Day 7 Test\_case\_Solution

BaseTest

package com.ibm.test;

import java.io.File;

import java.io.IOException;

import java.sql.Driver;

import java.util.Date;

import java.util.HashMap;

import java.util.concurrent.TimeUnit;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.By;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

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

import org.testng.Assert;

import org.testng.AssertJUnit;

import org.testng.Reporter;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

import com.ibm.pages.AdminPage;

import com.ibm.pages.BannersPage;

import com.ibm.pages.LoginPage;

import com.ibm.pages.MarketingPage;

import com.ibm.pages.SystemPage;

import com.ibm.pages.UserPage;

import com.ibm.utilities.ExcelUtil;

import com.ibm.utilities.PropertiesFileHandler;

public class BaseTest {

WebDriver driver;

WebDriverWait wait;

PropertiesFileHandler propFileHandler;

HashMap<String, String> data;

@BeforeSuite

public void preSetForTest() throws IOException {

String file = "./TestData/data.properties";

propFileHandler = new PropertiesFileHandler();

data = propFileHandler.getPropertiesAsMap(file);

}

@BeforeMethod

public void Initialization() {

System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");

driver = new ChromeDriver();

wait = new WebDriverWait(driver, 60);

driver.manage().window().maximize();

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

}

@AfterMethod public void closeBrowser()

{ driver.quit(); }

@Test(priority = 1, testName = "BannerListPageValidation")

public void TestCase1() throws IOException, InterruptedException {

String url = data.get("url"); String userName = data.get("user"); String

passWord = data.get("password");

driver.get(url);

LoginPage login = new LoginPage(driver, wait); // Login to Admin Page

login.enterEmailId(userName); login.enterPassWord(passWord);

login.clickOnLogin();

AdminPage admin = new AdminPage(driver, wait); admin.clickOnCatalog(); // ToSelect Catalog

BannersPage page = new BannersPage(driver, wait); page.clickOnBanners(); // To Select Banners

Thread.sleep(10000);

int noOfEntries= page.noOfEntriesInBannerList();

System.out.println("Number of Entries: " + noOfEntries);

page.clickOnPage1(); // To Check Page2 Thread.sleep(10000);

page.clickOnPage2(); // To Check Page1 Thread.sleep(10000);

page.clickOnNext(); // To Check Next Link

page.clickOnPrevious(); // To Check Previous link

}

@Test(priority = 2, testName = "EditCoupnsInMarketingTab")

public void TestCase2() throws IOException, InterruptedException {

try {

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

String userName = data.get("user");

String passWord = data.get("password");

String couponName= data.get("couponname");

String couponCode= data.get("code");

String expectedmsg= data.get("successmsg");

driver.get(url);

LoginPage login = new LoginPage(driver, wait); // Login to Admin Page

login.enterEmailId(userName); login.enterPassWord(passWord);

login.clickOnLogin();

MarketingPage marketing = new MarketingPage(driver, wait); // To Select Marketing

marketing.clickOnMarketing(); marketing.clickOnCoupons(); // To Select Marketing

marketing.clickOnAction(); //To select Action button under Coupon List

marketing.clickOnEdit(); //To Select Edit under Action

Thread.sleep(10000); marketing.enterCouponName(couponName); //To Edit Coupon Name

marketing.enterCode(couponCode); marketing.clickOnSave(); //To save Coupon

Thread.sleep(10000);

String actualmsg= marketing.getTextToVerifySuccessMsg();

System.out.println(actualmsg);

Assert.assertTrue(actualmsg.contains(expectedmsg),"Assertion on Updated coupon code");

} catch (Exception e) { // TODO: handle exception

System.out.println(e.getMessage());

Reporter.log("Negataive Credentials Failed"); //screenshot

TakesScreenshot ts=(TakesScreenshot) driver;

File file= ts.getScreenshotAs(OutputType.FILE);

Date date =new Date(); String currentDate=date.toString().replace(":", "-");

FileUtils.copyFile(file, new

File("./screenshots/Error\_"+currentDate+".png"));

Assert.fail(); }

}

@Test(priority = 3, testName = "AddNewMailInMarketingTab")

public void TestCase3() throws IOException, InterruptedException {

try {

String url = data.get("url"); String userName = data.get("user");

String passWord = data.get("password");

String customermail= data.get("customermailid");

String subject= data.get("Subject");

String expectedmsg1= data.get("Errormsg");

driver.get(url);

LoginPage login = new LoginPage(driver, wait); // Login to Admin Page

login.enterEmailId(userName); login.enterPassWord(passWord);

login.clickOnLogin();

MarketingPage marketing = new MarketingPage(driver, wait); // To Select Marketing

marketing.clickOnMarketing(); marketing.clickOnMail(); //To select Mail

marketing.clickOnAddMail(); //To Click on new mail +

marketing.fillToDetail(); //Fill To detail

marketing.fillCustomerMailId(customermail); //Fill Customermail id

marketing.fillSubject(subject); //Fill Subject

marketing.clickOnSaveMail();

//To get the Error message text to verify it is on the same page

String saveMailErrormsg= marketing.getTextToVerifyErrorMsg();

System.out.println(saveMailErrormsg);

Assert.assertTrue(saveMailErrormsg.contains(expectedmsg1),

"Assertion on Saving a New Mail");

} catch (Exception e) { // TODO: handle exception

System.out.println(e.getMessage());

Reporter.log("Negataive Credentials Failed"); //screenshot

TakesScreenshot ts=(TakesScreenshot) driver;

File file= ts.getScreenshotAs(OutputType.FILE);

Date date =new Date(); String currentDate=date.toString().replace(":", "-");

FileUtils.copyFile(file, new

File("./screenshots/Error\_"+currentDate+".png"));

Assert.fail(); } }

@Test(priority = 4, testName = "AddNewReturnsReasonInSystemTab")

public void TestCase4() throws IOException, InterruptedException {

try {

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

String userName = data.get("user");

String passWord = data.get("password");

String returnReason = data.get("returnreason");

String expectedmsg2 = data.get("successRetutnReasonMsg");

driver.get(url);

LoginPage login = new LoginPage(driver, wait); // Login to Admin Page

login.enterEmailId(userName);

login.enterPassWord(passWord);

login.clickOnLogin();

SystemPage system = new SystemPage(driver, wait); // To Select System Tab

system.clickOnSystem(); // To select System

Thread.sleep(5000);

system.clickOnReturns(); // To click on Returns

system.clickOnReturnReasons(); // To click on Return Reasons

system.clickOnAddNewReturnReason(); // To click on Add New Return reasons

system.enterReturnReasons(returnReason); // To Enter Return Reason to Add

system.clickOnReturnReasonSave(); // To Save Return Reason

// To get the Success message after Adding New Return Reason

String saveReturnReasonMsg = system.getTextToVerifyReturnreasonMsg();

System.out.println(saveReturnReasonMsg);

Assert.assertTrue(saveReturnReasonMsg.contains(expectedmsg2), "Assertion on Adding a New Return Reason");

// To get Row count of Return Reasons

int noOfReturnReason = system.noOfReturnReasonsList();

System.out.println("Number of Return Reasons: " + noOfReturnReason);

// To get the text for Return Reason name

String searchentry = system.getTextForReturnReason();

System.out.println("Return Reason Name: " + searchentry);

} catch (Exception e) { // TODO: handle exception

System.out.println(e.getMessage());

Reporter.log("Negataive Credentials Failed"); //screenshot

TakesScreenshot ts=(TakesScreenshot) driver;

File file= ts.getScreenshotAs(OutputType.FILE);

Date date =new Date(); String currentDate=date.toString().replace(":", "-");

FileUtils.copyFile(file, new

File("./screenshots/Error\_"+currentDate+".png"));

Assert.fail(); }

}

@Test(priority = 5, testName = "EditShippingLocationInSystemTab")

public void TestCase5() throws IOException, InterruptedException {

try {

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

String userName = data.get("user");

String passWord = data.get("password");

String negativeMessage = data.get("negativemsg");

driver.get(url);

LoginPage login = new LoginPage(driver, wait); // Login to Admin Page

login.enterEmailId(userName);

login.enterPassWord(passWord);

login.clickOnLogin();

SystemPage system = new SystemPage(driver, wait); // To Select System Tab

system.clickOnSystem(); // To select System

Thread.sleep(5000);

system.clickOnShipping(); //To click on Shipping

system.clickOnShippingLocation();//To click on Shipping Location

system.clickOnShippingLocationAction(); //To click on Action button under Shipping Locations

system.clickOnShippingLocationEdit(); //TO Edit Shipping Locations

system.clearCityNameText(); //To clear city name

//system.clearSortOrderNumber(); //To clear SortOrderNumber

system.clickOnShippingLocationsSave(); //To click on save

String tootilpmessage= system.getValidationForNoCityName(); //To get Tooltip Message

System.out.println("tooltip message: " + tootilpmessage);

Assert.assertTrue(tootilpmessage.contains(negativeMessage), "Assertion on Negative Test case Validation upon Saving Shipping Locations");

Reporter.log("Assertion on Negative Test case Validation upon Saving Shipping Locations");

} catch (Exception e) { // TODO: handle exception

System.out.println(e.getMessage());

Reporter.log("Negataive Credentials Failed"); //screenshot

TakesScreenshot ts=(TakesScreenshot) driver;

File file= ts.getScreenshotAs(OutputType.FILE);

Date date =new Date(); String currentDate=date.toString().replace(":", "-");

FileUtils.copyFile(file, new

File("./screenshots/Error\_"+currentDate+".png"));

Assert.fail(); }

}

@Test(priority = 6, testName = "ToDeleteTabInCatalogTabList")

public void TestCase6() throws IOException, InterruptedException {

try {

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

String userName = data.get("user");

String passWord = data.get("password");

String expectedDeletemsg = data.get("deletedtabmsg");

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

driver.get(url);

LoginPage login = new LoginPage(driver, wait); // Login to Admin Page

login.enterEmailId(userName);

login.enterPassWord(passWord);

login.clickOnLogin();

AdminPage admin = new AdminPage(driver, wait); // To Select System Tab

admin.clickOnCatalog(); //To select Catalog

admin.clickOnTabs(); //To select Tabs

admin.clickOnTabAction(); //To click on Action button

admin.clickOnTabDelete(); //To click on Delete

admin.clickOnDeleteIt(); //To delete a selected Tab

Thread.sleep(10000);

String deletedmsg= admin.getTextToVerifyDeletedTabMsg();

System.out.println(" message: " + deletedmsg);

Assert.assertTrue(deletedmsg.contains(expectedDeletemsg),"Assertion on Deleted Tab");

Reporter.log("Assertion on Deleting a Tab");

admin.clickOnLogout(); //Logout from Admin Page

String url1 = data.get("userpage");

driver.get(url1);

UserPage user= new UserPage(driver, wait);

String deletedTab = user.getTabNotPresentTextMsg();

Assert.assertFalse(deletedTab.contains(deletedTabName), "Assertion on Deleted Tab Not present in Userpage");

//System.out.println(" to verify Tab not present: " + deletedTab);

Reporter.log("Deleted Tab not present in User Page");

} catch (Exception e) { // TODO: handle exception

System.out.println(e.getMessage());

Reporter.log("Negataive Credentials Failed"); //screenshot

TakesScreenshot ts=(TakesScreenshot) driver;

File file= ts.getScreenshotAs(OutputType.FILE);

Date date =new Date(); String currentDate=date.toString().replace(":", "-");

FileUtils.copyFile(file, new

File("./screenshots/Error\_"+currentDate+".png"));

Assert.fail();

}

}

@Test(priority = 7, testName = "ToEditUsersCredentials")

public void TestCase7() throws IOException, InterruptedException {

try {

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

String userName = data.get("user");

String passWord = data.get("password");

String usersName = data.get("username");

String userPassword= data.get("userpassword");

String confirmUserPassword= data.get("confirmuserpassword");

String userSuccesspmessage= data.get("usersuccesspmessage");

String userMailId= data.get("usermailid");

driver.get(url);

LoginPage login = new LoginPage(driver, wait); // Login to Admin Page

login.enterEmailId(userName);

login.enterPassWord(passWord);

login.clickOnLogin();

SystemPage system = new SystemPage(driver, wait); // To Select System Tab

Thread.sleep(10000);

system.clickOnSystem();

system.clickOnUsers(); //To click on Users

system.clickOnUsersAction(); //To click on Users Action

system.clickOnEditUsers(); //To click on Users Edit

system.enteUsersName(usersName); //To click on Users Username

system.enteUsersPassword(userPassword); //To click on Users Password

system.enteUsersConfirmPassword(confirmUserPassword); //To click on Users confirm password

system.clickOnUsersSave(); //To click on Users Save

//To validate the presence of the Edited Users

String editedUsersMsg= system.getValidationToEditUsername();

Assert.assertTrue(editedUsersMsg.contains(userSuccesspmessage), "Assertion on Updating a user ");

Reporter.log("Assertion on Updating a user");

//To Logout from Admin page

AdminPage admin = new AdminPage(driver, wait);

admin.clickOnLogout();

// Login to admin portal using Edited user credential to verify the edited user have the access to admin portal

login.enterUserEmail(userMailId);

login.enterUserPassword(userPassword);

login.clickOnLogin();

} catch (Exception e) { // TODO: handle exception

System.out.println(e.getMessage());

Reporter.log("Negataive Credentials Failed"); //screenshot

TakesScreenshot ts=(TakesScreenshot) driver;

File file= ts.getScreenshotAs(OutputType.FILE);

Date date =new Date(); String currentDate=date.toString().replace(":", "-");

FileUtils.copyFile(file, new

File("./screenshots/Error\_"+currentDate+".png"));

Assert.fail();

}

}

}

LoginPage

package com.ibm.pages;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

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

import org.testng.annotations.Test;

public class LoginPage {

@FindBy(xpath="//\*[@name='email']")

WebElement emailEle;

@FindBy(xpath="//\*[@name='pword']")

WebElement passwordEle;

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

WebElement loginEle;

WebDriverWait wait;

WebDriver driver;

public LoginPage(WebDriver driver,WebDriverWait wait) {

PageFactory.initElements(driver, this);

this.driver=driver;

this.wait=wait;

}

public void enterEmailId(String user)

{

emailEle.sendKeys(user);

}

public void enterPassWord(String password)

{

passwordEle.sendKeys(password);

}

public void clickOnLogin()

{

loginEle.click();

}

public void enterUserEmail(String usermailid)

{

emailEle.sendKeys(usermailid);

}

public void enterUserPassword(String userpassword)

{

passwordEle.sendKeys(userpassword);

}

}

--

SystemPage

package com.ibm.pages;

import java.util.List;

import org.openqa.selenium.By;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindAll;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

import org.openqa.selenium.support.ui.ExpectedConditions;

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

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

public class SystemPage {

@FindBy(xpath = "//a[contains(text(),'System')]") // To select System

WebElement systemEle;

@FindBy(xpath="(//\*[contains(text(),'Returns')])[2]") //To click on Returns

WebElement returnsEle;

@FindBy(xpath="//a[contains(text(),'Return Reasons')]") //To click on Returns Reasons

WebElement returnReasonsEle;

@FindBy(xpath="//a[@title='Add New']") //To Add New Return Reasons

WebElement newReturnReasonEle;

@FindBy(xpath="//input[@name='name']") //To add reasons

WebElement addReturnReasonsEle;

@FindBy(xpath="//button[@title='Save']") //To save

WebElement saveReturnReasonEle;

@FindAll(@FindBy(xpath = "//\*[@id='page-wrapper']/div/div[2]")) //Success message upon saving Return Reason

WebElement addReturnReasonMsgEle;

@FindAll(@FindBy(xpath = "//table[@id='dataTableExample2']/tbody/tr")) //To get Row count for return reasons

List<WebElement> ReturnReasonCountEle;

@FindBy(xpath="//table[@id='dataTableExample2']/tbody/tr[7]/td[2]") //To get the Return Reason Name

WebElement returnReasonRowEle;

@FindBy(xpath="(//\*[contains(text(),'Shipping')])[1]") //To click on Shipping

WebElement shippingEle;

@FindBy(xpath="//\*[contains(text(),' Shipping Locations')]") //To click on Shipping Locations

WebElement shippingLocationEle;

@FindBy(xpath="(//button[@class='btn dropdown-toggle btn-primary'] )[1]") //To click on Action button under Shipping Locations

WebElement shippingActionEle;

@FindBy(linkText ="Edit") //To Edit the Shipping Locations

WebElement editShippingLocationEle;

@FindBy(xpath="//input[@name='name']") //To clear city name Text

WebElement cityNameEle;

@FindBy(xpath="//input[@name='sort']") //To clear sort order

WebElement sortOrderEle;

@FindBy(xpath="//button[@title='Save']") //To save

WebElement shippingLocationsSaveEle;

@FindBy(xpath="//\*[contains(text(),' Users')]") //To click on Users

WebElement usersEle;

@FindBy(xpath="(//button[@class='btn dropdown-toggle btn-primary'])[4]") //To click on Action button under Users

WebElement usersActionEle;

@FindBy(linkText ="Edit") //To Edit the Users

WebElement editUsersEle;

//To Edit the Users name

@FindBy(xpath="//input[@name='username']")

WebElement usersNameEle;

//To give Password

@FindBy(xpath="//input[@name='pword']")

WebElement usersPasswordEle;

//To confirm Password

@FindBy(xpath="//input[@name='cpword']")

WebElement confirmPasswordEle;

@FindBy(xpath="//button[@title='Save']") //To save Users

WebElement usersSaveEle;

@FindBy(xpath="//\*[@id='page-wrapper']/div/div[2]") //To get success message of users update

WebElement userSuccessMsgEle;

//\*\*\*\*\*\*\*\*\*\*\*\*\*

WebDriverWait wait;

WebDriver driver;

JavascriptExecutor js;

// Constructor

public SystemPage(WebDriver driver, WebDriverWait wait) {

PageFactory.initElements(driver, this);

this.driver = driver;

this.wait = wait;

this.js = (JavascriptExecutor) driver;

}

public void clickOnSystem()

{

systemEle.click();

}

public void clickOnReturns()

{

returnsEle.click();

}

public void clickOnReturnReasons()

{

returnReasonsEle.click();

}

public void clickOnAddNewReturnReason()

{

newReturnReasonEle.click();

}

public void enterReturnReasons(String returnreason)

{

addReturnReasonsEle.sendKeys(returnreason);

}

public void clickOnReturnReasonSave()

{

saveReturnReasonEle.click();

}

public String getTextToVerifyReturnreasonMsg()

{

wait.until(ExpectedConditions.presenceOfElementLocated(By.xpath("//div[contains(text(),'Success: You have successfully added return reason!')]")));

String saveReturnReasonMsg = addReturnReasonMsgEle.getText();

return saveReturnReasonMsg;

}

public Integer noOfReturnReasonsList() {

int noOfReturnReason = ReturnReasonCountEle.size();

return noOfReturnReason;

}

public String getTextForReturnReason()

{

String searchentry= null;

for(int i=1; i <=noOfReturnReasonsList(); i++)

{

searchentry= driver.findElement(By.xpath("//table[@id='dataTableExample2']/tbody/tr[" + i + "]/td[2]")).getText();

}

return searchentry;

}

public void clickOnShipping()

{

shippingEle.click();

}

public void clickOnShippingLocation()

{

shippingLocationEle.click();

}

public void clickOnShippingLocationAction()

{

shippingActionEle.click();

}

public void clickOnShippingLocationEdit()

{

editShippingLocationEle.click();

}

public void clearCityNameText()

{

cityNameEle.clear();

}

public void clearSortOrderNumber()

{

sortOrderEle.clear();

}

public void clickOnShippingLocationsSave()

{

shippingLocationsSaveEle.click();

}

public String getValidationForNoCityName()

{

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

return tootilpmessage;

}

public void clickOnUsers()

{

usersEle.click();

}

public void clickOnUsersAction()

{

usersActionEle.click();

}

public void clickOnEditUsers()

{

editUsersEle.click();

}

public void enteUsersName(String username)

{

usersNameEle.clear();

usersNameEle.sendKeys(username);

}

public void enteUsersPassword(String userpassword)

{

usersPasswordEle.sendKeys(userpassword);

}

public void enteUsersConfirmPassword(String confirmuserpassword)

{

confirmPasswordEle.sendKeys(confirmuserpassword);

}

public void clickOnUsersSave()

{

usersSaveEle.click();

}

public String getValidationToEditUsername()

{

String userSuccesspmessage= userSuccessMsgEle.getText();

return userSuccesspmessage;

}

}

AdminPage

package com.ibm.pages;

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.How;

import org.openqa.selenium.support.PageFactory;

import org.openqa.selenium.support.ui.ExpectedConditions;

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

public class AdminPage {

// To select Catalog

@FindBy(xpath = "//a[@href='#']")

WebElement catalogEle;

//To select Tabs

@FindBy(xpath="//a[contains(text(),'Tabs')]")

WebElement tabsEle;

//To select Action button for a Tab

@FindBy(xpath="(//button[contains(text(),'Action')])[1]")

WebElement tabActionEle;

//To Delete Tab

@FindBy(linkText ="Delete")

WebElement tabDeleteEle;

//To click on Delete It

@FindBy(xpath="//button[@class='confirm']")

WebElement DeleteItEle;

//Message for Deleted data in tab List

@FindBy(xpath="//\*[@id='page-wrapper']/div/div[2]")

WebElement deleteMsgEle;

//Logout

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

WebElement logoutEle;

WebDriverWait wait;

WebDriver driver;

public AdminPage(WebDriver driver,WebDriverWait wait) {

PageFactory.initElements(driver, this);

this.driver=driver;

this.wait=wait;

}

public void clickOnCatalog() {

catalogEle.click();

}

public void clickOnTabs()

{

tabsEle.click();

}

public void clickOnTabAction()

{

tabActionEle.click();

}

public void clickOnTabDelete()

{

tabDeleteEle.click();

}

public void clickOnDeleteIt()

{

DeleteItEle.click();

}

public String getTextToVerifyDeletedTabMsg() {

wait.until(ExpectedConditions.presenceOfElementLocated(By.xpath("//\*[contains(text(),'You have successfully deleted data!')]")));

String deletedmsg = deleteMsgEle.getText();

return deletedmsg ;

}

public void clickOnLogout()

{

logoutEle.click();

}

}

Data.properties

url=https://atozgroceries.com/admin

user=demo@atozgroceries.com

password=456789

couponname=10% OFF on GROC 2000

code=GROC

successmsg=Success: You have successfully updated coupon code!

customermailid=shobha.mce2003@gmail.com

Subject=Testing

Errormsg=Message is required!

returnreason=Not the same order

successRetutnReasonMsg=Success: You have successfully added return reason!

negativemsg=Please fill out this field.

deletedtabmsg=You have successfully deleted data!

userpage= https://atozgroceries.com/

deletedTabName=Tab\_To\_delete

username=shobha12

userpassword=123456

confirmuserpassword=123456

usersuccesspmessage=Success: You have successfully updated user!

[usermailid=testinguser1@gmail.com](mailto:usermailid=testinguser1@gmail.com)

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name=*"Suite"*>

<test thread-count=*"5"* name=*"Test"*>

<classes>

<class name=*"com.ibm.test.BaseTest"*>

<methods><exclude name=*"TestCase5"*></exclude></methods>

<methods><exclude name=*"TestCase6"*></exclude></methods>

<methods><include name=*"TestCase7"*></include></methods>

</class>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->