

Day 13 - Test case solution document

Code Used:

BaseTest.java

```
package com.ibm.test;

import java.io.File;
import java.io.IOException;
import java.sql.SQLException;
import java.util.Date;
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.testng.Assert;
import org.testng.Reporter;
import org.testng.annotations.DataProvider;
import org.testng.annotations.Optional;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;

import com.ibm.pages.AdminPage;
import com.ibm.pages.CategoryPage;
import com.ibm.pages.CouponPage;
import com.ibm.pages.DashboardPage;
import com.ibm.pages.DriverLaunch;
import com.ibm.pages.LoginPage;
import com.ibm.pages.ProductsPage;
import com.ibm.pages>ReturnsPage;
import com.ibm.pages.SettingsPage;
import com.ibm.pages.ShippingPage;
import com.ibm.pages.TabsPage;
import com.ibm.pages.UserPage;
import com.ibm.utilities.DBConnection;
import com.ibm.utilities.ExcelReader;

public class BaseTest extends DriverLaunch
{
    @Test(testName="EditProductQuantity",dataProvider="EditProductquantity")
    public void Day13TestCase(String old_name,String ProNewName,String
    ProMetaTitle,String ProModel,String ProGST,String ProPrice,String ProSpDisc,String
    ProQuantity,String ProTotalQty,String ProSort,String ProStatus,String ProTabs,String
    ProCategories,String ProImgPath,String ValidMsg,String SuccessMsg) throws
    InterruptedException, SQLException, IOException
    {
        LoginPage login=new LoginPage(driver,wait);
        DashboardPage dash = new DashboardPage(driver,wait);
```

```

ProductsPage prod=new ProductsPage(driver,wait);
UserPage user=new UserPage(driver,wait);
DBConnection dbconn=new DBConnection();
try
{
    String adminurl=hashData.get("adminurl");
    String username=hashData.get("username");
    String password=hashData.get("password");
    String userurl=hashData.get("userurl");

    //Navigating to Admin page of atozgroceries.com
    driver.get(adminurl);
    //logging into the admin page
    login.EnterEmail(username);
    login.EnterPassword(password);
    login.clickLogin();
    //Opening Catalog
    dash.OpenCatalog();
    //Opening Product
    prod.OpenProduct();
    //Editing Product and validating the Error message
    prod.EditProductquantity(old_name, ProNewName, ProMetaTitle,
ProModel, ProGST, ProPrice, ProSpDisc, ProQuantity,ProTotalQty, ProSort, ProStatus,
ProTabs, ProCategories, ProImgPath, ValidMsg);
    //Validating the scuccess message after editing the product
    prod.validateSuccessMsg(SuccessMsg);
    //validating the product updates in Admin portal
    prod.ValidateProTable(ProNewName,ProTotalQty);
    //logging out from admin portal
    login.ClickLogout();
    //navigating to User portal
    driver.get(userurl);
    //Adding the product to cart and validating the quantity in cart
    user.cartProQty(ProNewName, ProQuantity);
    //validating the modified product and its quantity in DB
    dbconn.DBProductVerify(ProNewName,ProTotalQty);
}
catch(Exception e)// catching if any exception occurs and takes a
screenshot of the page
{
    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/Day13_TestCase_Error_" + currentDate + ".jpg"));
    System.out.println(e.getMessage());
    Assert.fail();
}
finally
{
    driver.quit();
}
}

```

```

@DataProvider(name="EditProductquantity")
public Object[][] ProQuantity() throws IOException
{
    return ExcelReader.DataTable("./TestData/TestData.xlsx",
"EditProQuantity");
}
}

```

LoginPage.java

```

package com.ibm.pages;

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 LoginPage {

    @FindBy(name="email")
    WebElement emailEle;

    @FindBy(name="pword")
    WebElement passwordEle;

    @FindBy(xpath="/html/body/div/div/div[2]/form/button")
    WebElement loginEle;

    @FindBy(xpath="//a[@title='Logout']")
    WebElement logoutEle;

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

    @FindBy(xpath="//input[@id='tccheckbox']")
    WebElement chkboxEle;

    @FindBy(id="name")
    WebElement fullnameEle;

    @FindBy(id="pnum")
    WebElement phoneEle;

    @FindBy(id="password")
    WebElement userPwdEle;

    @FindBy(id="cpassword")
    WebElement confirmPwdEle;

    @FindBy(xpath="//button[@id='mem_signup']")
    WebElement signButEle;

    WebDriverWait wait;
}

```

```

    WebDriver driver;

    public LoginPage(WebDriver driver, WebDriverWait wait)
    {
        PageFactory.initElements(driver, this);
    }
    public void EnterEmail(String username)
    {
        emailEle.sendKeys(username);
    }
    public void EnterPassword(String password)
    {
        passwordEle.sendKeys(password);
    }
    public void clickLogin()
    {
        loginEle.click();
    }
    public void ClickLogout()
    {
        logoutEle.click();
    }

    //Signing up into the User Page
    public void signUp(String name, String phno, String pwd, String conPwd)
throws InterruptedException
{
    signUpEle.click();
    fullnameEle.sendKeys(name);
    phoneEle.sendKeys(phno);
    userPwdEle.sendKeys(pwd);
    confirmPwdEle.sendKeys(conPwd);
    chkboxEle.click();
    signButEle.click();
    Thread.sleep(5000);
    wait.until(ExpectedConditions.alertIsPresent());
    driver.switchTo().alert().accept();
    Thread.sleep(5000);
}
}

```

DashboardPage.java

```

package com.ibm.pages;

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 DashboardPage {

```

```

//Finding the WebElements
@FindBy(partialLinkText="Catalog")
WebElement catalogEle;

@FindBy(xpath="/html/body/div[1]/div[1]/div/ul/li[5]/a")
WebElement marketEle;

@FindBy(xpath="//*[contains(text(),'System')]")
WebElement systemEle;

@FindBy(xpath="//*[contains(text(),'Shipping')]")
WebElement shipEle;

WebDriver driver;
WebDriverWait wait;

//catalog page constructor
public DashboardPage(WebDriver driver, WebDriverWait wait)
{
    this.driver=driver;
    this.wait=wait;
    PageFactory.initElements(driver, this);
}

//Opencatalog method to click on catalog
public void OpenCatalog()
{
    wait.until(ExpectedConditions.visibilityOf(catalogEle));
    catalogEle.click();
}

//method for opening marketing
public void OpenMarketing()
{
    marketEle.click();
}

//method for opening system
public void OpenSystem()
{
    systemEle.click();
}

//method for Opening Shipping
public void OpenShipping()
{
    shipEle.click();
}
}

```

ProductsPage.java

```

package com.ibm.pages;

import java.util.List;

import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;

```

```

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.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.Assert;
import org.testng.Reporter;

public class ProductsPage {

    @FindBy(xpath="//*[contains(text(),'Products')]")
    WebElement prodEle;

    @FindBy(xpath="//*[contains(text(),'Product List')]")
    WebElement prodListEle;

    @FindBy(name="dataTableExample2_length")
    WebElement showEle;

    @FindBy(id="pro_name")
    WebElement proNameEle;
    @FindBy(id="meta_title")
    WebElement metaTitleEle;
    @FindBy(id="toTop")
    WebElement toTopEle;
    //@FindBy(xpath="//a[contains(text()='Data')]")
    @FindBy(xpath="/html/body/div[1]/div[3]/div/form/div/div[2]/div/div/div
/ul/li[2]/a")
    WebElement dataEle;
    @FindBy(id="model")
    WebElement modelEle;
    @FindBy(id="gst")
    WebElement gstEle;
    @FindBy(id="price2")
    WebElement priceEle;
    @FindBy(id="special_dis")
    WebElement spDisEle;
    @FindBy(id="price_after_special_dis2")
    WebElement SpPriceEle;
    @FindBy(id="quantity")
    WebElement quantityEle;
    @FindBy(id="total_quantity")
    WebElement totalQtyEle;
    @FindBy(id="sort_order")
    WebElement sortEle;
    @FindBy(id="status")
    WebElement statusEle;
    //@FindBy(xpath="//a[contains(text()='Link')]")
    @FindBy(xpath="/html/body/div[1]/div[3]/div/form/div[2]/div/div[2]/div/div
/ul/li[3]/a")
    WebElement linkEle;
    @FindBy(id="tabs")
    WebElement tabsEle;
}

```

```

@FindBy(id="categories")
WebElement categoryEle;
//@FindBy(xpath="//a[contains(text()='Image')]")
@FindBy(xpath="/html/body/div[1]/div[3]/div/form/div/div/div[2]/div/div[2]/div/div[2]/ul/li[4]/a")
WebElement imgEle;
@FindBy(xpath="//input[@id='pro_image']")
WebElement proImgEle;

@FindBy(xpath="//button[@title='Save']")
WebElement saveEle;

WebDriver driver;
WebDriverWait wait;
JavascriptExecutor js;

public ProductsPage(WebDriver driver, WebDriverWait wait)
{
    this.driver=driver;
    this.wait=wait;
    PageFactory.initElements(driver, this);
    js=(JavascriptExecutor) driver;
}
public void OpenProduct()
{
    prodEle.click();
}

public void validateSuccessMsg(String success)
{

    //wait.until(ExpectedConditions.presenceOfElementLocated(By.xpath("//div[@class='alert alert-success alert-dismissible']")));
    String message=js.executeScript("return
document.getElementsByClassName('alert alert-success alert-
dismissible')[0].textContent").toString().trim();
    Assert.assertTrue(message.contains(success), "Assertion on Success
Message after editing the product!!!");
    Reporter.log("Validation of Success Message after editing the
product!!!");
}

public void EditProductquantity(String old_name,String new_name,String
meta_Title,String model,String gst,String price,String SplDiscount,String qty,String
totalQty,String sort,String Status,String tabs,String categories,String
imgPath,String validmsg) throws InterruptedException
{

    wait.until(ExpectedConditions.visibilityOf(prodListEle));
    int
rowLen=driver.findElements(By.xpath("//table[@id='dataTableExample2']/tbody/tr")).size();
    for(int i=1;i<=rowLen;i++)
{
}
}

```

```

        String
prod_name=driver.findElement(By.xpath("//table[@id='dataTableExample2']/tbody/tr["+i+
"]/td[2]")).getText().trim();
        if(prod_name.equals(old_name))
{
            //clicking on the Action button of the desired category to
edit

        driver.findElement(By.xpath("//table[@id='dataTableExample2']/tbody/tr["+i+"]/
td[9]/descendant::button")).click();
            //clicking on the Edit button of the desired category to
edit

        driver.findElement(By.xpath("//table[@id='dataTableExample2']/tbody/tr["+i+"]/
td[9]/descendant::ul/li)[1]").click();
            //Clears the existing data in product tab
        proNameEle.clear();
        saveEle.click();
        String ReqMsg=js.executeScript("return
document.getElementById('pro_name').validationMessage").toString().trim();
        Assert.assertTrue(ReqMsg.contains(validmsg),"Assertion on
Error Message!!");
        Reporter.log("Validation of Error Message - Product Name
as mandatory field!!");

        js.executeScript("arguments[0].scrollIntoView(true);",metaTitleEle);
        metaTitleEle.clear();
        toTopEle.click();
        //sending new data in Product Tab
        proNameEle.sendKeys(new_name);

        js.executeScript("arguments[0].scrollIntoView(true);",metaTitleEle);
        metaTitleEle.sendKeys(meta_Title);
        toTopEle.click();
        //opening data tab
        js.executeScript("arguments[0].click()",dataEle);
        //clearing the existing data in Data tab

        js.executeScript("arguments[0].scrollIntoView(true);",modelEle);
        wait.until(ExpectedConditions.visibilityOf(modelEle));
        //Clears the existing data in Data tab
        modelEle.clear();
        gstEle.clear();
        priceEle.clear();
        spDisEle.clear();

        js.executeScript("arguments[0].scrollIntoView(true);",statusEle);
        SpPriceEle.clear();
        quantityEle.clear();
        totalQtyEle.clear();
        sortEle.clear();
        driver.findElement(By.id("toTop")).click();
        //sending new data to Data Tab

        js.executeScript("arguments[0].scrollIntoView(true);",modelEle);

```

```

        quantityEle.sendKeys(qty);
        totalQtyEle.sendKeys(totalQty);
        modelEle.sendKeys(model);
        gstEle.sendKeys(gst);
        priceEle.sendKeys(price);
        spDisEle.sendKeys(SplDiscount);

    js.executeScript("arguments[0].scrollIntoView(true);",statusEle);
    sortEle.sendKeys(sort);
    Select statusSel=new Select(statusEle);
    statusSel.selectByVisibleText(Status);
    driver.findElement(By.id("toTop")).click();
    //opening Link Tab
    js.executeScript("arguments[0].click()",linkEle);
    Thread.sleep(5000);//due to my browser prb, added this
    Select TabsSel=new Select(tabsEle);
    TabsSel.selectByVisibleText(tabs);
    Select CatSel=new Select(categoryEle);
    CatSel.selectByVisibleText("");
    CatSel.selectByVisibleText(categories);
    //opening Image tab
    js.executeScript("arguments[0].click()",imgEle);
    //wait.until(ExpectedConditions.visibilityOf(proImgEle));
    Thread.sleep(5000);//due to my browser prb, added this
    proImgEle.sendKeys(imgPath);
    //clicking on Save to save the product
    saveEle.click();
    break;
}
}
}
public void ValidateProTable(String proName,String totalQty)
{
    //getting rows
    List<WebElement>
rowEle=driver.findElements(By.xpath("//table[@id='dataTableExample2']/tbody/tr"));
    //finding no of rows
    int rows=rowEle.size();
    for(int i=1;i<=rows;i++)
    {
        //getting product name element
        WebElement
proNameEle=driver.findElement(By.xpath("//table[@id='dataTableExample2']/tbody/tr["+i+"]/td[2]"));
        String
proTotalQty=driver.findElement(By.xpath("//table[@id='dataTableExample2']/tbody/tr["+i+"]/td[6]")).getText().trim();
        //getting the text of the category name element
        String nameTxt=proNameEle.getText();
        //if the text is same as the new product name, doing assertion
        if(nameTxt.equals(proName))
        {
            Assert.assertTrue(nameTxt.contains(proName), "Assertion of
edited product presence in Admin Page products Table!!");
}

```

```

        Reporter.log("Validation of edited product presence in
Admin Page products Table!!!");
            break;
        }
        //validating the product quantity in admin table
        if(proTotalQty.equals(totalQty))
        {
            Assert.assertTrue(proTotalQty.contains(totalQty),
"Assertion of edited product quantity in Admin Page products Table!!!");
            Reporter.log("Validation of edited product quantity in
Admin Page products Table!!!");
            break;
        }
    }
}

```

UserPage.java

```

package com.ibm.pages;

import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
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.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.Assert;
import org.testng.Reporter;

public class UserPage
{
    //Finding the elements of the User Page
    @FindBy(xpath="//*[@class='click-categories flaticon-bars'][2]")
    WebElement catBarEle;
    @FindBy(xpath="//*[@class='category-drop-list-inner'][2]")
    WebElement CateList;
    @FindBy(xpath="/html/body/div[1]/div[5]/div/div[1]/div/nav/ul/li[2]/span")
    WebElement HdrEle;
    @FindBy(xpath="//input[@id='search-box']")
    WebElement mainSrchEle;
    @FindBy(xpath="//a[contains(text(),'Continue to payment')]")
    WebElement continuePayEle;
    @FindBy(id="email")
    WebElement userEmailEle;
    @FindBy(id="address")
    WebElement userAddEle;
    @FindBy(id="city")
    WebElement userCityEle;
    @FindBy(id="pincode")
    WebElement userPinEle;
}

```

```

@FindBy(xpath="//span[contains(text(),'order has been placed'))]")
WebElement orderPlacedEle;
@FindBy(xpath="//a[@id='confirm-order-id']")
WebElement confirmOrderEle;

//@FindBy(xpath="//a[@onclick='myAccount()']")
@FindBy(xpath="/html/body/header/div[1]/div/div[2]/div[3]/li/a")
WebElement myAccEle;
@FindBy(xpath="(//a[contains(text(),'Log Out')))[1]")
WebElement userLogout;
WebDriver driver;
WebDriverWait wait;
JavascriptExecutor js;
//Constructor
public UserPage(WebDriver driver,WebDriverWait wait)
{
    this.driver=driver;
    this.wait=wait;
    js=(JavascriptExecutor) driver;
    PageFactory.initElements(driver, this);
}

public void cartProQty(String ProName, String proqty) throws
InterruptedException
{
    mainSrchEle.click();
    Thread.sleep(3000);
    //Adding Product to cart

    driver.findElement(By.xpath("//*[@id='searchModal']/div/div/div[1]/input"))
    .sendKeys(ProName);
    Thread.sleep(10000);

    driver.findElement(By.xpath("//span[text()='"+ProName+"']/ancestor::div[@id='s
earchproducts-div']")).click();
    driver.findElement(By.xpath("//a[contains(text(),'Add To
Cart')]")).click();

    driver.findElement(By.xpath("/html/body/header/div[2]/div/div[3]/div/div/a
")).click();
    String quantity=driver.findElement(By.xpath("//*[@class='quantity-hover-
cart']/descendant::span")).getText().trim();
    System.out.println(quantity);
    Assert.assertTrue(quantity.contains(proqty), "Assertion on Product
Quantity!!!");
    Reporter.log("validating the quantity of product "+ProName+" added in
cart is "+proqty+"!!!");
}
}

DBConnection.java
package com.ibm.utilities;

import java.sql.Connection;
import java.sql.DriverManager;

```

```

import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import org.testng.Assert;
import org.testng.Reporter;

public class DBConnection {
    //method to validate edited product in Database
    public void DBProductVerify(String ProName, String totalQty) throws
SQLException
    {
        //Establishing the DB connection
        Connection c =
DriverManager.getConnection("jdbc:mysql://foodsonfinger.com:3306/foodsonfinger_atozgr
oceries",
                "foodsonfinger_atoz", "welcome@123");
        Statement s = c.createStatement();
        //Executing the required statement for getting the new category..
sending the new category name in where condition
        ResultSet rs = s.executeQuery("SELECT * from as_products where
name=\""+ProName+"\"");
        //the below while loop checks each and every line of the above
query result
        while(rs.next())
        {
            //doing assertion on Product edited in DB

            Assert.assertTrue(ProName.equals(rs.getString("name")), "Assertion on Edited
Product in Database!!!");
            Reporter.log("Assertion on Edited Product in Database!!!");
            //validating the product quantity in DB

            Assert.assertTrue(totalQty.equals(rs.getString("total_qty")), "Assertion on
Edited Product quantity in Database!!!");
            Reporter.log("Assertion on Edited Product quantity in
Database!!!");
        }
    }
}

```

DriverLaunch.java

```

package com.ibm.pages;

import java.io.IOException;
import java.util.HashMap;
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.ie.InternetExplorerDriver;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.annotations.AfterMethod;

```

```

import org.testng.annotations.BeforeMethod;
import org.testng.annotations.BeforeSuite;
import org.testng.annotations.Optional;
import org.testng.annotations.Parameters;

import com.ibm.utilities.PropertiesFileHandler;

public class DriverLaunch {

    public WebDriver driver;
    public WebDriverWait wait;
    public PropertiesFileHandler propFileHandle;
    public HashMap<String, String> hashData;

    @BeforeSuite
    public void preSetForTest() throws IOException
    {
        String file=".TestData/data.properties";
        propFileHandle =new PropertiesFileHandler();
        hashData=propFileHandle.getPropertiesAsMap(file);
    }
    @BeforeMethod
    @Parameters({"browser"})
    public void initialisation(@Optional("ch")String browser)
    {
        browserInitialization(browser);
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
        wait=new WebDriverWait(driver, 60);
    }
    @AfterMethod
    public void closingBrowser()
    {
        driver.quit();
    }

    public void browserInitialization(String browser)
    {
        switch(browser.toLowerCase())
        {
            case "ff":
                System.setProperty("webdriver.gecko.driver",
"./drivers/geckodriver.exe");
                driver=new FirefoxDriver();
                break;
            case "ch":
                System.setProperty("webdriver.chrome.driver",
"./drivers/chromedriver.exe");
                driver=new ChromeDriver();
                break;
            case "ie":
                System.setProperty("webdriver.ie.driver",
"./drivers/IEDriverServer.exe");
                driver=new InternetExplorerDriver();
        }
    }
}

```

```

        break;
    }
}
}

PropertiesFileHandler.java
package com.ibm.utilities;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.HashMap;
import java.util.Properties;
import java.util.Set;

public class PropertiesFileHandler {

    public HashMap<String, String> getPropertiesAsMap(String file) throws
IOException {
        HashMap<String, String> magentoMap = new HashMap<String, String>();

        FileInputStream fileIn = new FileInputStream(file);
        Properties prop = new Properties();
        prop.load(fileIn);

        Set<Object> keysProp = prop.keySet();
        for (Object key : keysProp) {
            magentoMap.put(key.toString(), prop.getProperty(key.toString()));
        }
        prop.clear();
        return magentoMap;
    }
}

```

data.properties

```

adminurl=https://atozgroceries.com/admin
userurl=https://atozgroceries.com
username=demo@atozgroceries.com
password=456789
fullname=HarryTest3
userphno=9000090003
userpwd=welcome
confirmPWD=welcome
proname1=Bislery Water
proname2=Basmathi Rice
useremail=harisha.sunny@gmail.com
useradd=Bangalore
usercity=MADIWALA
userpin=560029

```

testng.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

```

```

<suite name="Suite">
  <test thread-count="5" name="Test">
    <parameter name="browser" value="ff"></parameter>
    <classes>
      <class name="com.ibm.test.BaseTest">
        <methods><include name="Day10TestCase"></include></methods>
      </class>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->

```

ExcelReader.java

```

package com.ibm.utilities;

import java.io.FileInputStream;
import java.io.IOException;

import org.apache.poi.ss.usermodel.DataFormatter;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class ExcelReader {
    public static Object[][] DataTable(String WBLoc, String sheetName) throws IOException
    {
        XSSFWorbook TestDataWB=new XSSFWorbook(new FileInputStream(WBLoc));
        XSSFSheet Sheet=TestDataWB.getSheet(sheetName);
        Object[][] SheetInfo=new Object[Sheet.getPhysicalNumberOfRows()-
1][Sheet.getRow(0).getPhysicalNumberOfCells()];
        int rowcount=Sheet.getPhysicalNumberOfRows();
        int cellcount=Sheet.getRow(0).getPhysicalNumberOfCells();
        for(int i=1;i<rowcount;i++)
        {
            for(int j=0;j<cellcount;j++)
            {
                DataFormatter format=new DataFormatter();
                String cellvalue=format.formatCellValue(Sheet.getRow(i).getCell(j));
                SheetInfo[i-1][j]=cellvalue;
            }
        }
        return SheetInfo;
    }
}

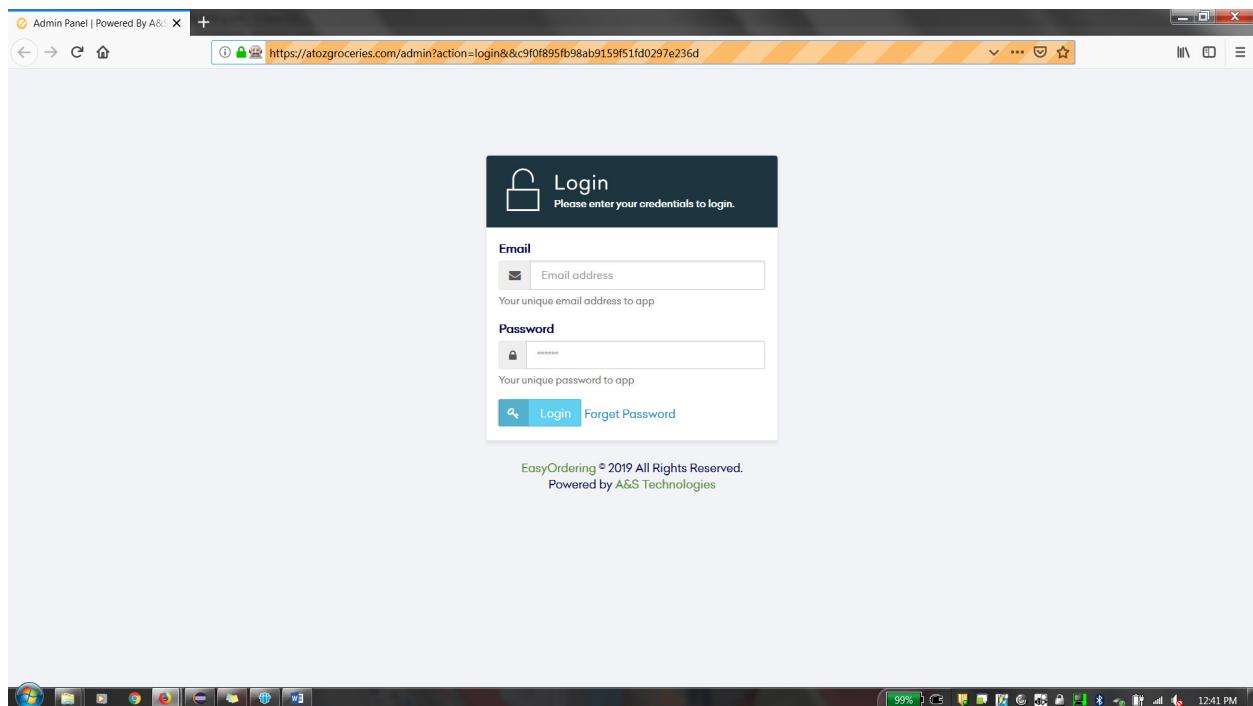
```

TestData.xlsx

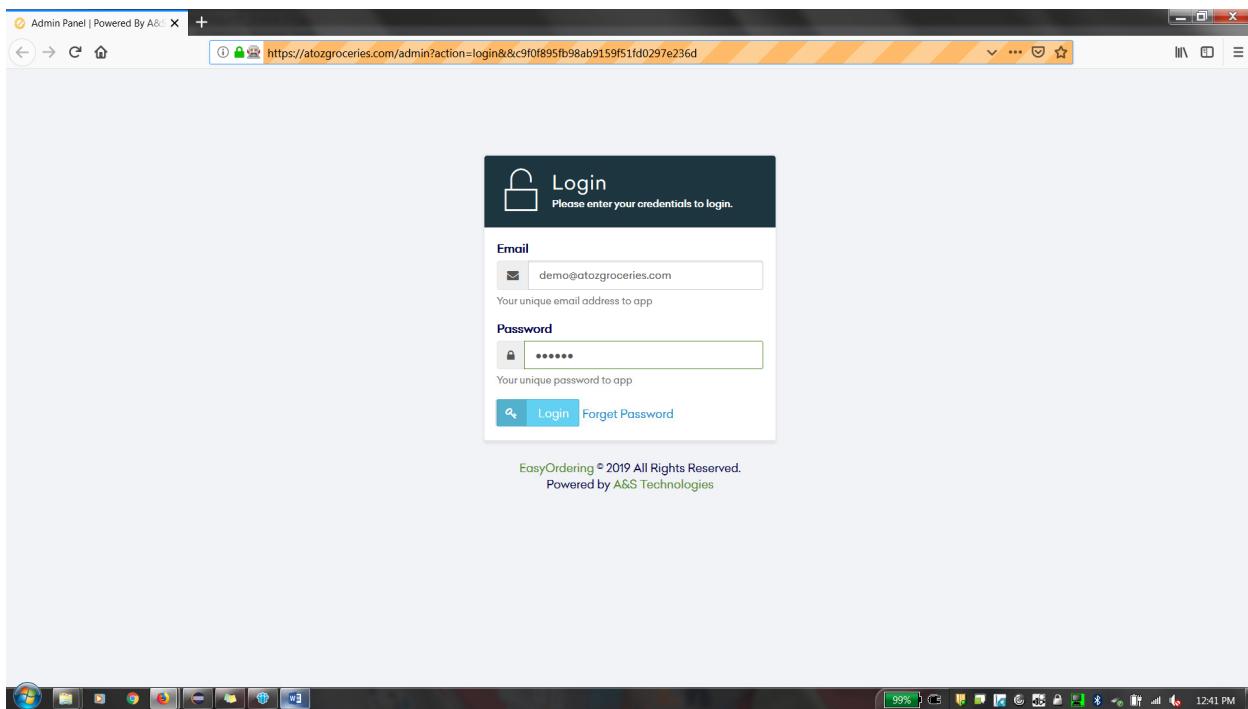
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	
1	old_name	ProNewName	ProMetaTitle	ProModel	ProGST	ProPrice	ProSpDisc	ProQuantity	ProTotalQty	ProSort	ProStatus	ProTabs	ProCategories	ProImgPath	ValidMsg	SuccessMsg
2	Banana	Tomato	Tomato	Tomato	4	45	0	5	5	1	Enabled	Test02	Vegetables	C:\Harisha\Selenium_Training\EclipseW\fill out this field	successfully updated product	
3																
4																
5																
6																
7																
8																
9																
10																
11																
12																
13																
14																
15																
16																
17																
18																
19																
20																
21																
22																
23																
24																
25																
26																

Step by step output:

Opening Admin portal



Providing login credentials for logging into admin portal



Admin portal after logging in

The screenshot shows the 'easyordering' Admin Panel Dashboard. The top bar of the browser indicates the URL as <https://atozgroceries.com/admin/dashboard?token=45c48cce2e2d7fbde1afc51c7c6ad26>. The main content area is titled 'Dashboard'. On the left, there is a sidebar with navigation links: 'Dashboard', 'Catalog', 'Sales', 'Customers', 'Marketing', 'System', and 'Reports'. The main dashboard area has four large cards: 'Total Orders' (5), 'Total Sales' (₹7,077.51), 'Total Customers' (9), and 'Total Products' (45). Below these cards is a section titled 'Latest Orders' with a table showing five recent orders. The table columns are: #, Order ID, Customer, Status, Total, Date Added, and Action. The orders listed are: #54 (Groceries, Pending, ₹3,975.00, 2019-01-04), #53 (Groceries, Pending, ₹3,975.00, 2019-01-04), #52 (Groceries, Pending, ₹3,405.00, 2019-01-04), #51 (Groceries, Pending, ₹3,405.00, 2019-01-03), and #50 (HarryTest3, Pending, ₹3,637.00, 2019-01-03). The bottom status bar shows battery level at 99% and the time as 12:41 PM.

#	Order ID	Customer	Status	Total	Date Added	Action
1	#54	Groceries	Pending	₹3,975.00	2019-01-04	
2	#53	Groceries	Pending	₹3,975.00	2019-01-04	
3	#52	Groceries	Pending	₹3,405.00	2019-01-04	
4	#51	Groceries	Pending	₹3,405.00	2019-01-03	
5	#50	HarryTest3	Pending	₹3,637.00	2019-01-03	

Opening Catalog to open Products

The screenshot shows the Admin Panel Dashboard with the following statistics:

- Total Orders: 30
- Total Sales: ₹35,745.00
- Total Customers: 51
- Total Products: 234

Below the dashboard, there is a section titled "Latest Orders" displaying the following table:

#	Order ID	Customer	Status	Total	Date Added	Action
1	#54	Groceries	Pending	₹3,975.00	2019-01-04	
2	#53	Groceries	Pending	₹3,975.00	2019-01-04	
3	#52	Groceries	Pending	₹3,405.00	2019-01-04	
4	#51	Groceries	Pending	₹3,405.00	2019-01-03	
5	#50	HarryTest3	Pending	₹3,637.00	2019-01-03	

Opening products

The screenshot shows the Admin Panel Products page with the following table:

#	Product Name	Image	Model	Price	Total Quantity	Sort Order	Status	Action
1	chocolates		NewArrival	238.00 260		1	Enabled	
2	hjwhjw		hhjghj	1.00 1	1	1	Enabled	
3	newP		excellent	522.5 550		1	Enabled	
4	Banana		Banana	82.4 82.4	0	1	Enabled	
5	mithaimate		Test	84.00 84	3	1	Enabled	
6	Badam-p01		Model-New	3.03 5.05	-119	0	Enabled	
7	Fresh Badham		Fresh	976.5 61	5	1	Enabled	
8	New Prod Req		1098	1,890.00 1		1	Disabled	

Selecting the product for modifying

The screenshot shows the Admin Panel of the easyordering system. The left sidebar has a 'Catalog' section with 'Products' selected. The main area is titled 'Products' and contains a 'Product List' table. The table has columns for #, Product Name, Image, Model, Price, Total Quantity, Sort Order, Status, and Action. There are 10 entries listed:

#	Product Name	Image	Model	Price	Total Quantity	Sort Order	Status	Action
1	chocolates		NewArrival	238.00 260		1	Enabled	Action
2	hjwhjw		hhjghj	1.00 1	1	1	Enabled	Action
3	newP		excellent	522.5 550		1	Enabled	Action
4	Banana		Banana	82.4 82.4	0	1	Enabled	Action
5	mithaimate		Test	84.00 84	3	1	Enabled	Edit Delete
6	Badam-p01		Model-New	3.03 5.05	-119	0	Enabled	Action
7	Fresh Badham		Fresh	976.5 51	5	1	Enabled	Action
8	New Prod Req			1098 1	1,890.00	1	Disabled	Action

Opening the product for modifying

The screenshot shows the Admin Panel with the 'Products' page open. The 'Products' tab is selected in the sidebar. The main area is titled 'Edit Product' and shows a form for editing a product. The 'Product' tab is active, showing the product name 'Banana' and a rich text editor for the description. The description text is 'good quality'.

Validating the error message by removing the product name

The screenshot shows the 'Edit Product' form in the easyordering Admin Panel. The 'Product Name' field is empty, and an error message 'Please fill out this field.' is displayed above it. Below the field is a rich text editor toolbar. The URL in the browser is https://atozgroceries.com/admin/products/edit/413?token=c9f0f895fb98ab9159f51fd0297e236d.

Clearing the existing data for the product

The screenshot shows the 'Edit Product' form in the easyordering Admin Panel. The 'Meta Title', 'Meta Tag Description', and 'Meta Tag Keywords' fields are empty. The URL in the browser is https://atozgroceries.com/admin/products/edit/413?token=c9f0f895fb98ab9159f51fd0297e236d.

Providing the new data for the product in product tab

The screenshot shows the 'Edit Product' interface. On the left, a sidebar menu is visible under the 'Catalog' section, with 'Products' selected. The main area displays a form with the following fields:

- Product Name:** Tomato
- Description:** good quality

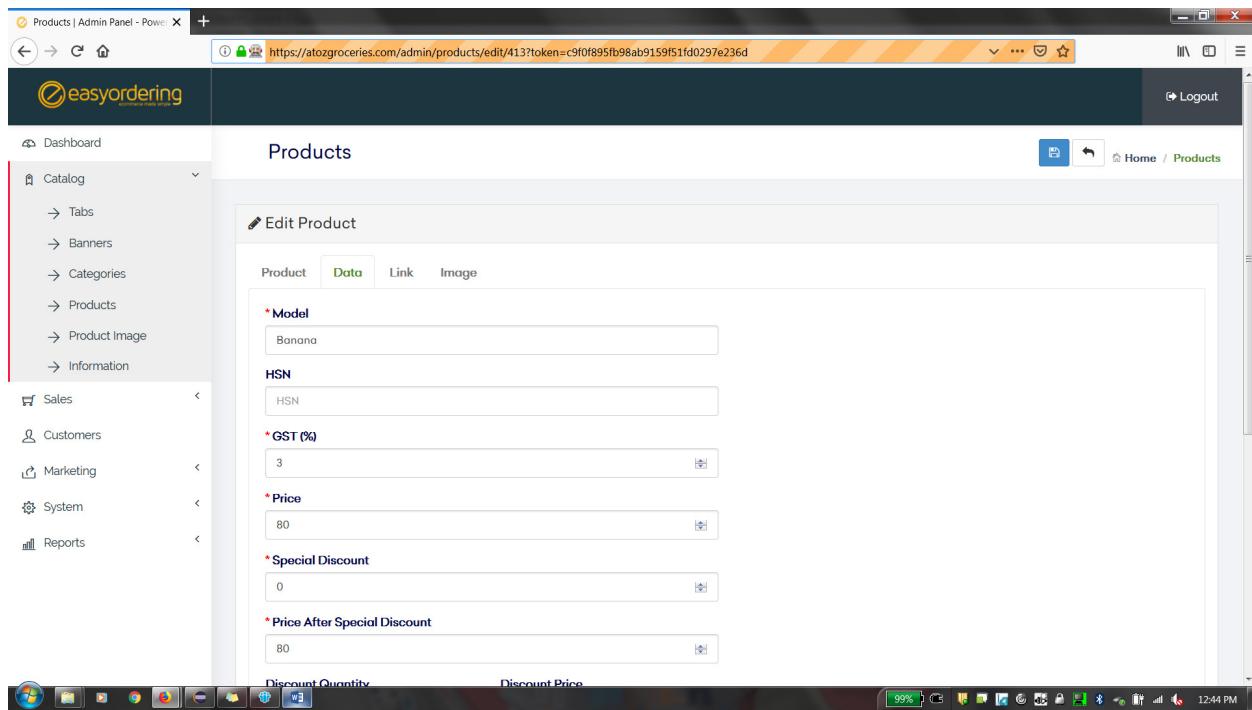
The browser address bar shows the URL: <https://atozgroceries.com/admin/products/edit/413?token=c9f0f895fb98ab9159f51fd0297e236d>.

The screenshot shows the 'Edit Product' interface with the 'Data' tab selected. The main area displays the following meta-information fields:

- Meta Title:** Tomato
- Meta Tag Description:** Meta Tag Description
- Meta Tag Keywords:** Meta Tag Keywords

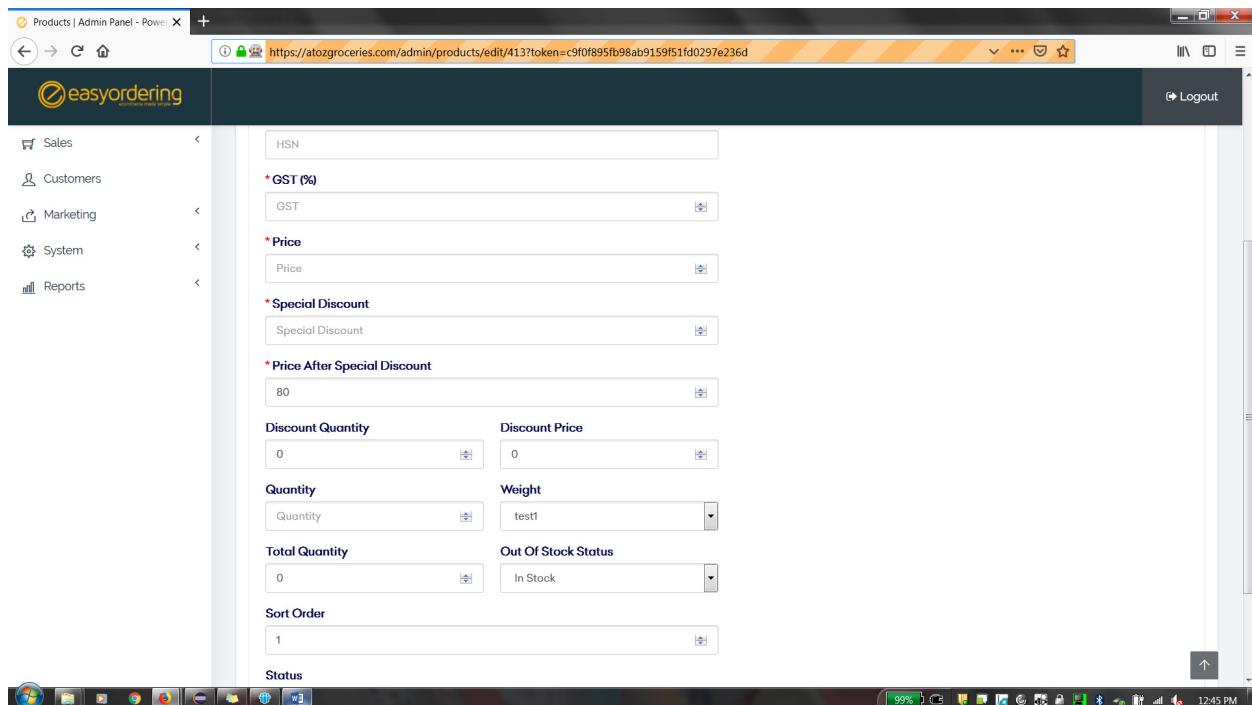
The browser address bar shows the URL: <https://atozgroceries.com/admin/products/edit/413?token=c9f0f895fb98ab9159f51fd0297e236d>.

Opening the data tab



The screenshot shows the 'Products' section of the easyordering Admin Panel. On the left, a sidebar lists categories like Catalog, Sales, Customers, Marketing, System, and Reports. The main area is titled 'Edit Product' and has tabs for Product, Data, Link, and Image. The 'Data' tab is selected. The form contains fields for Model (Banana), HSN (HSN), GST (%), Price (80), Special Discount (0), and Price After Special Discount (80). Below the form are sections for Discount Quantity (0) and Discount Price (0). The status bar at the bottom shows system information like battery level (99%) and time (12:44 PM).

Clearing the existing data in data tab



This screenshot shows the same 'Edit Product' page after clearing the data. The fields are now empty: Model (Banana), HSN (HSN), GST (%), Price (80), Special Discount (0), and Price After Special Discount (80). The sections for Discount Quantity (0) and Discount Price (0) are also empty. The status bar at the bottom shows system information like battery level (99%) and time (12:45 PM).

Products | Admin Panel - Powe... X

<https://atozgroceries.com/admin/products/edit/413?token=c9f0f895fb98ab9159f51fd0297e236d>

Logout

Reports

Price

*Special Discount

Special Discount

*Price After Special Discount

Price After Special Discount

Discount Quantity

Quantity

Weight

Total Quantity

Out Of Stock Status

Sort Order

Status

Enabled

Providing the new data for the the product in data tab

Products | Admin Panel - Powe... X

<https://atozgroceries.com/admin/products/edit/413?token=c9f0f895fb98ab9159f51fd0297e236d>

Logout

Dashboard

Catalog

- Tabs
- Banners
- Categories
- Products
- Product Image
- Information

Sales

Customers

Marketing

System

Reports

Products

Edit Product

Product Data Link Image

*Model

Tomato

HSN

HSN

*GST (%)

4

*Price

45

*Special Discount

0

*Price After Special Discount

45

Discount Quantity

Discount Price

Products | Admin Panel - Powe X +

https://atozgroceries.com/admin/products/edit/413?token=c9f0f895fb98ab9159f51fd0297e236d

Logout

easyordering

Reports

45

*Special Discount
0

*Price After Special Discount
45

Discount Quantity
0

Quantity
5

Discount Price
0

Weight
test1

Total Quantity
5

Out Of Stock Status
In Stock

Sort Order
1

Status
Enabled

Opening link tab

Products | Admin Panel - Powe X +

https://atozgroceries.com/admin/products/edit/413?token=c9f0f895fb98ab9159f51fd0297e236d

Logout

easyordering

Dashboard

Catalog

- Tabs
- Banners
- Categories
- Products
- Product Image
- Information

Sales

Customers

Marketing

System

Reports

Products

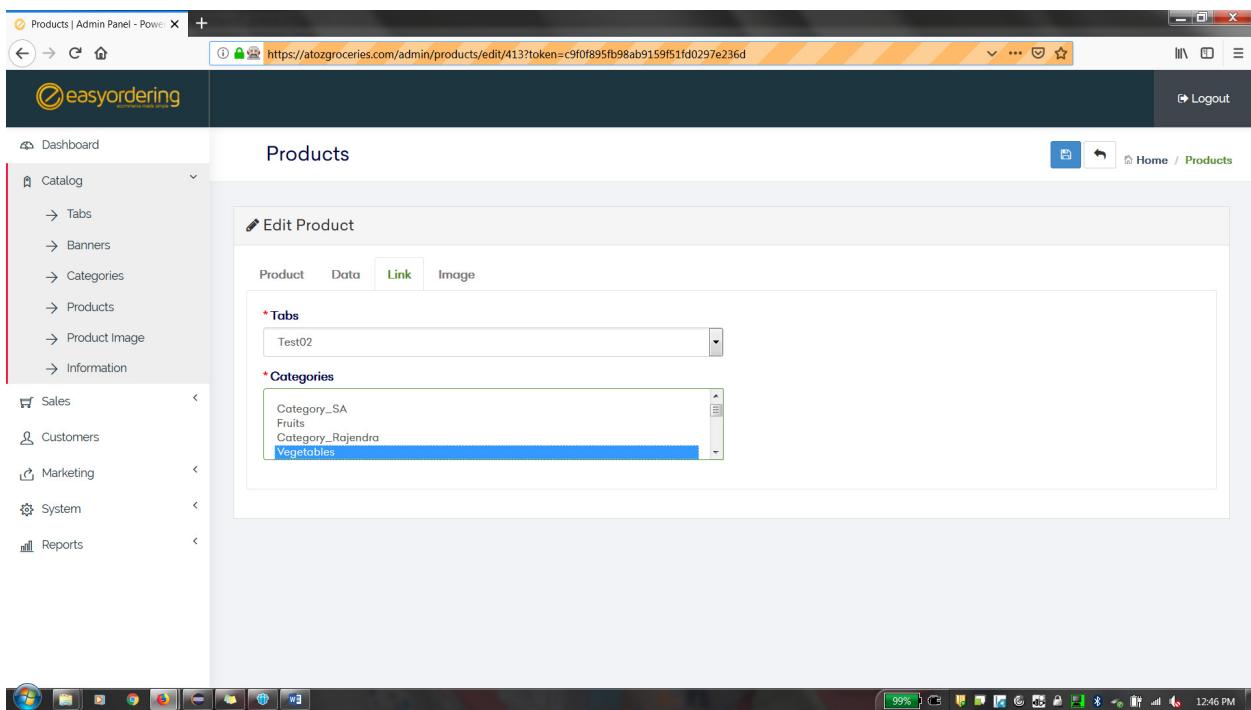
Edit Product

Product Data Link Image

*Tabs
Tab_SA

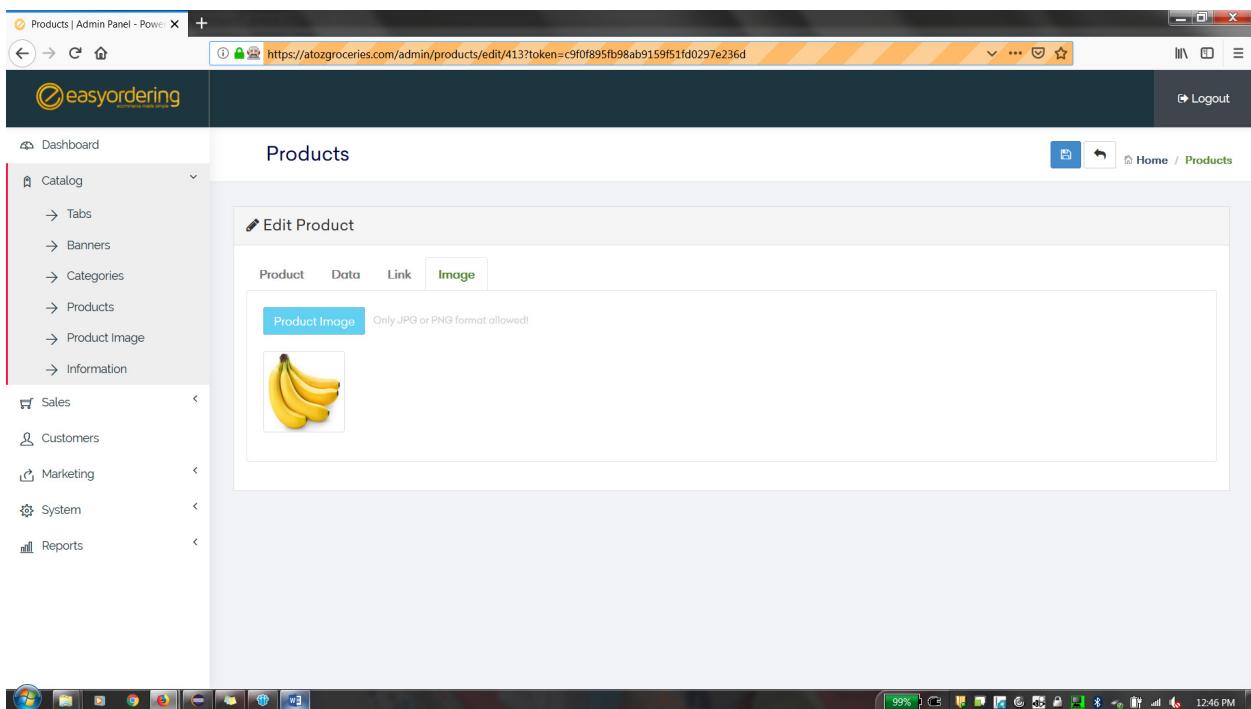
*Categories
Category_SA
Fruits
Category_Rajendra
Vegetables

Providing the new data for the the product in link tab



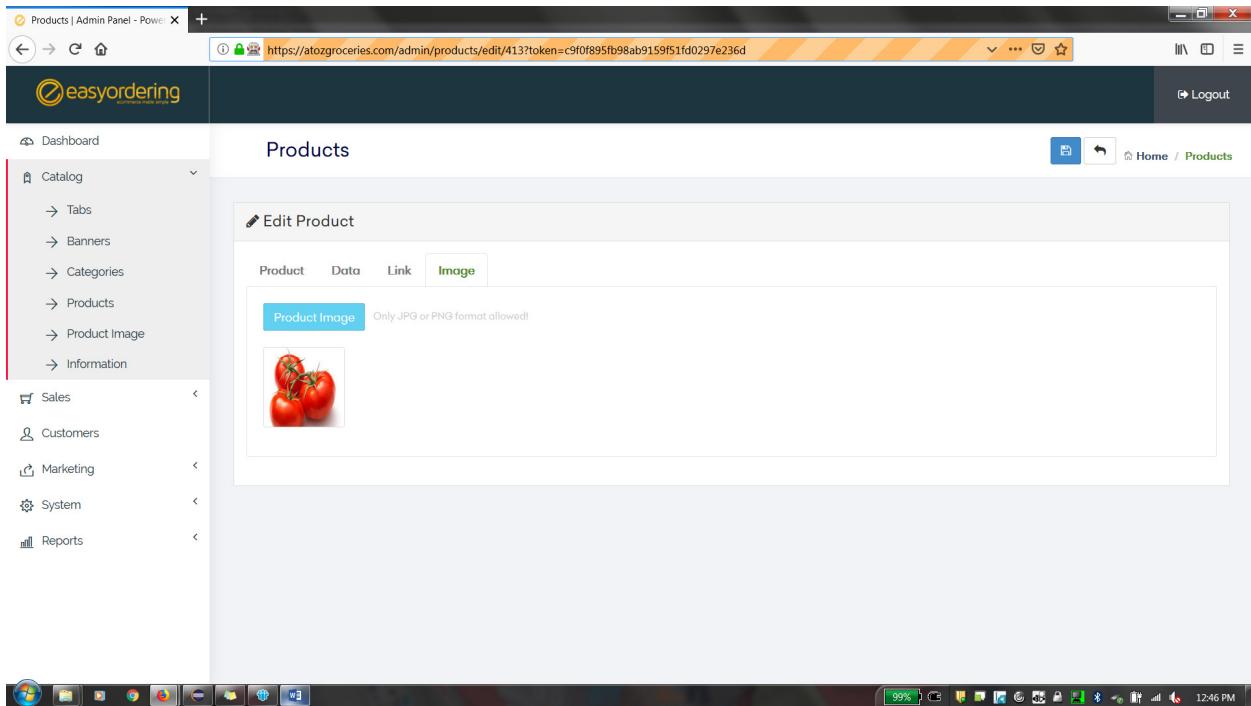
The screenshot shows the 'Edit Product' interface on a web browser. The left sidebar has a 'Catalog' section with 'Tabs' selected. The main area has tabs for 'Product', 'Data', 'Link', and 'Image', with 'Link' currently active. Under 'Link', there are fields for 'Tabs' (containing 'Test02') and 'Categories' (showing a dropdown menu with 'Category_SA', 'Fruits', 'Category_Rajendra', and 'Vegetables', where 'Vegetables' is highlighted). The status bar at the bottom shows a battery level of 99% and the time 12:46 PM.

Opening Image tab



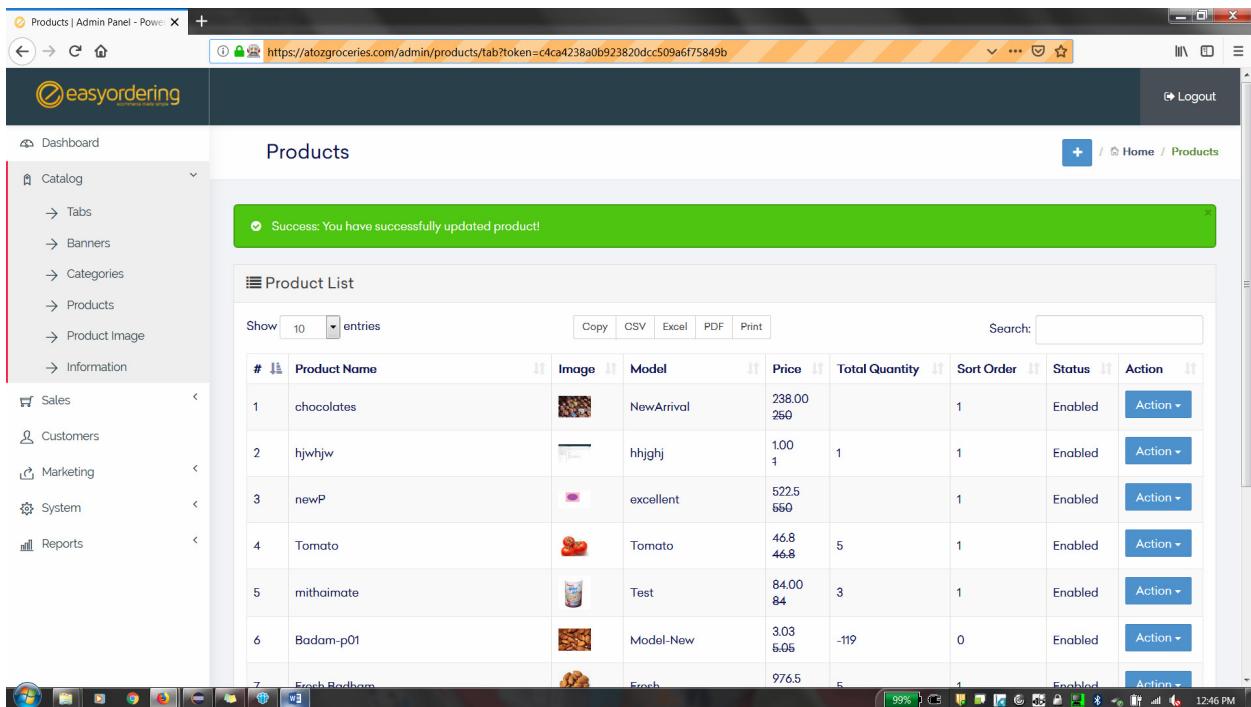
The screenshot shows the 'Edit Product' interface on a web browser. The left sidebar has a 'Catalog' section with 'Products' selected. The main area has tabs for 'Product', 'Data', 'Link', and 'Image', with 'Image' currently active. Under 'Image', there is a 'Product Image' field containing an image of three bananas. A tooltip message 'Only JPG or PNG format allowed!' is visible next to the image field. The status bar at the bottom shows a battery level of 99% and the time 12:46 PM.

Sending the new image for the product and clicking on “save” to save the product



The screenshot shows the 'Edit Product' interface. On the left, a sidebar menu is open under 'Catalog', showing options like 'Tabs', 'Banners', 'Categories', 'Products', 'Product Image', and 'Information'. The main area is titled 'Edit Product' and has tabs for 'Product', 'Data', 'Link', and 'Image'. The 'Image' tab is active, displaying a placeholder for a 'Product Image' with the note 'Only JPG or PNG format allowed!'. Below the placeholder is a thumbnail image of three red tomatoes. The status bar at the bottom of the screen shows various system icons and the time '12:46 PM'.

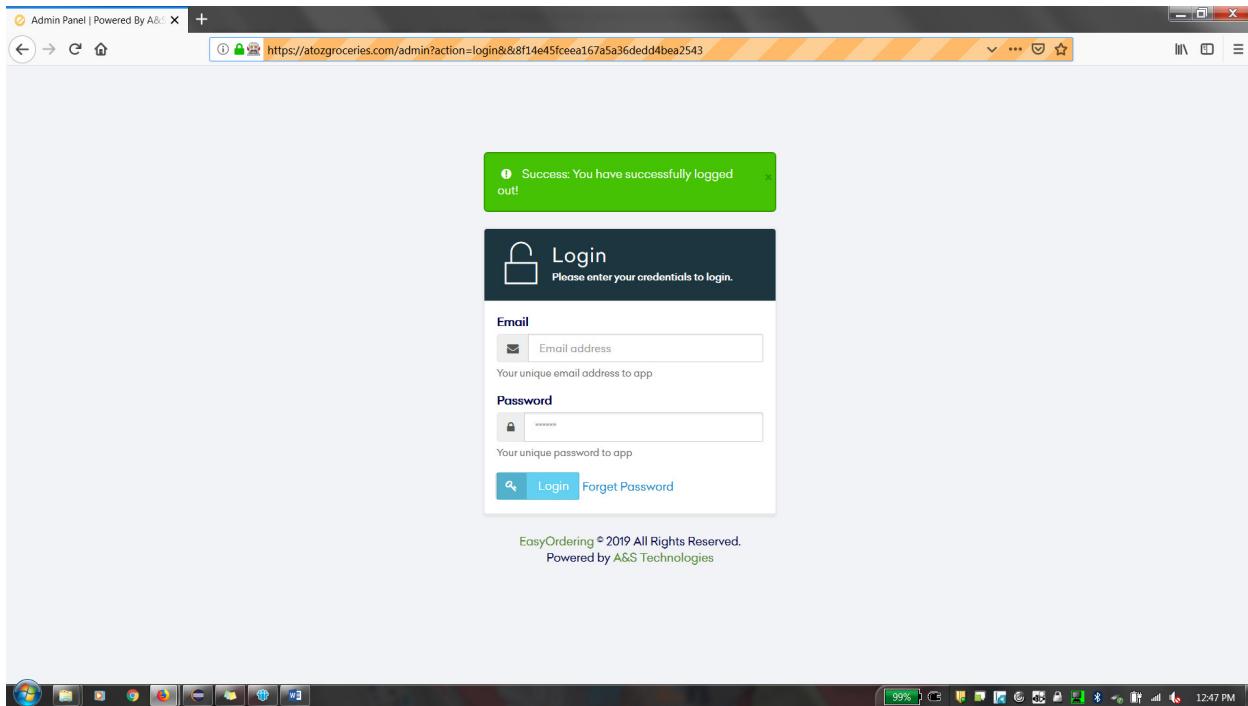
Validating the Success message and updated product “Tomato” with quantity 5



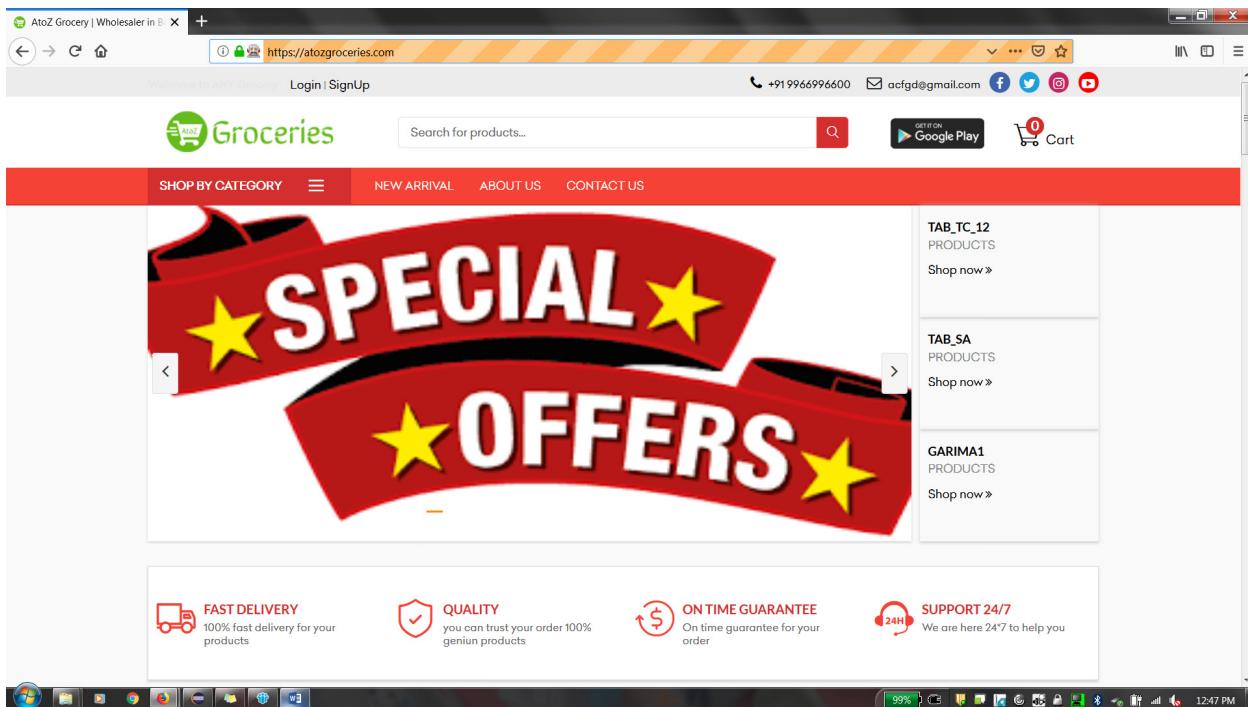
The screenshot shows the 'Product List' page. The sidebar menu is identical to the previous screenshot. The main area displays a table of products. A green success message at the top of the table area reads 'Success: You have successfully updated product!'. The table has columns: #, Product Name, Image, Model, Price, Total Quantity, Sort Order, Status, and Action. The 'Tomato' entry in the list is highlighted with a blue background. The 'Total Quantity' column for Tomato is listed as 5. The status bar at the bottom shows system icons and the time '12:46 PM'.

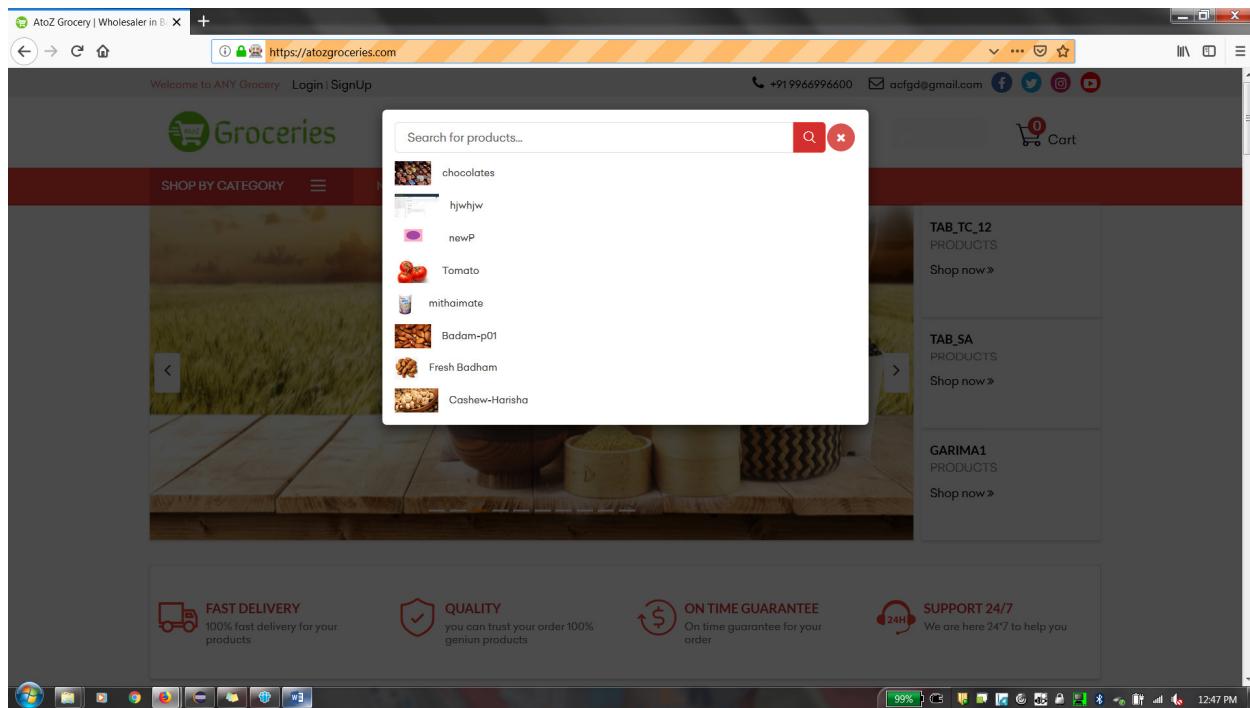
#	Product Name	Image	Model	Price	Total Quantity	Sort Order	Status	Action
1	chocolates		NewArrival	238.00 260		1	Enabled	Action
2	hjwhjw		hhjghj	1.00 1	1	1	Enabled	Action
3	newP		excellent	522.5 550		1	Enabled	Action
4	Tomato		Tomato	46.8 46.8	5	1	Enabled	Action
5	mithaimate		Test	84.00 84	3	1	Enabled	Action
6	Badam-p01		Model-New	3.03 5.05	-119	0	Enabled	Action
7	Fresh Badham		Fresh	976.5			Enabled	Action

Logging out from admin portal

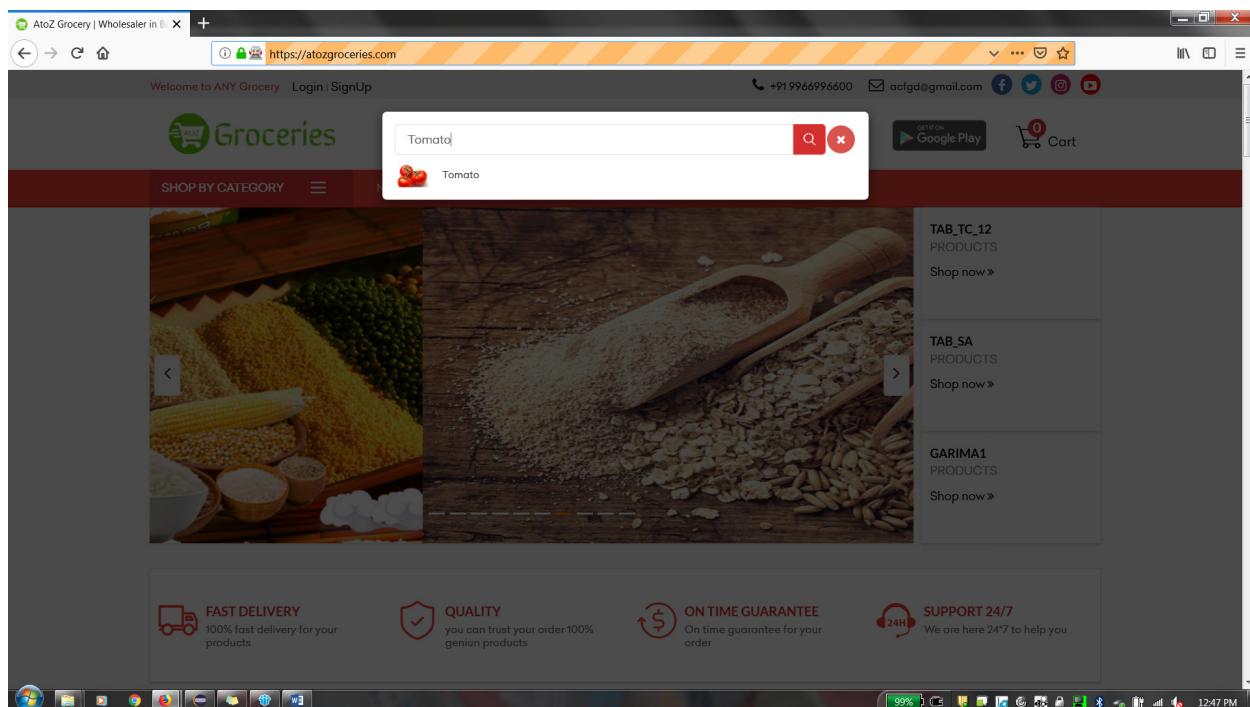


Opening the User portal and clicking on Search for products..





Sending the updated product as input for validating the quantity



Opening the product and adding to cart

The screenshot shows a product page for a Tomato. The main image displays three ripe red tomatoes. To the right of the image, a green button indicates "0% OFF". The product name "Tomato" is displayed in bold black text. Below the name, it says "Discounted price: ₹45.00" and "MRP: ₹45.00". The availability status is shown as "In stock". A blue "Add To Cart" button is prominently displayed. Below the button, a section titled "QUICK OVERVIEW" contains the text "good quality". At the bottom of the page, there is a "RELATED PRODUCTS" section featuring thumbnail images of other items.

Success message after adding to cart

The screenshot shows the same Tomato product page as the previous one, but with a green success message banner at the top stating "Successfully added Tomato 5 test1 to the basket". The rest of the page content, including the product image, details, and related products section, remains identical to the first screenshot.

Opening cart and validating the quantity of the product

The screenshot shows a web browser displaying a product page for 'Tomato' on the 'Groceries' section of the 'AtoZ Grocery' website. The product image shows three ripe red tomatoes. The price is listed as 'Discounted price: ₹45.00' and 'MRP: ₹46.00'. The availability status is 'In stock'. Below the product details, there is a 'QUICK OVERVIEW' section with the text 'good quality'. On the right side of the screen, a 'My Cart (5 items)' summary is visible, showing one item: Tomato, Quantity: 5, Price: ₹45.00. The total sub-total for the cart is ₹225.00. The browser interface includes a navigation bar with back, forward, and search buttons, and a status bar at the bottom showing system information like battery level and time.

Output from console

The screenshot shows the Eclipse IDE interface with the 'SeleniumFramework' project open. The 'testng.xml' configuration file is displayed in the center, defining a suite named 'Test' with a single test case 'Day13TestCase'. The 'AdminPage.java' class under the 'com.ibm.pages' package is shown in the code editor. The 'Console' tab at the bottom displays several error messages in red, primarily related to JavaScript errors on the AtoZ Groceries website, such as 'TypeError: document.getElementById(...)', and a warning message: 'WARNING: [Parent 10380, Gecko_IOThread] WARNING: pipe error: 109: file z:/build/src/ipc/chromium/src/chrome/common/ipc_channel_win.cc, line 346'. The status bar at the bottom indicates the system is at 95% battery and the time is 2:28 PM.

Automated report of index.html

Test results
1 suite

All suites

Suite

Info

- C:\Harisha\Selenium_Training\EclipseWorkSpace\SeleniumFramework\testing.xml
- 1 test
- 0 groups
- 0 times
- Reporter output (highlighted)
- Ignored methods
- Chronological view

Results

- 1 method, 1 passed
- Passed methods (show)

Reporter output for Suite

```
Day13TestCase(Banana, Tomato, Tomato, Tomato, 4, 45, 0, 5, 5, 1, Enabled, Test02, Vegetables, C:\Harisha\Selenium...) 
Validation of Error Message - Product Name as mandatory field!! Validation of Success Message after editing the product!! Validation of edited product presence in Admin Page products Table!! validating the quantity of product Tomato added in cart is 5!! Assertion on Edited Product in Database!! 
Assertion on Edited Product quantity in Database!!
```



Automated report of Emailable report

TestNG Report

Test # Passed # Skipped # Failed Time (ms) Included Groups Excluded Groups

Suite					
Test	1	0	0	90,489	
Class	Method	Start	Time (ms)		
Suite					
Test — passed					
com.ibm.test.BaseTest	Day13TestCase	[1546592193528]	[8174]		

Test

com.ibm.test.BaseTest#Day13TestCase

Parameter #1	Parameter #2	Parameter #3	Parameter #4	Parameter #5	Parameter #6	Parameter #7	Parameter #8	Parameter #9	Parameter #10	Parameter #11	Parameter #12	Parameter #13	Parameter #14	Parameter #15	Parameter #16
Banana	Tomato	Tomato	Tomato	4	45	0	5	5	1	Enabled	Test02	Vegetables	C:\Harisha\Selenium_Training\EclipseWorkSpace\SeleniumFramework\ TestData\tomato.jpg	fill out this field	successfully updated product

Messages

```
Validation of Error Message - Product Name as mandatory field!! 
Validation of Success Message after editing the product!! 
Validation of edited product presence in Admin Page products Table!! 
validating the quantity of product Tomato added in cart is 5!! 
Assertion on Edited Product in Database!! 
Assertion on Edited Product quantity in Database!!
```

[back to summary](#)



*** End of the document ***