver.getPageSource().contains("Login"));

            Reporter.log("Login Link Not Present");

            System.out.println("Login Link  is not Present");


            //Validate successful log out with the presence of Login / Sign Up links


            user.clickOnCustomerMyAccount();

            Thread.sleep(50000);

            user.clickOnCustomerLogout();

            Thread.sleep(50000);


            Assert.assertTrue(driver.getPageSource().contains("Login"));

            Reporter.log("Login Link is Present");

            System.out.println("Login Link  is  Present");



      Thread.sleep(50000);


            Assert.assertTrue(driver.getPageSource().contains("SignUp"));

            Reporter.log("Sign Up Link is Present");

            System.out.println("Sign Up Link  is  Present");


            //To validate Presence of a Customer Record on Database


       String actualCustomerName =  DBUtil.singleDataQuery("SELECT name FROM as_customer WHERE name='shobha' ");

       System.out.println(actualCustomerName);

       Assert.assertEquals("shobha",actualCustomerName.trim());

       Reporter.log("Assertion on customer record  present in Database");
```

```
              System.out.println("Customer Name = shobha is  available");


            } catch (Exception e) { // TODO: handle exception
                                    System.out.println(e.getMessage());
                                    Reporter.log("Negataive Credentials Failed"); //screenshot
                                    TakesScreenshot ts=(TakesScreenshot) driver;
                                    File file= ts.getScreenshotAs(OutputType.FILE);
                                    Date date =new Date(); String currentDate=date.toString().replace(":", "-");
                                    FileUtils.copyFile(file, new
                                    File("./screenshots/Day_10_Test_case_Error_"+currentDate+".jpg"));
                                    System.out.println(e.getMessage());
                                    Assert.fail();
                                    }


  }
}
```

UserPage

```
package com.ibm.pages;


import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;


public class UserPage {
	@FindBy(xpath="//*[@id='mm-0']/div[5]/div/div/div[2]/div/ul/li") // To find the Dleeted Tab
	//@FindBy(xpath="//h5[contains(text(),'Tab_To_delete')]")
	WebElement tabNotPresentEle;



	@FindBy(xpath="//a[@class='register']") //To Sign Up
	WebElement signUpEle;

	@FindBy(xpath="//input[@id='name']") //To enter Fullname
	WebElement fullNameEle;

	@FindBy(xpath="//input[@id='pnum']") //To enter phone number
	WebElement phoneNumberEle;

	@FindBy(xpath="//input[@id='password']") //To enter password
	WebElement passwordEle;

	@FindBy(xpath="//input[@id='cpassword']") //To enter password
	WebElement confirmPasswordEle;

	@FindBy(xpath="//input[@id='tccheckbox']") //To Agree to Terms and Conditions
	WebElement agreeTermsCheckBoxEle;

	@FindBy(xpath="//*[@id='mem_signup']") //To  complete Sign up
	WebElement completeSignUpEle;

	@FindBy(xpath="(//*[@placeholder='Search for products...'])[1]") //To  Search Products
	WebElement searchProductEle;

	@FindBy(xpath="(//*[@placeholder='Search for products...'])[2]")  //( do a SendKeys on this
in Method)-
```

```
	WebElement sendTextToSearch;


	@FindBy(xpath="//*[text()='ORANGE IDLY FINE RICE 25kg']/ancestor::a") //To  Search Products - search for rice and click on first product

	WebElement searchForProduct1Ele;


	@FindBy(xpath="//div[@class='button-detail']/a") //Add to  cart

	WebElement addTocart1Ele;




	@FindBy(xpath="//*[text()='HI-TECH RAW RICE 25kg']/ancestor::a") //To  Search Products - search for Dall and click on second product

	WebElement searchForProduct2Ele;


	@FindBy(xpath="//*[@id='bigCart']")

	WebElement cartEle;


	@FindBy(xpath="//*[@id='complete-cart']/div[1]/div[2]")

	WebElement product1InCartEle;


	@FindBy(xpath="//*[@id='complete-cart']/div[2]/div[2]")

	WebElement product2InCartEle;


	@FindBy(xpath="//a[text()='Login']")

	WebElement loginEle;


	@FindBy(xpath="//input[@id='pnum2']")

	WebElement customerLoginPhno;


	@FindBy(xpath="//input[@id='pword2']")

	WebElement customerLoginPwd;
```

```
@FindBy(xpath="//button[@id='mem_login']")
WebElement clickOnCustomerLogin;

@FindBy(xpath="//small[@id='pword_err2']")
WebElement invalidCustomerLogin;

@FindBy(xpath="(//a[@class='close'])[2]")
WebElement closeCustomerLogin;

@FindBy(xpath="/html/body/header/div[1]/div/div[2]/div[3]/li/a")
WebElement myAccount;

@FindBy(xpath="//a[contains(text(),'Log Out')]")
WebElement customerLogout;



WebDriverWait wait;
WebDriver driver;

public UserPage(WebDriver driver,WebDriverWait wait) {
        PageFactory.initElements(driver, this);
        this.driver=driver;
        this.wait=wait;
}

        public String getTabNotPresentTextMsg() {
        wait.until(ExpectedConditions.presenceOfElementLocated(By.xpath("//h5")));
        String deletedTab = tabNotPresentEle.getText();
        return deletedTab ;
        }
```

```
public void clickOnSignUp()
{
	signUpEle.click();
}

public void enterUserFullName(String fullname)
{
	fullNameEle.sendKeys(fullname);
}

public void enterUserPhoneNumber(String phonenumber)
{
	phoneNumberEle.sendKeys(phonenumber);
}

public void enterUserSignUpPassword(String pwd)
{
	passwordEle.sendKeys(pwd);
}

public void enterUserConfirmPassword(String cpwd)
{
	confirmPasswordEle.sendKeys(cpwd);
}

public void clickCheckBox()
{
	agreeTermsCheckBoxEle.click();
}

public void clickOncompSignUp()
```

```
		{
			completeSignUpEle.click();
		}

		public void clickOnSearchProducts()
		{
			searchProductEle.click();
		}
		public void enterSearchProducts(String searchtext )
		{
			sendTextToSearch.sendKeys(searchtext);
		}

		public void cliickOnAddToCart()
		{
			addTocart1Ele.click();
		}

		public void clickOnProduct2()
		{
			searchForProduct2Ele.click();
		}

		public void clickOnCart()
		{
			cartEle.click();
		}

	public String getValidationAddedProduct1InCart()

		{
```

```
                        String product1Incart= product1InCartEle.getText();
                        return product1Incart;
                }

public String getValidationAddedProduct2InCart()

                {
                        String product2Incart= product2InCartEle.getText();
                        return product2Incart;
                }

public void clickOnLogin()
{
        loginEle.click();
}

public void enterCustomerPhoneNmunber(String customerphonenumber)
{
        customerLoginPhno.sendKeys(customerphonenumber);
}

public void enterCustomerLoginPwd(String customerpassword)
{
        customerLoginPwd.sendKeys(customerpassword);
}



public void clickOnCustomerLogin()
{
        clickOnCustomerLogin.click();
}
```

```
public String getTextForCustomerIvalidLogin()


{
        String invalidcuslogin= invalidCustomerLogin.getText();
        return invalidcuslogin;
}

public void clickOnCustomerLoginClose()
{
        closeCustomerLogin.click();
}

public void clickOnCustomerMyAccount()
{
        myAccount.click();
}

public void clickOnCustomerLogout()
{
        customerLogout.click();
}
}
```

Data properties

```
userpage= https://atozgroceries.com/
fullname=shobha
phonenumber=9998887770
pwd=welcometoatoz
```

```
cpwd=welcometoatoz

customerphonenumber=9998887770

customerpassword=welcome1234
```