

Test Objective:

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

Testcase: Tc11

Process: Add a mail

Steps to Execute:

Step1: Navigate on to admin panel->Marketing->Mail

Step2: Populate only subject and Save.

Step3: Populate the other mandatory fields.

Expected Results:

Validate the presence of Mail header.

Validate the error message.

Validate the success message.

Validate the presence of record in the admin panel table.

Validate the presence of record in the DB table.

Source Code:

File1: ValidateAddEmail.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.testng.Assert;
import org.testng.Reporter;
import org.testng.annotations.Test;
import com.ibm.groceriespages.MaillistPage;
import com.ibm.groceriespages.PageDashboard;
import com.ibm.groceriespages.PageLogin;
import com.ibm.groceriespages.SendMailPage;
import com.ibm.initialization.WebDriverLaunch;
import com.ibm.utilities.DatabaseConnection;
import com.ibm.utilities.GetScreenshot;

public class ValidateAddEmail extends WebDriverLaunch {
```

```

@Test(priority = 1, testName = "ValidateEmailAdd", groups =
"low")

    public void validateAddEmailDbase() throws IOException,
InterruptedException, SQLException {
        String url = data.get("url");
        String userName = data.get("username");
        String password = data.get("password");
        String email = data.get("customerEmail");
        String subject = data.get("subject");
        String message = data.get("message");
        String expMessage = data.get("emailMsg");
        String mailHeader = data.get("mailHeader");
        String expsendmailPageTitle = data.get("sendmailPageTitle");
        String expAddressMessage = data.get("errorAddressMessage");
        String emailTablename = data.get("emailTable");
        // Launching the web site for atozgroceries
        driver.get(url);
        GetScreenshot screen = new GetScreenshot();

        PageLogin login = new PageLogin(driver, wait);
        // To enter email address and password and clickon login
button
        login.enterEmailAddress(userName);
        login.enterPassword(password);
        screen.takeScreenshot(driver);
        login.clickOnLogin();
        // Checking logout link is displayed

        Assert.assertTrue(driver.findElement(By.partialLinkText("Logout"))
.isDisplayed());

        PageDashboard dashboard = new PageDashboard(driver, wait);
        // To click on Catalog
        dashboard.clickOnMarketing();

        // To click on Mail link
        dashboard.clickOnMail();
        screen.takeScreenshot(driver);

        // Verify the presence of mail header

        Assert.assertTrue(driver.getPageSource().contains(mailHeader));

        MailListPage maillistObj = new MailListPage(driver, wait);

```

```

        // Calling method to click on Add email button
        maillistObj.clickOnAdd();
        screen.takeScreenshot(driver);
        // Verifying the page title of Send Mail page
        Assert.assertEquals(driver.getTitle(),
expsendmailPageTitle);

        SendMailPage addemailObj = new SendMailPage(driver, wait);
        // Calling method to populate subject
        String pageSource = addemailObj.populateSubject(subject);
        screen.takeScreenshot(driver);
        // Checking the expected error message is displayed
        Assert.assertTrue(pageSource.contains(expAddressMessage));

        // Calling method to enter email,subject
        pageSource = addemailObj.populateMandatory(email, message);
        screen.takeScreenshot(driver);

        if (pageSource.contains(expMessage + email)) {
            Reporter.Log(expMessage + email);
            System.out.println(expMessage + email);
            // checking whether success message is displayed or
not
            Assert.assertTrue(pageSource.contains(expMessage +
email));

            // Checking the presence of record in the admin panel
table
            Assert.assertEquals(

                driver.findElement(By.xpath("//table[@id='dataTableExample2']/tbody/tr[1]/td[2]")).getText(),
                    email);
            Assert.assertEquals(

                driver.findElement(By.xpath("//table[@id='dataTableExample2']/tbody/tr[1]/td[3]")).getText(),
                    subject);

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

            ResultSet rs = st.executeQuery("select *from " +
emailTablename + " where to_mail=" + "" + email + "");

```

```

        // To verify the presence of record in DB table
        if (rs.next()) {
            System.out.println("Added Record in email
table:");

            System.out.println(rs.getString("message"));
            Assert.assertEquals(rs.getString("message"),
message);

            System.out.println(rs.getString("subject"));
            Assert.assertEquals(rs.getString("subject"),
subject);

            System.out.println(rs.getString("to_mail"));
            Assert.assertEquals(rs.getString("to_mail"),
email);

        }

    }

}

```

File2: PageLogin.java

```

package com.ibm.groceriespages;

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

public class PageLogin {

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

```

```
@FindBy(name="pword")
```

```
WebElement passEle;
```

```
//@FindBy(className="btn btn-labeled btn-info m-b-5")
```

```
@FindBy(xpath="//button[@class='btn btn-labeled btn-info m-b-5']")
```

```
WebElement loginEle;
```

```
WebDriverWait wait;
```

```
WebDriver driver;
```

```
public PageLogin(WebDriver driver,WebDriverWait wait)
```

```
{
```

```
    PageFactory.initElements(driver, this);
```

```
    this.driver=driver;
```

```
    this.wait=wait;
```

```
}
```

```
//To enter email address
```

```
public void enterEmailAddress(String userName)
```

```
{
```

```
    emailEle.sendKeys(userName);
```

```
}
```

```
//To enter apssword
```

```
public void enterPassword(String password)
```

```
{
```

```
    passEle.sendKeys(password);
```

```
}
```

```

        //To click on Login button
        public void clickOnLogin()
        {
            loginEle.click();
        }
    }
}

```

File3: PageDashboard.java

```

package com.ibm.groceriespages;

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

public class PageDashboard {

    // @FindBy(xpath="//a[@class='material-ripple']")
    @FindBy(xpath = "//a[text()=' Catalog']")
    WebElement catalogEle;

    @FindBy(xpath = "//a[text()=' Products']")
    WebElement productEle;
}

```

```
By prodElt = By.xpath("//a[text()=' Products']");
```

```
WebDriverWait wait;
```

```
WebDriver driver;
```

```
// Locating elements for Marketing link
```

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

```
WebElement marketEle;
```

```
// Locating Pushnotification link
```

```
@FindBy(xpath = "//a[text()=' Push Notification']")
```

```
WebElement notificatonEle;
```

```
// Locating System link
```

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

```
WebElement systemEle;
```

```
// Locating Returns link
```

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

```
// @FindBy(xpath("//a[contains(text(),'Returns')]"))
```

```
WebElement returnsEle;
```

```
// Locating Shipping link
```

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

```
WebElement shippingEle;
```

```
// Locating ShippingLocatons link
```

```
@FindBy(xpath = "//a[text()=' Shipping Locations']")
```

```
WebElement locatoinEle;
```

```
@FindBy(xpath = "//a[text()=' Return Actions']")
```

```
WebElement actionsEle;
```

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

```
WebElement settingsEle;
```

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

```
WebElement mailEle;
```

```
public PageDashboard(WebDriver driver, WebDriverWait wait) {
```

```
    PageFactory.initElements(driver, this);
```

```
    this.driver = driver;
```

```
    this.wait = wait;
```

```
}
```

```
// To click on Catalog
```

```
public void clickOnCatalog() {
```

```
    catalogEle.click();
```

```
    wait.until(ExpectedConditions.presenceOfElementLocated(prodElt));
```

```
}
```

```
// To click on Products
```

```
public void clickOnProducts() {
```

```
    productEle.click();
```

```
}
```



```
// To click on Marketing link  
public void clickOnMarketing() {
```

```
    marketEle.click();
```

```
}
```

```
// To click on Push Notificatoin link  
public void clickOnPushNotification() {
```

```
    notificatonEle.click();
```

```
}
```

```
// TO click on System link  
public void clickOnsystem() {
```

```
    systemEle.click();
```

```
}
```

```
// To click on Retuns link
```

```
public void clickOnReturns() {
```

```
    driver.findElement(By.partialLinkText("Returns")).click();
```

```
    // returnsEle.click();
```

```
}
```

```
// To click on Return actoins link
```

```
public void clickOnRetrunActions()
```

```
{
```

```

        actionsEle.click();
    }

    public void clickOnShipping() {
        driver.findElement(By.partialLinkText("Shipping")).click();
        // shippingEle.click();

    }

    public void clickOnShippingLocations() {
        locatoinEle.click();

    }

    // To click on Settings link
    public void clickOnSettings() {
        settingsEle.click();
    }

    public void clickOnMail() {
        mailEle.click();
    }

}

```

File4: MailListPage.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;

public class MailListPage {

    @FindBy(xpath = "//a[@title='Add New']")
    WebElement addnewEle;

    WebDriverWait wait;
    WebDriver driver;

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

    // Method to add email
    public void clickOnAdd() {
        addnewEle.click();
    }

}
```

File5: SendMailPage.java

```
package com.ibm.groceriespages;

import java.io.IOException;

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 SendMailPage {

    // To locate To drop down field
    @FindBy(xpath = "//select[@name='to']")
    WebElement toEle;

    // To locate Customer E-mail text box
    @FindBy(xpath = "//input[@id='email']")
    WebElement emailEle;

    // To locate Subject text box
    @FindBy(xpath = "//input[@name='subject']")
    WebElement subjectEle;

    /*
     * To locate message web element
     *
     * @FindBy(xpath="//div[@class='note-editable panel-body']")
    WebElement
        * messageEle;
    */

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

    WebDriverWait wait;
    WebDriver driver;
```

```

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

    // Method to add email
    public String addMail(String customerEmail, String subject,
String message) throws InterruptedException {

        JavascriptExecutor js = (JavascriptExecutor) driver;

        // To select the value for dropdown 'To'
        Select toSelect = new Select(toEle);
        toSelect.selectByIndex(2);

        // To enter customer email
        emailEle.sendKeys(customerEmail);
        // To enter subject
        subjectEle.sendKeys(subject);
        // To enter message

        js.executeScript("document.getElementById('summernote').value='We
lcome All'");
        // To click on Save button
        saveEle.click();
        Thread.sleep(1000);
        return driver.getPageSource();
    }

    //Method to enter only subject
    public String populateSubject(String subject)
    {

        // To enter subject
        subjectEle.sendKeys(subject);

        // To click on Save button
        saveEle.click();

        return driver.getPageSource();
    }

```

```

    public String populateMandatory(String customerEmail,String
message)
    {
        JavascriptExecutor js = (JavascriptExecutor) driver;

        // To enter customer email
        emailEle.sendKeys(customerEmail);

        // To enter message

        //js.executeScript("document.getElementById('summernote').value='
Welcome All'");

        js.executeScript("document.getElementById('summernote').value='"+
message+"'");

        // To click on Save button
        saveEle.click();

        return driver.getPageSource();
    }
}

```

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_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;
    }
}

```

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: 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"));

    }
}

```

}

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
searchDisplayMessage=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
expectedShoppingCartMsg=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
```

File11:

```
<?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="ValidatingAddEmail">
    <parameter name="browser" value="firefox"></parameter>
    <classes>
      <class name="com.ibm.groceries.ValidateAddEmail"/>
    </classes>
  </test> <!-- ValidatingAddEmail -->
</suite> <!-- Suite -->
```

Test Result:

Validated the presence of Mail header.

Validated the error message.

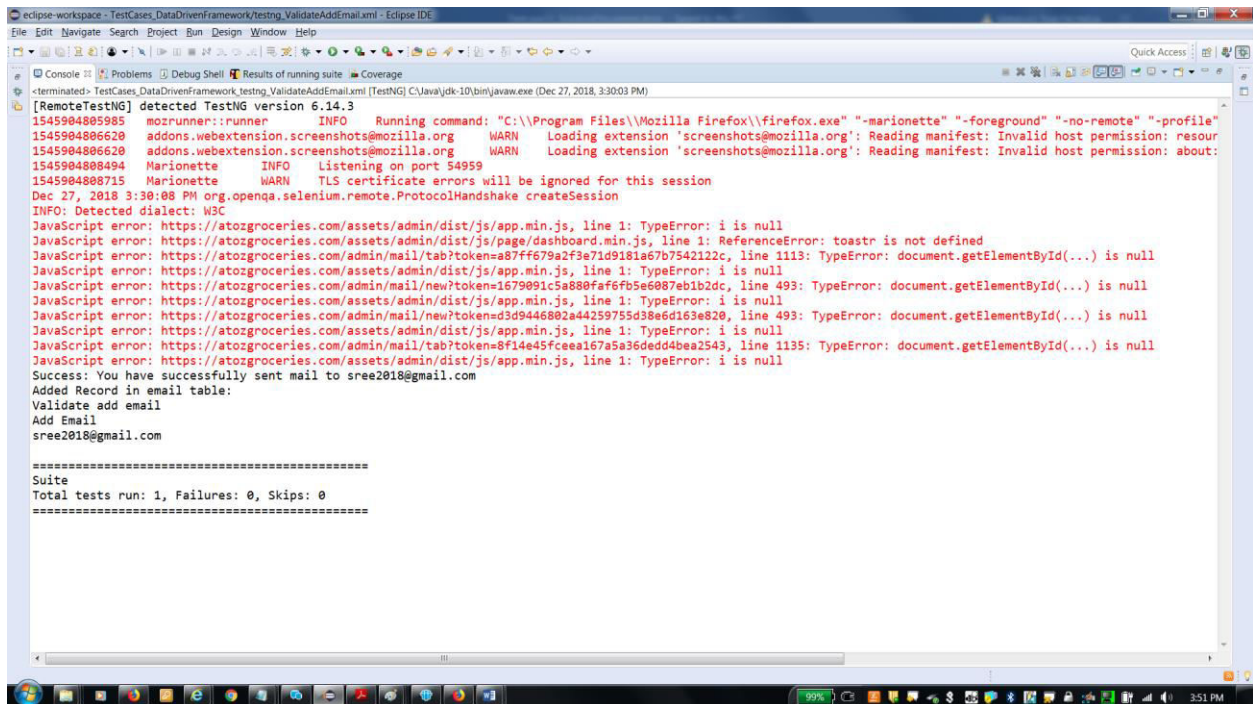
Validated the success message.

Validated the presence of record in the admin panel table.

Validated the presence of record in the DB table.

Screenshots:

Console Output:

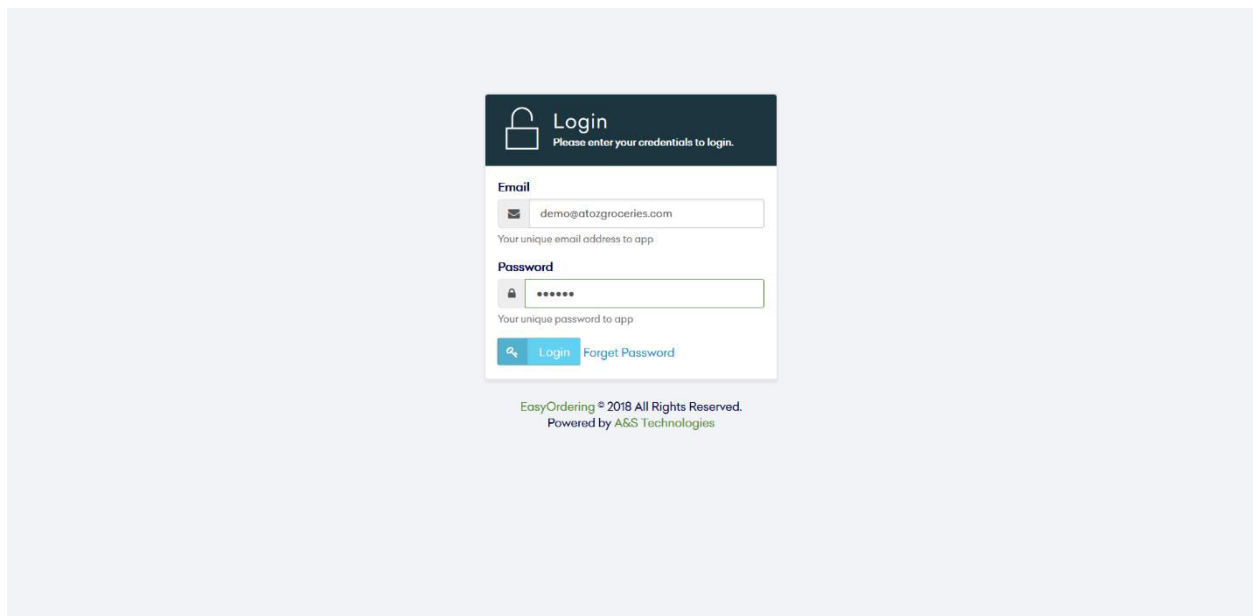


```
eclipse-workspace - TestCases_DataDrivenFramework/testing_Va... - Eclipse IDE
File Edit Navigate Search Project Run Design Window Help

[RemoteTestNG] detected TestNG version 6.14.3
1545904805985 mozrunner::runner INFO Running command: "C:\Program Files\Mozilla Firefox\firefox.exe" "-marionette" "-foreground" "-no-remote" "-profile"
1545904806620 addons.webextension.screenshots@mozilla.org WARN Loading extension 'screenshots@mozilla.org': Reading manifest: Invalid host permission: resour
1545904806620 addons.webextension.screenshots@mozilla.org WARN Loading extension 'screenshots@mozilla.org': Reading manifest: Invalid host permission: about:
1545904808494 Marionette INFO Listening on port 54959
1545904808715 Marionette WARN TLS certificate errors will be ignored for this session
Dec 27, 2018 3:30:08 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
JavaScript error: https://atozgroceries.com/assets/admin/dist/js/app.min.js, line 1: TypeError: i is null
JavaScript error: https://atozgroceries.com/assets/admin/dist/js/page/dashboard.min.js, line 1: ReferenceError: toastr is not defined
JavaScript error: https://atozgroceries.com/admin/mail/tab?token=a87ff679a2f3e71d9181a67b7542122c, line 1113: TypeError: document.getElementById(...) is null
JavaScript error: https://atozgroceries.com/assets/admin/dist/js/app.min.js, line 1: TypeError: i is null
JavaScript error: https://atozgroceries.com/admin/mail/new?token=1679091c5a880faf6fb5e6087eb1b2dc, line 493: TypeError: document.getElementById(...) is null
JavaScript error: https://atozgroceries.com/assets/admin/dist/js/app.min.js, line 1: TypeError: i is null
JavaScript error: https://atozgroceries.com/admin/mail/new?token=d3d9446802a44259755d38e6d163e820, line 493: TypeError: document.getElementById(...) is null
JavaScript error: https://atozgroceries.com/assets/admin/dist/js/app.min.js, line 1: TypeError: i is null
JavaScript error: https://atozgroceries.com/admin/mail/tab?token=8f14e45fcee167a5a36dedd4bea2543, line 1135: TypeError: document.getElementById(...) is null
JavaScript error: https://atozgroceries.com/assets/admin/dist/js/app.min.js, line 1: TypeError: i is null
Success: You have successfully sent mail to sree2018@gmail.com
Added Record in email table:
Validate add email
Add Email
sree2018@gmail.com

=====
Suite
Total tests run: 1, Failures: 0, Skips: 0
=====
```

Login page is displayed for admin portal



Mail List page is displayed.

The screenshot shows the 'Mail List' page in the 'easyordering' application. The left sidebar contains navigation links: Dashboard, Catalog, Sales, Customers, Marketing (highlighted with a red bar), System, and Reports. The Marketing menu is expanded, showing 'Coupons', 'Mail', and 'Push Notification'. The main content area is titled 'Mail' and contains a 'Mail List' table. Above the table are controls for 'Show 10 entries', export options (Copy, CSV, Excel, PDF, Print), and a search bar. The table has columns for #, Mail Type, Subject, Date Added, and Action. It lists 8 entries, including individual email addresses and an 'All Customers' entry.

#	Mail Type	Subject	Date Added	Action
1	abc7@gmail.com	subject7	2018-12-27 14:11:59	Action ▾
2	abc6@gmail.com	subject6	2018-12-27 14:09:38	Action ▾
3	abc5@gmail.com	subject5	2018-12-27 13:08:07	Action ▾
4	abc4@gmail.com	subject4	2018-12-27 13:05:19	Action ▾
5	abc3@gmail.com	subject3	2018-12-27 13:03:04	Action ▾
6	abc@gmail.com	subject2	2018-12-27 12:34:26	Action ▾
7	email1@gmail.com	sub1	2018-12-27 11:51:37	Action ▾
8	All Customers	Adding email	2018-12-26 18:50:49	Action ▾

Expected validation message is displayed

The screenshot shows the 'Send Mail' form in the 'easyordering' application. The left sidebar is the same as the previous screenshot. The main content area is titled 'Mail' and contains the 'Send Mail' form. The form has fields for 'From' (Default), 'To' (empty), and 'Subject' (Add Email). Below these fields is a 'Message' section with a rich text editor toolbar and a text area. A red error message 'To address is required!' is displayed below the 'To' field.

Send Mail

From
Default

To
Customer E-Mail

*** Subject**
Add Email

*** Message**

To address is required!

Success message is displayed with added record

The screenshot shows the 'Mail' management page in the EasyOrdering system. A green success message at the top states: 'Success: You have successfully sent mail to sree2018@gmail.com!'. Below this is a 'Mail List' table with 7 entries. The table columns are: #, Mail Type, Subject, Date Added, and Action. The 'Action' column contains a dropdown menu labeled 'Action'.

#	Mail Type	Subject	Date Added	Action
1	sree2018@gmail.com	Add Email	2018-12-27 15:30:33	Action
2	abc7@gmail.com	subject7	2018-12-27 14:11:59	Action
3	abc6@gmail.com	subject6	2018-12-27 14:09:38	Action
4	abc5@gmail.com	subject5	2018-12-27 13:08:07	Action
5	abc4@gmail.com	subject4	2018-12-27 13:05:19	Action
6	abc3@gmail.com	subject3	2018-12-27 13:03:04	Action
7	abc@gmail.com	subject2	2018-12-27 12:34:26	Action

The screenshot shows a TestNG report for the test 'ValidatingAddEmail'. The report is displayed in a table format with columns: Test, # Passed, # Skipped, # Failed, Time (ms), Included Groups, and Excluded Groups. The test 'ValidatingAddEmail' has 1 passed, 0 skipped, and 0 failed results. Below the table, there is a section for 'ValidatingAddEmail' with a 'Messages' box containing the success message: 'Success: You have successfully sent mail to sree2018@gmail.com'. A link 'back to summary' is also present.

Test	# Passed	# Skipped	# Failed	Time (ms)	Included Groups	Excluded Groups
ValidatingAddEmail	1	0	0	28,390		

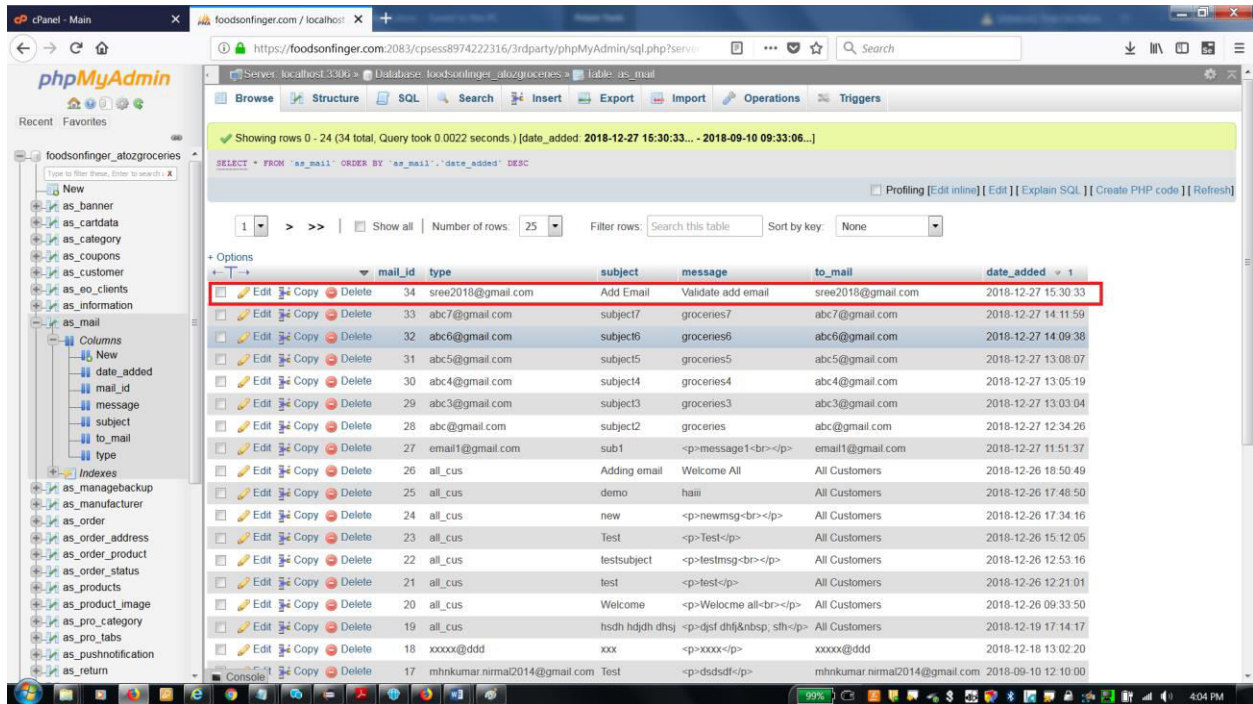
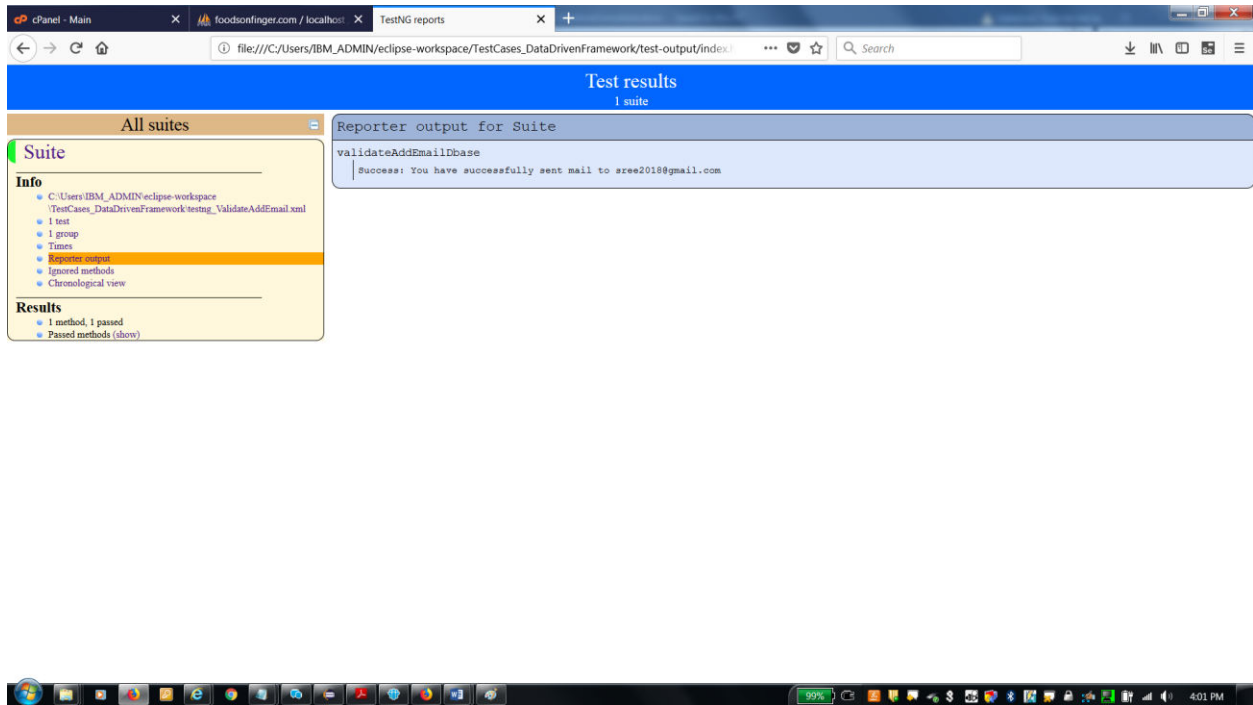
ValidatingAddEmail

com.ibm.groceries.ValidateAddEmail#validateAddEmailDbase

Messages

Success: You have successfully sent mail to sree2018@gmail.com

[back to summary](#)



Flow Information:

Initially a property file 'groceries.properties' is created in Test Data folder of the project and list of key, values for admin page url,user name,password,customer email,subject,message and 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 PageLogin class is defined with methods to locate the web elements email,password and login buttons. PageFactory is used in the constructor of this class. The method sendKeys is used to enter the email address and password. The method click is used to click on login button.

The class 'PageDashboard' is defined with methods to locate the web elements Marketing, Mail. PageFactory is used in the constructor of this class. The method click is used to click on Marketing, Mail links.

The class 'MailListPage' is defined with method clickOnAdd to click on 'Add New' link which are located using 'FindBy' annotation . PageFactory method is used in the in the constructor of this class.

The class 'SendMailPage' is used to locate the elements customer email,subject,message by using annotation FindBy. PageFactory is used in the constructor of this class.The method pouplateSubject is defined to enter subject only and click on Save button. The method populateMandatory is defined to enter customer email,subject and click on Save button and pagesource is returned by these two methods.

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

The annotation @Test is used with method validateAddEmailDbase in the class ValidateAddEmail. This method is used to get the values for username,password , customer email,subject,message and validation message etc. The get method is called to launch the web url for admin portal . The objects for classes PageLogin,PageDashboard, , MailListPage, SendMailPage, 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 the added record of mail from 'as_mail table' and next method used on result set object.

Finally the Assert is used to verify the success message and error message while adding mail. Also validated the presence of the record for added mail on admin portal and DB table too.