Test Objective:

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

Testcase: Tc12

Process: Place an Order,DB and Negative Check

Steps to Execute:

1.Launch the user page https://atozgroceries.com

2.Without login, add a product to cart and proceed to checkout

3.It should ask for "Please enter your Phone Number to Login/Signup".

4.Type some alphabets and verify the validation

5.Fill all mandatory and place an order.

Expected Results:

1.Validate the text "Please enter your phone Number to Login/Signup".

2.Assert the validation message "Invalid or incorrect phone number!"

3.Validate your order is placed under My Account→My Orders

4.Verify the presence of order placed on the database.

Source Code:

File1: `PlaceOrder.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.CheckOutPage;
import com.ibm.groceriespages.GroceriesUserPage;
import com.ibm.groceriespages.MyOrdersPage;
import com.ibm.groceriespages.OrderSuccessPage;
import com.ibm.initialization.WebDriverLaunch;
```

```
import com.ibm.utilities.DatabaseConnection;
import com.ibm.utilities.GetScreenshot;

public class PlaceOrder extends WebDriverLaunch {

	@Test(priority = 1, testName = "PlaceOrder", groups = "low")
	public void myOrders() throws IOException, InterruptedException, SQLException
{
		String userPage = data.get("userPageUrl");
		String checkoutHeader = data.get("checkoutHeader");
		String text = data.get("text");
		String phoneNum = data.get("phoneNum");
		String password = data.get("pwd");
		String invalidPhoneMsg = data.get("invalidPhoneMsg");
		String fullname = data.get("fullnamevalue");
		String mailid = data.get("mailidvalue");
		String address = data.get("addressvalue");
		String pincode = data.get("pincodevalue");
		String orderMessage = data.get("orderMessage");

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

		GroceriesUserPage userpage = new GroceriesUserPage(driver, wait);
		String pageSource = userpage.gotoCheckOut();
		screen.takeScreenshot(driver);

		// Verifying checkout header is displayed
		Thread.sleep(2000);
		Assert.assertTrue(pageSource.contains(checkoutHeader));
		CheckOutPage checkObj = new CheckOutPage(driver, wait);

		// To verify the validation invalid phone number
		pageSource = checkObj.verifyinvalidPhonenumber(text, password);
		Thread.sleep(2000);
		screen.takeScreenshot(driver);
		Assert.assertTrue(pageSource.contains(invalidPhoneMsg));

		checkObj.fillMandatory(phoneNum, password);
		// To verify full name text box is displayed

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

		checkObj.enterDeliveryAddress(fullname, mailid, address, pincode);

		Thread.sleep(2000);
		// Verifying agree check box is displayed

	Assert.assertTrue(driver.findElement(By.xpath("//input[@id='tc']")).isDisplaye
d());

		pageSource = checkObj.confirmOrder();
```

```
            Thread.sleep(2000);

            //
wait.until(ExpectedConditions.presenceOfElementLocated(By.xpath("//a[contains(text(),
'My
            // Account')]")));
            // Verifying for message of order success
            // Assert.assertTrue(pageSource.contains(orderMessage));

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

            // To click on MyOrders
            OrderSuccessPage orderObj = new OrderSuccessPage(driver, wait);
            orderObj.clickOMyAccount();
            screen.takeScreenshot(driver);

            MyOrdersPage myorderObj = new MyOrdersPage(driver, wait);
            String orderid = myorderObj.orderId();
            orderid = orderid.replace("#", "");
            int id = Integer.parseInt(orderid);
            String customer = myorderObj.customer();
            String noProducts = myorderObj.numberOfProducts();
            String status = myorderObj.status();
            String date_added = myorderObj.dateadded();
            System.out.println("Order Details: ");
            System.out.println(orderid);
            System.out.println(customer);
            System.out.println(noProducts);
            System.out.println(status);
            System.out.println(date_added);

            DatabaseConnection conn = new DatabaseConnection();
            Statement st = conn.connectDatabase();


            ResultSet rs = st.executeQuery("select *from as_order" + " where
order_id=" + id);
            if (rs.next()) {
                  Assert.assertEquals(rs.getInt("order_id"), id);
            }

      }
}
```

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;
```

```java
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;
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();
	}

}
```

File3: CheckOutPage.java

```
package com.ibm.groceriespages;

import java.io.IOException;

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 org.testng.Assert;

import com.ibm.utilities.GetScreenshot;

public class CheckOutPage {

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

```
WebElement phonenumEle;

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

// To locate Next button
@FindBy(xpath = "//button[@id='next-id']")
WebElement nextEle;

// 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 Continue Payment button
@FindBy(xpath = "//a[text()='Continue to payment']")
WebElement continuepaymnetEle;

// To locate agree check box
@FindBy(xpath = "//input[@id='tc']")
WebElement agreeEle;

// To locate Confirm order button
@FindBy(xpath = "//a[@id='confirm-order-id']")
WebElement confirmorderEle;

WebDriverWait wait;
WebDriver driver;

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

```
	// Verify the validation message for invalid phone number field
	public String verifyinvalidPhonenumber(String text, String password) {
		phonenumEle.sendKeys(text);
		passwordEle.sendKeys(password);
		nextEle.click();
		return driver.getPageSource();

	}

	public void fillMandatory(String phoneNum, String password) {
		phonenumEle.clear();
		passwordEle.clear();
		phonenumEle.sendKeys(phoneNum);
		passwordEle.sendKeys(password);
		nextEle.click();
	}

	public void enterDeliveryAddress(String fullname, String mailid, String
address, String pincode)
			throws IOException, InterruptedException {
		fullnameEle.clear();
		emailEle.clear();
		addressEle.clear();

		fullnameEle.sendKeys(fullname);
		emailEle.sendKeys(mailid);
		addressEle.sendKeys(address);

		// To select city
		Select citySelect = new Select(cityEle);
		citySelect.selectByIndex(10);

		// To enter pin code
		pincodeEle.clear();
		pincodeEle.sendKeys(pincode);

		// To select place
		Select placeSelect = new Select(placeEle);
		placeSelect.selectByIndex(0);

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

		// To click on continue to payment button
		continuepaymnetEle.click();
	}

	public String confirmOrder() {
		agreeEle.click();
		confirmorderEle.click();
		return driver.getPageSource();
	}

}
```

File4: OrderSuccessPage.java

```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 OrderSuccessPage {

	/*
	 * @FindBy(xpath ="//a[contains(text(),'My Account')]") WebElement
myaccountEle;
	 */

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

	WebDriverWait wait;
	WebDriver driver;

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

	// Method to click on MyAccount
	public void clickOMyAccount() throws IOException, InterruptedException {

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

}
```

File5: MyOrdersPage.java

```java
package com.ibm.groceriespages;

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;
```

```java
public class MyOrdersPage {

	@FindBy(xpath="//table[@class='table table-bordered table-
hover']/tbody/tr[1]/td[1]")
	WebElement orderIdEle;


	@FindBy(xpath="//table[@class='table table-bordered table-
hover']/tbody/tr[1]/td[2]")
	WebElement customerEle;


	@FindBy(xpath="//table[@class='table table-bordered table-
hover']/tbody/tr[1]/td[3]")
	WebElement noProductsEle;

	@FindBy(xpath="//table[@class='table table-bordered table-
hover']/tbody/tr[1]/td[4]")
	WebElement statusEle;

	@FindBy(xpath="//table[@class='table table-bordered table-
hover']/tbody/tr[1]/td[6]")
	WebElement dateaddedEle;

	WebDriverWait wait;
	WebDriver driver;

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



	public String orderId()
	{
		String orderid=orderIdEle.getText().toString();
		return orderid;
	}

	public String customer()
	{
		String customer=customerEle.getText().toString();
		return customer;
	}


	public String numberOfProducts()
	{
		String noProducts=noProductsEle.getText().toString();
		return noProducts;
	}
```

```
	public String status()
	{
		String status=statusEle.getText().toString();
		return status;
	}

	public String dateadded()

	{
		String date1=dateaddedEle.getText().toString();
		return date1;
	}
}
```

File6: 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_atoz
groceries","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_atoz
groceries","foodsonfinger_atoz","welcome@123");
		Statement st=c.createStatement();
		ResultSet rs=st.executeQuery(query);

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

}
```

File7: 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"));


	}

}
```

File8: 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();
        }

}
```

File9: 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;
            }
        }
}
```

File10: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=Benguluru
pincodevalue=555666
orderMessage=Thank you! Your order has been placed successfully
```

File11:testing_PlaceOrder.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="PlaceAnOrder">
  <parameter name="browser" value="firefox"></parameter>
    <classes>
      <class name="com.ibm.groceries.PlaceOrder"/>
    </classes>
  </test> <!-- PlaceAnOrder -->
</suite> <!-- Suite -->
```

Test Result:

1.Validatedthe text “Please enter your phone Number to Login/Signup”.

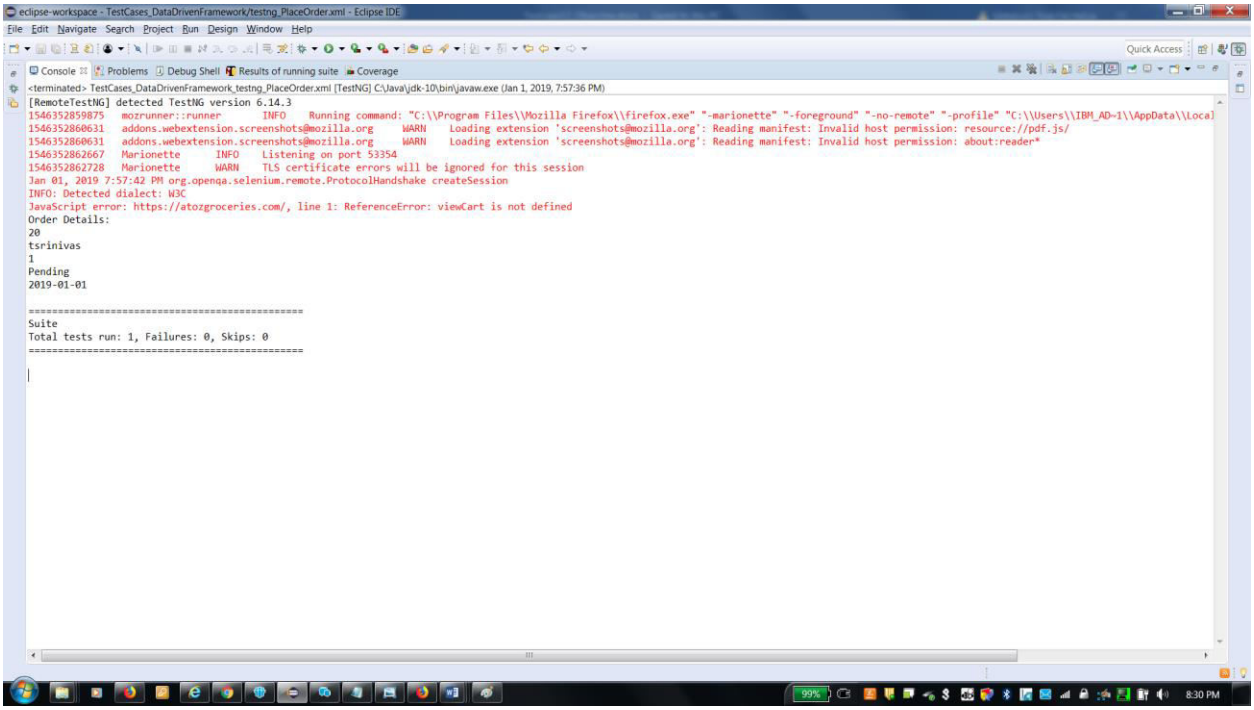
2.Done Assertion on the validation message “Invalid or incorrect phone number!”

3.Validated that the order is placed under My Account→My Orders

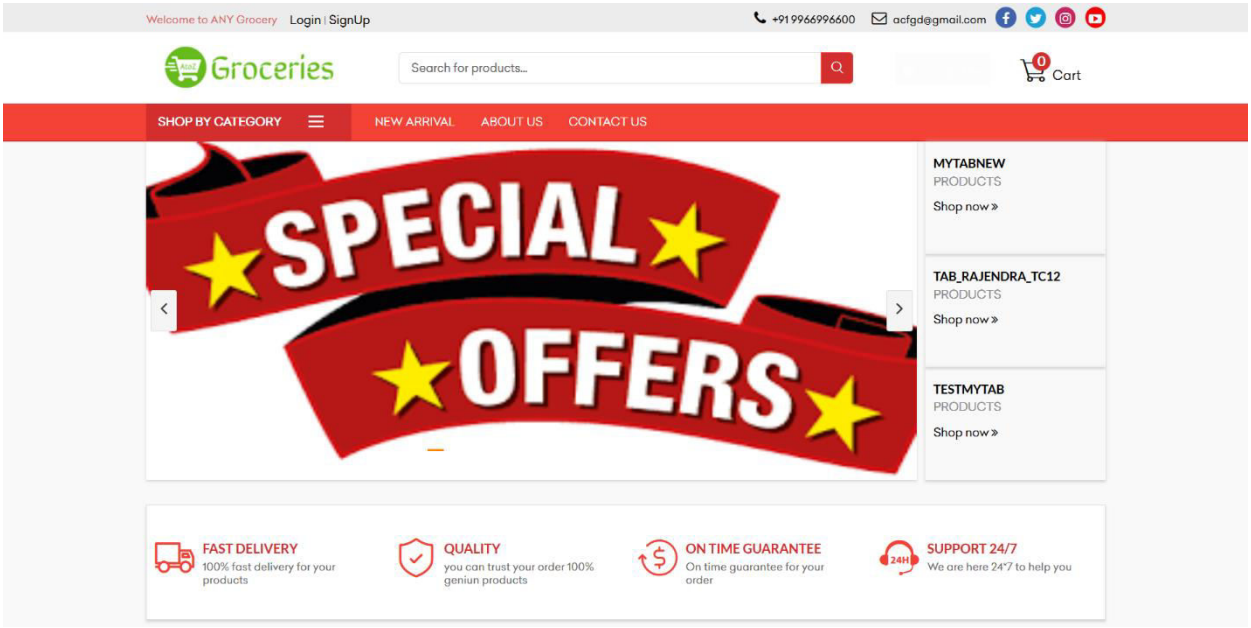
4.Verified the presence of order placed on the database.

Screenshots:

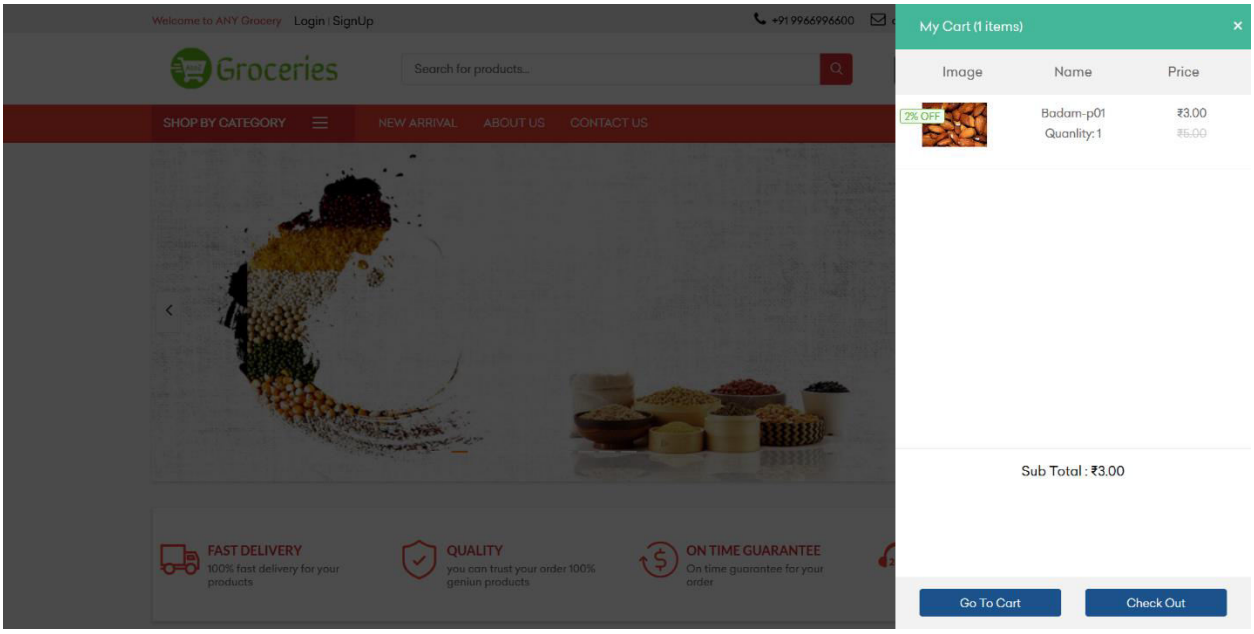
ConsoleOutput:



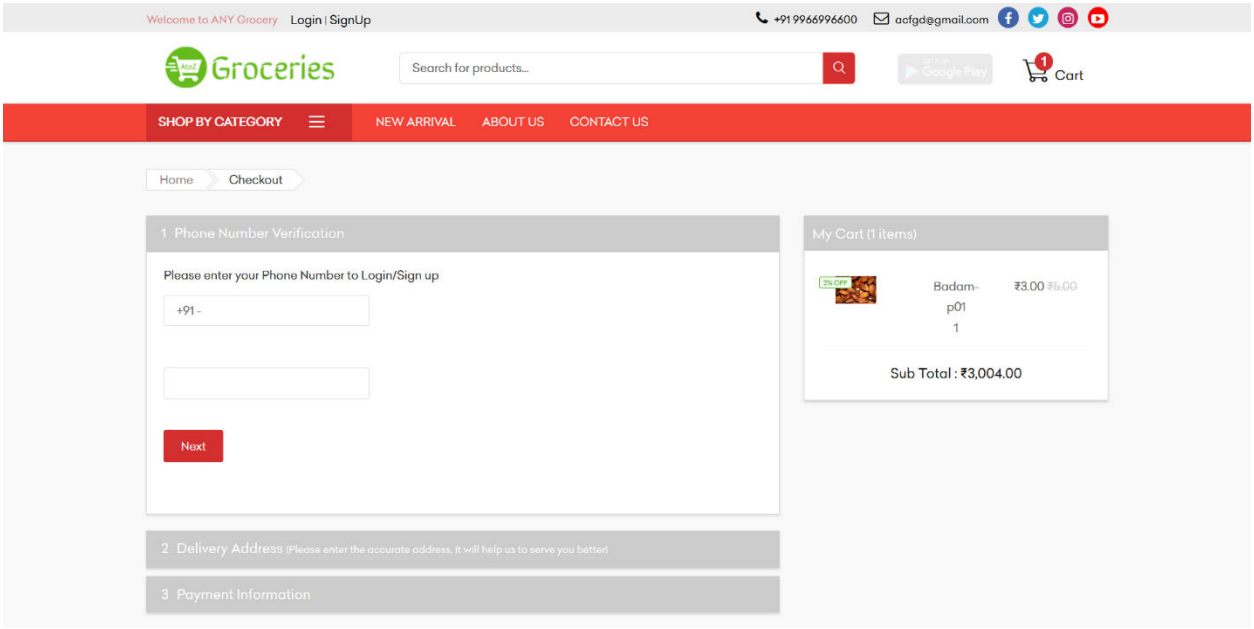
Userportal is displayed



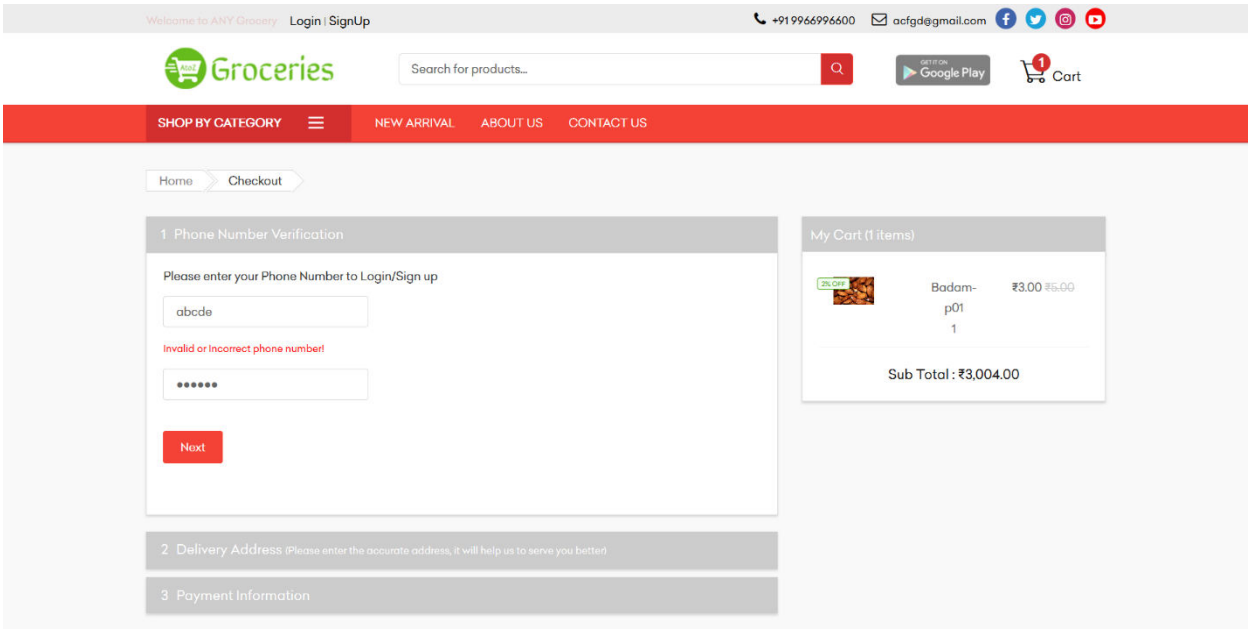
Product is added to cart



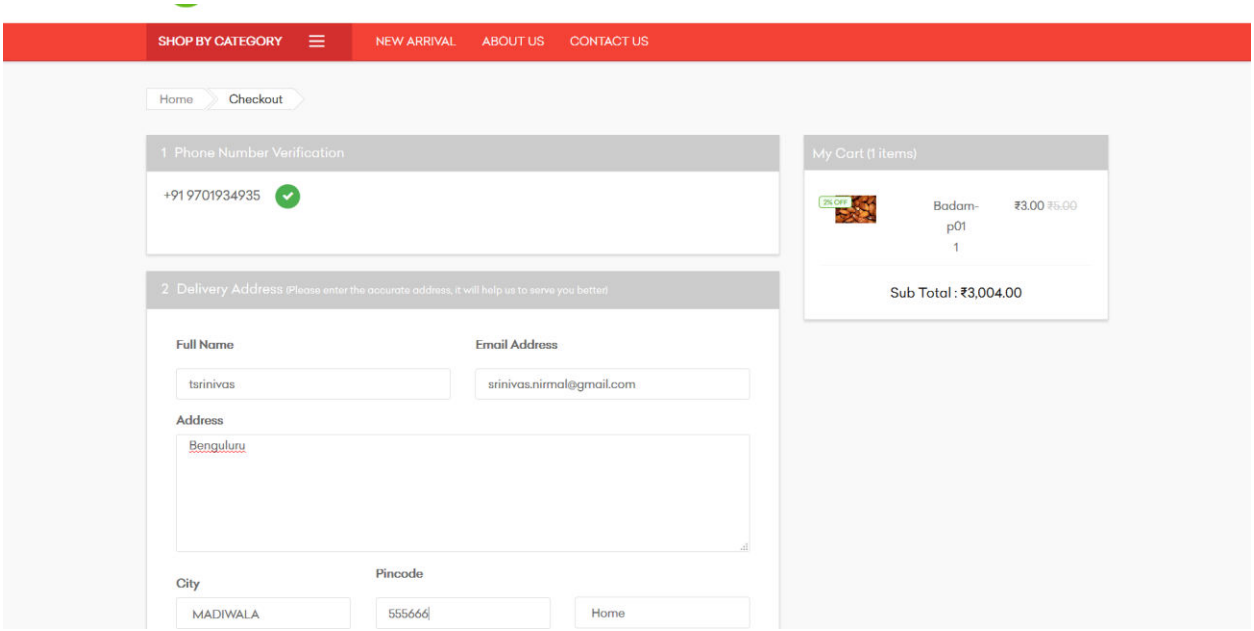
Login page is displayed to checkout



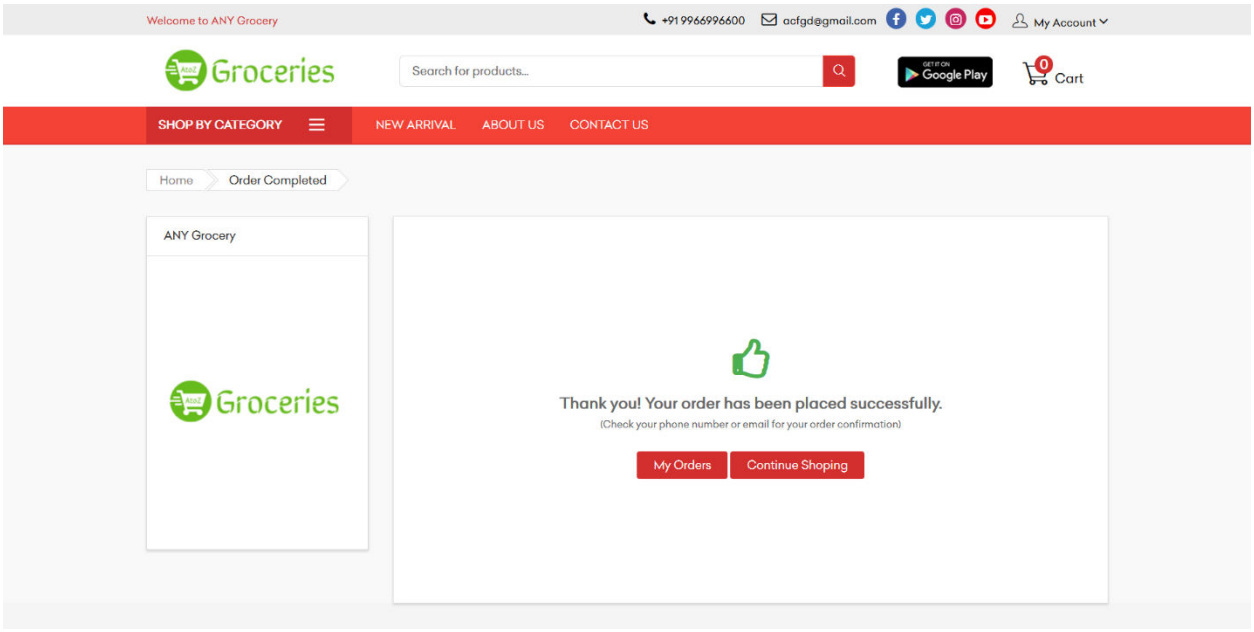
Validation message is displayed on entering invalid phone number



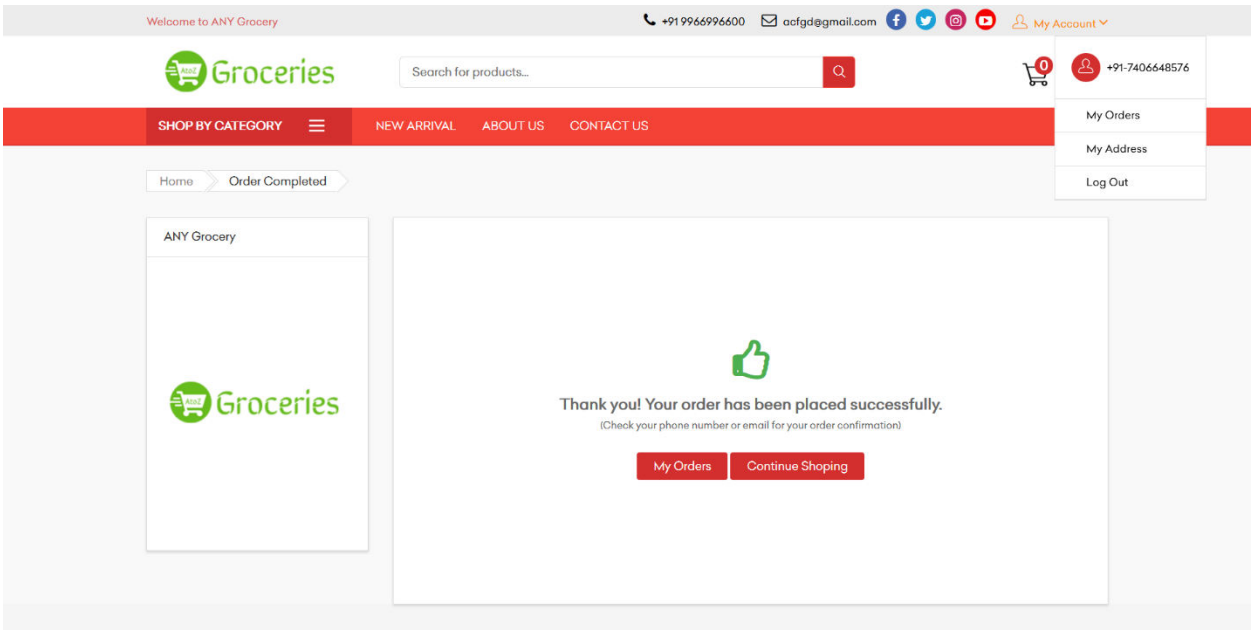
Delivery Address details are displayed



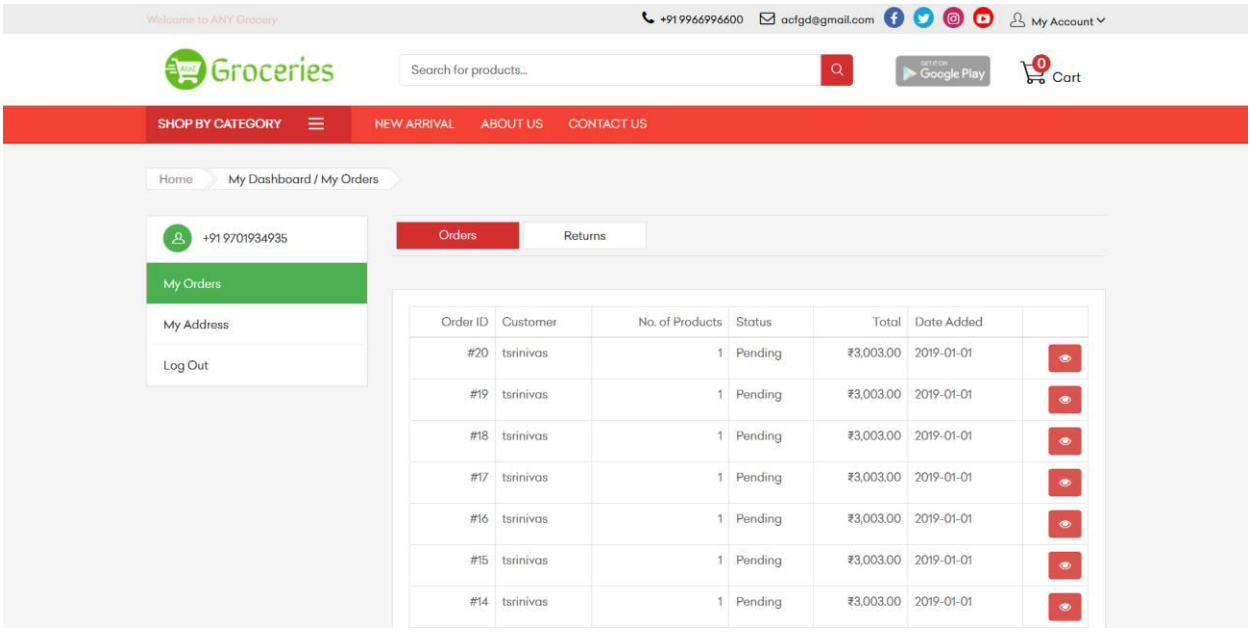
Confirmation message is displayed with order successfully placed

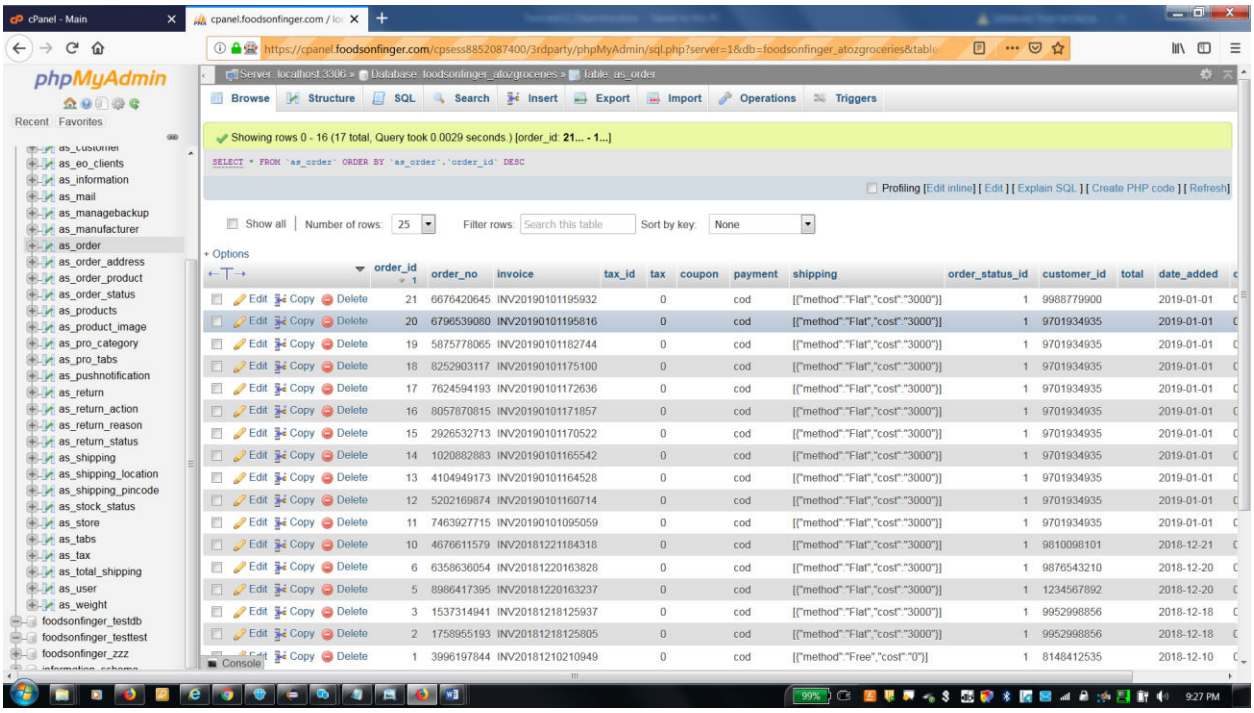


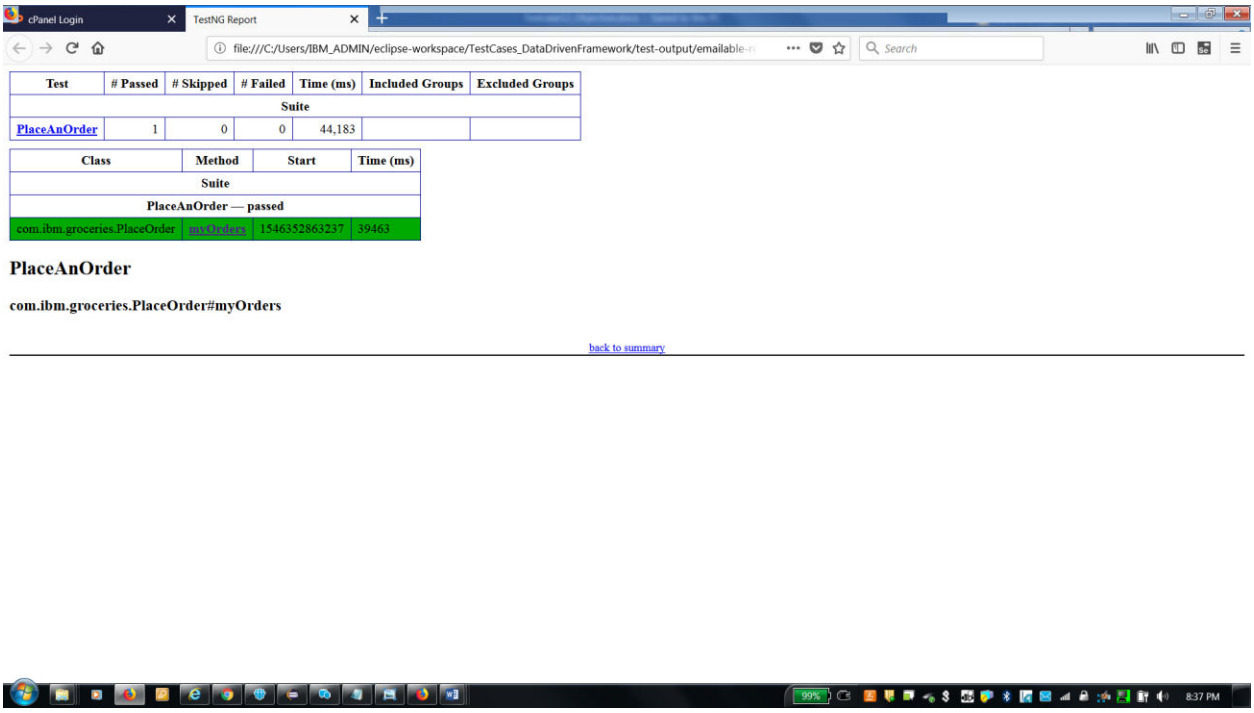
My Account→My Orders is displayed



Placed order is displayed as first row in the table







cPanel Login | TestNG Report

file:///C:/Users/IBM_ADMIN/eclipse-workspace/TestCases_DataDrivenFramework/test-output/emailable-r

Search

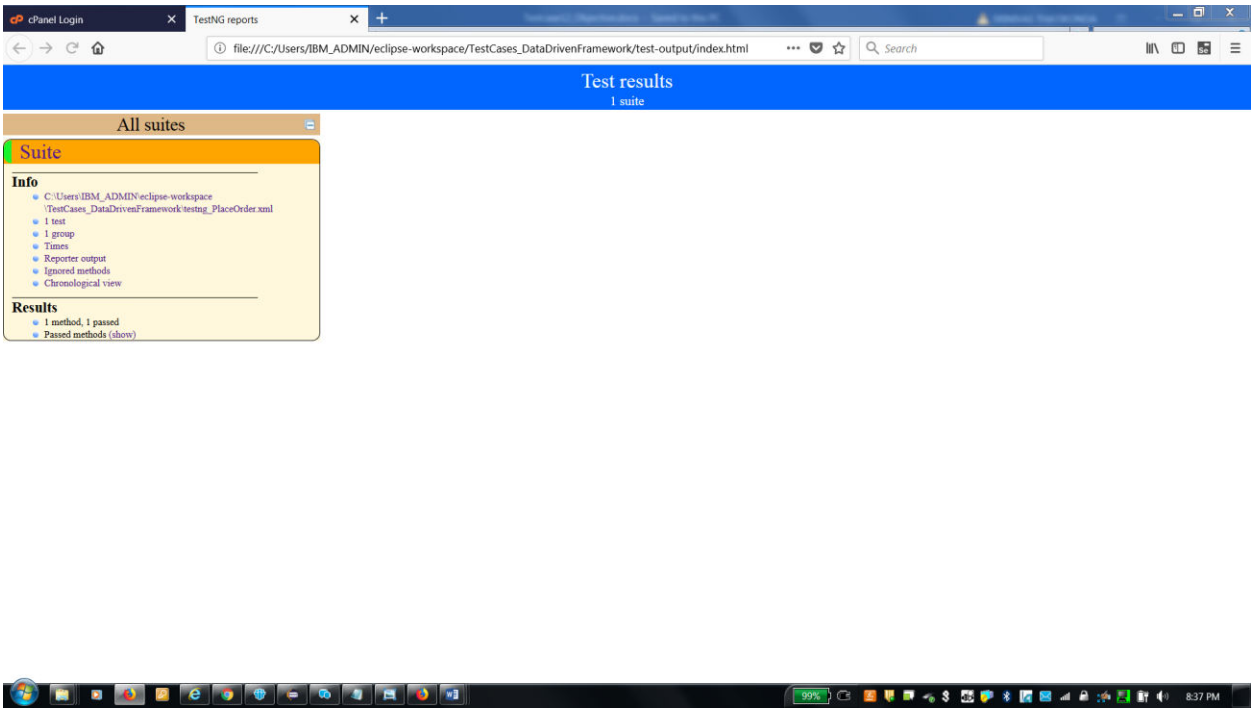
Test	# Passed	# Skipped	# Failed	Time (ms)	Included Groups	Excluded Groups
Suite						
PlaceAnOrder	1	0	0	44,183		

Class	Method	Start	Time (ms)
Suite			
PlaceAnOrder — passed			
com.ibm.groceries.PlaceOrder	myOrders	1546352863237	39463

PlaceAnOrder

com.ibm.groceries.PlaceOrder#myOrders

back to summary



cPanel Login | TestNG reports

file:///C:/Users/IBM_ADMIN/eclipse-workspace/TestCases_DataDrivenFramework/test-output/index.html

Search

Test results

1 suite

All suites

Suite

Info

- C:\Users\IBM_ADMIN\eclipse-workspace \TestCases_DataDrivenFramework\testng_PlaceOrder.xml
- 1 test
- 1 group
- Times
- Reporter output
- Ignored methods
- Chronological view

Results

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

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, user name, password, invalid phone number validation message 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 web elements add to cart button, cart link and checkout element. PageFactory is used in the constructor of this class. The method gotoCheckOut method is defined to click on these elements and pagesource is returned.

The CheckOutPage class is used to locate the web elements phone number,password,next button,full name,email,address,city,pin code, place,continue payment button,confirm order button. PageFactory is used in the constructor of this class. The method verifyinvalidPhonenumber is defined to verify the expected validation message for invalid phone number.The method fillMandatory is defined to enter values for the mandatory fields and click method is used to click on continue payment button, confirm order button.

The method enterDeliveryAddress is defined to enter values for fullname,email, address,city,pincode, place by using sendKeys method.

The method confirmOrder is defined to click on agree check box and confirm order button.

The OrderSuccessPage class is used to locate MyAccount,My Order links. PageFactory is used in the constructor of this class.The method clickOMyAccount is defined to click on these two elements.

MyOrdersPage is defined to locate to table row elements order id,customer, number of products, status, date added.The methods orderId,customer,numberOfProducts,status,dateadded are defined to get the values for the above elements.

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_PlaceOrder xml file is having the parameter ‘firefox’ to launch Firefox browser.

The annotation @Test is used with method myOrders in the class PlaceOrder. This method is used to get the values for user portal url,phonenumber, user name, password, invalid phone number validation message etc. The get method is called to launch the web url for user portal . The objects for classes

GroceriesUserPage , CheckOutPage , OrderSuccessPage , MyOrdersPage ,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 order created in as_order table by using next method used on result set object.

Finally the Assert is used to verify the order id from the database matches with order displayed on the user portal while placing an order.