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/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[2]/div/div/div[2]/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/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[2]/div/div/div[2]/div/div
/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[@clas
s='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")).siz
e();
		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="(//span[@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/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/div/a
")).click();
		String quantity=driver.findElement(By.xpath("//*[@class='quanlity-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 catergory 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

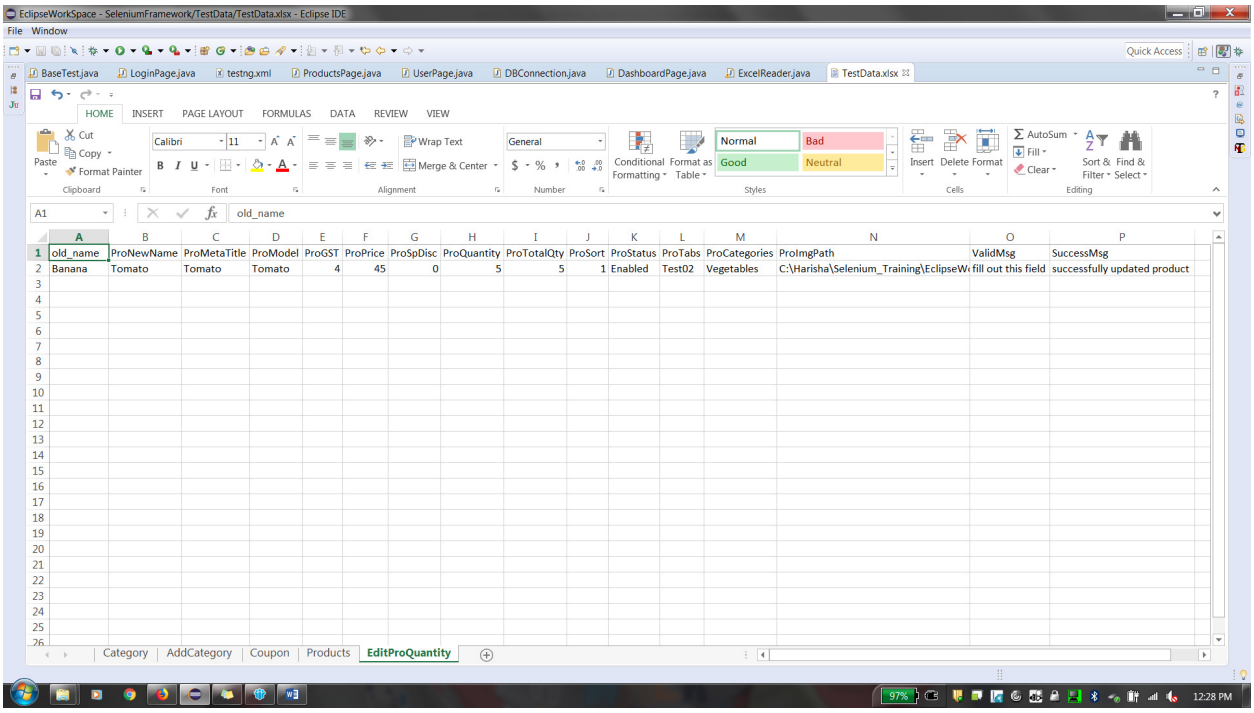
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
```

```
<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

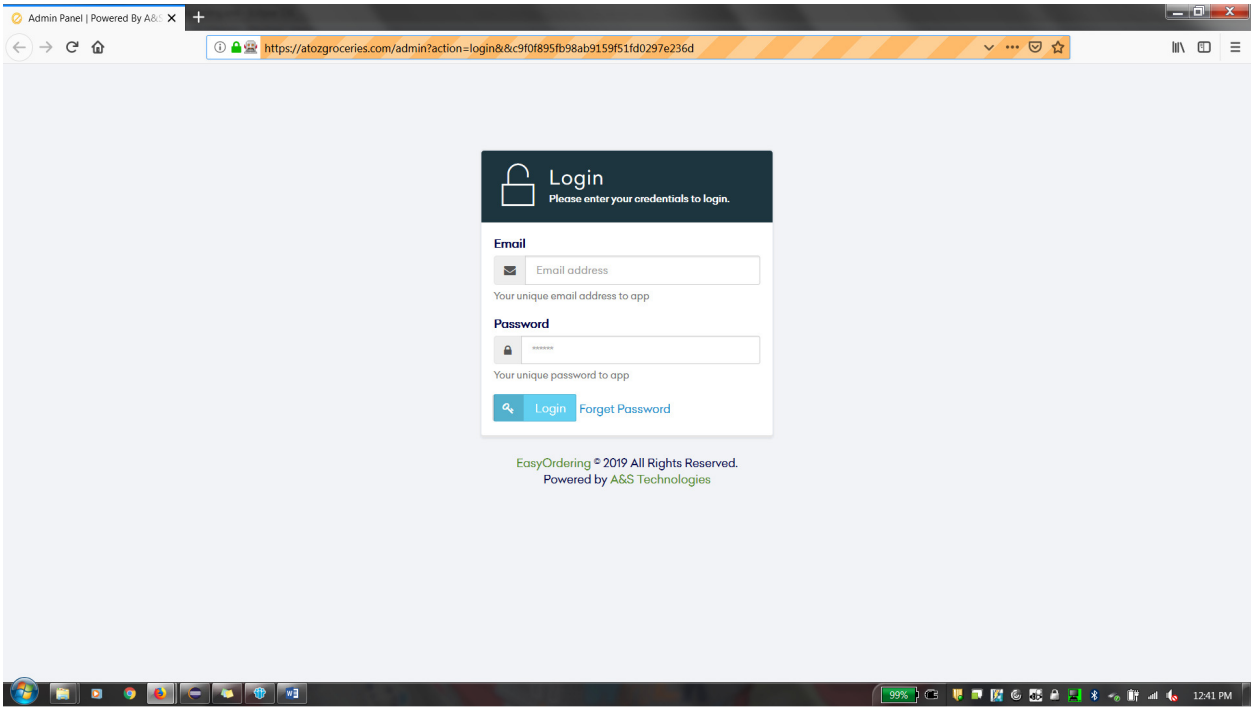
```
package com.ibm.utilities;

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
    {

        XSSFWorkbook TestDataWB=new XSSFWorkbook(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

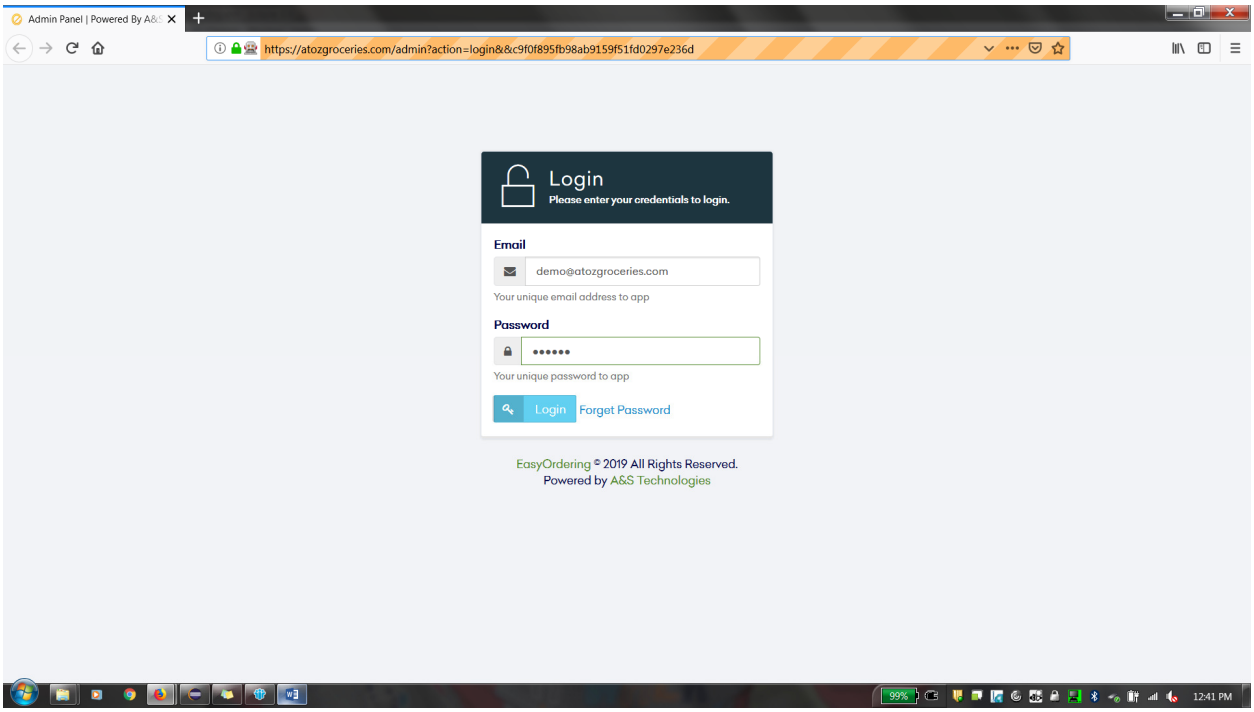


Step by step output:

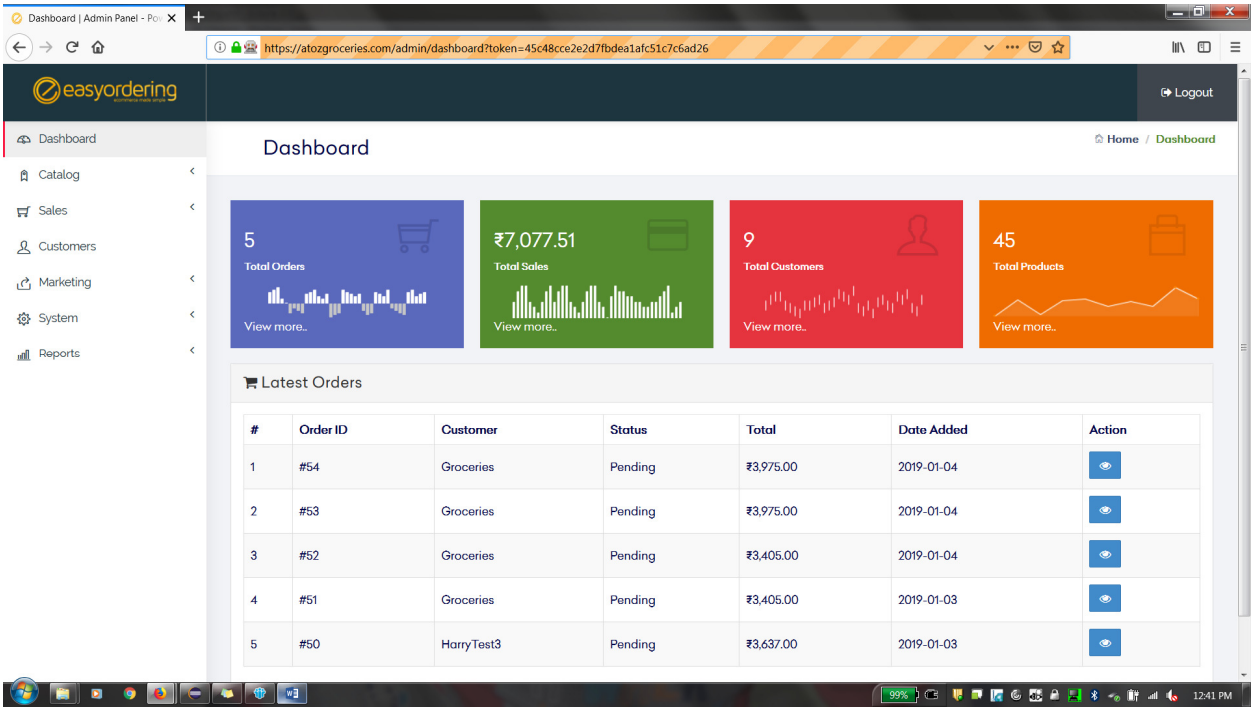
Opening Admin portal



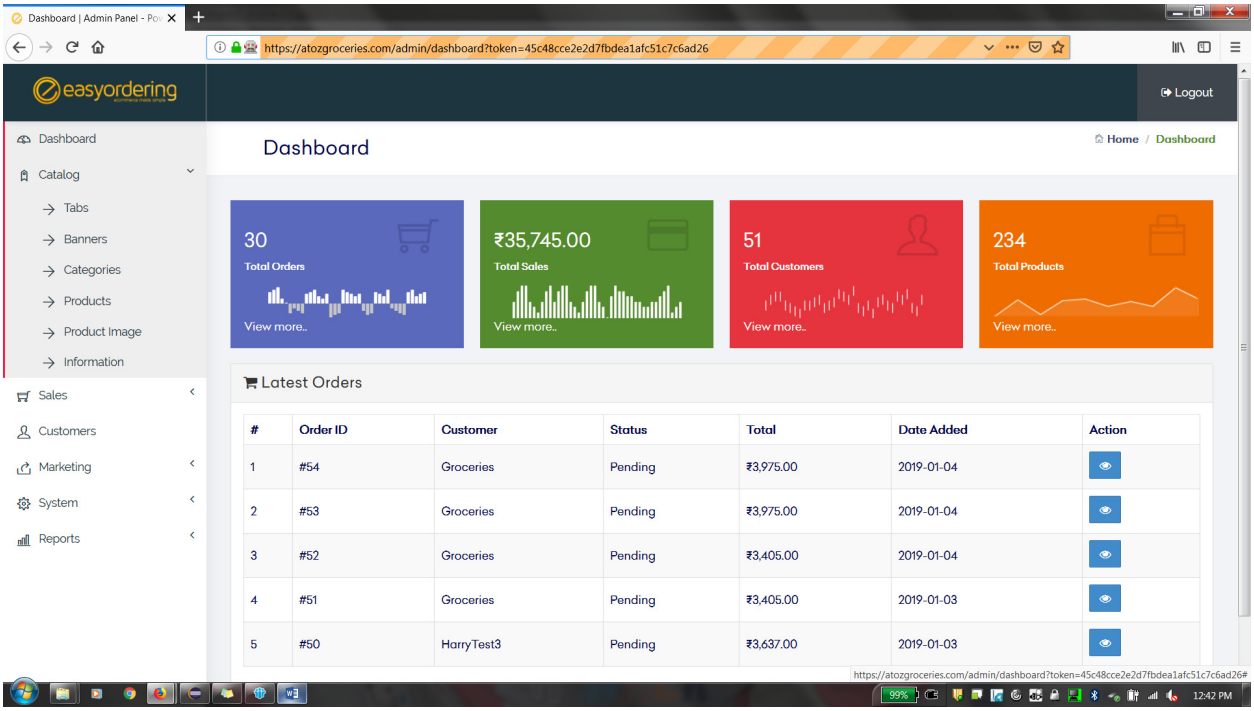
Providing login credentials for logging into admin portal



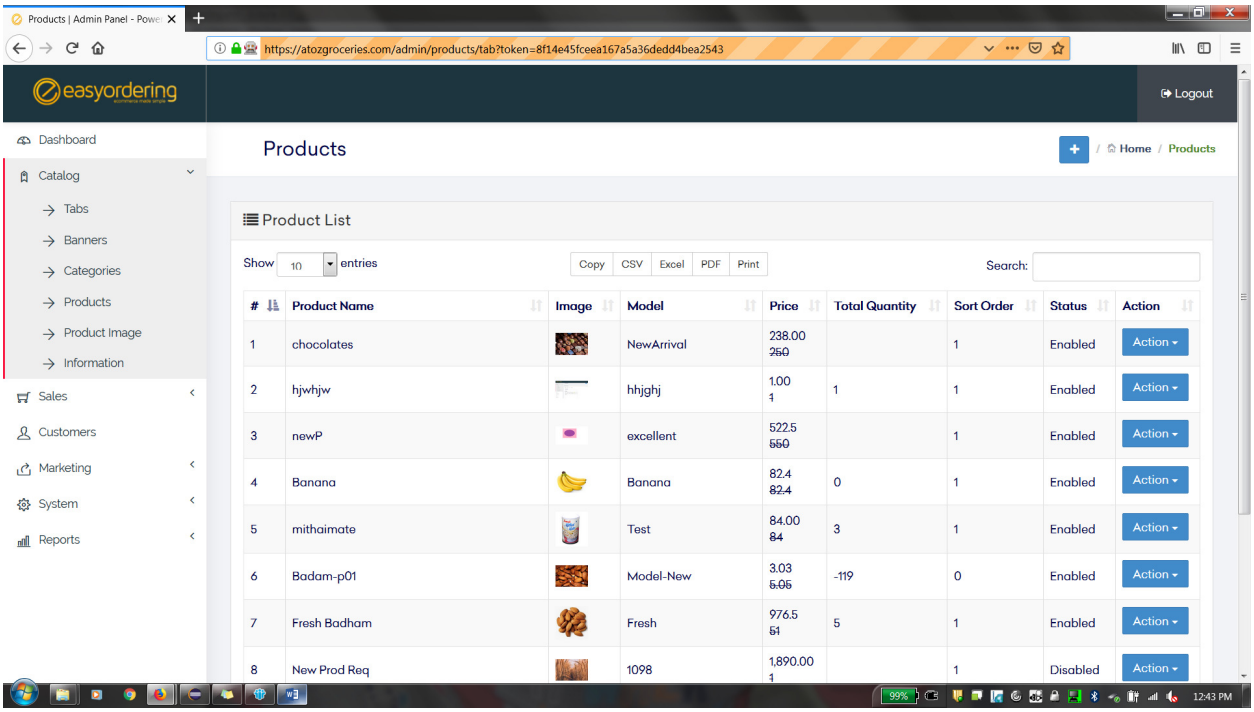
Admin portal after logging in



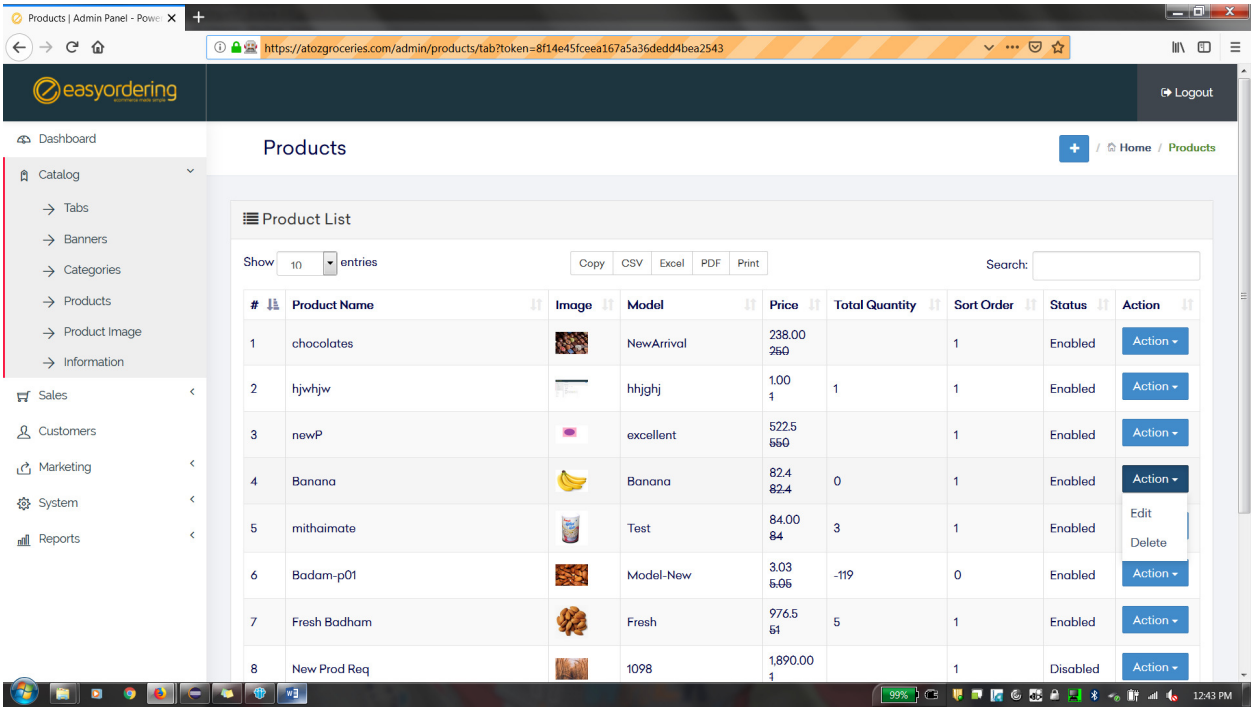
Opening Catalog to open Products



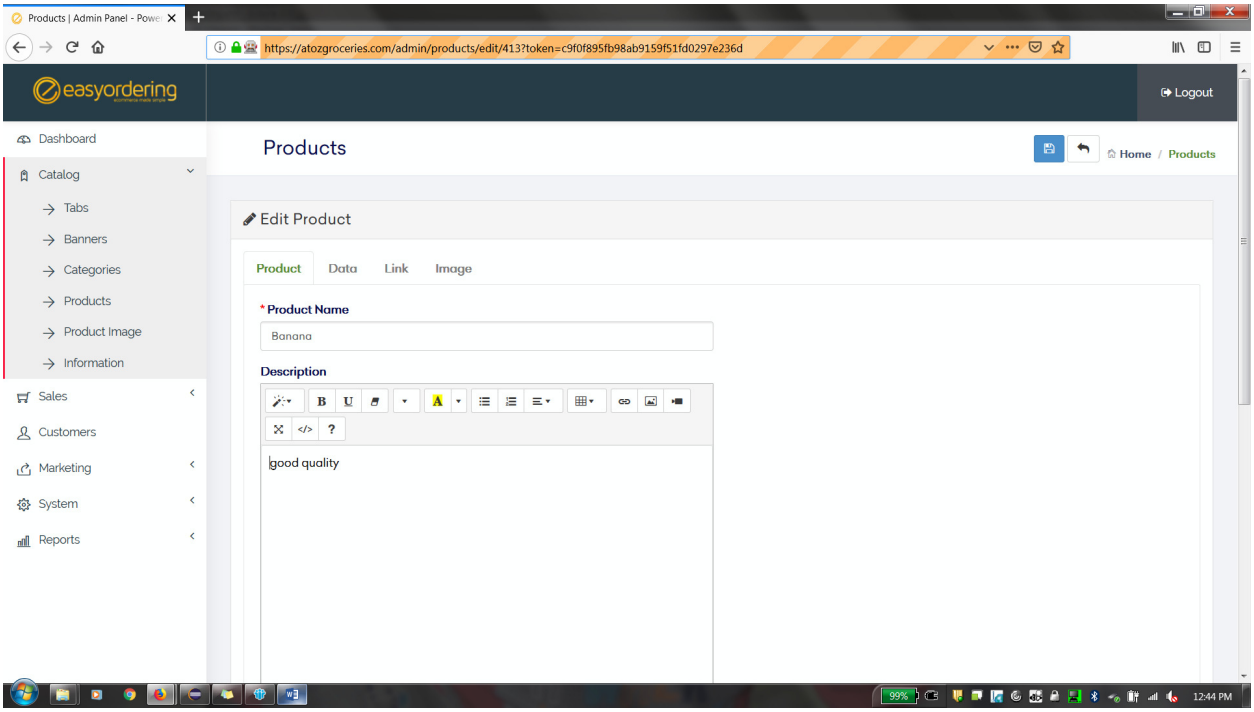
Opening products



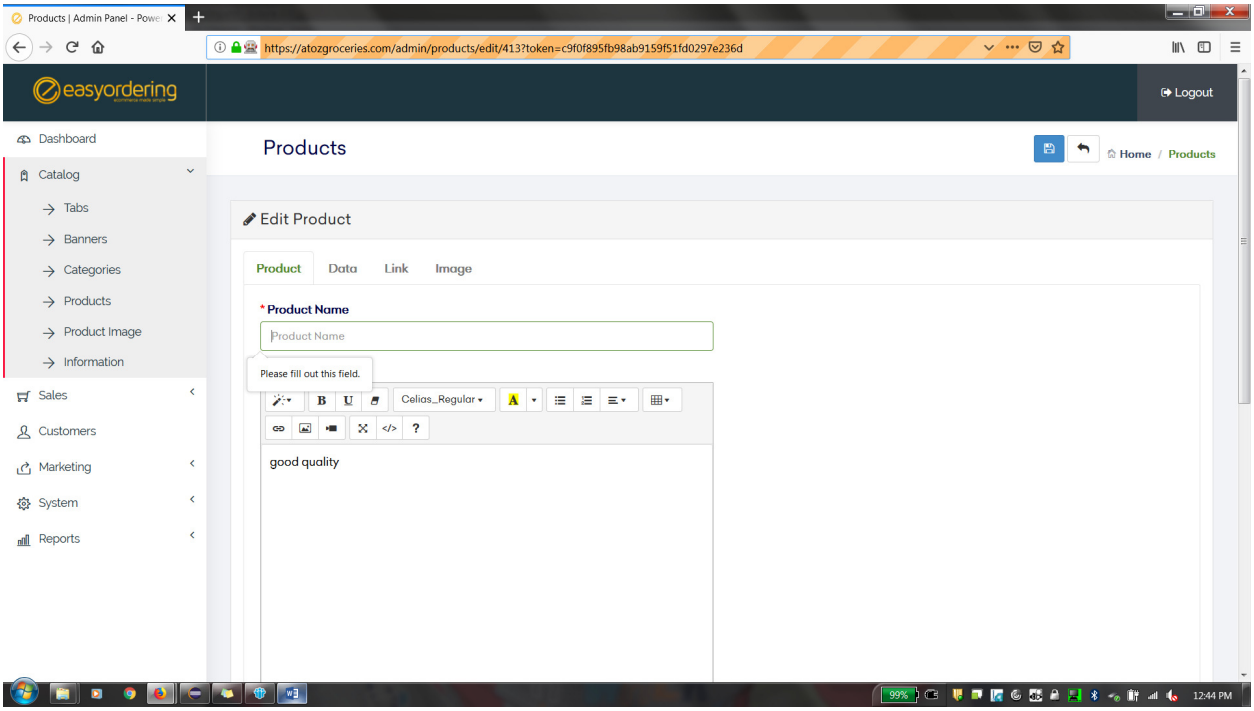
Selecting the product for modifying



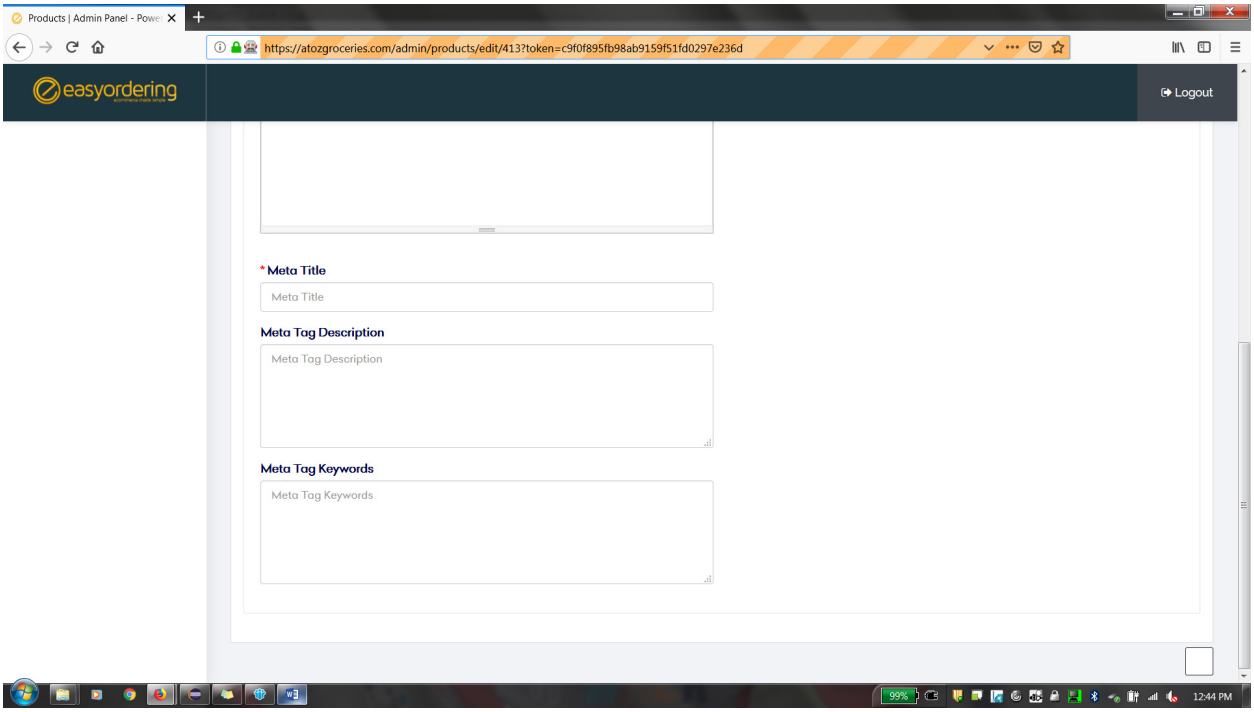
Opening the product for modifying



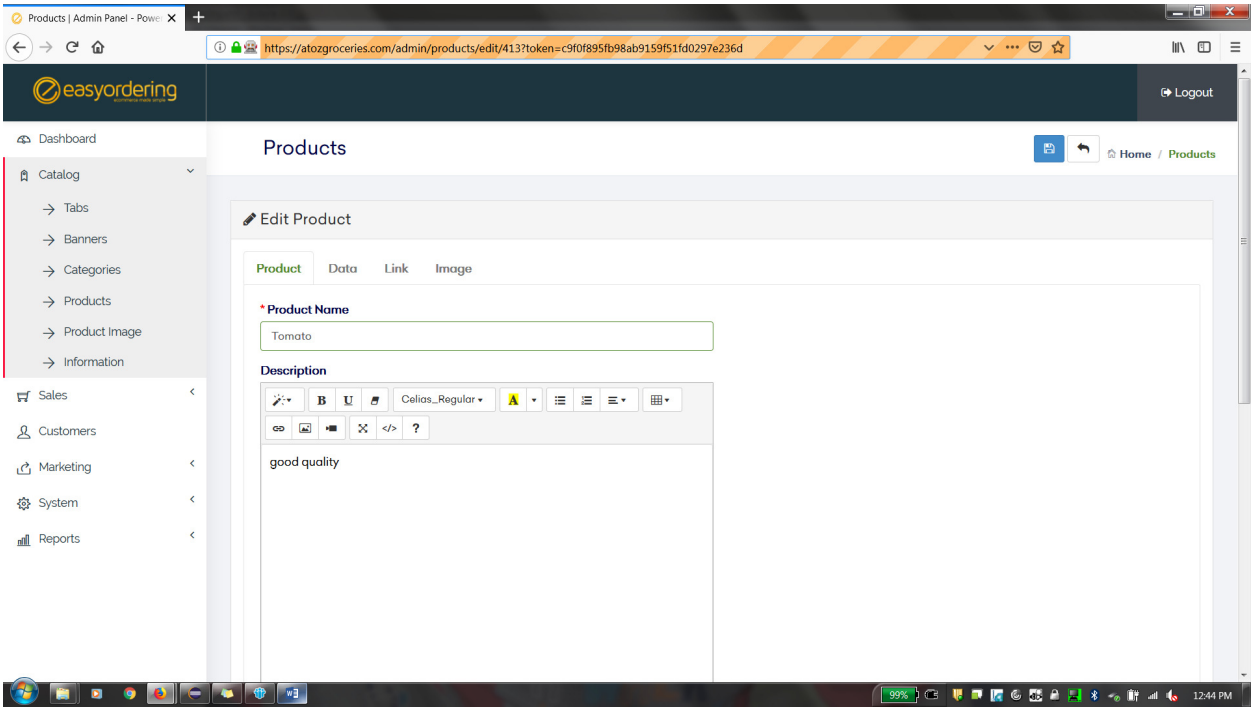
Validating the error message by removing the product name

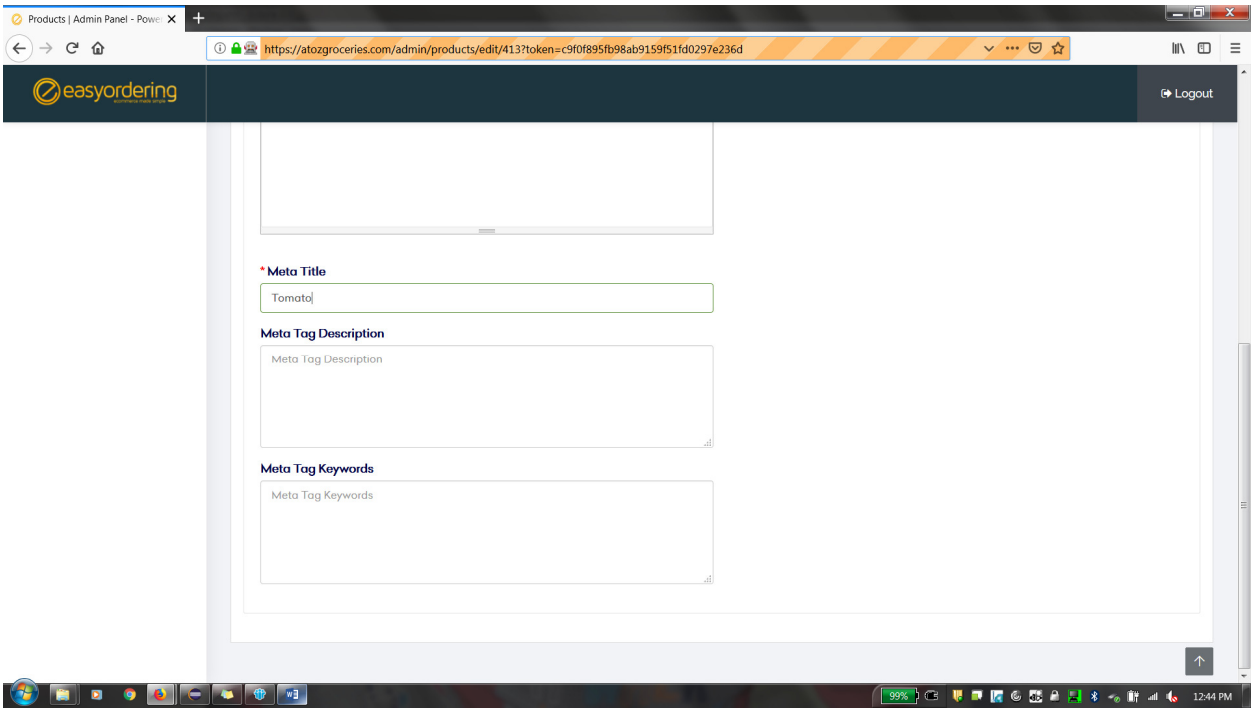


Clearing the existing data for the product

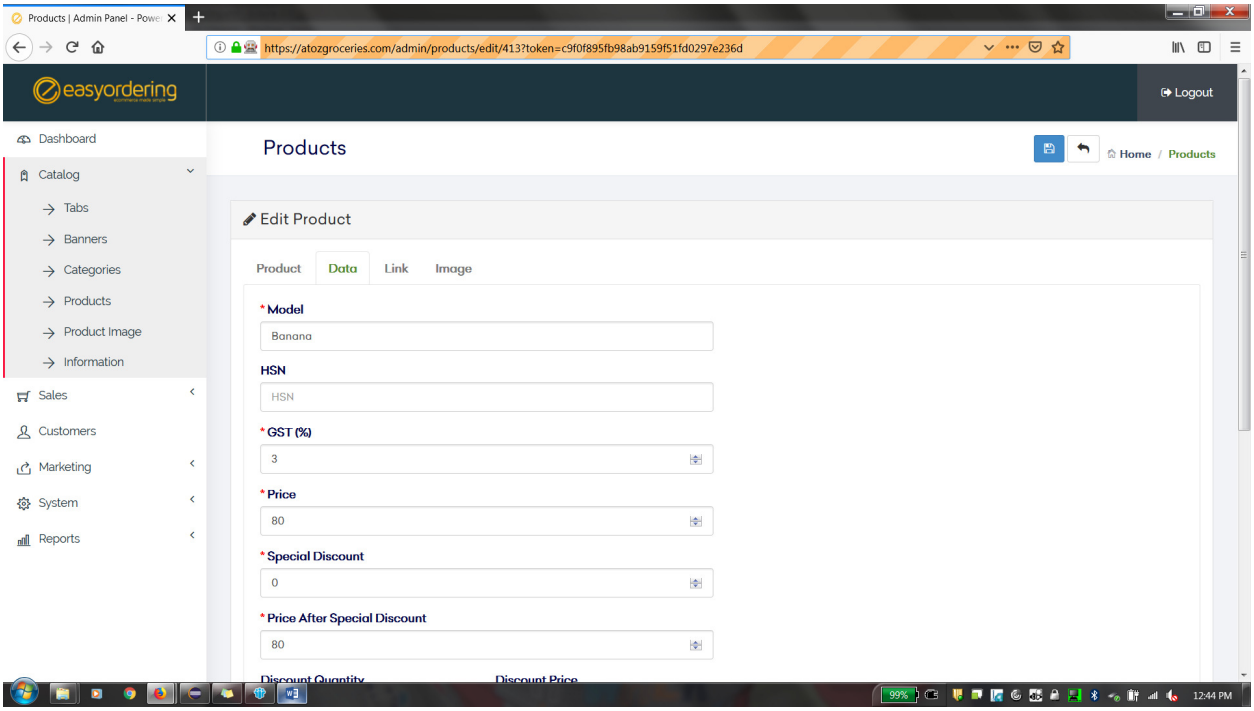


Providing the new data for the the product in product tab

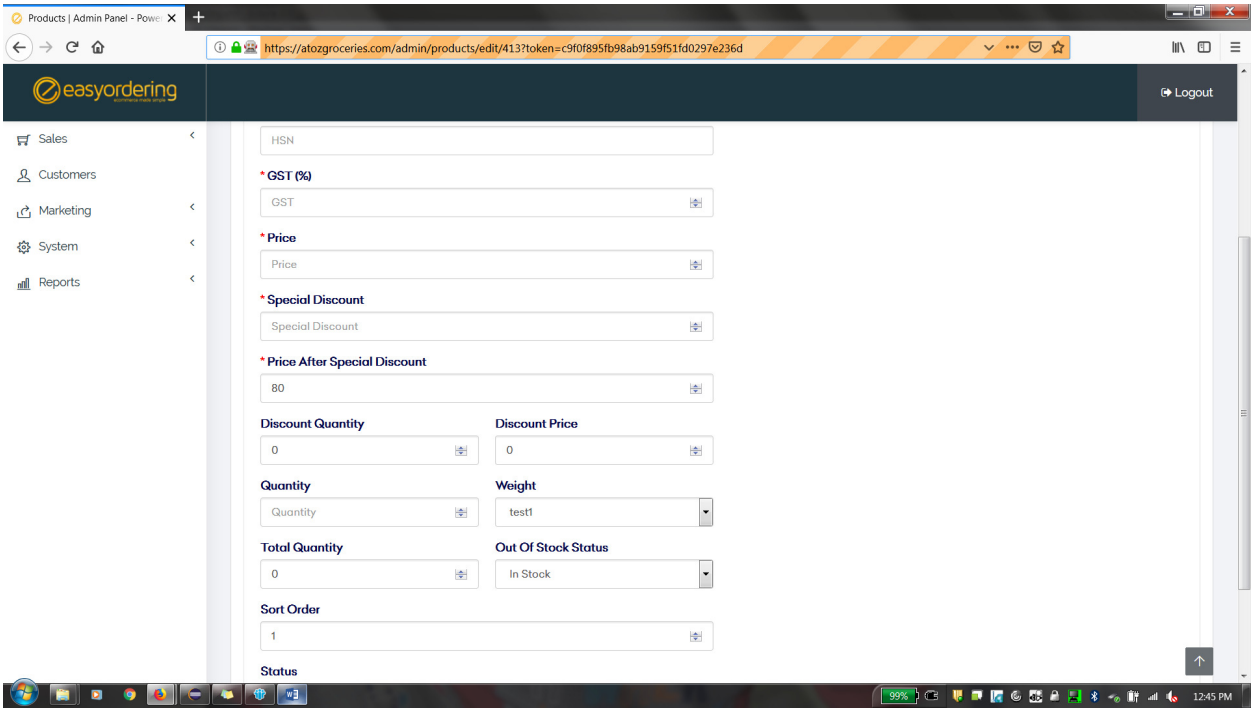


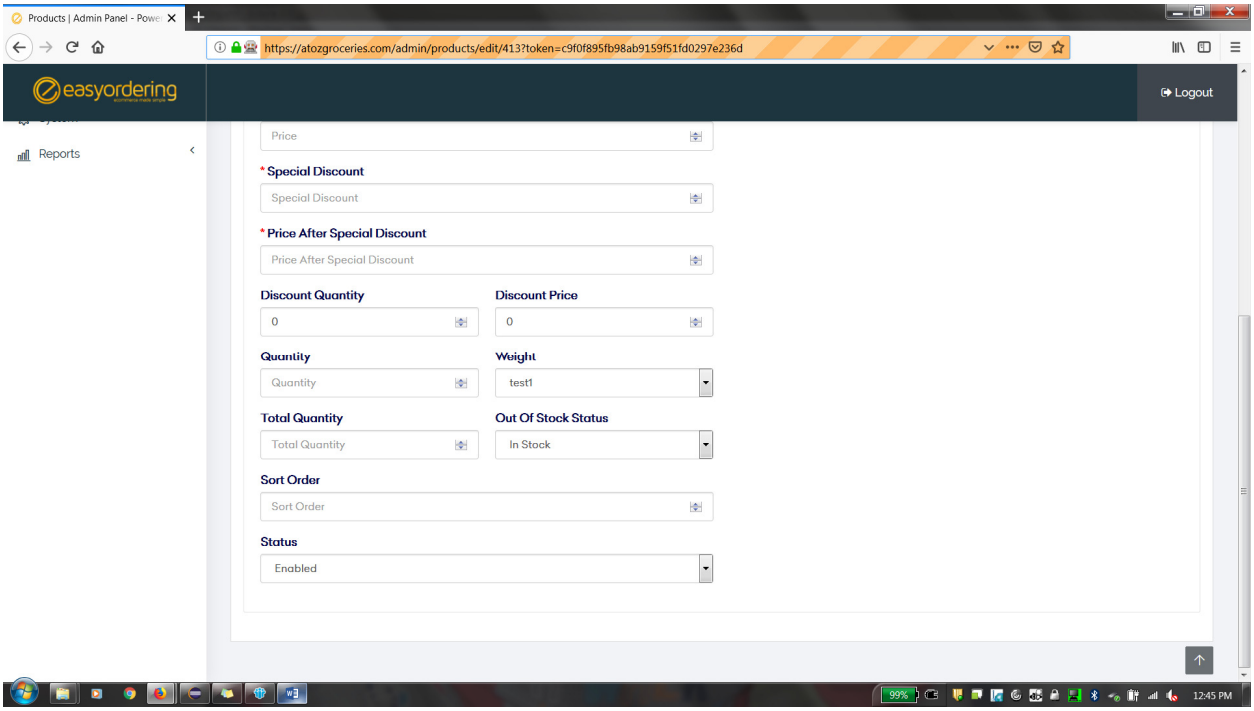


Opening the data tab

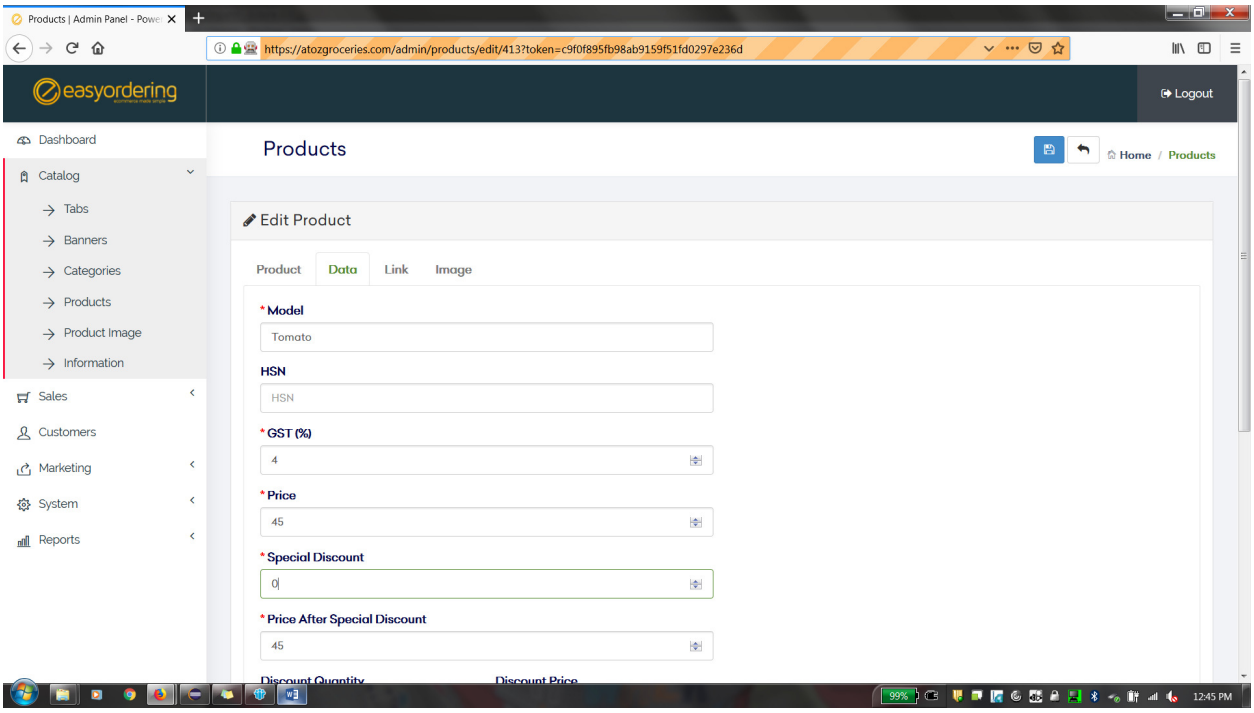


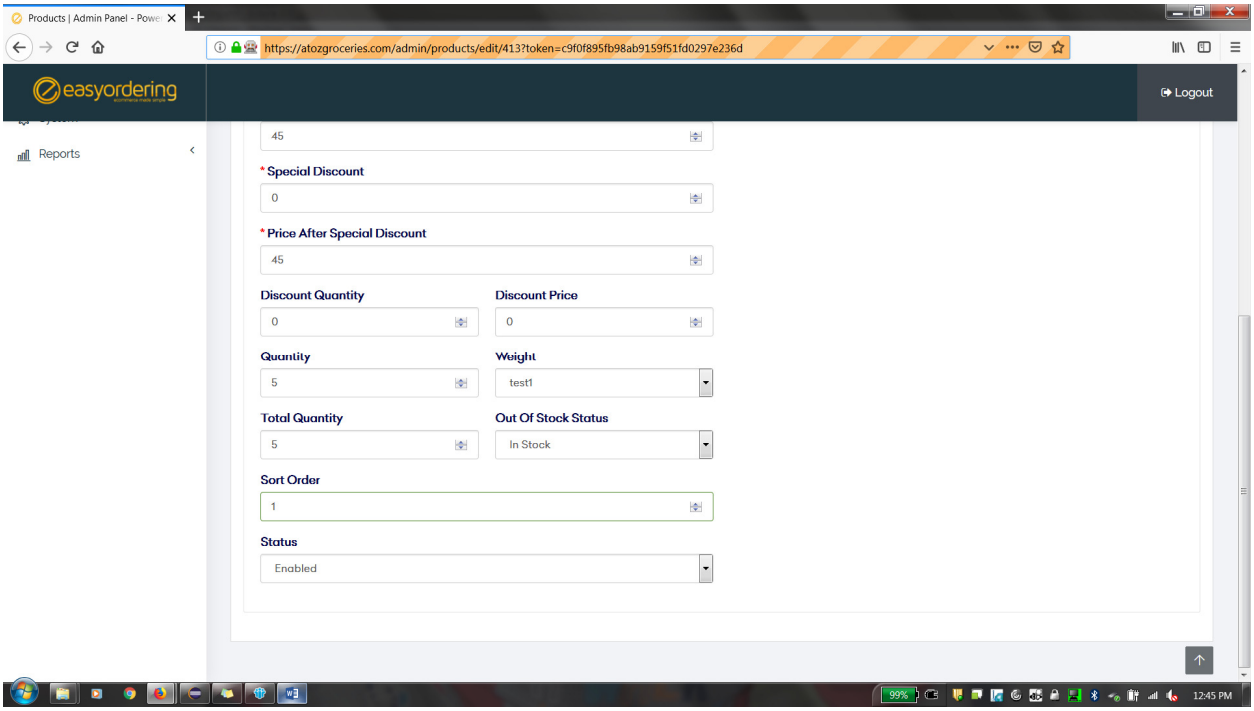
Clearing the existing data in data tab



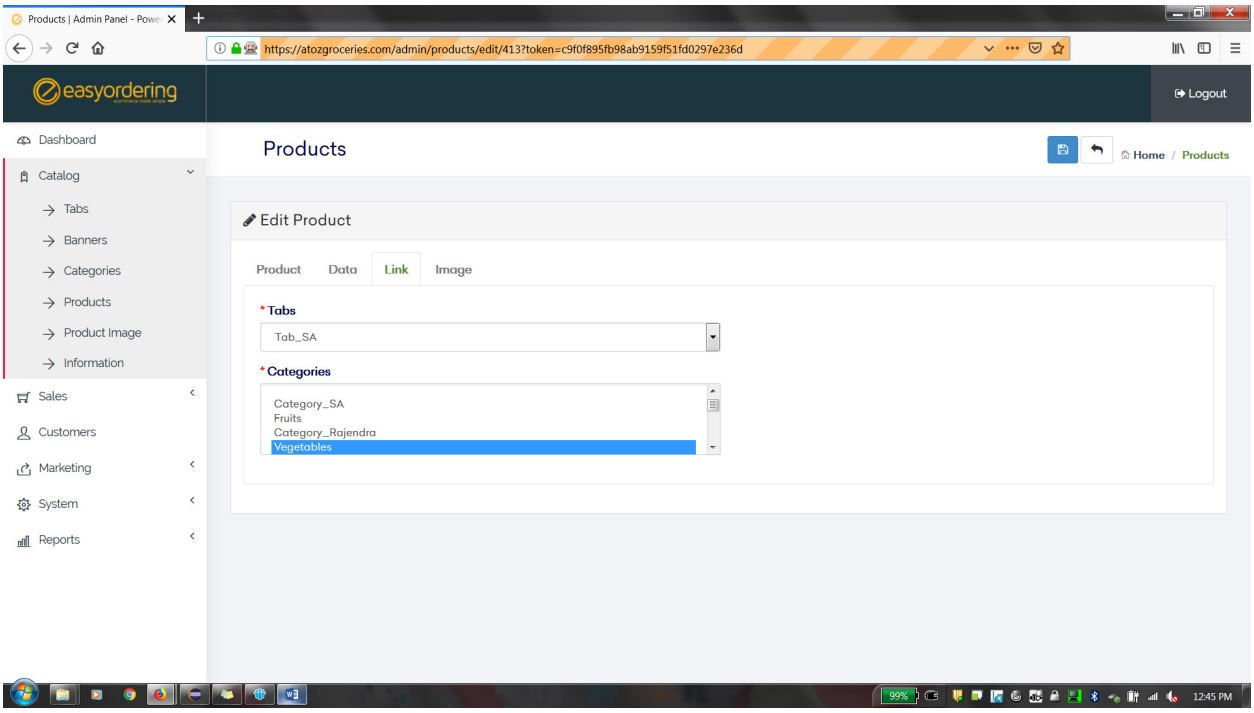


Providing the new data for the the product in data tab

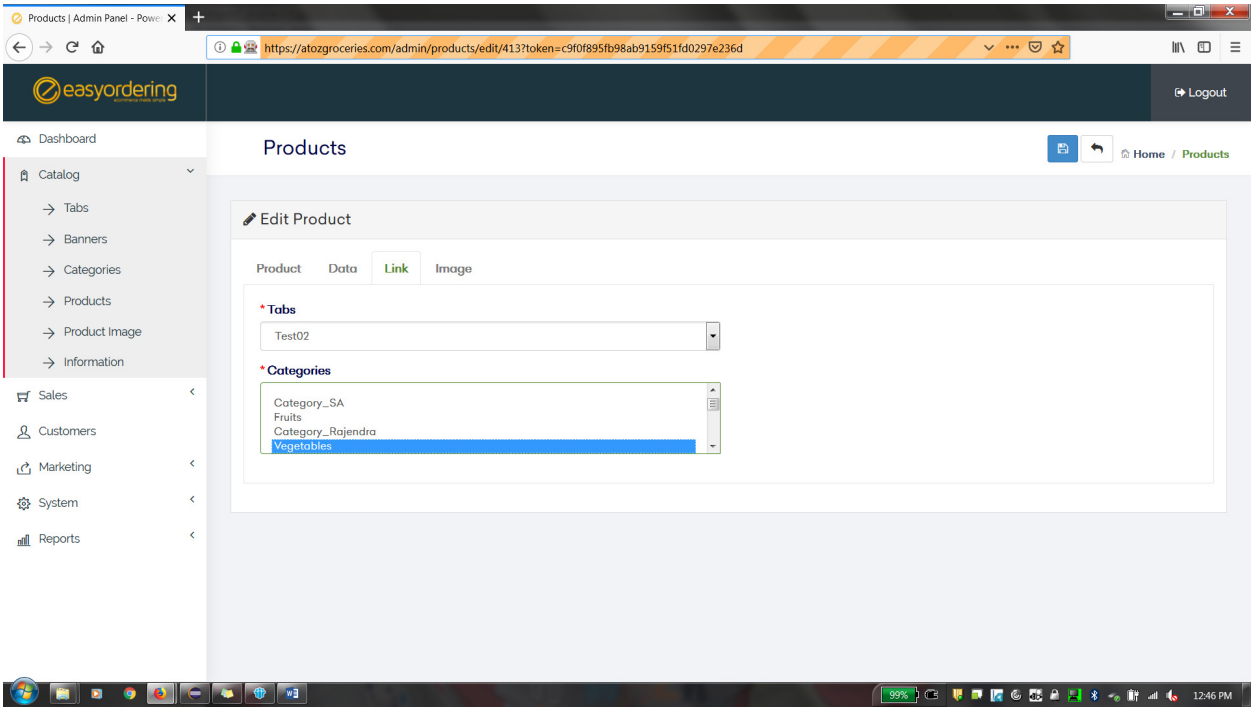




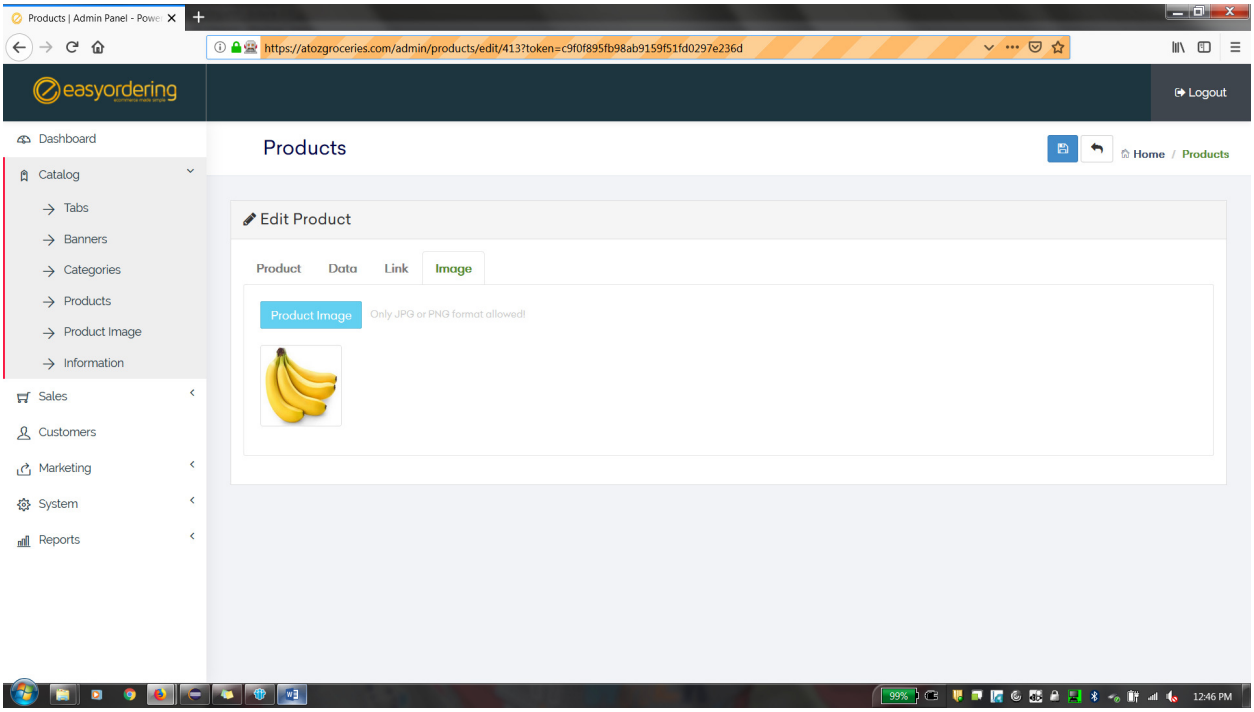
Opening link tab



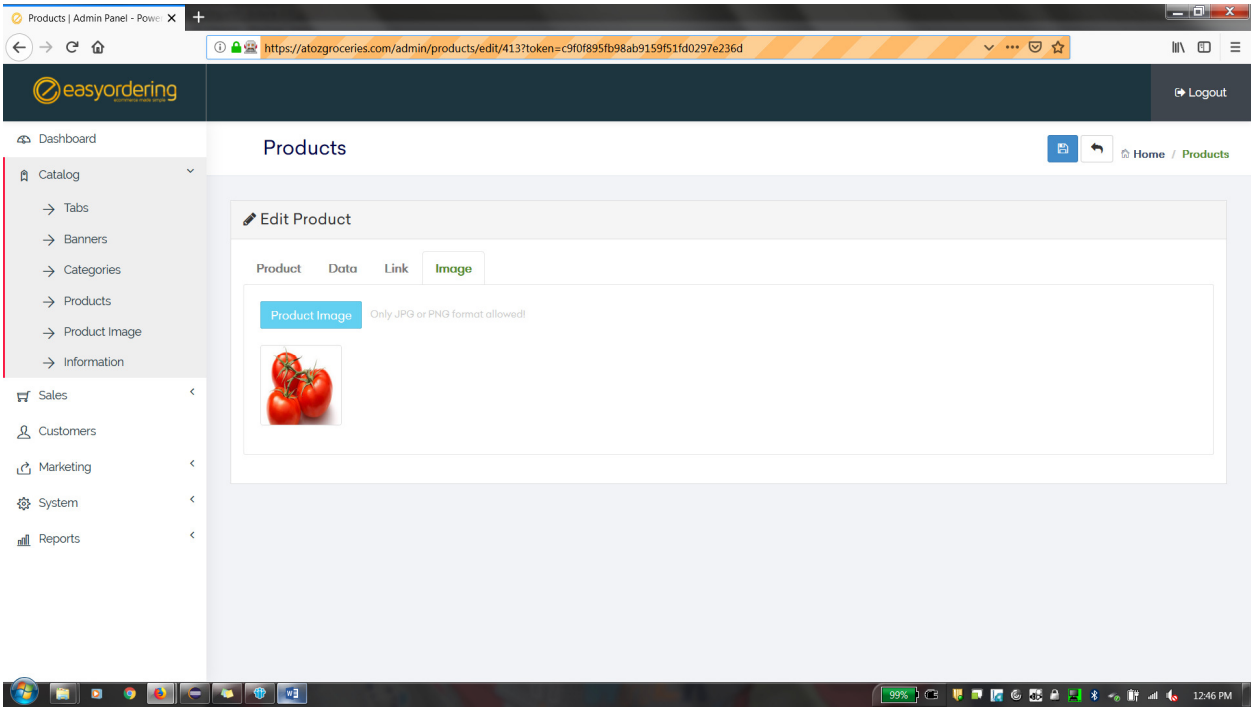
Providing the new data for the the product in link tab



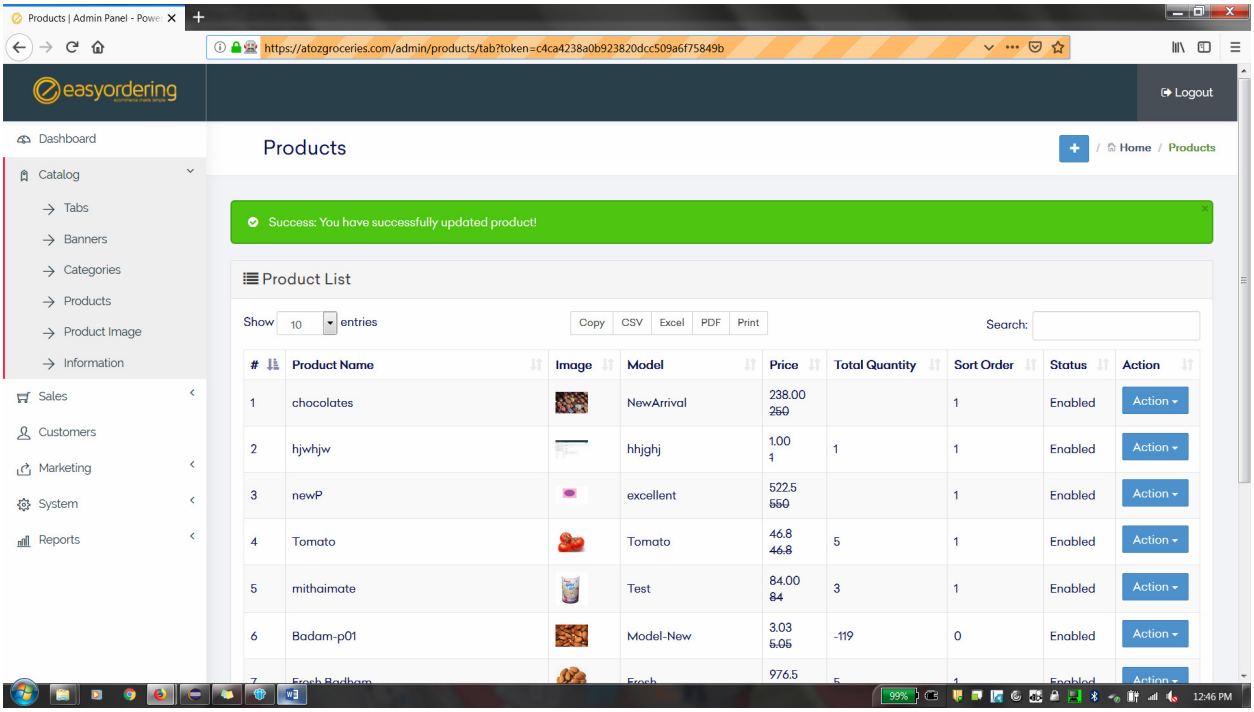
Opening Image tab



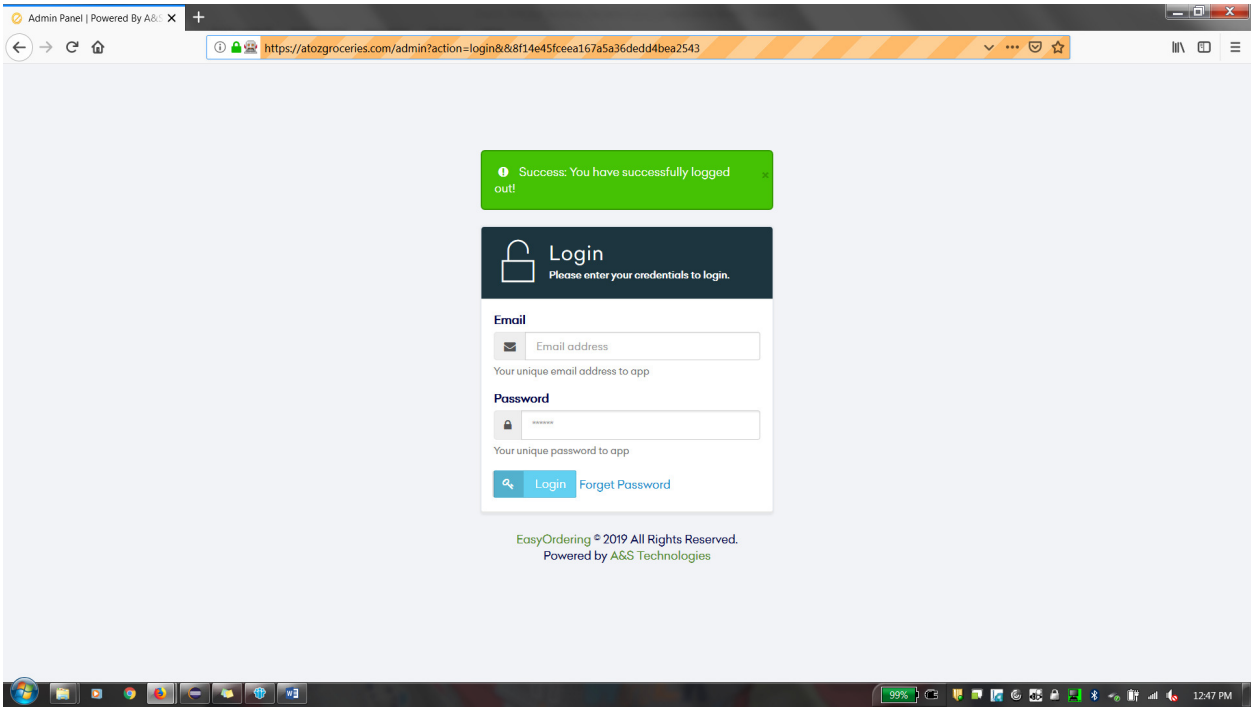
Sending the new image for the product and clicking on “save” to save the product



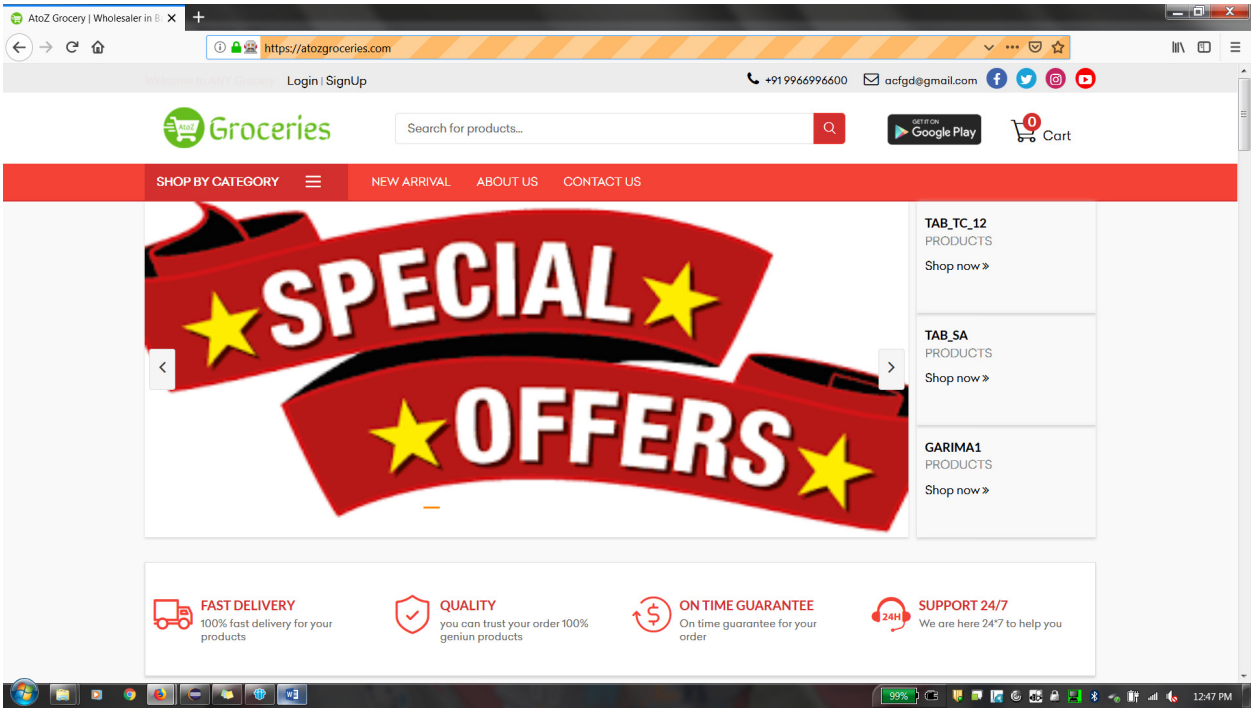
Validating the Success message and updated product “Tomato” with quantity 5

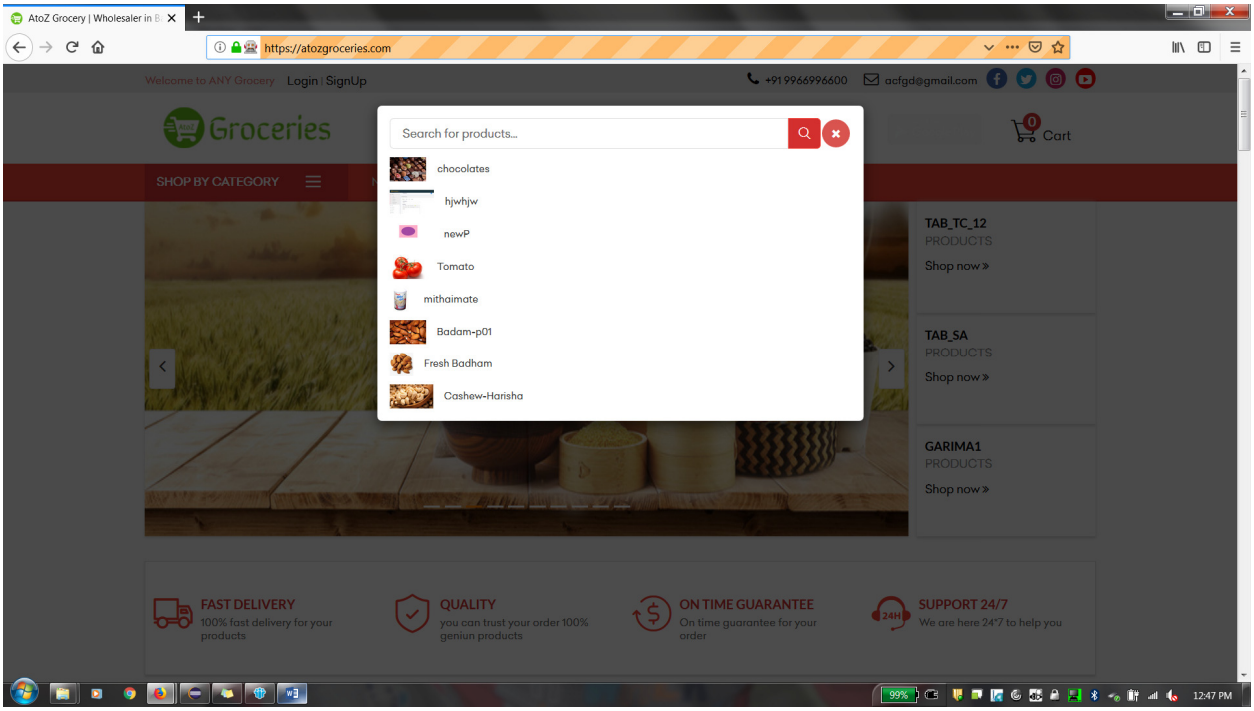


Logging out from admin portal

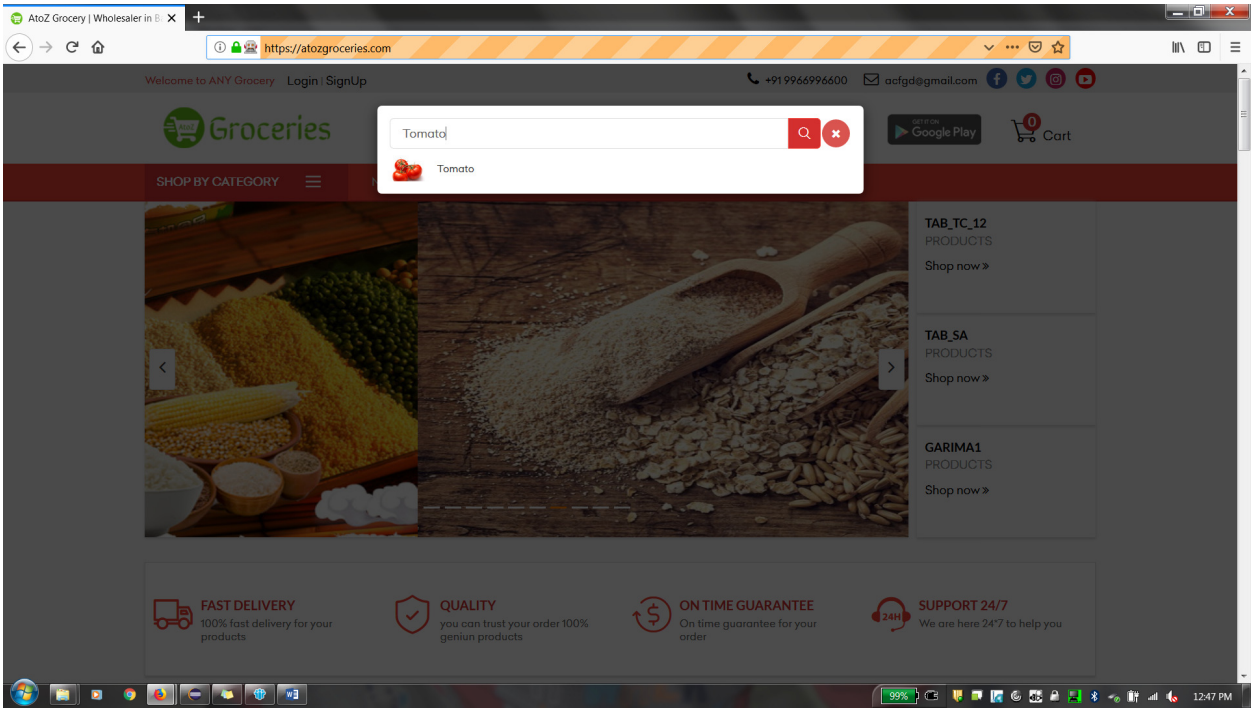


Opening the User portal and clicking on Search for products..

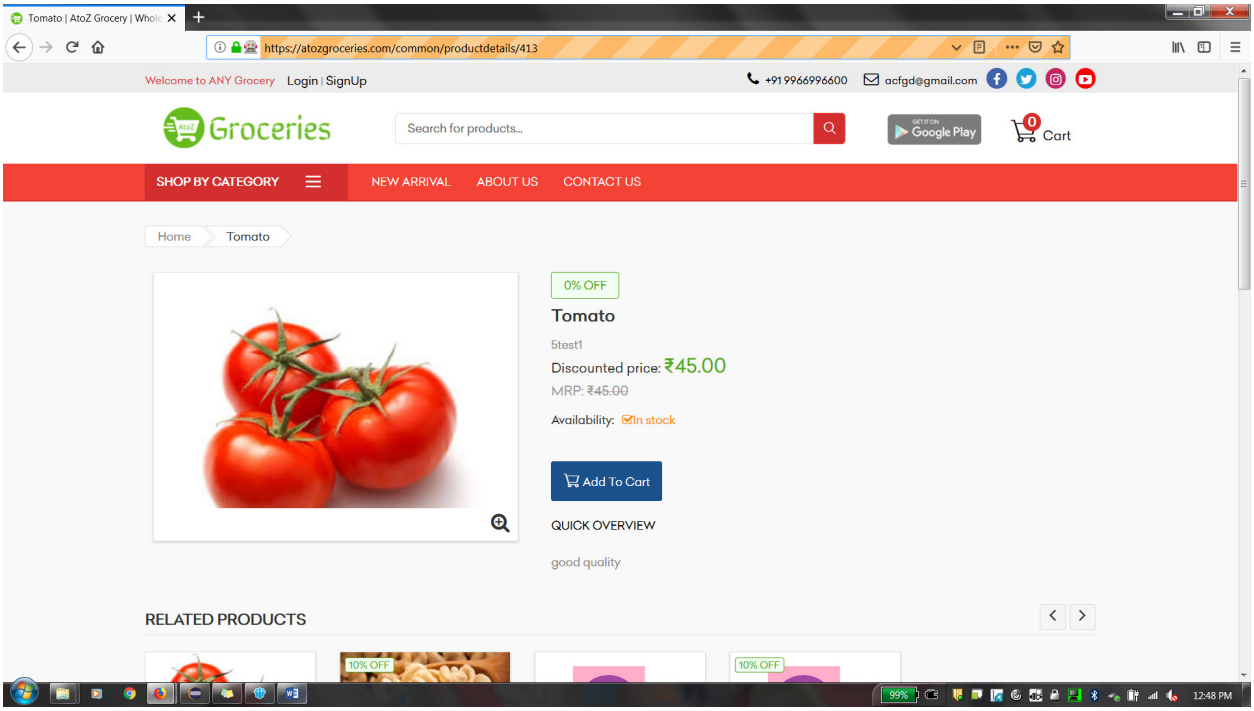




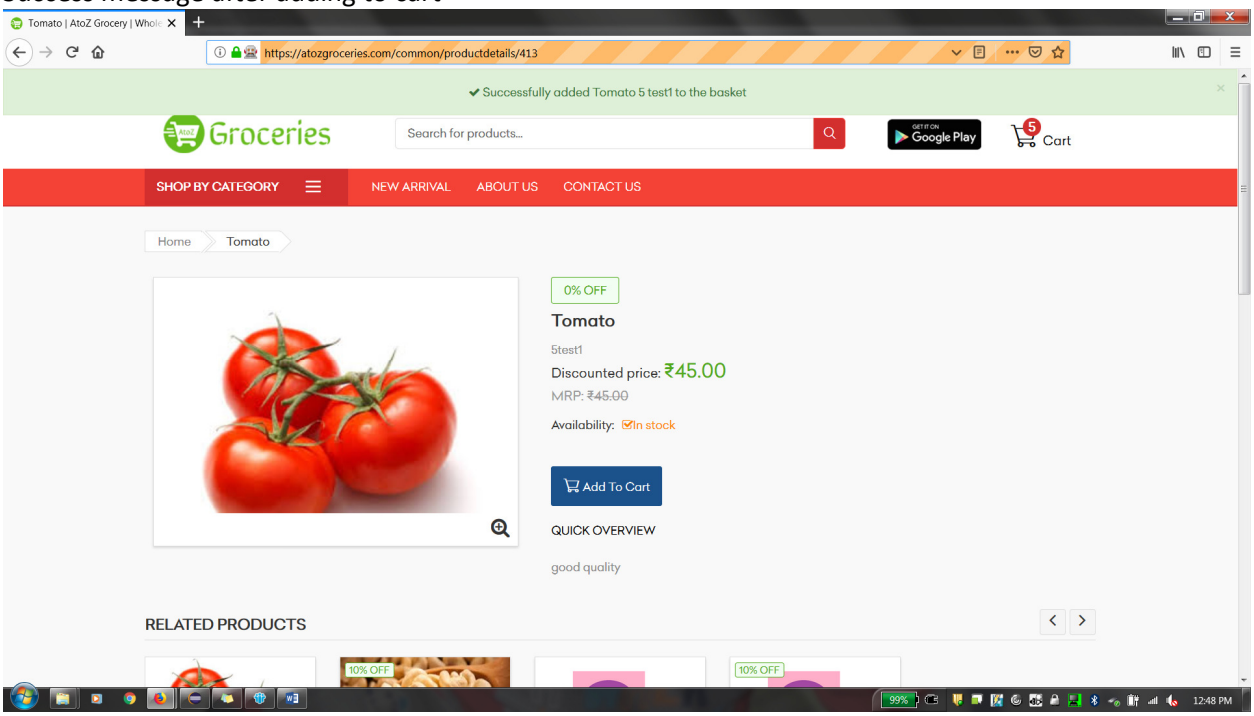
Sending the updated product as input for validating the quantity



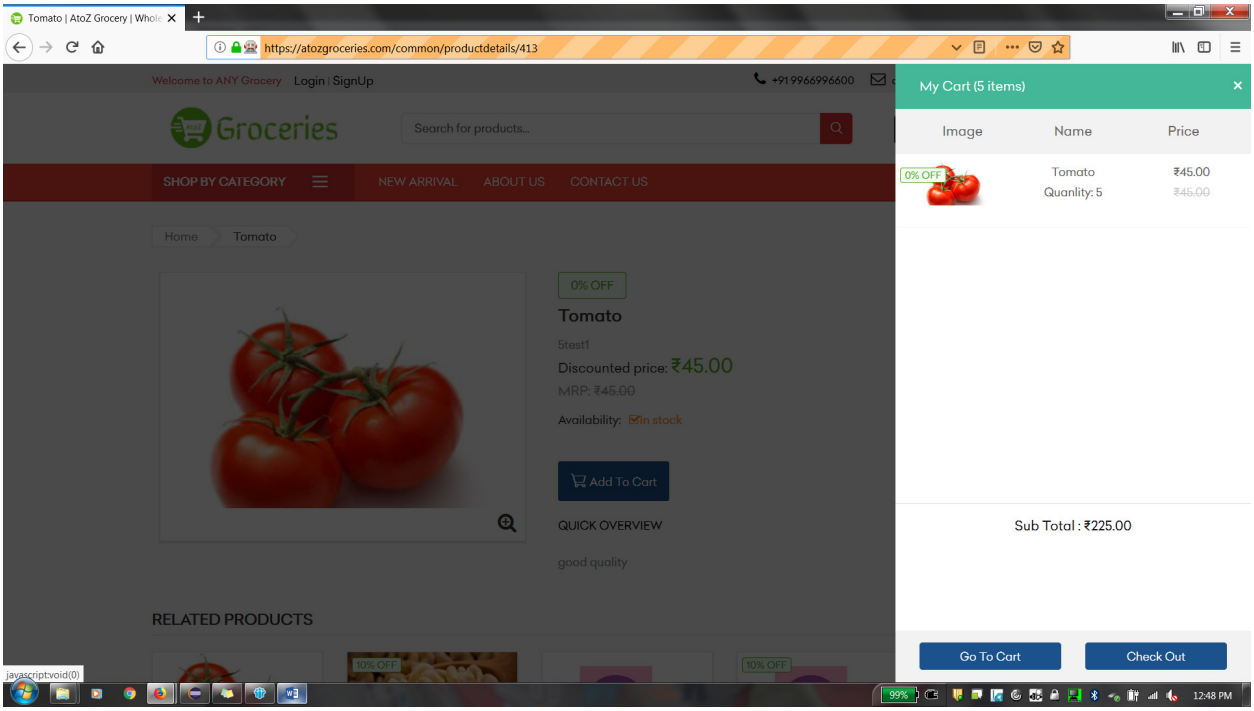
Opening the product and adding to cart



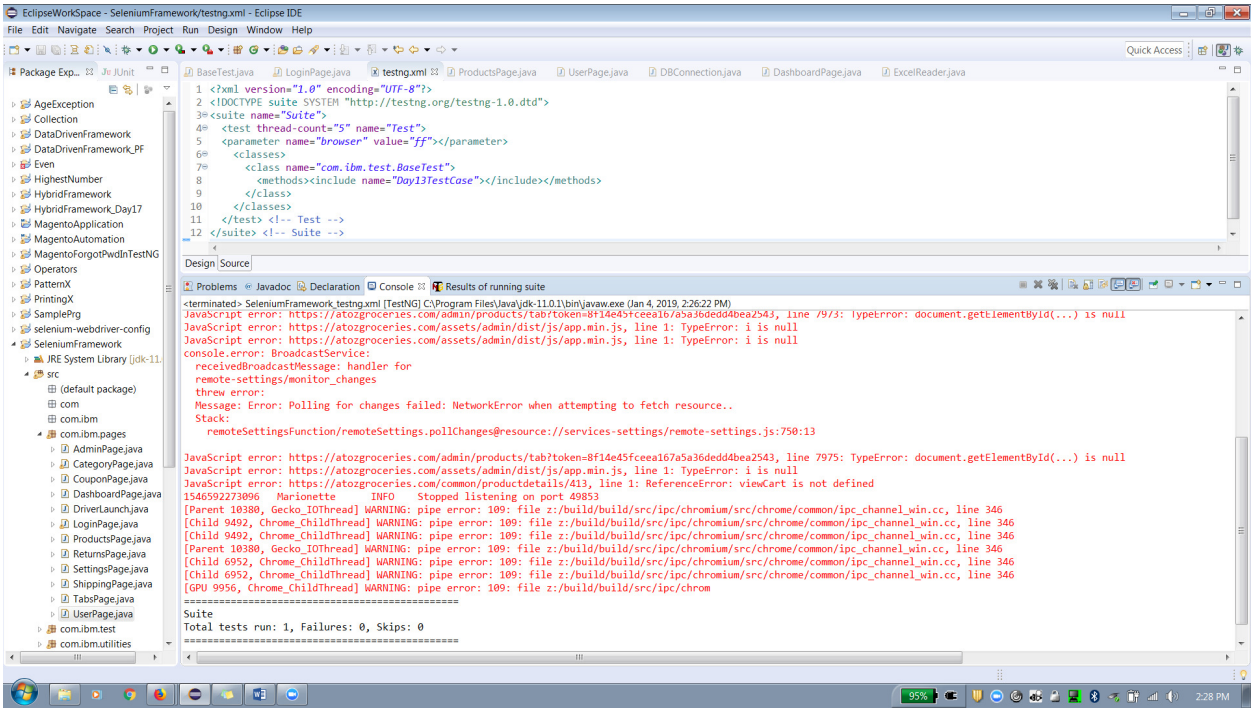
Success message after adding to cart



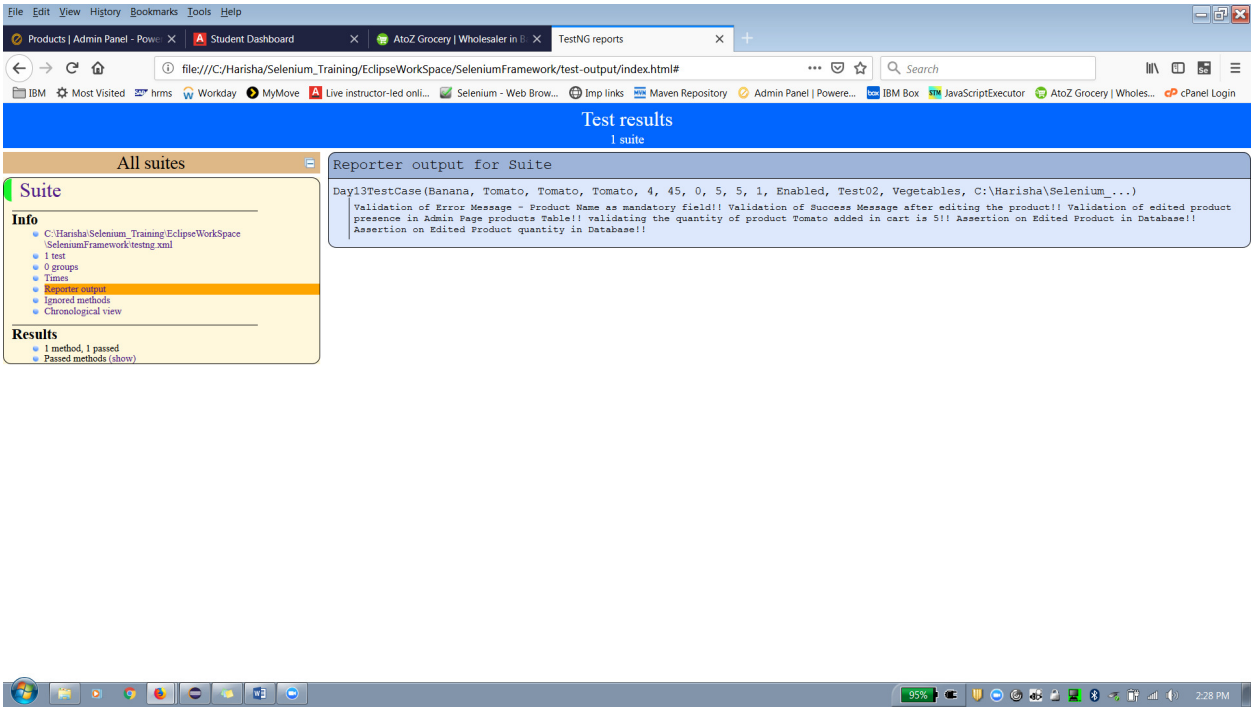
Opening cart and validating the quantity of the product



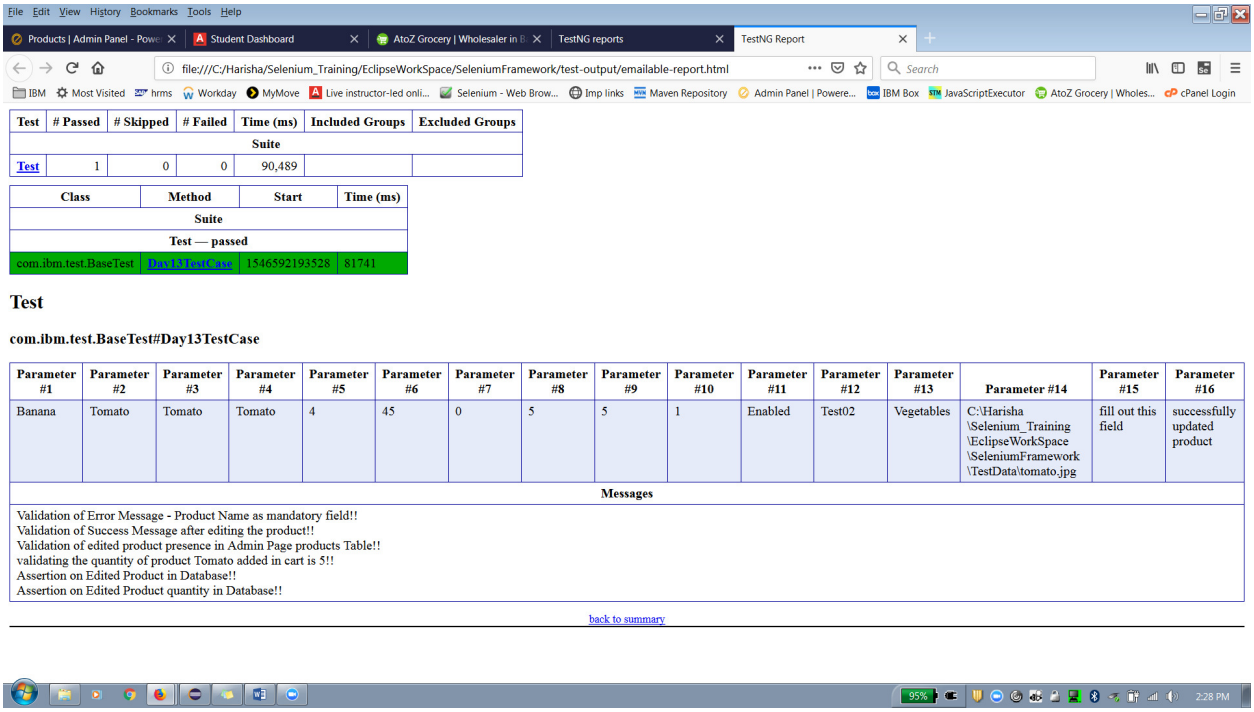
Output from console



Automated report of index.html



Automated report of Emailable 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	81741

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 *****