URL=https://iiq72user.sit01.service.dev/identityiq/login.jsf?prompt=true

Userid=571246

Password=admin

ChromeDriver=C:\\Users\\moharas2\\eclipse-workspace\\iiqbdd\\chromedriver\\chromedriver.exe

Screenshot=C:\\Users\\moharas2\\eclipse-workspace\\iiqbdd\\

https://www.techbeamers.com/generate-reports-selenium-webdriver/

package ocp.stepDefinition;

import java.io.IOException;

import org.openqa.selenium.By;

import cucumber.api.java.en.And;

import cucumber.api.java.en.Then;

import cucumber.api.java.en.When;

import ocp.testBase.TestBasic;

import ocp.testBase.TestStorage;

@SuppressWarnings("unused")

public class Test8HomepageValidation extends TestBasic{

public static TestStorage ts = new TestStorage();

private static int i = 0;

private static int ij = 0;

private static int val0 ;

private static int jvalue = 0 ;

private static int y = 1;

public Test8HomepageValidation() throws IOException {

super();

// TODO Auto-generated constructor stub

}

@When("^Verifying Attribute \"([^\"]\*)\" is Present in Homepage$")

public void verifyingAttributeIsPresentInHomepage(String arg1) throws Throwable {

try {

String Value = getValue(xProp.getProperty("start")+arg1+xProp.getProperty("end"));

if (Value.equalsIgnoreCase(arg1)) {

System.out.println("Element present" + arg1);

logger(pass, "Attribute Validation", 0, arg1+" is Present");

}

else

{

System.out.println("Element not present" + arg1);

logger(fail, "Attribute Validation", 0, arg1+" is not Present");

}

}catch(Exception e)

{

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

logger(fail, "Attribute Validation", 0, "Error in finding attribute");

}

}

@Then("^Attribute \"([^\"]\*)\" Should Be Present Immediate Left To The ReqAttribute \"([^\"]\*)\" Column$")

public void attributeShouldBePresentImmediateLeftToTheReqAttributeColumn(String arg1, String arg2)

throws Throwable {

smallWait();

try{

int attribute1 = 0;

int attribute2 = 0;

String value1 = getLocationLeftHeader(xProp.getProperty("filter\_head"), arg1);

String value2 = getLocationRightHeader(xProp.getProperty("filter\_head"), arg2);

attribute1 = Integer.parseInt(value1)+1;

attribute2 = Integer.parseInt(value2);

if (attribute1==attribute2)

{

screenshot(pass);

logger(pass, "Attribute Position Validation", 0, arg1+" is Present immediate left to "+arg2);

}

else {

screenshot(pass);

logger(fail, "Attribute Position Validation", 0, arg1+" is Not Present immediate left to "+arg2);

}

}catch(Exception e)

{

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

logger(fail, "Attribute Position Validation", 0, "Error in finding elements");

}

}

@When("^Click on Profile To Move To Homepage$")

public void clickOnProfileToMoveToHomepage() throws Throwable {

smallWait();

try{

click(xProp.getProperty("start")+"Profiles"+xProp.getProperty("end"));

screenshot(pass);

logger(pass , "Profile Click", 0, "Clicked on Profile ICON/button" );

}catch(Exception e)

{

screenshot(fail);

logger(fail, "Profile Click", 0, "Unable to find the Profile ICON/button in the page" );

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

}

}

@And("^OFF Profile Toggle In The Homepage$")

public void offProfileToggleInTheHomepage() throws Throwable {

smallWait();

try {

click(xProp.getProperty("tog\_pro"));

screenshot(pass);

logger(pass , "Profile Toggle", 0, "Clicked on Profile Toggle to OFF" );

}catch (Exception e)

{

screenshot(fail);

logger(fail, "Profile toggle", 0, "Unable to find the Profile toggle in Homepage" );

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

}

}

@SuppressWarnings("static-access")

@Then("^Searching For the Profile and Validate CMF\_Location CMF\_Status and Segment Value In Homepage$")

public void searchingForTheProfileAndValidateCMF\_LocationCMF\_StatusAndSegmentValueInHomepage() throws Throwable {

waitTime();

try{

//searching the text box and passing the value

for( ; i < ts.StorageSize(); )

{ sendingValue(xProp.getProperty("tbox\_search\_profile"),ts.getVal(i) );

System.out.println("value passed to the input box "+ts.getVal(i));

logger(pass, "Profile Search", 0, "Passing values to the text box");

screenshot(pass);

try{

//click on search button

smallWait();

click(xProp.getProperty("btn\_search\_profile"));

System.out.println("Clicked on search");

logger(pass, "Profile Search", 1, "Clicked on search button in Home page");

screenshot(pass);

try {

waitTime();

String val1 = getValue(xProp.getProperty("home\_cmf\_loc"));

System.out.println("Got the location value");

if (val1.equalsIgnoreCase(ts.getVal(i+2)))

{

click(xProp.getProperty("home\_cmf\_loc"));

screenshot(pass);

logger(pass, "CMF Location", 2, "CMF Location is passed for the "+ts.getVal(i)+" profile");

}

else {

screenshot(pass);

logger(fail, "CMF Location", 2, "CMF Location failed for the "+ts.getVal(i)+" profile");

}

try {

smallWait();

String val2 = getValue(xProp.getProperty("home\_cmf\_sta"));

System.out.println("Got the status value");

if (val2.equalsIgnoreCase(ts.getVal(i+1)))

{ click(xProp.getProperty("home\_cmf\_sta"));

screenshot(pass);

logger(pass, "CMF Status", 3, "CMF Status is passed for the "+ts.getVal(i)+" profile");

}

else {

screenshot(pass);

logger(fail, "CMF Status", 3, "CMF Status failed for the "+ts.getVal(i)+" profile");

} try {

smallWait();

String val3 = getValue(xProp.getProperty("home\_segment"));

System.out.println("got the segment value");

if (val3.equalsIgnoreCase(ts.getVal(i+3))) { click(xProp.getProperty("home\_segment"));

screenshot(pass);

logger(pass, "Segment", 4, "segment is passed for the "+ts.getVal(i)+" profile");

}

else {

screenshot(pass);

logger(fail, "Segment", 4, "Segment failed for the "+ts.getVal(i)+" profile"); }

}catch (Exception e) {

logger(fail,"Segment", 4,"ERROR ON FINDING THE ELEMENT");

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

}catch (Exception e) {

logger(fail,"CMF Status", 3,"ERROR ON FINDING THE ELEMENT");

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

}catch(Exception e) {

logger(fail,"CMF Location", 2,"ERROR ON FINDING THE ELEMENT");

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

}

}catch (Exception e){

//if search button not enabled

logger(fail, "Profile Search", 1, "Unable to Click on search button in Home page");

screenshot(fail);

System.out.println("Error message: "+e.getLocalizedMessage());}

i=i+4;

break;

}

}catch(Exception ex)

{

//if text box or parameter not found

logger(fail, "Profile Search", 0, "unable to passing values to the text box");

screenshot(fail);

System.out.println("Error message "+ ex.getLocalizedMessage());

}}

@SuppressWarnings("static-access")

@And("^Adding RM \"([^\"]\*)\" RA \"([^\"]\*)\" To The Submission$")

public void addingRMRAToTheSubmission(String arg1, String arg2) throws Throwable {

smallWait();

String value = "cap";

try {

driver.findElement(By.xpath("//div[@class='hero\_\_details']")).click();

clickSendValuesAndEnter(xProp.getProperty("txt\_rm\_inp"), configProp.getProperty(arg1));

waitTime();

driver.findElement(By.xpath("//div[@class='hero\_\_details']")).click();

clickSendValuesAndEnter(xProp.getProperty("txt\_ra\_inp"), configProp.getProperty(arg2));

value="pass";

screenshot(pass);

smallWait();

if (((arg1.equalsIgnoreCase("RM1")) && (arg2.equalsIgnoreCase("RA1")))

|| ((arg1.equalsIgnoreCase("RM1")) && (arg2.equalsIgnoreCase("RA2")))

|| ((arg1.equalsIgnoreCase("RM1")) && (arg2.equalsIgnoreCase("RA3")))

|| ((arg1.equalsIgnoreCase("RM1")) && (arg2.equalsIgnoreCase("RA4")))

|| ((arg1.equalsIgnoreCase("RM1")) && (arg2.equalsIgnoreCase("RA5"))))

{

ts.r1Storage(profile);

}

else if (((arg1.equalsIgnoreCase("RM2")) && (arg2.equalsIgnoreCase("RA1")))

|| ((arg1.equalsIgnoreCase("RM2")) && (arg2.equalsIgnoreCase("RA2")))

|| ((arg1.equalsIgnoreCase("RM2")) && (arg2.equalsIgnoreCase("RA3")))

|| ((arg1.equalsIgnoreCase("RM2")) && (arg2.equalsIgnoreCase("RA4")))

|| ((arg1.equalsIgnoreCase("RM2")) && (arg2.equalsIgnoreCase("RA5"))))

{

ts.r2Storage(profile);

}

else if (((arg1.equalsIgnoreCase("RM3")) && (arg2.equalsIgnoreCase("RA1")))

|| ((arg1.equalsIgnoreCase("RM3")) && (arg2.equalsIgnoreCase("RA2")))

|| ((arg1.equalsIgnoreCase("RM3")) && (arg2.equalsIgnoreCase("RA3")))

|| ((arg1.equalsIgnoreCase("RM3")) && (arg2.equalsIgnoreCase("RA4")))

|| ((arg1.equalsIgnoreCase("RM3")) && (arg2.equalsIgnoreCase("RA5"))))

{

ts.r3Storage(profile);

}

else if (((arg1.equalsIgnoreCase("RM4")) && (arg2.equalsIgnoreCase("RA1")))

|| ((arg1.equalsIgnoreCase("RM4")) && (arg2.equalsIgnoreCase("RA2")))

|| ((arg1.equalsIgnoreCase("RM4")) && (arg2.equalsIgnoreCase("RA3")))

|| ((arg1.equalsIgnoreCase("RM4")) && (arg2.equalsIgnoreCase("RA4")))

|| ((arg1.equalsIgnoreCase("RM4")) && (arg2.equalsIgnoreCase("RA5"))))

{

ts.r4Storage(profile);

}

else if (((arg1.equalsIgnoreCase("RM5")) && (arg2.equalsIgnoreCase("RA1")))

|| ((arg1.equalsIgnoreCase("RM5")) && (arg2.equalsIgnoreCase("RA2")))

|| ((arg1.equalsIgnoreCase("RM5")) && (arg2.equalsIgnoreCase("RA3")))

|| ((arg1.equalsIgnoreCase("RM5")) && (arg2.equalsIgnoreCase("RA4")))

|| ((arg1.equalsIgnoreCase("RM5")) && (arg2.equalsIgnoreCase("RA5"))))

{

ts.r5Storage(profile);

}

}catch (Exception e)

{

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

logger(fail, "RM", 0, "unable to passing values to the text box");

screenshot(fail);

}

screenshot(value);

}

@SuppressWarnings("static-access")

@When("^Adding Values \"([^\"]\*)\"$")

public static void addingValues(String arg1)

{

ts.r2Storage(arg1);

}

@SuppressWarnings("static-access")

@When("^Making Decision on Profile As Submission \"([^\"]\*)\" Approver$")

public void makingDecisionOnProfileAsSubmissionApprover(String args1) throws Throwable {

smallWait();

try {

click(xProp.getProperty("icon\_profile"));

waitTime();waitTime();

click(xProp.getProperty("tog\_pro"));

sendingValue(xProp.getProperty("tbox\_search\_profile"), ts.getVal(ij));

click(xProp.getProperty("btn\_search\_profile"));

smallWait();smallWait();

click(xProp.getProperty("start")+ts.getVal(ij)+xProp.getProperty("end"));

ij=ij+1;

smallWait();

click(xProp.getProperty("lab\_sub\_his"));

click(xProp.getProperty("sub\_title"));

smallWait();waitTime();

System.out.println(ij+" \n"+ts.getVal(ij));

switchTab();

click(xProp.getProperty("but\_sub"));

if (args1.equalsIgnoreCase("APPROVED")) {

selectFromDropdown(xProp.getProperty("drop\_decision"), "Approved");

smallWait();

click(xProp.getProperty("but\_dec\_submit"));

smallWait();

click(xProp.getProperty("pop\_but\_dec\_confirm"));

smallWait();

System.out.println(147);

}

else if (args1.equalsIgnoreCase("For Rework"))

{

selectFromDropdown(xProp.getProperty("drop\_decision"), "For Rework");

smallWait();

click(xProp.getProperty("but\_dec\_submit"));

smallWait();

click(xProp.getProperty("pop\_but\_dec\_confirm"));

smallWait();

System.out.println(148);

}

else if (args1.equalsIgnoreCase("Declined"))

{

selectFromDropdown(xProp.getProperty("drop\_decision"), "Declined");

smallWait();

click(xProp.getProperty("but\_dec\_submit"));

smallWait();

click(xProp.getProperty("pop\_but\_dec\_confirm"));

smallWait();

System.out.println(149);

}

else

{

smallWait();

}

screenshot(pass);

logger(pass, "Decision Making", 0, "PASSED Second scenario");

}catch(Exception e)

{

screenshot(fail);

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

logger(fail, "Decision Making", 0, "FAILED Second scenario");

}

}

@And("^Changing The Search Method To RA\_RM\_RCM$")

public void changingTheSearchMethodToRA\_RM\_RCM() throws Throwable {

smallWait();

try {

selectFromDropdown(xProp.getProperty("drop\_search"), "RM, R&A or RCM");

System.out.println("Search option changed");

logger(pass, "Search Option", 0, "Search Option changed to RM, R&A or RCM");

screenshot(pass);

}catch (Exception e)

{

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

screenshot(fail);

logger(fail, "Search Option", 0, "Error occrued");

}

}

@When("^Click on Search Text Box Enter the Value \"([^\"]\*)\"$")

public void clickOnSearchTextBoxEnterTheValue(String arg1) throws Throwable {

waitTime();

try{

//searching the text box and passing the value

sendingValue(xProp.getProperty("tbox\_search\_profile"),configProp.getProperty(arg1) );

logger(pass, "Profile Search by User", 0, "Passing values to the text box");

screenshot(pass);

try{

//click on search button

click(xProp.getProperty("btn\_search\_profile"));

logger(pass, "Profile Search by User", 1, "Clicked on search button in Home page");

screenshot(pass);

}catch (Exception e){

//if search button not enabled

logger(fail, "Profile Search by User", 1, "Unable to Click on search button in Home page");

screenshot(fail);

System.out.println("Error message: "+e.getLocalizedMessage());

}

}catch(Exception ex)

{

//if text box or parameter not found

logger(fail, "Profile Search", 0, "unable to passing values to the text box");

screenshot(fail);

System.out.println("Error message "+ ex.getLocalizedMessage());

}

}

}

package ocp.stepDefinition;

import java.io.IOException;

import cucumber.api.java.en.Given;

import cucumber.api.java.en.Then;

import ocp.testBase.TestBasic;

import ocp.testBase.TestStorage;

public class Test8SubProg extends TestBasic

{

static TestStorage ts = new TestStorage();

public Test8SubProg() throws IOException {

super();

// TODO Auto-generated constructor stub

}

private static int val0;

@SuppressWarnings("static-access")

@Then("^Validate All The Mentioned Profiles Are \"([^\"]\*)\" Prsent$")

public void validateAllTheMentionedProfilesArePrsent(String arg1) throws Throwable {

if (arg1.equalsIgnoreCase("RM1"))

{

val0 = ts.r1Size();

for(int val1=0;val1<val0;val1++)

{

Boolean value = objCheck(xProp.getProperty("start")+ts.getr1Val1(val1)+xProp.getProperty("end"));

if(value.equals(true))

{

System.out.println("passed "+val1);

screenshot(pass);

logger(pass, "Profile Check", val1, "PASSED Able to find "+ts.getr1Val1(val1));

}else

{

System.out.println("failed "+val1); screenshot(fail); logger(fail,"profile Check",val1,"FAILED "+ts.getr1Val1(val1));

}

}

}

else if (arg1.equalsIgnoreCase("RM2"))

{

val0 = ts.r2Size();

for(int val1=0;val1<val0;val1++)

{

Boolean value = objCheck(xProp.getProperty("start")+ts.getr2Val1(val1)+xProp.getProperty("end"));

if(value.equals(true))

{

System.out.println("passed "+val1);

screenshot(pass);

logger(pass, "Profile Check", val1, "PASSED Able to find "+ts.getr2Val1(val1));

}else

{

System.out.println("failed "+val1); screenshot(fail); logger(fail,"profile Check",val1,"FAILED "+ts.getr2Val1(val1));

}

}

}

else if (arg1.equalsIgnoreCase("RM3"))

{

val0 = ts.r3Size();

for(int val1=0;val1<val0;val1++)

{

Boolean value = objCheck(xProp.getProperty("start")+ts.getr3Val1(val1)+xProp.getProperty("end"));

if(value.equals(true))

{

System.out.println("passed "+val1);

screenshot(pass);

logger(pass, "Profile Check", val1, "PASSED Able to find "+ts.getr3Val1(val1));

}else

{

System.out.println("failed "+val1); screenshot(fail); logger(fail,"profile Check",val1,"FAILED "+ts.getr3Val1(val1));

}

}

}

else if (arg1.equalsIgnoreCase("RM4"))

{

val0 = ts.r4Size();

for(int val1=0;val1<val0;val1++)

{

Boolean value = objCheck(xProp.getProperty("start")+ts.getr4Val1(val1)+xProp.getProperty("end"));

if(value.equals(true))

{

System.out.println("passed "+val1);

screenshot(pass);

logger(pass, "Profile Check", val1, "PASSED Able to find "+ts.getr4Val1(val1));

}else

{

System.out.println("failed "+val1); screenshot(fail); logger(fail,"profile Check",val1,"FAILED "+ts.getr4Val1(val1));

}

}

}

else if (arg1.equalsIgnoreCase("RM5"))

{

val0 = ts.r5Size();

for(int val1=0;val1<val0;val1++)

{

Boolean value = objCheck(xProp.getProperty("start")+ts.getr5Val1(val1)+xProp.getProperty("end"));

if(value.equals(true))

{

System.out.println("passed "+val1);

screenshot(pass);

logger(pass, "Profile Check", val1, "PASSED Able to find "+ts.getr5Val1(val1));

}else

{

System.out.println("failed "+val1); screenshot(fail); logger(fail,"profile Check",val1,"FAILED "+ts.getr5Val1(val1));

}

}

}

}

@SuppressWarnings("static-access")

@Given("^Get Profile Values$")

public void getProfileValues() throws Throwable {

for(int i=0;i<ts.StorageSize();i++) {

System.out.println(ts.getVal(i));

}

}

}

--

package ocp.stepDefinition;

import java.io.IOException;

import cucumber.api.java.en.Then;

import ocp.testBase.TestBasic;

import ocp.testBase.TestPartyProfileStorage;

public class Test9 extends TestBasic{

TestPartyProfileStorage tp = new TestPartyProfileStorage();

public Test9() throws IOException {

super();

}

private static String input;

private static String decision;

@SuppressWarnings("static-access")

@Then("^Validate All The Mentioned Profiles Are Present for user as Participant \"([^\"]\*)\"$")

public void validateAllTheMentionedProfilesArePresentForUserAsParticipant(String arg1) throws Throwable {

if (arg1.equalsIgnoreCase("USER1"))

{

for(int x=0; x<tp.u1Size();x++)

{

sendingValue(xProp.getProperty("tbox\_search\_profile"), configProp.getProperty("creator"));

click(xProp.getProperty("btn\_search\_profile"));

Boolean value = objCheck(xProp.getProperty("start")+tp.u1getValue(x)+xProp.getProperty("end"));

input = tp.u1getValue(x);

if(value.equals(true))

{

System.out.println("Passed - profile found in list: "+tp.u1getValue(x));

screenshot(pass);

logger(pass, "Participant Profile", 0, "Able to find profile ("+tp.u1getValue(x)+") in homepage When Profile Toggle is ON");

decision = tp.approvalType(x);

makeDecisionOnProfileSubmission(decision, input);

}

else

{

System.out.println("Failed - profile found in list: "+tp.u1getValue(x));

screenshot(fail);

logger(fail, "Participant Profile", 0, "Unable to find profile ("+tp.u1getValue(x)+") in homepage When Profile Toggle is ON");

}

}

}

else if (arg1.equalsIgnoreCase("USER2"))

{

for(int x=0; x<tp.u2Size();x++)

{

sendingValue(xProp.getProperty("tbox\_search\_profile"), configProp.getProperty("creator"));

click(xProp.getProperty("btn\_search\_profile"));

Boolean value = objCheck(xProp.getProperty("start")+tp.u2getValue(x)+xProp.getProperty("end"));

input = tp.u2getValue(x);

if(value.equals(true))

{

System.out.println("Passed - profile found in list: "+tp.u2getValue(x));

screenshot(pass);

logger(pass, "Participant Profile", 0, "Able to find profile ("+tp.u2getValue(x)+") in homepage When Profile Toggle is ON");

decision = tp.approvalType(x);

makeDecisionOnProfileSubmission(decision, input);

}

else

{

System.out.println("Failed - profile found in list: "+tp.u2getValue(x));

screenshot(fail);

logger(fail, "Participant Profile", 0, "Unable to find profile ("+tp.u2getValue(x)+") in homepage When Profile Toggle is ON");

}

}

}

else if (arg1.equalsIgnoreCase("USER3"))

{

for(int x=0; x<tp.u3Size();x++)

{

sendingValue(xProp.getProperty("tbox\_search\_profile"), configProp.getProperty("creator"));

click(xProp.getProperty("btn\_search\_profile"));

Boolean value = objCheck(xProp.getProperty("start")+tp.u3getValue(x)+xProp.getProperty("end"));

input = tp.u3getValue(x);

if(value.equals(true))

{

System.out.println("Passed - profile found in list: "+tp.u3getValue(x));

screenshot(pass);

logger(pass, "Participant Profile", 0, "Able to find profile ("+tp.u3getValue(x)+") in homepage When Profile Toggle is ON");

decision = tp.approvalType(x);

makeDecisionOnProfileSubmission(decision, input);

}

else

{

System.out.println("Failed - profile found in list: "+tp.u3getValue(x));

screenshot(fail);

logger(fail, "Participant Profile", 0, "Unable to find profile ("+tp.u3getValue(x)+") in homepage When Profile Toggle is ON");

}

}

}

else if (arg1.equalsIgnoreCase("USER4"))

{

for(int x=0; x<tp.u4Size();x++)

{

sendingValue(xProp.getProperty("tbox\_search\_profile"), configProp.getProperty("creator"));

click(xProp.getProperty("btn\_search\_profile"));

Boolean value = objCheck(xProp.getProperty("start")+tp.u4getValue(x)+xProp.getProperty("end"));

input = tp.u4getValue(x);

if(value.equals(true))

{

System.out.println("Passed - profile found in list: "+tp.u4getValue(x));

screenshot(pass);

logger(pass, "Participant Profile", 0, "Able to find profile ("+tp.u4getValue(x)+") in homepage When Profile Toggle is ON");

decision = tp.approvalType(x);

makeDecisionOnProfileSubmission(decision, input);

}

else

{

System.out.println("Failed - profile found in list: "+tp.u4getValue(x));

screenshot(fail);

logger(fail, "Participant Profile", 0, "Unable to find profile ("+tp.u4getValue(x)+") in homepage When Profile Toggle is ON");

}

}

}

else if (arg1.equalsIgnoreCase("USER5"))

{

for(int x=0; x<tp.u5Size();x++)

{

sendingValue(xProp.getProperty("tbox\_search\_profile"), configProp.getProperty("creator"));

click(xProp.getProperty("btn\_search\_profile"));

Boolean value = objCheck(xProp.getProperty("start")+tp.u5getValue(x)+xProp.getProperty("end"));

input = tp.u5getValue(x);

if(value.equals(true))

{

System.out.println("Passed - profile found in list: "+tp.u5getValue(x));

screenshot(pass);

logger(pass, "Participant Profile", 0, "Able to find profile ("+tp.u5getValue(x)+") in homepage When Profile Toggle is ON");

decision = tp.approvalType(x);

makeDecisionOnProfileSubmission(decision, input);

}

else

{

System.out.println("Failed - profile found in list: "+tp.u5getValue(x));

screenshot(fail);

logger(fail, "Participant Profile", 0, "Unable to find profile ("+tp.u5getValue(x)+") in homepage When Profile Toggle is ON");

}

}

}

}

public void makeDecisionOnProfileSubmission(String args1, String profileIn) throws Throwable {

try {

click(xProp.getProperty("icon\_profile"));

waitTime();waitTime();

clearValue(xProp.getProperty("tbox\_search\_profile"));

sendingValue(xProp.getProperty("tbox\_search\_profile"), profileIn);

click(xProp.getProperty("btn\_search\_profile"));

smallWait();smallWait();

click(xProp.getProperty("start")+profileIn+xProp.getProperty("end"));

waitTime();smallWait();

driver.navigate().refresh();

click(xProp.getProperty("sub\_title"));

smallWait();waitTime();

System.out.println(profileIn);

switchTab();

driver.navigate().refresh();waitTime();

click(xProp.getProperty("but\_sub"));

screenshot(pass);

if (args1.equalsIgnoreCase("APPROVED")) {

selectFromDropdown(xProp.getProperty("drop\_decision"), "Approved");

smallWait();

screenshot(pass);

click(xProp.getProperty("but\_dec\_submit"));

smallWait();

click(xProp.getProperty("pop\_but\_dec\_confirm"));

smallWait();

System.out.println(147);

click(xProp.getProperty("icon\_profile"));

}

else if (args1.equalsIgnoreCase("For Rework"))

{

selectFromDropdown(xProp.getProperty("drop\_decision"), "For Rework");

smallWait();

click(xProp.getProperty("but\_dec\_submit"));

smallWait();screenshot(pass);

click(xProp.getProperty("pop\_but\_dec\_confirm"));

smallWait();

System.out.println(148);

click(xProp.getProperty("icon\_profile"));

}

else if (args1.equalsIgnoreCase("Declined"))

{

selectFromDropdown(xProp.getProperty("drop\_decision"), "Declined");

smallWait();

click(xProp.getProperty("but\_dec\_submit"));

smallWait();screenshot(pass);

click(xProp.getProperty("pop\_but\_dec\_confirm"));

smallWait();

System.out.println(149);

click(xProp.getProperty("icon\_profile"));

}

else

{

smallWait();

}

System.out.println("Done --");

screenshot(pass);

logger(pass, "Decision Making", 0, "PASSED Second scenario");

}catch(Exception e)

{

screenshot(fail);

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

logger(fail, "Decision Making", 0, "FAILED Second scenario");

}

}

}

--

package ocp.storage;

import java.io.IOException;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.Date;

import ocp.testBase.TestBasic;

public class HomeStorage extends TestBasic {

public HomeStorage() throws IOException {

super();

}

public static ArrayList<String> proId = new ArrayList<String>();

public static ArrayList<String> proName = new ArrayList<String>();

public static ArrayList<String> cmfLocation = new ArrayList<String>();

public static ArrayList<String> cmfStatus = new ArrayList<String>();

public static ArrayList<String> ccr = new ArrayList<String>();

public static ArrayList<String> segment = new ArrayList<String>();

public static ArrayList<Date> nextReviewDate = new ArrayList<Date>();

public static ArrayList<Date> rpiDisDate = new ArrayList<Date>();

////

public static void hStoreprofileId(String value) {

proId.add(value);

}

public static void hStoreProfileName(String value)

{

proName.add(value);

}

public static void hStoreCmfLoc(String value) {

cmfLocation.add(value);

}

public static void hStoreCmfStatus(String value) {

cmfStatus.add(value);

}

public static void hStoreCCR(String value) {

ccr.add(value);

}

public static void hStoreSegment(String value) {

segment.add(value);

}

public static void hStoreNextReviewDate(String value) throws ParseException {

if(value.equals(null))

{

nextReviewDate.add(null);

}

else if (value.contentEquals(""))

{

SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");

Date date = format.parse("01/01/2019");

nextReviewDate.add(date);

}

else {

SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");

Date date = format.parse(value);

nextReviewDate.add(date);

}

}

public static void hStoreRpiDiscussionDate(String value) throws ParseException {

if(value.equals(null))

{

rpiDisDate.add(null);

}

else if(value.equals(""))

{

SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");

Date date1 = format.parse("01/01/2019");

rpiDisDate.add(date1);

}

else {

SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");

Date date1 = format.parse(value);

rpiDisDate.add(date1);

}

}

public static int hSizeProfileId() {

int value = proId.size(); return value;

}

public static int hSizeProfileName() {

int value = proName.size(); return value;

}

public static int hSizeCmfLoc() {

int value = cmfLocation.size(); return value;

}

public static int hSizeCmfStatus() {

int value = cmfStatus.size(); return value;

}

public static int hSizeCcr() {

int val = ccr.size(); return val;

}

public static int hSizeSegment() {

int v = segment.size(); return v;

}

public static int hSizeNextReviewDate() {

int value = nextReviewDate.size(); return value;

}

public static int hSizeRpiDisDate() {

int alue = rpiDisDate.size(); return alue;

}

public static String getSegment(int a) {

String value = segment.get(a); return value;

}

public static String getCCR(int a) {

String value = ccr.get(a); return value;

}

public static String getCmfStatu(int a) {

String value = cmfStatus.get(a); return value;

}

public static String getCmfLoca(int a) {

String value = cmfLocation.get(a); return value;

}

public static String getProName(int a) {

String value = proName.get(a); return value;

}

public static String getProId(int a) {

String value = proId.get(a); return value;

}

public static String getNextReviewDate(int a) {

SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");

Date date = nextReviewDate.get(a);

String value = format.format(date);

if(value.equals("01/01/2019"))

{

value="";

}

return value;

}

public static String getRpiDiscussionDate(int a) {

SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");

Date date = rpiDisDate.get(a);

String value = format.format(date);

if(value.equals("01/01/2019"))

{

value="";

}

return value;

}

public static void sortAscending() {

Collections.sort(proId);

Collections.sort(proName);

Collections.sort(cmfLocation);

Collections.sort(cmfStatus);

Collections.sort(ccr);

Collections.sort(segment);

Collections.sort(nextReviewDate);

Collections.sort(rpiDisDate);

}

@SuppressWarnings({ "unchecked", "rawtypes" })

public static void sortDescending() {

Comparator c = Collections.reverseOrder();

Collections.sort(proId,c);

Collections.sort(proName,c);

Collections.sort(cmfLocation,c);

Collections.sort(cmfStatus,c);

Collections.sort(ccr,c);

Collections.sort(segment,c);

Collections.sort(nextReviewDate,c);

Collections.sort(rpiDisDate,c);

}

}

--

package ocp.testBase;

import gherkin.formatter.Reporter;

import java.awt.AWTException;

import java.awt.Robot;

import java.awt.event.KeyEvent;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Iterator;

import java.util.List;

import java.util.Properties;

import java.util.Set;

import java.util.concurrent.TimeUnit;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.By;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.Keys;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.chrome.ChromeOptions;

import org.openqa.selenium.interactions.Actions;

import org.openqa.selenium.remote.server.handler.ImplicitlyWait;

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

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

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

import com.relevantcodes.extentreports.ExtentReports;

import com.relevantcodes.extentreports.ExtentTest;

import com.relevantcodes.extentreports.LogStatus;

@SuppressWarnings("unused")

public class TestBasic {

public static WebDriver driver;

public static Properties configProp;

public static Properties xProp;

public static Properties resLog;

public static BufferedReader brConfig;

public static BufferedReader brXpath;

public static BufferedWriter brRes;

public static BufferedWriter logs;

public static FileWriter logfile;

public static BufferedWriter errorLog;

public static FileWriter errorLogfile;

public static String cur\_date = getCurrentDate();

public static String cur\_time = getCurrentTime();

public static String pass = "Pass";

public static String fail = "Fail";

public static String error = "Error";

public static String member = null;

public String proName = null;

public static String profile;

public static String line ="\*\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*----------------------------------------\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"

+ "Date|Time|Status|UserID|MethodName|StepNo|Comments\n";

public TestBasic() throws IOException{

readConfig();

}

public static void highlight(WebDriver driver, WebElement ele)

{

JavascriptExecutor js = (JavascriptExecutor)driver;

js.executeScript("arguments[0].setAttribute('style', 'background: #ffcccc; border: 2px solid red; 5px;');", ele);

try{

Thread.sleep(50);

}catch (Exception e)

{

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

}

js.executeScript("arguments[0].setAttribute('style','border: solid 0px white;padding: 5px;');", ele);

}

//Wait COnditions for object/element visible

public static void waitForCondition(WebElement string)

{

try

{

WebDriverWait wait = new WebDriverWait(driver, 10);

wait.until(ExpectedConditions.visibilityOf(string));

}

catch (Exception ex)

{

ex.printStackTrace();

}

}

public static void waitTime() throws InterruptedException{

Thread.sleep(1000);

}

//Configuration file and Property file reading

public static void readConfig() throws IOException

{

brConfig = new BufferedReader(new FileReader("./Config/config.properties"));

configProp = new Properties();

configProp.load(brConfig);

brXpath = new BufferedReader(new FileReader("./Config/xpath.properties"));

xProp = new Properties();

xProp.load(brXpath);

}

public static int getSelectedOption(String Xpath) throws InterruptedException

{

waitTime();

Select select = new Select(driver.findElement(By.xpath(Xpath)));

WebElement option = select.getFirstSelectedOption();

String defaultItem = option.getText();

int count = Integer.parseInt(defaultItem);

return count ;

}

public static int totalNoProfiles(String xpath) throws InterruptedException {

waitTime();waitTime();

String cnt = getValue(xpath);

String [] c = cnt.split("\\s");

int count = Integer.parseInt(c[0]);

return count;

}

//log file produce

public static void writeLog(){

try {

logfile = new FileWriter(System.getProperty("user.dir")+"/log/LogReport"+cur\_date+".log",true);

logs = new BufferedWriter(logfile);

logs.write(line);

} catch (IOException e) {

System.out.println("File not Found or File Path not found\n"+e.getMessage());

}

}

public static void logger(String status,String method,int subStep,String comment) throws IOException{

String c\_date = getCurrentDate();

String c\_time = getCurrentTime();

if(status.equalsIgnoreCase("PASS")){

logs.write(c\_date+"|"+c\_time+"|PASS|"+member+"|"+method+"|"+subStep+"|"+comment+"\n");

}

else if (status.equalsIgnoreCase("FAIL"))

{

logs.write(c\_date+"|"+c\_time+"|FAIL|"+member+"|"+method+"|"+subStep+"|"+comment+"\n");

}

}

public static void errorWriteLog(String status,String clsname,String subStep,String comments) {

try {

errorLogfile = new FileWriter(configProp.getProperty("Log\_file\_path")+configProp.getProperty("Log\_file\_name"));

errorLog = new BufferedWriter(logfile);

} catch (IOException e) {

System.out.println("File not Found or File Path not found\n"+e.getMessage());

}

}

public static void closelog() throws IOException{

logs.close();

}

// browser Setup

public static void startBrowser()

{

System.setProperty(configProp.getProperty("browser"),configProp.getProperty("path"));

driver = new ChromeDriver();

driver.manage().timeouts().implicitlyWait(50, TimeUnit.SECONDS);

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

}

//launching URL

public static void navigateUserURL(String user, String pwd, String url) throws InterruptedException

{

Thread.sleep(5000);

String Urel = configProp.getProperty("url").replace("https://", "https://"+user+":"+pwd+"@");

System.out.println(Urel);

driver.navigate().to(Urel);

waitTime();waitTime();waitTime();

driver.navigate().refresh();

}

public static void switchTab() throws AWTException

{

String parentWindowHandler=driver.getWindowHandle();// Store your parent window

String subWindowHandler = null;

Set<String> handles = driver.getWindowHandles(); // get all window handles

Iterator<String> iterator = handles.iterator();

while (iterator.hasNext()){

subWindowHandler = iterator.next();

}

driver.switchTo().window(subWindowHandler);

}

public static void profileURL(String profile)

{

String url = configProp.getProperty("url")+profile;

driver.get(url);

}

//Get text Values from attribute(element)

public static String getValue(String xpath)

{

WebElement xp = driver.findElement(By.xpath(xpath));

highlight(driver, xp);

waitForCondition(xp);

return xp.getText();

}

//get attribute values from element

public static String getAttributeValue(String xpath, String value)

{

WebElement xp = driver.findElement(By.xpath(xpath));

highlight(driver, xp);

waitForCondition(xp);

return xp.getAttribute(value);

}

//Obj exist or nNot

public static boolean objCheck(String xpath)

{

boolean chk = false;

try {

WebElement xp = driver.findElement(By.xpath(xpath));

highlight(driver, xp);

waitForCondition(xp);

waitTime();

if (xp.isDisplayed()) {

chk = true;

} else {

chk = false;

}

} catch (Exception e) {

System.out.println("Check Obj Exists Error: " + e.getLocalizedMessage());

}

return chk;

}

//Drop down selection

public static void selectFromDropdown(String xpath, String value) throws InterruptedException {

WebElement xp = driver.findElement(By.xpath(xpath));

highlight(driver, xp);

waitForCondition(xp);

waitTime();

int indexNum = 0;

List<WebElement> lstOption = xp.findElements(By.tagName("option"));

if (lstOption.size() > 0) {

for (int cnt = 0; cnt < lstOption.size(); cnt++) {

if (lstOption.get(cnt).getText().equalsIgnoreCase(value)) {

indexNum = cnt;

break;

}

}

}

smallWait();

System.out.println(indexNum + value);

Select sel = new Select(xp);

sel.selectByIndex(indexNum);

}

public static String getLocationLeftHeader(String xpat, String value) throws Throwable{

WebElement xp = driver.findElement(By.xpath(xpat));

waitForCondition(xp);

smallWait();

int indexNum1 = 0;

List<WebElement> lstOption = xp.findElements(By.tagName("th"));

if (lstOption.size() > 0) {

for (int cnt = 0; cnt < lstOption.size(); cnt++) {

if (lstOption.get(cnt).getText().equalsIgnoreCase(value)) {

indexNum1 = cnt;

break;

}

}

}

int i = indexNum1+1;

String left = Integer.toString(indexNum1);

String xpath=xpat+"/th["+i+"]" ;

WebElement ele = driver.findElement(By.xpath(xpath));

highlight(driver, ele);

return left;

}

public static String getLocationRightHeader(String xpat, String value) throws Throwable{

WebElement xp = driver.findElement(By.xpath(xpat));

waitForCondition(xp);

smallWait();

int indexNum1 = 0;

List<WebElement> lstOption = xp.findElements(By.tagName("th"));

if (lstOption.size() > 0) {

for (int cnt = 0; cnt < lstOption.size(); cnt++) {

if (lstOption.get(cnt).getText().equalsIgnoreCase(value)) {

indexNum1 = cnt;

break;

}

}

}

int i = indexNum1+1;

String right = Integer.toString(indexNum1);

String xpath=xpat+"/th["+i+"]" ;

WebElement ele = driver.findElement(By.xpath(xpath));

highlight(driver, ele);

return right;

}

//Click on obj

public static void click(String xpath)

{

WebElement xp = driver.findElement(By.xpath(xpath));

highlight(driver, xp);

waitForCondition(xp);

try

{

Actions act = new Actions(driver);

act.moveToElement(xp).build().perform();

act.click().build().perform();

}catch (Exception e){

try {

driver.navigate().refresh();

waitForCondition(xp);

Actions act = new Actions(driver);

act.moveToElement(xp).build().perform();

act.click().build().perform();

}catch (Exception er) {

System.out.println("Error "+er.getLocalizedMessage());

}

System.out.println("Error "+e.getLocalizedMessage());

}

}

//clear the value

public static void clearValue (String xpath)

{

WebElement xp = driver.findElement(By.xpath(xpath));

highlight(driver, xp);

waitForCondition(xp);

try

{

xp.clear();

}catch (Exception e)

{

System.out.println("Error: "+e.getMessage());

}

}

public static void sendingValue(String xpath, String value)

{

WebElement xp = driver.findElement(By.xpath(xpath));

highlight(driver, xp);

waitForCondition(xp);

try{

xp.sendKeys(value);

}catch (Exception e){

System.out.println("Error in Entering Value "+ e.getLocalizedMessage());

}

}

public static void clickSendValuesAndEnter(String xpath, String value)

{

try{

WebElement xp = driver.findElement(By.xpath(xpath));

xp.click();

xp.sendKeys(value);

waitTime(); waitTime();waitTime();

xp.sendKeys(Keys.TAB);

}catch (Exception e){

System.out.println("Error in Click Send Value and Enter "+ e.getLocalizedMessage());

}

}

public static void addValueAndTab(String xpath, String value)

{

WebElement xp = driver.findElement(By.xpath(xpath));

highlight(driver, xp);

waitForCondition(xp);

try{

xp.sendKeys(value);

waitTime(); smallWait();

xp.sendKeys(Keys.TAB);

}catch (Exception e){

System.out.println("Error in Add Value and Tab "+ e.getLocalizedMessage());

}

}

public static String getCurrentDate()

{

Date date = new Date();

String curDate = new SimpleDateFormat("ddMMyyyy").format(date);

return curDate;

}

public static String getCurrentTime()

{

Date date = new Date();

String curTime = new SimpleDateFormat("hhmmss").format(date);

return curTime;

}

public static void userPermission(String k) throws InterruptedException

{

waitTime();

switch (k) {

case "Re":

click(xProp.getProperty("cbox\_read"));

break;

case "Wr":

click(xProp.getProperty("cbox\_write"));

break;

case "Ap":

click(xProp.getProperty("cbox\_approve"));

break;

case "Ad":

click(xProp.getProperty("cbox\_admin"));

break;

case "ReWr":

smallWait();

click(xProp.getProperty("cbox\_read"));

click(xProp.getProperty("cbox\_write"));

break;

case "ReAd":

smallWait();

click(xProp.getProperty("cbox\_read"));

click(xProp.getProperty("cbox\_admin"));

break;

case "ReAp":

smallWait();

click(xProp.getProperty("cbox\_read"));

click(xProp.getProperty("cbox\_approve"));

break;

case "WrAp":

smallWait();

click(xProp.getProperty("cbox\_write"));

click(xProp.getProperty("cbox\_approve"));

break;

case "WrAd":

smallWait();

click(xProp.getProperty("cbox\_write"));

click(xProp.getProperty("cbox\_approve"));

break;

case "ApAd":

smallWait();

click(xProp.getProperty("cbox\_approve"));

click(xProp.getProperty("cbox\_admin"));

break;

case "ReWrAp":

smallWait();

click(xProp.getProperty("cbox\_read"));

click(xProp.getProperty("cbox\_write"));

click(xProp.getProperty("cbox\_approve"));

break;

case "ReWrAd":

smallWait();

click(xProp.getProperty("cbox\_read"));

click(xProp.getProperty("cbox\_write"));

click(xProp.getProperty("cbox\_admin"));

break;

case "ReApAd":

smallWait();

click(xProp.getProperty("cbox\_read"));

click(xProp.getProperty("cbox\_approve"));

click(xProp.getProperty("cbox\_admin"));

break;

case "WrApAd":

smallWait();

click(xProp.getProperty("cbox\_write"));

click(xProp.getProperty("cbox\_approve"));

click(xProp.getProperty("cbox\_admin"));

break;

case "ReWrApAd":

smallWait();

click(xProp.getProperty("cbox\_read"));

click(xProp.getProperty("cbox\_write"));

click(xProp.getProperty("cbox\_approve"));

click(xProp.getProperty("cbox\_admin"));

break;

default:

break;

}

}

// taking screenshot

public static void screenshot(String inp) throws IOException, InterruptedException{

waitTime();Thread.sleep(3000);

String cr\_time = getCurrentTime();

String cr\_date = getCurrentDate();

File srcFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);

FileUtils.copyFile(srcFile, new File(configProp.getProperty("Screenshot\_path")+cr\_date+" "+inp+" "+cr\_time+" "+member+".jpeg"));

}

public static void smallWait() throws InterruptedException{

Thread.sleep(500);

}

public static void closeBrowser()

{

driver.quit();

}

}

package ocp.testBase;

import java.io.IOException;

import java.util.ArrayList;

public class TestPartyProfileStorage extends TestBasic{

static ArrayList<String> arg1 = new ArrayList<String>();

static ArrayList<String> arg2 = new ArrayList<String>();

static ArrayList<String> arg3 = new ArrayList<String>();

static ArrayList<String> arg4 = new ArrayList<String>();

static ArrayList<String> arg5 = new ArrayList<String>();

public TestPartyProfileStorage() throws IOException {

super();

// TODO Auto-generated constructor stub

}

//Stroring values

public static void u1StoreValue(String args) {

arg1.add(args);

}

public static void u2StoreValue(String args) {

arg2.add(args);

}

public static void u3StoreValue(String args) {

arg3.add(args);

}

public static void u4StoreValue(String args) {

arg4.add(args);

}

public static void u5StoreValue(String args) {

arg5.add(args);

}

//Get size of the Array

public static int u1Size()

{

int value = arg1.size();

return value;

}

public static int u2Size()

{

int value = arg2.size();

return value;

}

public static int u3Size()

{

int value = arg3.size();

return value;

}

public static int u4Size()

{

int value = arg4.size();

return value;

}

public static int u5Size()

{

int value = arg5.size();

return value;

}

//get value from the arrayList

public static String u2getValue(int value) {

String val = arg2.get(value);

return val;

}

public static String u1getValue(int value) {

String val = arg1.get(value);

return val;

}

public static String u3getValue(int value) {

String val = arg3.get(value);

return val;

}

public static String u4getValue(int value) {

String val = arg4.get(value);

return val;

}

public static String u5getValue(int value) {

String val = arg5.get(value);

return val;

}

public static String approvalType(int value)

{

String value1 = null;

if(value==0) {

value1 = "Approved";

}

else if(value==1) {

value1 = "For Rework";

}

else if(value==2) {

value1 = "Declined";

}

else if(value==3) {

value1 = "Noting";

}

else if(value==4) {

value1 = "";

}

return value1;

}

}

--

**package** ocp.testBase;

**import** java.util.ArrayList;

**public** **class** TestStorage {

**static** ArrayList<String> *arguments* = **new** ArrayList<String>();

**static** ArrayList<String> *RA1args* = **new** ArrayList<String>();

**static** ArrayList<String> *RA2args* = **new** ArrayList<String>();

**static** ArrayList<String> *RA3args* = **new** ArrayList<String>();

**static** ArrayList<String> *RA4args* = **new** ArrayList<String>();

**static** ArrayList<String> *RA5args* = **new** ArrayList<String>();

**public** **static** **void** StoreProfileCreationValue(String value)

{

*arguments*.add(value);

}

**public** **static** **int** StorageSize()

{

**int** y = *arguments*.size();

**return** y;

}

**public** **static** String getVal(**int** val) {

String value = *arguments*.get(val);

**return** value;

}

**public** **static** **void** r1Storage(String val)

{

*RA1args*.add(val);

}

**public** **static** **int** r1Size()

{

**int** val = *RA1args*.size();

**return** val;

}

**public** **static** String getr1Val1(**int** val) {

String value = *RA1args*.get(val);

**return** value;

}

**public** **static** **void** r2Storage(String val)

{

*RA2args*.add(val);

}

**public** **static** **int** r2Size()

{

**int** val = *RA2args*.size();

**return** val;

}

**public** **static** String getr2Val1(**int** val) {

String value = *RA2args*.get(val);

**return** value;

}

**public** **static** **void** r3Storage(String val)

{

*RA3args*.add(val);

}

**public** **static** **int** r3Size()

{

**int** val = *RA3args*.size();

**return** val;

}

**public** **static** String getr3Val1(**int** val) {

String value = *RA3args*.get(val);

**return** value;

}

**public** **static** **void** r4Storage(String val)

{

*RA4args*.add(val);

}

**public** **static** **int** r4Size()

{

**int** val = *RA4args*.size();

**return** val;

}

**public** **static** String getr4Val1(**int** val) {

String value = *RA4args*.get(val);

**return** value;

}

**public** **static** **void** r5Storage(String val)

{

*RA5args*.add(val);

}

**public** **static** **int** r5Size()

{

**int** val = *RA5args*.size();

**return** val;

}

**public** **static** String getr5Val1(**int** val) {

String value = *RA5args*.get(val);

**return** value;

}

}

--

package ocp.testRunner;

import org.junit.runner.RunWith;

import cucumber.api.CucumberOptions;

import cucumber.api.junit.Cucumber;

@RunWith(Cucumber.class)

@CucumberOptions(features="Features",

glue={"ocp.stepDefinition"},

tags={"@Test1366"}

)

public class TestRunner {

}

--

TestName=TC\_001

Description= sample test or smoke test

profile=/profile/3914/view

#creator=AUTOMATIONSKTEST

creator=SORTING

date=23042019

time=110804

#title=AutomationSubmission

title=Automated Submission

sub\_date=23052019

party=Vamsi

30042019-094257

23042019-110804

19042019-104755

Screenshot\_path=./Screenshot/

OutboundPath=./reports/

path=./drivers/chromedriver.exe

browser=webdriver.chrome.driver

user1=satheeshkumar sathasivam

user2=nayakr

user3=vamsi

user4=ocpwriter

user5=ocppnvuser242

ra1=satheeshkumar sathasivam

ra2=nayakr

ra3=vamsi

ra4=ocpwriter

ra5=ocppnvuser242

rm1=satheeshkumar sathasivam

rm2=nayakr

rm3=vamsi

rm4=ocpwriter

rm5=ocppnvuser242

url=https://ocp-test.dev.anz

user=sathasisadmin

pwd=OCPassword\*98

user\_name=satheeshkumar sathasivam

#admin\_user=nayakr3admin

admin\_user=thotakv1admin

admin\_pwd=Password@123

#RA

writer\_user=ocpwriter

writer\_pwd=Password@123

#RM

pnv\_user=ocppnvuser242

pnv\_pwd=Password@123

#credit

cre\_user=nayakr3admin

cre\_pwd=Password$123

#cmf\_status

cmf\_status1=GRP & CP

cmf\_status2=CP

cmf\_status3=AP

cmf\_status4=GRP (Not CP)

#cmf\_location

cmf\_loc1=Australia Adelaide

cmf\_loc2=Australia Brisbane

cmf\_loc3=Australia Melbourne

cmf\_loc4=Australia Perth

cmf\_loc5=Australia Sydney

cmf\_loc6=American Samoa

cmf\_loc7=Cambodia

cmf\_loc8=China

cmf\_loc9=Cook Islands

cmf\_loc10=Fiji

cmf\_loc11=France

cmf\_loc12=Germany

cmf\_loc13=Guam

cmf\_loc14=Hong Kong

cmf\_loc15=India

cmf\_loc16=Indonesia

cmf\_loc17=Japan

cmf\_loc18=Kiribati

cmf\_loc19=Laos

cmf\_loc20=Malaysia

cmf\_loc21=Myanmar

cmf\_loc22=New Caledonia

cmf\_loc23=New Zealand

cmf\_loc24=Papua New Guinea

cmf\_loc25=Philippines

cmf\_loc26=Samoa

cmf\_loc27=Singapore

cmf\_loc28=Solomon Islands

cmf\_loc29=South Korea

cmf\_loc30=Taiwan

cmf\_loc31=Thailand

cmf\_loc32=Timor-Leste

cmf\_loc33=Tonga

cmf\_loc34=United Arab Emirates

cmf\_loc35=United Kingdom

cmf\_loc36=United States

cmf\_loc37=Vanuatu

cmf\_loc38=Vietnam

#-----------------------

#icis\_l5\_values

option1=CORPORATE NSW

option2=CORPORATE QLD

option3=CORPORATE SA

option4=CORPORATE VIC

option5=CORPORATE WA

option6=BANKS

option7=DIVERSIFIED FINANCIALS

option8=FUND MGR

option9=INSURANCE

option10=CONSUMER & SERVICES

option11=GOVERNMENT, EDUCATION & DEFENCE

option12=DIVERSIFIED INDUSTRIALS

option13=FOOD, BEVERAGE & AGRI

option14=GLOBAL SUBSIDIARIES

option15=TELECOMM, MEDIA, ENTERTAINMENT & TECH

option16=TRANSPORT & LOGISTICS

option17=ANZ HEALTH INSTITUTIONAL

option18=INSTITUTIONAL PROPERTY

option19=GLOBAL COMMODITIES COMPANIES

option20=MINERAL MINING

option21=OIL AND GAS

option22=RESOURCE OTHERS

option23=UTILITIES & INFRASTRUCTURE

#CAD LEVEL

cad\_level=Corp 1

cad\_level1=Corp

cad\_level2=Corp 0.2

cad\_level3=Corp 0.4

cad\_level4=Corp 0.5

cad\_level5=Corp 0.8

cad\_level6=Corp 1

cad\_level7=Corp 4

cad\_level8=Corp 8

cad\_level9=Corp 15

cad\_level10=Corp 20

cad\_level11=Corp 35

cad\_level12=Corp 60

cad\_level13=Corp 115

cad\_level14=Corp 250

cad\_level15=Corp 500

cad\_level16=Corp 750

cad\_level17=Corp 1000

cad\_level18=Corp 1500

cad\_level19=Corp 2500

cad\_level20=Corp 4000

cad\_level21=CFS 10

cad\_level22=CFS 75

cad\_level23=CFS 140

cad\_level24=CFS 500

cad\_level25=CFS 1000

cad\_level26=CFS 2000

cad\_level27=CFS 2500

cad\_level28=CFS 4000

cad\_level29=CFS 8000

cad\_level30=CFS 15000

#Approval Type

app\_type1=For Approval

app\_type2=For Support

app\_type3=For Noting

app\_type4=For No Objections

#useraccess permissions to the profile

ResearchAnalyst=Re

ReleaseManager=Wr

SystemAdmin=ApAd

party\_add\_user=satheeshkumar sathasivam

exrep\_path=./Config/extent-config.xml

excelPath=./DataFiles/OCP.xlsx

DataSheet=sheet1

txt1\_pop=If you have selected the new participant to receive an email notification they will receive an email request for approval by 23-May-2019. The submission will not be closed until the new participant has actioned the request.

txt2\_pop=Do you want to proceed?

profile1=30042019025803

profile2=07052019100956

profile3=23042019110804

profile4=10052019095202

profile5=30042019094257

profile6=09052019050505

profile7=06052019010915

profile8=02052019114451

profile9=03052019212012

--

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

<extentreports>

<configuration>

<!-- report theme --> <!-- standard, dark -->

<theme>standard</theme>

<!-- document encoding --> <!-- defaults to UTF-8 -->

<encoding>UTF-8</encoding>

<!-- protocol for script and stylesheets --> <!-- defaults to https -->

<protocol>https</protocol>

<!-- title of the document -->

<documentTitle>ToolsQA - Cucumber Framework</documentTitle>

<!-- report name - displayed at top-nav -->

<reportName>ToolsQA - Cucumber Report</reportName>

<!-- global date format override --> <!-- defaults to yyyy-MM-dd -->

<dateFormat>yyyy-MM-dd</dateFormat>

<!-- global time format override --> <!-- defaults to HH:mm:ss -->

<timeFormat>HH:mm:ss</timeFormat>

<!-- custom javascript -->

<scripts>

<![CDATA[

$(document).ready(function() {

});

]]>

</scripts>

<!-- custom styles -->

<styles>

<![CDATA[

]]>

</styles>

</configuration>

</extentreports>

--

profile\_value=//a[@href='/profile/3902/view']

start=//\*[contains(text(),'

end=')]

body\_start=//table[@class='profileSummaryTable table table--fixed']/tbody/tr[

body\_mid=]/td[

body\_end=]

home\_but\_next=//div[@class='btn-group']/button[@title='Next']

head\_proid=//tr[@class='tableHeader\_\_columns']/th[1]/div

head\_proname=//tr[@class='tableHeader\_\_columns']/th[2]/div

head\_cmfloca=//tr[@class='tableHeader\_\_columns']/th[3]/div

head\_cmfsta=//tr[@class='tableHeader\_\_columns']/th[4]/div

head\_ccr=//tr[@class='tableHeader\_\_columns']/th[5]/div

head\_segment=//tr[@class='tableHeader\_\_columns']/th[6]/div

head\_nxtrevDate=//tr[@class='tableHeader\_\_columns']/th[9]/div

head\_rpidisDate=//tr[@class='tableHeader\_\_columns']/th[10]/div

#HomePage xpaths

icon\_profile=//\*[@class='header\_\_container']//span[contains(text(),'Profiles')]

drop\_search=//select[@class='search\_\_mask-select']

btn\_create\_profile=//span[@class='button\_\_label'][contains(text(),'Create')]

btn\_profile=//span[contains(text(),'Profiles')]

tbox\_search\_profile=//input[@class='search\_\_input form-control']

btn\_search\_profile=//button[@class='button btn button--primary search\_\_submit btn btn-secondary']/span[contains(text(),'Search')]

tog\_pro=//span[@class='toggle\_\_pseudo']

filter\_head=//thead[@class='tableHeader profileSummaryTable\_\_header']/tr[3][@class='tableHeader\_\_columns']

text\_profile=//table[@class='profileSummaryTable table--fixed table']/tbody/tr/td[2]/Strong/a

home\_cmf\_loc=//table[@class='profileSummaryTable table table--fixed']/tbody/tr/td[3]

home\_cmf\_sta=//table[@class='profileSummaryTable table table--fixed']/tbody/tr/td[4]

home\_segment=//table[@class='profileSummaryTable table table--fixed']/tbody/tr/td[6]

pend\_profile\_start=//table[@class='profileSummaryTable table--fixed table']/tbody/tr[

pend\_profile\_end=]/td[2]/Strong/a

#pending task icon yellow

pending\_task\_icon\_Start=//table[@class='profileSummaryTable table--fixed table']//tr[

pending\_task\_icon\_End=]//button[@class='toolTip profileSummaryTable\_\_alert profileSummaryTable\_\_alert--task']/span[@class='icon icon--xsmall']

#rework task icon pink

rew\_icon\_start=//tbody/tr[

rew\_icon\_end=]//button[@class='toolTip profileSummaryTable\_\_alert profileSummaryTable\_\_alert--rework']/span

#sections

lab\_sub\_his=//section[@id='Submission History']/label

#Submission Table Xpaths

sub\_status=//table[@class='submissionHistory-table table table-striped']/tbody/tr/td[5]/span

sub\_title=//table[@class='submissionHistory-table table table-striped']/tbody/tr/td[3]/a

#New Profile Creation

tbox\_cust\_name=//div[@class='formGroup\_\_field-container']/input[@name='CustomerName']

tbox\_razor\_id=//div[@class='formGroup\_\_field-container']/input[@name='CustomerIdentifier']

drop\_cmf\_status=//div[@class='formGroup\_\_field-container']/select[@name='CmfStatus']

drop\_cmf\_location=//div[@class='formGroup\_\_field-container']/select[@name='CmfLocation']

drop\_icis\_l5=//select[@name='Segment']

btn\_create=//span[@class='submitButton\_\_label']

#AfterProfile Name Path

profile\_name\_value=//div[@class='hero\_\_title-container']/h2

#Profile Page SysAdmin

button\_profile\_not\_restrict=//button[@class='iconButton hero\_\_iconButton'][@title='Restrict access to this profile']

button\_profile\_restrict=//button[@class='iconButton hero\_\_iconButton iconButton--toggled'][@title='Remove restricted access']

button\_Permissions=//button[@title='Permissions']

#Permissions

#Add Users to Profile

button\_addUser=//button[@class='button btn button--cta button--icon btn btn-secondary']/span[@class='button\_\_label']

tbox\_user=//div[@class='rbt-input-hint-container']/input[1]

cbox\_read=//div[@id='Read']//label[@class='checkbox']/span[2]

cbox\_write=//div[@id='Write']//label[@class='checkbox']/span[2]

cbox\_approve=//div[@id='Approve']//label[@class='checkbox']/span[2]

cbox\_admin=//div[@id='Admin']//label[@class='checkbox']/span[2]

but\_permission\_save=//button[@class='submitButton undefined btn btn-primary']/span

#Removing Users

but\_rm=//\*[contains(text(),'OCPPNVUSER242')]

but\_ra=//\*[contains(text(),'OCPWriter')]

but\_sa=//\*[contains(text(),'nayakr ')]

but\_remove\_user=//span[contains(text(),'Remove User')]

but\_rm\_usr\_cfrm=//div[@class='react-confirm-alert-button-group']/button[contains(text(),'Yes')]

No\_of\_Profiles=//i[@class='tableHeader\_\_results']

Page\_per\_Profile=//select[@class='tableHeader\_\_rows-select form-control-sm form-control']

#Submissions

#Home

but\_sub=//Section[@id='Submission']/label

but\_submission=//button[@title='New Submission']

sub\_page\_cnfrm=//span[contains(text(),'Submission')][@class='section\_\_title-inner']

#icons in submission page

icon\_print=//button[@title='Print (p)']/span

#Submission Description

tbox\_sub\_title=//input[@name='SubmissionSummary']

cal\_exp\_due=//input[@name='SubmissionDateOfEvent']

drop\_cad\_level=//select[@name='SubmissionCadLevel']

tog\_pro\_up\_req=//input[@class='checkbox\_\_real'][@value='false']

tod\_pro\_up\_noreq=//label[@class='toggle']/span

tod\_pro\_up\_no=//input[@class='checkbox\_\_real'][@value='true']

but\_del\_cng=//button/span[contains(text(),'Delete Changes')]

txt\_box\_off=//input[@name='confirmAction']

but\_cfm=//button/span[contains(text(),'I understand and wish to proceed')]

but\_off\_can=//button/span[contains(text(),'Cancel')]

#recall

but\_recall=//span[contains(text(),'Recall')][@class='button\_\_label']

but\_recall\_cfrm=//button[contains(text(),'Confirm')]

#Schedule review

cbox\_sub\_review=//input[@name='SubmissionIsScheduledReview']/following-sibling::span[2]

cbox\_sub\_extnd=//input[@name='SubmissionIsExtendedScheduledReview']/following-sibling::span[2]

#submission Purpose Box

tbox\_sub\_pur=//div[@class='fr-element fr-view form-control']

#Basic Info Section

enable\_basic\_info=//span[contains(text(),'Basic Info')][@class='section\_\_title-inner']

cal\_nxt\_sch\_rev\_dt=//input[@name='NextReviewDate']

txt\_rm\_inp=//div[@id='RelationshipManagerList']//div[@class='form-control rbt-input-multi rbt-input']

txt\_ra\_inp=//div[@id='RnaAnalystList']//div[@class='form-control rbt-input-multi rbt-input']

drop\_ccr=//select[@name='Ccr']

inp\_rpi\_date=//input[@name='RpiDiscussionDate']

#add participant

but\_add\_parti=//button[@class='button btn button--primary button--icon submissionRoles\_\_btn btn btn-secondary']

tbox\_parti\_name=//div[@id='AppUserId']/label/div[2]/div/div/div/input[1]

drop\_parti\_type=//select[@class='form-control'][@name='Role']

but\_parti\_save=//div[@class='form\_\_actions']/button/span[contains(text(),'Save')]

txt\_add\_party=//div[@class='react-confirm-alert-body']/div

but\_add\_party\_cntne=//div[@class='react-confirm-alert-button-group']/button[contains(text(),'Continue')]

Edit\_party=//button/span[contains(text(),'Edit')]

#Decision Making

drop\_decision=//select[@name='Decision'][@class='form-control']

but\_dec\_save=//button[@class='button btn button--primary button--icon btn btn-secondary']//span[contains(text(),'Save')]

but\_dec\_cancel=//button[@class='button btn button--inverted-secondary button--icon btn btn-secondary']//span[contains(text(),'Close')]

but\_dec\_close=//button[@class='button btn button--primary button--icon btn btn-secondary']/span[2]

but\_dec\_submit=//button[@class='button btn button--cta button--icon btn btn-secondary']/span[@class='button\_\_label'][contains(text(),'Submit')]

pop\_but\_dec\_confirm=//div[@class='react-confirm-alert-button-group']/button[contains(text(),'Confirm')]

pop\_but\_dec\_cancel=//div[@class='react-confirm-alert-button-group']/button[contains(text(),'Cancel')]

#

but\_cancel=//button/span[contains(text(),'Cancel Submission')]

but\_save=//span[contains(text(),'Save')][@class='button\_\_label']

but\_save\_close=//button/span[contains(text(),'Save and Close')]

but\_submit=//button/span[contains(text(),'Submit')]

#After click on submit

but\_sav\_sub=//div[@class='react-confirm-alert-button-group']/button[contains(text(),'Yes')]

but\_sub\_close=//button/span[contains(text(),'Close')]

txt\_party\_user=//table[@class='table']/tbody/tr[3]/td/strong

--

@Bug1893

Feature: To validate PrintIcon is present in Submmission page when profile update toggle is OFF

Scenario Outline: Users Creating New Profiles , Creating New Submission , Turn OFF profile update toggle

& Validating PRINT ICON

Given Launching OCP Application and Login As User "<users>"

When move to profile creation page

Then entering values in the fields for creating new profile with CMF Status "<cmf\_status>" CMF\_Location "<cmf\_loca>" ICIS\_Lfive "<icis\_l5>"

And Click on Submit to Create new profile

Then Validate whether new profile created or not

Then Moving To Submission Page For Creating New Submission

When Entering The Submission Title

And Entering The Expected Due Date "<DueDate>"

And Entering CAD Level value

Then Updating Profile Toggle Type "<type>"

Then Validating Print Icon in submission page

Examples:

|users| cmf\_status|cmf\_loca|icis\_l5|DueDate |type|

|user|cmf\_status1|cmf\_loc6|option10|30102019|OFF|

|admin|cmf\_status2|cmf\_loc6|option8|30082019|OFF|

|RA|cmf\_status3|cmf\_loc6|option9|30092019|OFF|

|RM|cmf\_status1|cmf\_loc6|option1|30112019|OFF|

--

