Test Objective:

The objective is to automate the below testcase using selenium WebDriver and framework concepts.

Testcase: Tc13

Process: Sign in and Fill address

Steps to Execute:

1.Sign in on user page: https://atozgroceries.com

2.Go to MyAccount

3.Go to My Address

4.Without filling the address details

5.Update the details and verify the error message

6.Fill the address details.

7.Update the details

Expected Results:

1.Validate the error message

2.Validate the success message

3.Validate the presence of updated address on the database under my account in the user page

4.Verify the presence of updated address on the database.

Source code:

File1: `UpdateAddress.java`

```
package com.ibm.groceries;

import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import org.openqa.selenium.By;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.testng.Assert;
import org.testng.Reporter;
import org.testng.annotations.Test;

import com.ibm.groceriespages.AccountPage;
import com.ibm.groceriespages.GroceriesUserPage;
import com.ibm.groceriespages.MyAddressPage;
import com.ibm.initialization.WebDriverLaunch;
import com.ibm.utilities.DatabaseConnection;
import com.ibm.utilities.GetScreenshot;
```

```
public class UpdateAddress extends WebDriverLaunch {

	@Test(priority = 1, testName = "UpdateAddress", groups = "low")
	public void updateAddress() throws IOException, InterruptedException,
SQLException {
		String userPage = data.get("userPageUrl");
		String phoneNum = data.get("phoneNum");
		String password = data.get("pwd");
		String expErrorMessage = data.get("expErrorMessage");
		String fullname = data.get("fullnamevalue");
		String mailid = data.get("mailidvalue");
		String address = data.get("addressvalue");
		String pincode = data.get("pincodevalue");
		String cityName = data.get("cityName");
		String placeName = data.get("placeName");

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

		// To launch Groceries user page
		driver.get(userPage);
		GetScreenshot screen = new GetScreenshot();
		screen.takeScreenshot(driver);

		GroceriesUserPage userpage = new GroceriesUserPage(driver, wait);
		// Calling method to click on Login link on uer portal
		userpage.clickOnLogin();

		// Calling method to sign in
		userpage.singIn(phoneNum, password);


	wait.until(ExpectedConditions.presenceOfElementLocated(By.partialLinkText("My
Account")));
		screen.takeScreenshot(driver);

		AccountPage accountObj = new AccountPage(driver, wait);
		// calling method to click on My address link
		accountObj.clickOMyAccount();

		// verifying full name text box is displayed

	Assert.assertTrue(driver.findElement(By.xpath("//input[@id='name']")).isDispla
yed());
		MyAddressPage addressObj = new MyAddressPage(driver, wait);

		// calling method to update with out filling address
		String actualtooltipMessage = addressObj.fillAddressEmpty();
		screen.takeScreenshot(driver);

		// Verifying the expected tooltip error message displayed
		Assert.assertEquals(actualtooltipMessage, expErrorMessage);
		System.out.println(actualtooltipMessage);

		// Calling method to enter address details
```

```
            String pageSource = addressObj.enterAddress(fullname, mailid, address,
pincode, cityName, placeName);
            Thread.sleep(2000);
            Assert.assertTrue(pageSource.contains(expAddressUpdateMessage));
            System.out.println(expAddressUpdateMessage);
            Reporter.log(expAddressUpdateMessage);

            // Calling method to click on My account->My Address
            addressObj.clickOnMyAddress();
            screen.takeScreenshot(driver);

            Thread.sleep(2000);
            // Verifying the updated address details
            Assert.assertEquals(addressObj.getFullName(), fullname);
            Assert.assertEquals(addressObj.getEmail(), mailid);
            Assert.assertEquals(addressObj.getAddress(), address);
            Assert.assertEquals(addressObj.getPinCode(), pincode);
            Assert.assertEquals(addressObj.getCity(), cityName);
            Assert.assertEquals(addressObj.getPlace(), placeName);

            DatabaseConnection conn = new DatabaseConnection();
            Statement st = conn.connectDatabase();
            // Verifying the address details in table
            ResultSet rs = st.executeQuery("select *from as_customer where name=" +
"'"+fullname+"'");
            if (rs.next()) {
                  Assert.assertEquals(rs.getString("name"), fullname);
                  Assert.assertEquals(rs.getString("email"), mailid);
                  Assert.assertEquals(rs.getString("address"), address);
                  Assert.assertEquals(rs.getString("city"), cityName);
                  Assert.assertEquals(rs.getString("pincode"), pincode);
                  Assert.assertEquals(rs.getString("type"), placeName);
            }

      }
}
```

File2: GroceriesUserPage.java

```
package com.ibm.groceriespages;


import java.io.IOException;

import org.openqa.selenium.Alert;

import org.openqa.selenium.By;

import org.openqa.selenium.Keys;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;
```

```
import org.openqa.selenium.interactions.Actions;
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;
import com.ibm.utilities.GetScreenshot;

public class GroceriesUserPage {

	@FindBy(xpath = "//input[@placeholder='Search for products...']")
	WebElement searchEle;

	@FindBy(xpath = "(//div[@class='input-group']/descendant::input[1])[2]")
	WebElement searchEle2;

	@FindBy(xpath = "//div[@id='searchproducts-div']/descendant::a[1]")
	WebElement productNewLink;

	// To locate email link on user page
	@FindBy(xpath = "(//div[@class='header-mid-right-content']/descendant::a[1])[2]")
	WebElement emailLink;

	// To locate phone link on user page
	@FindBy(xpath = "(//div[@class='header-mid-right-content']/descendant::a[1])[1]")
	WebElement phoneLink;

	// To locate SignUp link
	@FindBy(xpath = "//a[text()='SignUp']")
	WebElement signupEle;
```

```
// To locate fullname text box
@FindBy(xpath = "//input[@id='name']")
WebElement fullnameEle;

// To locate phonnumber text box on signupuser page
@FindBy(xpath = "//input[@id='pnum']")
WebElement phonenumEle;

// To locate password element on signup user page
@FindBy(xpath = "//input[@id='password']")
WebElement passwordEle;

// To locate confirm password text box
@FindBy(xpath = "//input[@id='cpassword']")
WebElement confirmpasswordEle;

// TO locate agree terms check box
@FindBy(xpath = "//input[@id='tccheckbox']")
WebElement agreeCheckEle;

// To locate sign up button
@FindBy(xpath = "//button[@id='mem_signup']")
WebElement signupButton;

// To locate addto cart link
@FindBy(xpath = "//a[@id='addtocart_cartbtn336']")
WebElement addcartEle;
```

```
// To locate addto cart link for placing order
@FindBy(xpath = "//a[@id='addtocart_cartbtn403']")
WebElement addcartEle2;

// To locate Cart icon link
// @FindBy(xpath="//a[text()=' Cart']")
// WebElement cartEle;

@FindBy(xpath = "//div[@class='header-bottom-right']/descendant::a[1]")
WebElement cartEle;

// To locate Go to Cart icon link
@FindBy(xpath = "//a[text()='Go To Cart']")
WebElement gotocartEle;

// To locate Check Out button
@FindBy(xpath = "//a[text()='Check Out']")
WebElement checkoutEle;

// To locate Login link button
@FindBy(xpath = "//a[text()='Login']")
WebElement loginLink;

//To locate phone number2 on sign in user page
@FindBy(xpath="//input[@id='pnum2']")
WebElement phonenumberEle;

// To locate password element on sign in user page
@FindBy(xpath = "//input[@id='pword2']")
```

```
WebElement pwordEle;

// To locate login button on sign in user page
        @FindBy(xpath = "//button[@id='mem_login']")
        WebElement loginButton;



WebDriverWait wait;
WebDriver driver;

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

public String searchProduct(String proudctName) throws InterruptedException {

        searchEle.sendKeys(proudctName);
        searchEle.click();

        // To enter product name in search text box of popup
        searchEle2.sendKeys(proudctName);
        Thread.sleep(10000);
        productNewLink.click();

        return driver.getPageSource();

}
```

```
// To verify the new address on user page
public String verifyAddress() {
	return driver.getPageSource();
}

// To verify the new email link is on user page
public boolean verifyEmail(String newemail) {

	if (emailLink.getText().contains(newemail))

		return true;
	else
		return false;
}

// To verify the new phone number is user page
public boolean verifyPhone(String newphone) {

	if (phoneLink.getText().contains(newphone))

		return true;
	else
		return false;

}

public void signUp(String fullName, String phoneNum, String password, String confirmPassword)
		throws InterruptedException {
```

```
        // To click signUp link
        signupEle.click();
        // To enter the value for full name
        fullnameEle.sendKeys(fullName);
        // To enter the value for phone number
        phonenumEle.sendKeys(phoneNum);
        // To enter the value for password
        passwordEle.sendKeys(password);
        // To enter the value for confirm password
        confirmpasswordEle.sendKeys(confirmPassword);

        // To click on check box for agree terms
        agreeCheckEle.click();

        // To click on SignUp button
        signupButton.click();

        // To click on Ok button of Alert box after signup
        Alert alertBox = driver.switchTo().alert();
        String text = alertBox.getText();
        Thread.sleep(2000);
        alertBox.accept();

}

public void addProductToCart() throws InterruptedException, IOException {
        GetScreenshot screenObj = new GetScreenshot();
        // To scroll down using key down
```

```
		Actions actions = new Actions(driver);
		for (int i = 1; i <= 4; i++)
			actions.sendKeys(Keys.ARROW_DOWN).build().perform();

		// To click on Add to Cart link
		addcartEle.click();

		// To click on Cart link at top right corner
		cartEle.click();

		screenObj.takeScreenshot(driver);

		// To click on Got TO Cart link
		gotocartEle.click();

		screenObj.takeScreenshot(driver);
}

public String gotoCheckOut() throws IOException, InterruptedException {
		GetScreenshot screenObj = new GetScreenshot();

		// To scroll down using key down
		Actions actions = new Actions(driver);
		for (int i = 1; i <= 4; i++)
			actions.sendKeys(Keys.ARROW_DOWN).build().perform();

		// To click on Add to Cart link
		addcartEle2.click();
		// To click on Cart link at top right corner
```

```
		cartEle.click();
		screenObj.takeScreenshot(driver);
		wait.until(ExpectedConditions.presenceOfElementLocated(By.linkText("Check Out")));
		// To click on Check Out button
		checkoutEle.click();
		screenObj.takeScreenshot(driver);
		return driver.getPageSource();
	}

	//To click on login link
	public void clickOnLogin()
	{
		loginLink.click();
	}

	//Method to signin from user portal
	public void singIn(String phoneNum,String password) throws IOException, InterruptedException
	{
		GetScreenshot screenObj = new GetScreenshot();
		phonenumberEle.sendKeys(phoneNum);
		pwordEle.sendKeys(password);
		screenObj.takeScreenshot(driver);
		loginButton.click();
	}

}
```

File3:AccountPage.java

```
package com.ibm.groceriespages;
```

```
import java.io.IOException;

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.WebDriverWait;

import com.ibm.utilities.GetScreenshot;

public class AccountPage {

	@FindBy(xpath = "//a[contains(text(),'My Address')]")
	WebElement myaddressEle;

	WebDriver driver;
	WebDriverWait wait;

	@FindBy(linkText = "Log Out")
	WebElement logOutEle;

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

```
        // Method to click on My address link
        public void clickOMyAccount() throws IOException, InterruptedException {

                driver.findElement(By.partialLinkText("My Account")).click();
                GetScreenshot screen = new GetScreenshot();
                screen.takeScreenshot(driver);
                myaddressEle.click();
        }

}
```

File4:MyAddressPage.java

```
package com.ibm.groceriespages;

import java.io.IOException;

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.Select;
import org.openqa.selenium.support.ui.WebDriverWait;

import com.ibm.utilities.GetScreenshot;

public class MyAddressPage {

        // To locate full name
        @FindBy(xpath = "//input[@id='name']")
        WebElement fullnameEle;

        // To locate email
        @FindBy(xpath = "//input[@id='email']")
        WebElement emailEle;

        // To locate address
        @FindBy(xpath = "//textarea[@id='address']")
        WebElement addressEle;

        // To locate city drop down
        @FindBy(xpath = "//select[@id='city']")
        WebElement cityEle;
```

```
	// To locate pin code
	@FindBy(xpath = "//input[@id='pincode']")
	WebElement pincodeEle;

	// To locate home or office drop down
	@FindBy(xpath = "//select[@id='type']")
	WebElement placeEle;

	// To locate Update button
	@FindBy(xpath = "//button[@class='button button-check-out']")
	WebElement updatEle;

	WebDriverWait wait;
	WebDriver driver;

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

	// To update with out filling address
	public String fillAddressEmpty() throws IOException, InterruptedException {
		fullnameEle.clear();
		emailEle.clear();
		addressEle.clear();

		// To select city as blank
		Select citySelect = new Select(cityEle);
		citySelect.selectByIndex(0);
		pincodeEle.clear();

		GetScreenshot screen = new GetScreenshot();
		screen.takeScreenshot(driver);

		// To click on update buttton
		updatEle.click();

		JavascriptExecutor js = (JavascriptExecutor) driver;
		String tooltipMessage = js.executeScript("return
document.getElementsByName('name')[0].validationMessage;")
				.toString();
		return tooltipMessage;

	}

	public String enterAddress(String fullname, String mailid, String address,
String pincode, String cityName,
			String placeName) throws IOException, InterruptedException {
		fullnameEle.sendKeys(fullname);
		emailEle.sendKeys(mailid);
		addressEle.sendKeys(address);

		// To select city
```

```
		Select citySelect = new Select(cityEle);
		// citySelect.selectByIndex(10);
		citySelect.selectByVisibleText(cityName);
		// To enter pin code
		pincodeEle.sendKeys(pincode);

		// To select place
		Select placeSelect = new Select(placeEle);
		// placeSelect.selectByIndex(0);
		placeSelect.selectByVisibleText(placeName);
		GetScreenshot screen = new GetScreenshot();
		screen.takeScreenshot(driver);

		// To click on continue to payment button
		updatEle.click();
		return driver.getPageSource();
	}

	// to click on My address link
	public void clickOnMyAddress() {
		driver.findElement(By.partialLinkText("My Account")).click();

		driver.findElement(By.xpath("//a[contains(text(),'My
Address')]")).click();

	}

	// To get updated full name
	public String getFullName() {

		JavascriptExecutor js = (JavascriptExecutor) driver;
		String name = js.executeScript("return
document.getElementsByName('name')[0].value;").toString();
		return name;
	}

	// TO get updated email
	public String getEmail() {
		JavascriptExecutor js = (JavascriptExecutor) driver;
		String email = js.executeScript("return
document.getElementsByName('email')[0].value;").toString();
		return email;
	}

	// TO get updated address
	public String getAddress() {
		JavascriptExecutor js = (JavascriptExecutor) driver;
		String address = js.executeScript("return
document.getElementsByName('address')[0].value;").toString();
		return address;
	}

	// To Get updated City
	public String getCity() {
		// To get city
```

```
		JavascriptExecutor js = (JavascriptExecutor) driver;
		String city = js.executeScript("return
document.getElementsByName('city')[0].value;").toString();
		return city;

	}

	// TO get updated pin code
	public String getPinCode() {
		JavascriptExecutor js = (JavascriptExecutor) driver;
		String pincode = js.executeScript("return
document.getElementsByName('pincode')[0].value;").toString();
		return pincode;

	}

	public String getPlace() {
		// To get place
		JavascriptExecutor js = (JavascriptExecutor) driver;
		String place = js.executeScript("return
document.getElementsByName('type')[0].value;").toString();
		return place;

	}
}
```

File5:DatabaseConnection.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;


public class DatabaseConnection {


	public Statement connectDatabase()throws SQLException

	{
```

```
	Connection c=DriverManager.getConnection("jdbc:mysql://foodsonfinger.com:3306/foodsonfinger_atozgroceries","foodsonfinger_atoz","welcome@123");
	Statement stmt=c.createStatement();


	return stmt;
	}



	public int countRecords(String query)throws SQLException
	{
		int count=0;
		Connection c=DriverManager.getConnection("jdbc:mysql://foodsonfinger.com:3306/foodsonfinger_atozgroceries","foodsonfinger_atoz","welcome@123");
		Statement st=c.createStatement();
		ResultSet rs=st.executeQuery(query);


		if(rs.next())
		{
		count=rs.getInt(1);
		}
		return count;
	}


}
```

File6:GetScreenshot.java

```
package com.ibm.utilities;

import java.io.File;
import java.io.IOException;
import java.util.Date;

import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;

public class GetScreenshot {

	public void takeScreenshot(WebDriver driver)throws IOException, InterruptedException
	{
		Thread.sleep(2000);
		TakesScreenshot ts=(TakesScreenshot)driver;
		File file=ts.getScreenshotAs(OutputType.FILE);
		Date date=new Date();
		String currentDate=date.toString().replaceAll(":", "-");
		FileUtils.copyFile(file,new File("./screenshots/Error_"+currentDate+".png"));



	}

}
```

File7: PropertiesFileHandler.java

```
package com.ibm.utilities;
import java.io.File;
```

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
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;
	}

	public void setKeyAndValue(String file,String key,String value) throws IOException
	{
		FileInputStream fileIn = new FileInputStream(file);
		Properties prop = new Properties();
		prop.load(fileIn);
```

```
                prop.setProperty(key, value);

                FileOutputStream fOut=new FileOutputStream(file);
                prop.store(fOut, "Test Result");
                fOut.close();
                fileIn.close();
        }

}
```

File8:

WebDriverLaunch.java

```
package com.ibm.initialization;
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.BeforeTest;
import org.testng.annotations.Optional;
import org.testng.annotations.Parameters;
import com.ibm.utilities.PropertiesFileHandler;
```

```
public class WebDriverLaunch {
	public WebDriver driver;
	public WebDriverWait wait;
	public PropertiesFileHandler propFileHandler;
	public HashMap<String, String> data;

	//Getting the keys from properties file
	//@BeforeSuite
	@BeforeSuite(groups= {"high","low"})
		public void preSetForTest() throws IOException {
		String file = "./TestData/groceries.properties";
		propFileHandler = new PropertiesFileHandler();
		data = propFileHandler.getPropertiesAsMap(file);
	}


	//@BeforeTest

	@BeforeMethod(groups= {"high","low"})
	@Parameters({"browser"})
	public void Initialization(@Optional("ff")String browser) {
		browserInitialization(browser);
		wait = new WebDriverWait(driver, 60);
		driver.manage().window().maximize();
		driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
	}


	//@AfterMethod
```

```
	//Closing driver
	@AfterMethod(groups= {"high","low"})
	public void closeBrowser() {
		driver.quit();
	}

	//Setting path for webdriver
	public void browserInitialization(String browser)
	{
		switch (browser.toLowerCase()) {
		case "ff":
		case "firefox":
			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;
		default:
			System.out.println("No browser Available "+browser);
			break;
		}
	}
}
```

File9:groceries.properties

url=https://atozgroceries.com/admin

username=demo@atozgroceries.com

password=456789

expectedMessage=Success: You have successfully deleted data!

notifyName=DemoNotification

notifyMessage=Adding Notification

expNotificationMessage=Success: You have successfully sent push notification to your app!

imagePath=C:\\Users\\IBM_ADMIN\\eclipse-workspace\\TestCases_DataDrivenFramework\\TestData\\globe.jpg

searchForKeyword=refund

searchDispalyMessage=Search with keyword is successful

noMatchDisplayMessage=No matching records found

searchForShipKeyword=Discount

prodNameNew=ProductNew100

modelNameNew=Model-New

expectedEditProductMessage=Success: You have successfully updated product!

productNotUpdated=Product update failed

userPageUrl=https://atozgroceries.com

userPageMsg=New Product found on User page

addressNew=Koramangala,Bengaluru

emailNew=info@atozgroceries.com

phoneNew=9797979797

expectedEditSettingMessage=successfully updated Store!

addressFoundMsg=new Address found on user page.

emailFoundMsg=new Email found on user page.

phoneFoundMsg=new Phone number found on user page.

fullName=srinivast

phoneNum=9701934935

passwd=dasara

confirmPassword=dasara

quantity=30

expectedShopingCartMsg=You have successfully updated cart items!

titleNameNew=Title-New

tagDescriptionNew=Tag-New

keywordNew=keyword-New

hsnNew=Hsn-New

tableProducts=as_products

productName=Badam-p01

modifiedProductConsole=Modified details of product from DataBase:

customerEmail=sree2018@gmail.com

subject=Add Email

message=Validate add email

emailMessage=Success: You have successfully sent mail to all customer!

emailMsg=Success: You have successfully sent mail to

beforeMsg=Email count before adding email:

afterMsg=Email count after adding email:

mailHeader=Mail List

sendmailPageTitle=Mail | Admin Panel - Powered By A&S

errorAddressMessage=To address is required!

emailTable=as_mail

shippingPincodeHeader=Shipping Pincode List

expltooltipMessage=Please enter a number.

expltooltipMessage2=Please fill out this field.

orderStatusHeader=Order Status List

newTabName=MyTabNew

editTabMessage=Success: You have successfully updated tab!

productName=Prod-Icecream

prodcutDesc=Vanila

productMetaTitle=Beverages

prodMetaTagDesc=MetagTagIce

prodMetaTagKeyword=MetaKeywordIce

model=model-ice

hsn=hsn-ice

gst=2

price=10

specialDiscount=2

priceafterspecialdiscount=6

discountQuantity=2

discountPrice=4

quantity=6

totalQuantity=15

updateTabMsg=Updated tab name is found

checkoutHeader=Please enter your Phone Number to Login/Sign up

text=abcde

invalidPhoneMsg=Invalid or Incorrect phone number!

phoneNum=9701934935

pwd=dasara

fullnamevalue=tsrinivas

mailidvalue=srinivas.nirmal@gmail.com

addressvalue=Madiwala area

pincodevalue=555788

orderMessage=Thank you! Your order has been placed successfully

expErrorMessage=Please fill out this field.

expAddressUpdateMessage=You have successfully updated address details!

cityName=Bangalore_JT7

placeName=Office

File10:testing_UpdateAddress.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="UpdatingAddress">
  <parameter name="browser" value="firefox"></parameter>
    <classes>
      <class name="com.ibm.groceries.UpdateAddress"/>
    </classes>
  </test> <!-- UpdatingAddress -->
</suite> <!-- Suite -->
```

Test Result:

1.Validated the error message

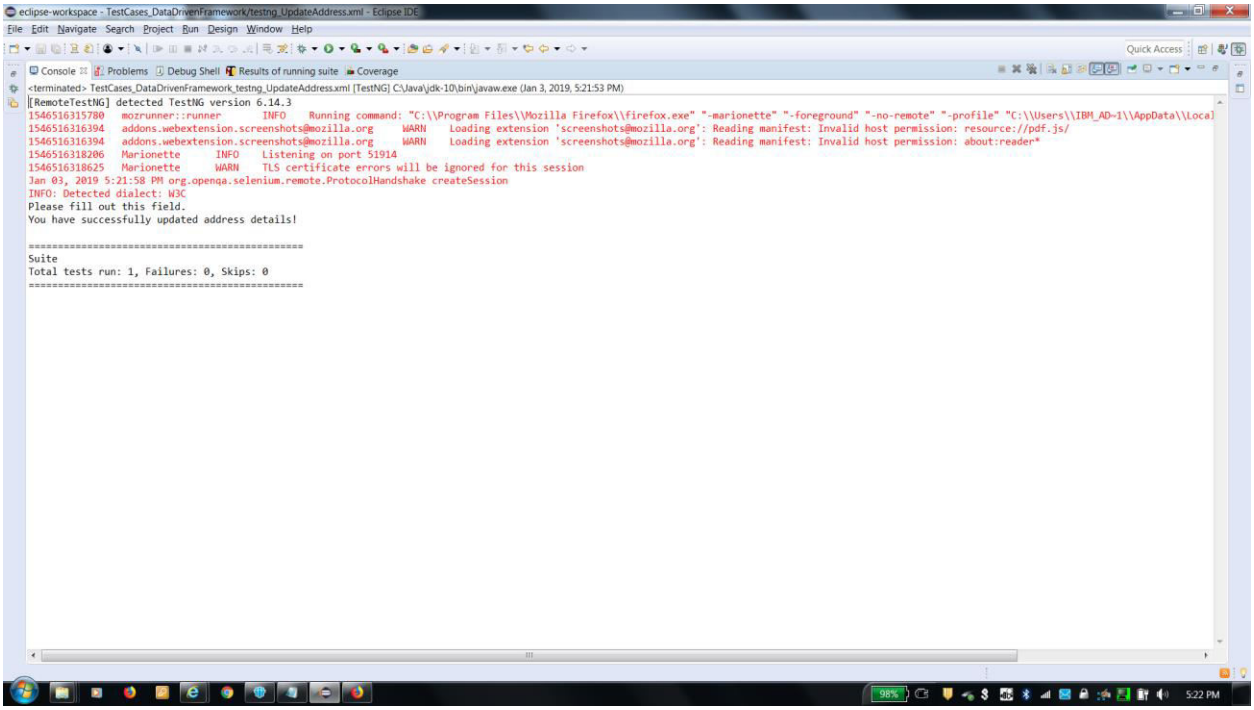
2.Validate the success message

3.Validated the presence of updated address on the database under my account in the user page

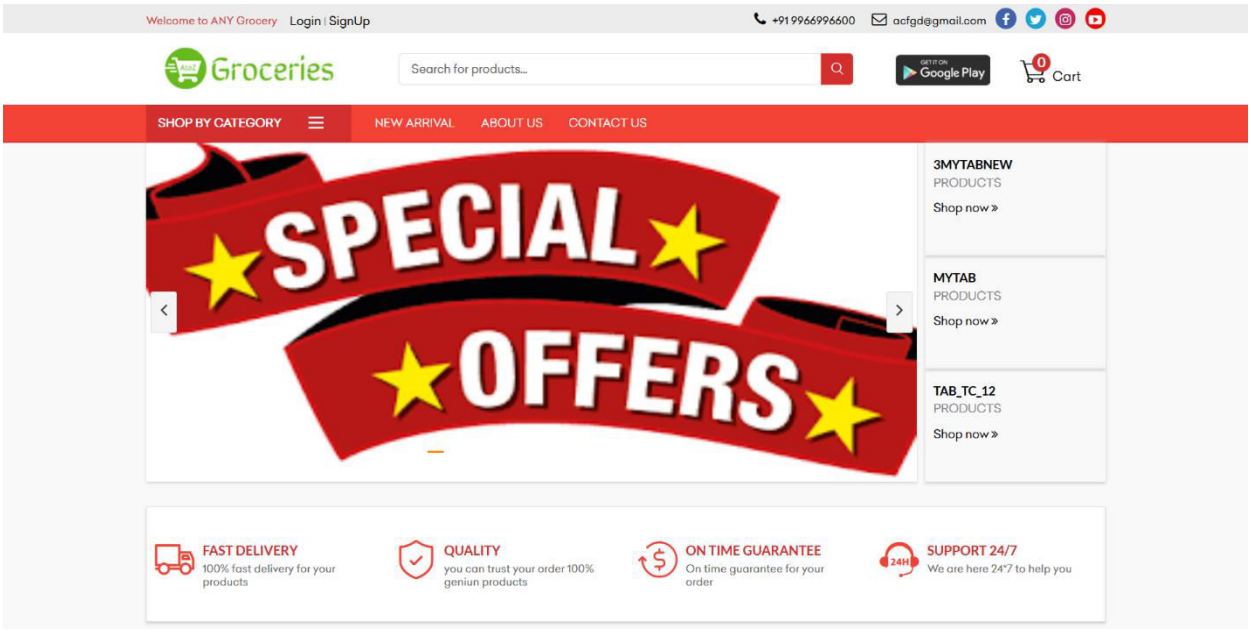
4.Verified the presence of updated address on the database.

Screenshots:

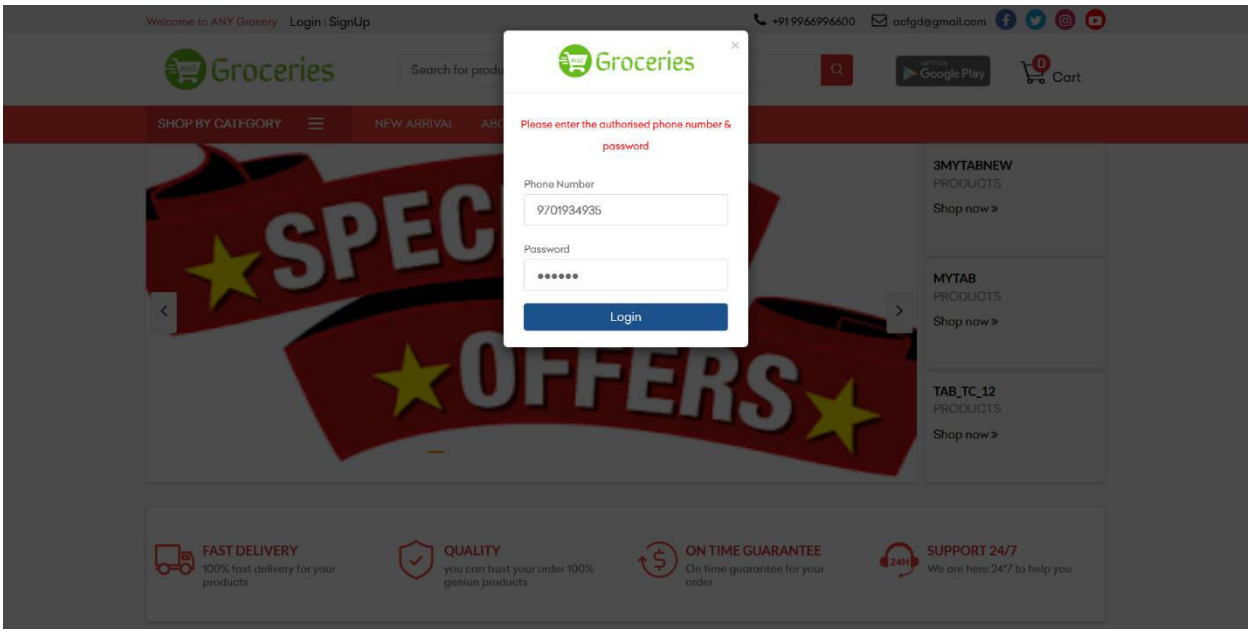
Console output:



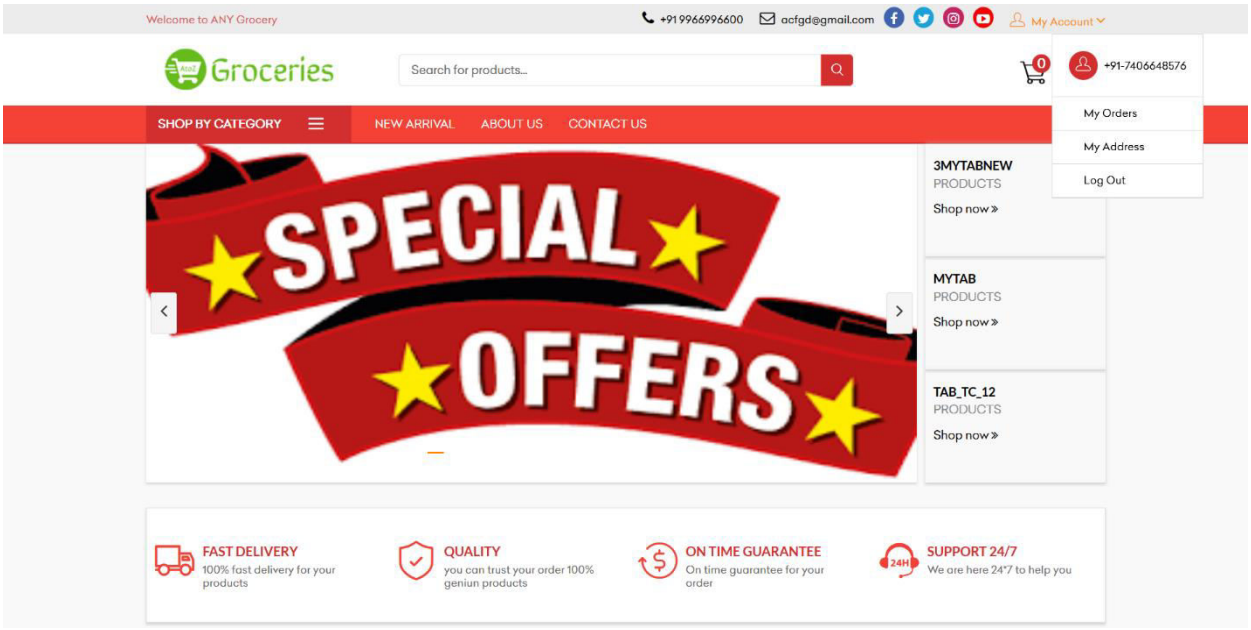
User portal is displayed



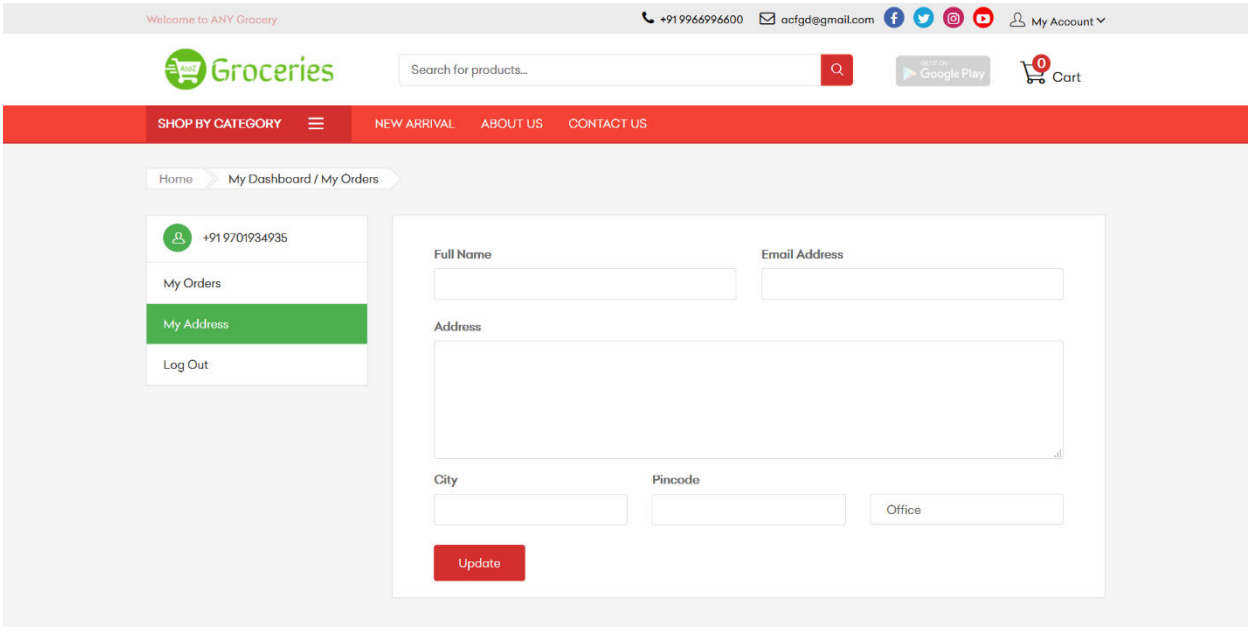
Sign In page is displayed



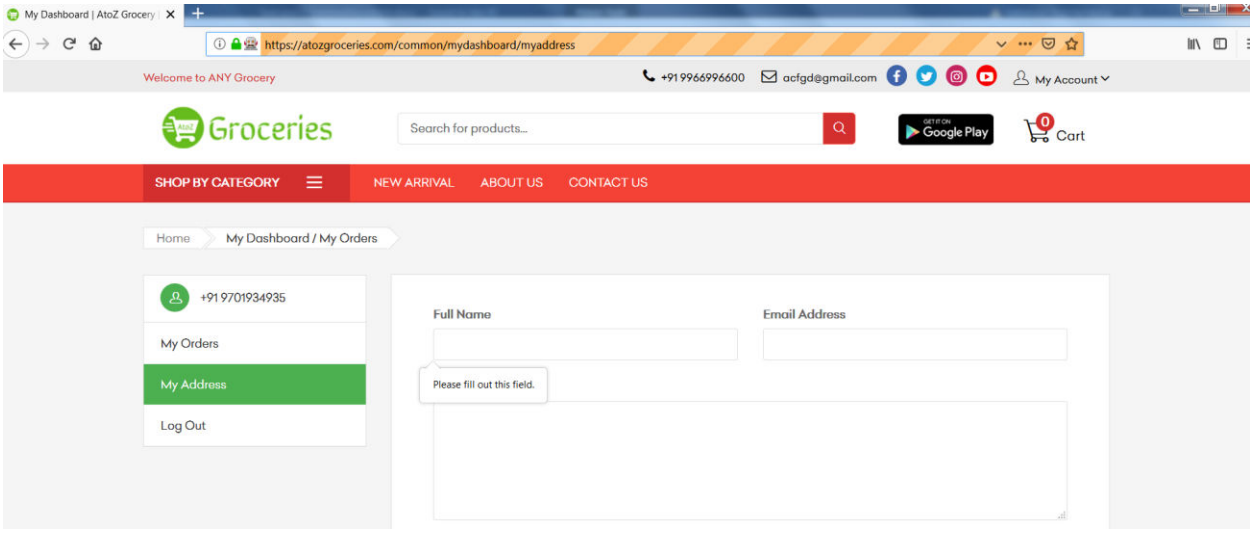
My account->My address is displayed



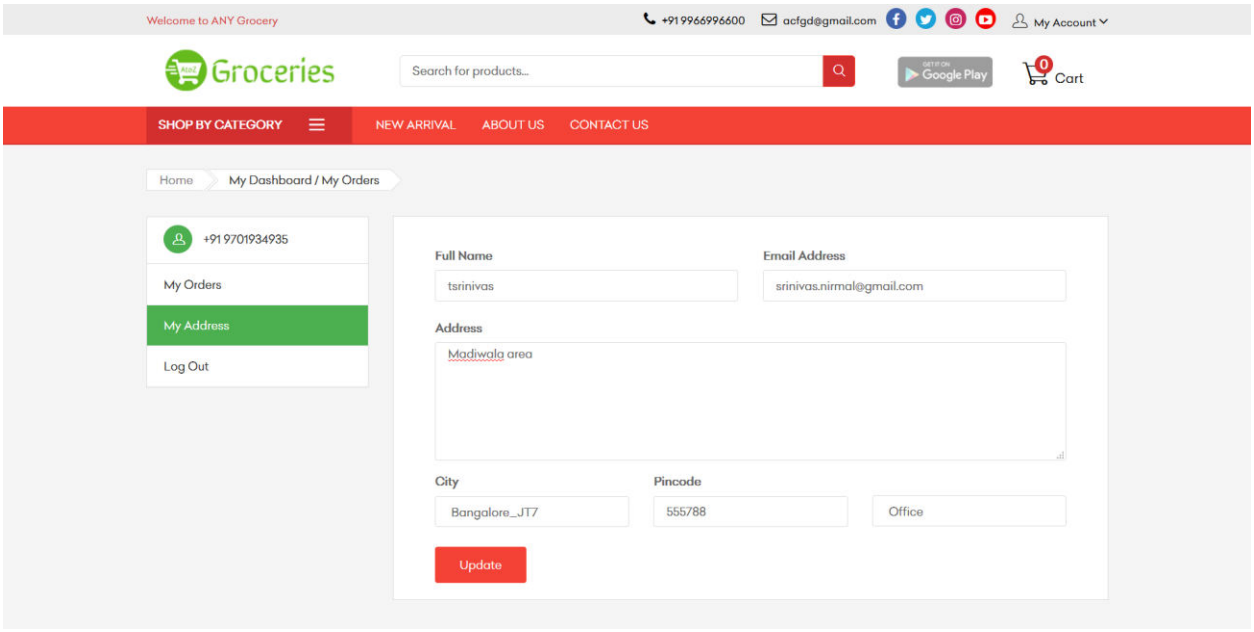
My Address page is displayed with out address details



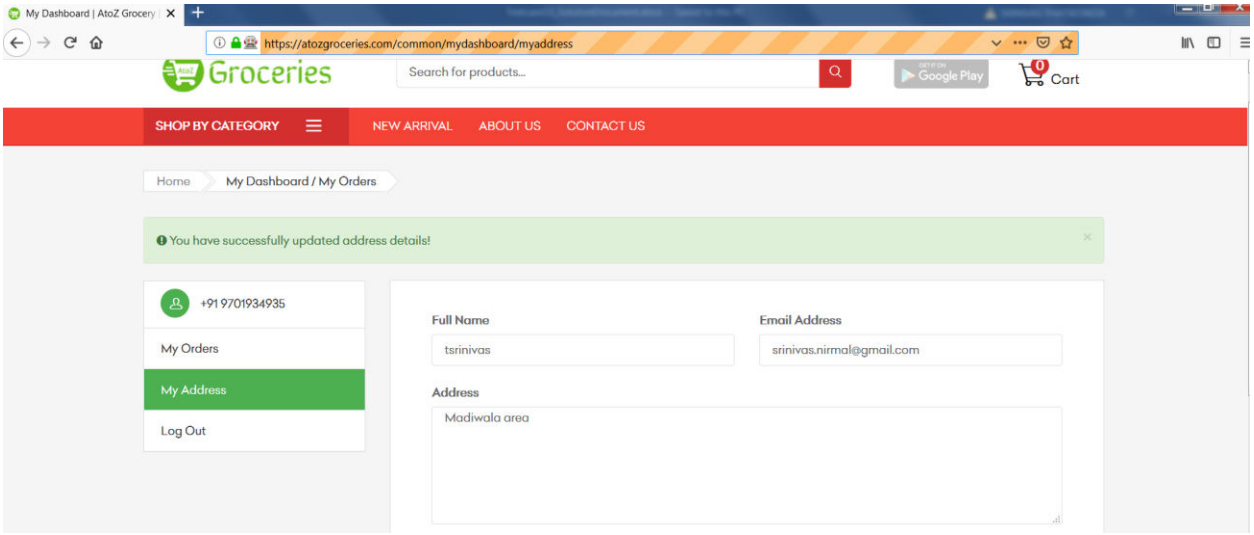
Error message is displayed

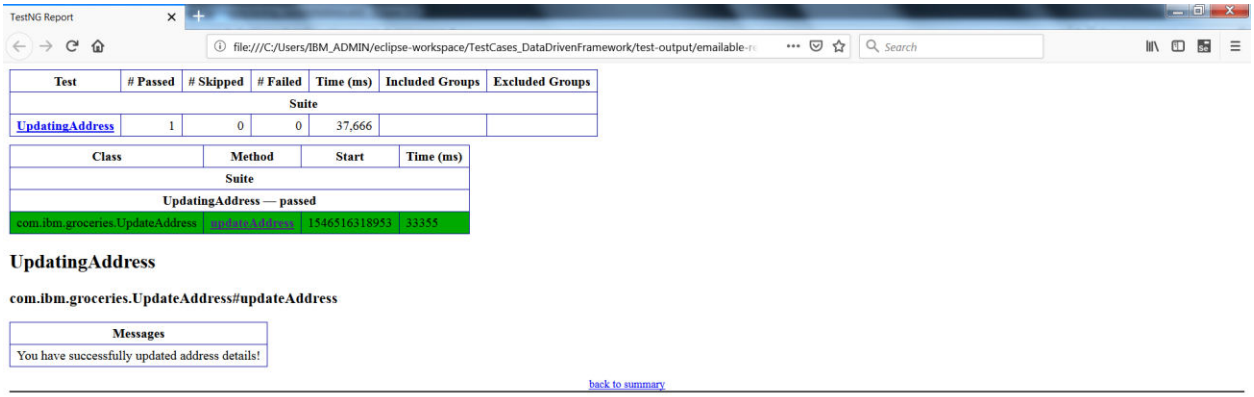


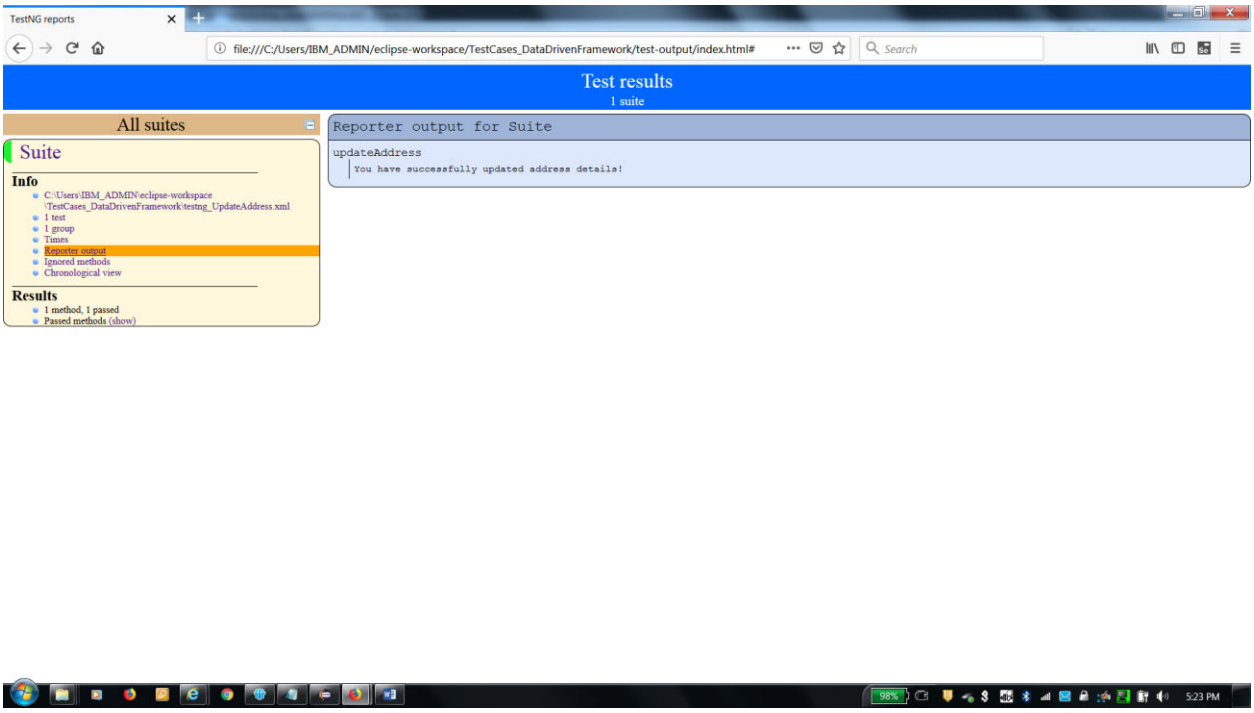
My Address page is displayed with entered details



Success message is displayed







Flow Information:

Initially a property file 'groceries.properties' is created in Test Data folder of the project and list of key, values for user portal page url,phonenumber,password, expected error message,expected success message,fullname,mail id,address,pin code,city, place etc. are stored in this property file.

Under the class 'WebDriverLaunch' ,a Test NG annotation 'Before Suite' is used to invoke the method 'preSetFortTest' which is used to instantiate object for class PropertiesFileHandler to return the list of these key,values in to a HashMap.

The Test NG annotation 'Before Method' is used by passing the parameter for browser to select and to invoke the method 'Initialization' which is used to call another method and to create webdriver object for the firefoxdriver or chromedriver or internetexplorer by using switch statement and setting path for webdriver exe.

The GroceriesUserPage class is defined with methods to locate the login link,phone number,password,login button. PageFactory is used in the constructor of this class. The method clickOnLogin method is defined to click on login link and singIn method is used to enter the phone number,password and login button.

The AccountPage class is defined to locate My Account and My Addres links. PageFactory is used in the constructor of this class. The method clickOnMyAccount is defined to click on these two links.

MyAddressPage class is defined to locate fullname,mail id,address,pin code,city, place and Update button.The method fillAddressEmpty is defined to empty the values for fullname,mail id,address,pin code,city and returns tool tip error message on clicking update button.

The method enterAddress is defind to enter or select values for fullname,mail id,address,pin code,city, place and returns success message on clicking update button.

The method clickOnMyAddress is defined to click on My Account,My Address links.

The getmethods are used to return the updated for fullname,mail id,address,pin code,city, place.

The class DatabaseConnection defines the method connectDatabase to connect with database and returns statement object.

The class GetScreenshot defines the method takeScreenshot to save screenshot of output screen with.png file format.

The testng_UpdateAddress xml file is having the parameter 'firefox' to launch Firefox browser.

The annotation @Test is used with method updateAddress in the class UpdateAddress. This method is used to get the values page url,phonenumber,password, expected error message,expected success message,fullname,mail id,address,pin code,city, place. The get method is called to launch the web url for user portal . The objects for classes GroceriesUserPage , AccountPage, MyAddressPage, ,DatabaseConnection, GetScreenshot are created and the above necessary methods are called on these objects. The executeQuery method is used to execute the query to retrieve record of updated address created in as_customer table by using next method used on result set object.

Finally Assertion is used to verify the success message of updated address and presence of updated address details on portal and database table as_customer.