**Project**-

actionpackage

/First\_RMSProject/src/actionpackage/actionanalyst.java

/First\_RMSProject/src/actionpackage/actionclass.java

/First\_RMSProject/src/actionpackage/actiongatekeeper.java

/First\_RMSProject/src/actionpackage/actionrequestor.java

rmspackage

/First\_RMSProject/src/rmspackage/Delete.java

/First\_RMSProject/src/rmspackage/ExcelUtils.java

/First\_RMSProject/src/rmspackage/Fillfeedback.java

/First\_RMSProject/src/rmspackage/FillTimesheet.java

/First\_RMSProject/src/rmspackage/Loginclass.java

/First\_RMSProject/src/rmspackage/New\_Request.java

/First\_RMSProject/src/rmspackage/Reports.java

/First\_RMSProject/src/rmspackage/Req\_deliverables.java

/First\_RMSProject/src/rmspackage/Request\_details.java

/First\_RMSProject/src/rmspackage/View\_request.java

/First\_RMSProject/src/rmspackage/Viewfeedback.java

/First\_RMSProject/src/rmspackage/ViewTimesheet.java

**Actionanalyst.java-**

**package** actionpackage;

**import** java.awt.AWTException;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.logging.Logger;

**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.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.PageFactory;

**import** org.testng.annotations.Test;

**import** rmspackage.ExcelUtils;

**import** rmspackage.FillTimesheet;

**import** rmspackage.Fillfeedback;

**import** rmspackage.Loginclass;

**import** rmspackage.New\_Request;

**import** rmspackage.Reports;

**import** rmspackage.Req\_deliverables;

**import** rmspackage.Request\_details;

**import** rmspackage.ViewTimesheet;

**import** rmspackage.View\_request;

**import** rmspackage.Viewfeedback;

**public** **class** actionanalyst {

WebDriver driver;

Loginclass objlogin;

New\_Request objreq;

View\_request objview;

Request\_details objdetails;

Req\_deliverables objdeli;

//Login to application using username and password

**static** Logger *log* = Logger.*getLogger*(actionclass.**class**.getName());

@Test

**public** **void** enterurl() **throws** IOException, InterruptedException

{

//Loginclass objurl=PageFactory.initElements(driver,Loginclass.class);

//System.setProperty("webdriver.chrome.driver","C:\\Windows\\System32\\chromedriver.exe");

/\*driver = new InternetExplorerDriver();

DesiredCapabilities capabilities = DesiredCapabilities.internetExplorer();

capabilities.setCapability(InternetExplorerDriver.

INTRODUCE\_FLAKINESS\_BY\_IGNORING\_SECURITY\_DOMAINS,true);

driver = new InternetExplorerDriver(capabilities);\*/

//DOMConfigurator.configure("log.xml");

// driver =new ChromeDriver();

driver=**new** FirefoxDriver();

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

Thread.*sleep*(1000);

*log*.info("Opening URL");

Loginclass objurl=PageFactory.*initElements*(driver,Loginclass.**class**);

ExcelUtils.*setexcel*("C:\\RMS\_workspace\\First\_RMSProject\\RMS data.xlsx", "Sheet1");

objurl.logintourl();

}

@Test

**public** **void** loginanalyst() **throws** InterruptedException{

*log*.info("Logging analyst");

Loginclass objloginanalyst=PageFactory.*initElements*(driver,Loginclass.**class**);

System.***out***.println("\n" +"///---actionclass.loginanalyst()---///");

Thread.*sleep*(1000);

objloginanalyst.enteruser(5);

Thread.*sleep*(1000);

objloginanalyst.enterpswd(5);

objloginanalyst.log();

Thread.*sleep*(2000);

}

//Create new request

@Test

**public** **void** newreq() **throws** InterruptedException{

New\_Request objreq=PageFactory.*initElements*(driver, New\_Request.**class**);

System.***out***.println("\n" +"///--- actionclass.newreq() ---///");

Thread.*sleep*(5000);

objreq.raise\_Req();

}

// Check visibility and functionality on view request page

@Test

**public** **void** viewreq() **throws** InterruptedException

{

View\_request objview=PageFactory.*initElements*(driver, View\_request.**class**);

System.***out***.println("\n" +"///--- actionclass.viewreq() ---///");

objview.unassigned();

objview.inprogress();

objview.onhold();

objview.completed();

objview.closed();

objview.cancelled();

objview.deadline\_TBD();

}

//Check visibility and functionality on request details page

@Test

**public** **void** reqdetails() **throws** InterruptedException

{

Request\_details objdetails=PageFactory.*initElements*(driver, Request\_details.**class**);

System.***out***.println("\n" + "///--- actionclass.reqdetails() ---///");

objdetails.unassignedreq();

//objdetails.inprogressreq();

/\*objdetails.onholdreq();

objdetails.cancelreq();

objdetails.TBDreq();\*/

objdetails.completedreq();

objdetails.closedreq();

}

//Check visibility and functionality on request deliverable page

@Test

**public** **void** Request\_deliverable() **throws** InterruptedException, AWTException

{

Req\_deliverables objdeli=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

System.***out***.println("\n" +"///--- actionclass.Request\_deliverable() ---///");

/\*objdeli.upload();

objdeli.addcomment();

objdeli.download();

objdeli.deletedeliverable();\*/

objdeli.deliverable\_action();

/\*objdeli.actionunassigned1();

objdeli.actioninprogress();

objdeli.actiononhold();

objdeli.actioncompleted();

objdeli.actionclosed();

objdeli.actionTBD();

objdeli.actioncancelled();\*/

}

//Check visibility and functionality on Fill Timesheet page

@Test

**public** **void** timesheet1() **throws** InterruptedException, AWTException

{

System.***out***.println("\n" + "///--- actionclass.timesheet1() ---///");

**if**(driver.findElements(By.*linkText*("Timesheet")).size()>0)

{

FillTimesheet objfill=PageFactory.*initElements*(driver, FillTimesheet.**class**);

objfill.fill\_timesheet();

//objfill.fill\_activity();

//objfill.reset();

ViewTimesheet objview\_timesheet=PageFactory.*initElements*(driver, ViewTimesheet.**class**);

objview\_timesheet.view\_timesheet();

}

**else**

{

System.***out***.println("\n" + "Timesheet module not present");

}

}

//Check visibility of reports

@Test

**public** **void** Reports\_visibility() **throws** InterruptedException

{

System.***out***.println("\n" + "///actionclass.Reports() ---///");

Reports objrep=PageFactory.*initElements*(driver, Reports.**class**);

objrep.reports\_TM();

}

//Ckeck visibility and functionaility of View feedback module

@Test

**public** **void** feedback() **throws** InterruptedException

{

System.***out***.println("\n" + "///---actionclass.feedback() ---///");

**boolean** feedbackpresent = driver.findElements(By.*linkText*("Feedback")).size()>0;

**if**(feedbackpresent)

{

Viewfeedback objfeedback=PageFactory.*initElements*(driver, Viewfeedback.**class**);

objfeedback.View\_feedback1();

Fillfeedback objfillback=PageFactory.*initElements*(driver, Fillfeedback.**class**);

objfillback.fill\_feedback1();

}

**else**

{

System.***out***.println("Feedback module is not present");

}

}

@Test

**public** **void** logout() **throws** InterruptedException

{

*log*.info("Executing logout operation");

Loginclass objlogout=PageFactory.*initElements*(driver,Loginclass.**class**);

objlogout.logout();

Thread.*sleep*(2000);

}

//Take screenshot and Close browser after test is completed

@Test

**public** **void** closetest() **throws** IOException

{

*log*.info("Taking screenshot and closing window");

File scrFile=((TakesScreenshot)driver).getScreenshotAs(OutputType.***FILE***);

FileUtils.*copyFile*(scrFile, **new** File("C://Users//chaman.preet//Desktop//screenshotana.png"));

driver.quit();

}

}

**Actionclass.java**

**package** actionpackage;

**import** java.awt.AWTException;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.logging.Logger;

**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.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.PageFactory;

**import** org.testng.annotations.BeforeMethod;

**import** org.testng.annotations.Test;

**import** rmspackage.Loginclass;

**import** rmspackage.New\_Request;

**import** rmspackage.Reports;

**import** rmspackage.Req\_deliverables;

**import** rmspackage.View\_request;

**import** rmspackage.Request\_details;

**import** rmspackage.ExcelUtils;

**import** rmspackage.FillTimesheet;

**import** rmspackage.ViewTimesheet;

**import** rmspackage.Viewfeedback;

**import** rmspackage.Fillfeedback;

**public** **class** actionclass {

WebDriver driver;

Loginclass objlogin;

New\_Request objreq;

View\_request objview;

Request\_details objdetails;

Req\_deliverables objdeli;

//Login to application using username and password

**static** Logger *log* = Logger.*getLogger*(actionclass.**class**.getName());

@Test

**public** **void** enterurl() **throws** IOException, InterruptedException

{

//Loginclass objurl=PageFactory.initElements(driver,Loginclass.class);

//System.setProperty("webdriver.chrome.driver","C:\\Windows\\System32\\chromedriver.exe");

/\*driver = new InternetExplorerDriver();

DesiredCapabilities capabilities = DesiredCapabilities.internetExplorer();

capabilities.setCapability(InternetExplorerDriver.

INTRODUCE\_FLAKINESS\_BY\_IGNORING\_SECURITY\_DOMAINS,true);

driver = new InternetExplorerDriver(capabilities);\*/

//DOMConfigurator.configure("log.xml");

// driver =new ChromeDriver();

driver=**new** FirefoxDriver();

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

Thread.*sleep*(1000);

*log*.info("Opening URL");

Loginclass objurl=PageFactory.*initElements*(driver,Loginclass.**class**);

ExcelUtils.*setexcel*("C:\\RMS\_workspace\\First\_RMSProject\\RMS data.xlsx", "Sheet1");

objurl.logintourl();

}

@Test

**public** **void** loginteam() **throws** IOException, InterruptedException {

*log*.info("logging team manager");

Loginclass objlogin=PageFactory.*initElements*(driver,Loginclass.**class**);

System.***out***.println("\n" +"///--- actionclass.loginteam() ---///");

// for(int i=2;i<=5;i++)

// {

objlogin.enteruser(2);

Thread.*sleep*(1000);

objlogin.enterpswd(2);

objlogin.log();

Thread.*sleep*(2000);

// }

}

//Create new request

@Test

**public** **void** newreq() **throws** InterruptedException{

New\_Request objreq=PageFactory.*initElements*(driver, New\_Request.**class**);

System.***out***.println("\n" +"///--- actionclass.newreq() ---///");

Thread.*sleep*(5000);

objreq.raise\_Req();

}

// Check visibility and functionality on view request page

@Test

**public** **void** viewreq() **throws** InterruptedException

{

View\_request objview=PageFactory.*initElements*(driver, View\_request.**class**);

System.***out***.println("\n" +"///--- actionclass.viewreq() ---///");

objview.unassigned();

objview.inprogress();

objview.onhold();

objview.completed();

objview.closed();

objview.cancelled();

objview.deadline\_TBD();

}

//Check visibility and functionality on request details page

@Test

**public** **void** reqdetails() **throws** InterruptedException

{

Request\_details objdetails=PageFactory.*initElements*(driver, Request\_details.**class**);

System.***out***.println("\n" + "///--- actionclass.reqdetails() ---///");

objdetails.unassignedreq();

//objdetails.inprogressreq();

/\*objdetails.onholdreq();

objdetails.cancelreq();

objdetails.TBDreq();\*/

objdetails.completedreq();

objdetails.closedreq();

}

//Check visibility and functionality on request deliverable page

@Test

**public** **void** Request\_deliverable() **throws** InterruptedException, AWTException

{

Req\_deliverables objdeli=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

System.***out***.println("\n" +"///--- actionclass.Request\_deliverable() ---///");

/\*objdeli.upload();

objdeli.addcomment();

objdeli.download();

objdeli.deletedeliverable();\*/

objdeli.deliverable\_action();

/\*objdeli.actionunassigned1();

objdeli.actioninprogress();

objdeli.actiononhold();

objdeli.actioncompleted();

objdeli.actionclosed();

objdeli.actionTBD();

objdeli.actioncancelled();\*/

}

//Check visibility and functionality on Fill Timesheet page

@Test

**public** **void** timesheet1() **throws** InterruptedException, AWTException

{

System.***out***.println("\n" + "///--- actionclass.timesheet1() ---///");

**if**(driver.findElements(By.*linkText*("Timesheet")).size()>0)

{

FillTimesheet objfill=PageFactory.*initElements*(driver, FillTimesheet.**class**);

objfill.fill\_timesheet();

//objfill.fill\_activity();

//objfill.reset();

ViewTimesheet objview\_timesheet=PageFactory.*initElements*(driver, ViewTimesheet.**class**);

objview\_timesheet.view\_timesheet();

}

**else**

{

System.***out***.println("\n" + "Timesheet module not present");

}

}

//Check visibility of reports

@Test

**public** **void** Reports\_visibility() **throws** InterruptedException

{

System.***out***.println("\n" + "///actionclass.Reports() ---///");

Reports objrep=PageFactory.*initElements*(driver, Reports.**class**);

objrep.reports\_TM();

}

//Ckeck visibility and functionaility of View feedback module

@Test

**public** **void** feedback() **throws** InterruptedException

{

System.***out***.println("\n" + "///---actionclass.feedback() ---///");

**boolean** feedbackpresent = driver.findElements(By.*linkText*("Feedback")).size()>0;

**if**(feedbackpresent)

{

Viewfeedback objfeedback=PageFactory.*initElements*(driver, Viewfeedback.**class**);

objfeedback.View\_feedback1();

Fillfeedback objfillback=PageFactory.*initElements*(driver, Fillfeedback.**class**);

objfillback.fill\_feedback1();

}

**else**

{

System.***out***.println("Feedback module is not present");

}

}

@Test

**public** **void** logout() **throws** InterruptedException

{

*log*.info("Executing logout operation");

Loginclass objlogout=PageFactory.*initElements*(driver,Loginclass.**class**);

objlogout.logout();

Thread.*sleep*(2000);

}

//Take screenshot and Close browser after test is completed

@Test

**public** **void** closetest() **throws** IOException

{

*log*.info("Taking screenshot and closing window");

File scrFile=((TakesScreenshot)driver).getScreenshotAs(OutputType.***FILE***);

FileUtils.*copyFile*(scrFile, **new** File("C://Users//chaman.preet//Desktop//screenshot.png"));

driver.close();

}

}

**Actiongatekeeper.java**

**package** actionpackage;

**import** org.testng.annotations.Test;

**import** java.awt.AWTException;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.logging.Logger;

**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.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.PageFactory;

**import** rmspackage.ExcelUtils;

**import** rmspackage.FillTimesheet;

**import** rmspackage.Fillfeedback;

**import** rmspackage.Loginclass;

**import** rmspackage.New\_Request;

**import** rmspackage.Reports;

**import** rmspackage.Req\_deliverables;

**import** rmspackage.Request\_details;

**import** rmspackage.ViewTimesheet;

**import** rmspackage.View\_request;

**import** rmspackage.Viewfeedback;

**public** **class** actiongatekeeper {

WebDriver driver;

Loginclass objlogin;

New\_Request objreq;

View\_request objview;

Request\_details objdetails;

Req\_deliverables objdeli;

//Login to application using username and password

**static** Logger *log* = Logger.*getLogger*(actionclass.**class**.getName());

@Test

**public** **void** enterurl() **throws** IOException, InterruptedException

{

//Loginclass objurl=PageFactory.initElements(driver,Loginclass.class);

//System.setProperty("webdriver.chrome.driver","C:\\Windows\\System32\\chromedriver.exe");

/\*driver = new InternetExplorerDriver();

DesiredCapabilities capabilities = DesiredCapabilities.internetExplorer();

capabilities.setCapability(InternetExplorerDriver.

INTRODUCE\_FLAKINESS\_BY\_IGNORING\_SECURITY\_DOMAINS,true);

driver = new InternetExplorerDriver(capabilities);\*/

//DOMConfigurator.configure("log.xml");

// driver =new ChromeDriver();

driver=**new** FirefoxDriver();

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

Thread.*sleep*(1000);

*log*.info("Opening URL");

Loginclass objurl=PageFactory.*initElements*(driver,Loginclass.**class**);

ExcelUtils.*setexcel*("C:\\RMS\_workspace\\First\_RMSProject\\RMS data.xlsx", "Sheet1");

objurl.logintourl();

}

@Test

**public** **void** logingate() **throws** InterruptedException{

*log*.info("Logging gatekeeper");

Loginclass objlogingate=PageFactory.*initElements*(driver,Loginclass.**class**);

System.***out***.println("\n" +"///---actionclass.logingate()---///");

Thread.*sleep*(1000);

objlogingate.enteruser(4);

Thread.*sleep*(1000);

objlogingate.enterpswd(4);

objlogingate.log();

Thread.*sleep*(2000);

}

//Create new request

@Test

**public** **void** newreq() **throws** InterruptedException{

New\_Request objreq=PageFactory.*initElements*(driver, New\_Request.**class**);

System.***out***.println("\n" +"///--- actionclass.newreq() ---///");

Thread.*sleep*(5000);

objreq.raise\_Req();

}

// Check visibility and functionality on view request page

@Test

**public** **void** viewreq() **throws** InterruptedException

{

View\_request objview=PageFactory.*initElements*(driver, View\_request.**class**);

System.***out***.println("\n" +"///--- actionclass.viewreq() ---///");

objview.unassigned();

objview.inprogress();

objview.onhold();

objview.completed();

objview.closed();

objview.cancelled();

objview.deadline\_TBD();

}

//Check visibility and functionality on request details page

@Test

**public** **void** reqdetails() **throws** InterruptedException

{

Request\_details objdetails=PageFactory.*initElements*(driver, Request\_details.**class**);

System.***out***.println("\n" + "///--- actionclass.reqdetails() ---///");

objdetails.unassignedreq();

//objdetails.inprogressreq();

/\*objdetails.onholdreq();

objdetails.cancelreq();

objdetails.TBDreq();\*/

objdetails.completedreq();

objdetails.closedreq();

}

//Check visibility and functionality on request deliverable page

@Test

**public** **void** Request\_deliverable() **throws** InterruptedException, AWTException

{

Req\_deliverables objdeli=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

System.***out***.println("\n" +"///--- actionclass.Request\_deliverable() ---///");

/\*objdeli.upload();

objdeli.addcomment();

objdeli.download();

objdeli.deletedeliverable();\*/

objdeli.deliverable\_action();

/\*objdeli.actionunassigned1();

objdeli.actioninprogress();

objdeli.actiononhold();

objdeli.actioncompleted();

objdeli.actionclosed();

objdeli.actionTBD();

objdeli.actioncancelled();\*/

}

//Check visibility and functionality on Fill Timesheet page

@Test

**public** **void** timesheet1() **throws** InterruptedException, AWTException

{

System.***out***.println("\n" + "///--- actionclass.timesheet1() ---///");

**if**(driver.findElements(By.*linkText*("Timesheet")).size()>0)

{

FillTimesheet objfill=PageFactory.*initElements*(driver, FillTimesheet.**class**);

objfill.fill\_timesheet();

//objfill.fill\_activity();

//objfill.reset();

ViewTimesheet objview\_timesheet=PageFactory.*initElements*(driver, ViewTimesheet.**class**);

objview\_timesheet.view\_timesheet();

}

**else**

{

System.***out***.println("\n" + "Timesheet module not present");

}

}

//Check visibility of reports

@Test

**public** **void** Reports\_visibility() **throws** InterruptedException

{

System.***out***.println("\n" + "///actionclass.Reports() ---///");

Reports objrep=PageFactory.*initElements*(driver, Reports.**class**);

objrep.reports\_TM();

}

//Ckeck visibility and functionaility of View feedback module

@Test

**public** **void** feedback() **throws** InterruptedException

{

System.***out***.println("\n" + "///---actionclass.feedback() ---///");

**boolean** feedbackpresent = driver.findElements(By.*linkText*("Feedback")).size()>0;

**if**(feedbackpresent)

{

Viewfeedback objfeedback=PageFactory.*initElements*(driver, Viewfeedback.**class**);

objfeedback.View\_feedback1();

Fillfeedback objfillback=PageFactory.*initElements*(driver, Fillfeedback.**class**);

objfillback.fill\_feedback1();

}

**else**

{

System.***out***.println("Feedback module is not present");

}

}

@Test

**public** **void** logout() **throws** InterruptedException

{

*log*.info("Executing logout operation");

Loginclass objlogout=PageFactory.*initElements*(driver,Loginclass.**class**);

objlogout.logout();

Thread.*sleep*(2000);

}

//Take screenshot and Close browser after test is completed

@Test

**public** **void** closetest() **throws** IOException

{

*log*.info("Taking screenshot and closing window");

File scrFile=((TakesScreenshot)driver).getScreenshotAs(OutputType.***FILE***);

FileUtils.*copyFile*(scrFile, **new** File("C://Users//chaman.preet//Desktop//screenshotgate.png"));

driver.quit();

}

}

**Actionrequestor.java**

**package** actionpackage;

**import** java.awt.AWTException;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.logging.Logger;

**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.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.PageFactory;

**import** org.testng.annotations.Test;

**import** rmspackage.ExcelUtils;

**import** rmspackage.FillTimesheet;

**import** rmspackage.Fillfeedback;

**import** rmspackage.Loginclass;

**import** rmspackage.New\_Request;

**import** rmspackage.Reports;

**import** rmspackage.Req\_deliverables;

**import** rmspackage.Request\_details;

**import** rmspackage.ViewTimesheet;

**import** rmspackage.View\_request;

**import** rmspackage.Viewfeedback;

**public** **class** actionrequestor {

WebDriver driver;

Loginclass objlogin;

New\_Request objreq;

View\_request objview;

Request\_details objdetails;

Req\_deliverables objdeli;

//Login to application using username and password

**static** Logger *log* = Logger.*getLogger*(actionclass.**class**.getName());

@Test

**public** **void** enterurl() **throws** IOException, InterruptedException

{

//Loginclass objurl=PageFactory.initElements(driver,Loginclass.class);

//System.setProperty("webdriver.chrome.driver","C:\\Windows\\System32\\chromedriver.exe");

/\*driver = new InternetExplorerDriver();

DesiredCapabilities capabilities = DesiredCapabilities.internetExplorer();

capabilities.setCapability(InternetExplorerDriver.

INTRODUCE\_FLAKINESS\_BY\_IGNORING\_SECURITY\_DOMAINS,true);

driver = new InternetExplorerDriver(capabilities);\*/

//DOMConfigurator.configure("log.xml");

// driver =new ChromeDriver();

driver=**new** FirefoxDriver();

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

Thread.*sleep*(1000);

*log*.info("Opening URL");

Loginclass objurl=PageFactory.*initElements*(driver,Loginclass.**class**);

ExcelUtils.*setexcel*("C:\\RMS\_workspace\\First\_RMSProject\\RMS data.xlsx", "Sheet1");

objurl.logintourl();

}

@Test

**public** **void** loginreq() **throws** InterruptedException, IOException{

ExcelUtils.*setexcel*("C:\\RMS\_workspace\\First\_RMSProject\\RMS data.xlsx", "Sheet1");

*log*.info("Logging requestor");

Loginclass objloginreq=PageFactory.*initElements*(driver,Loginclass.**class**);

System.***out***.println("\n" +"///---actionclass.loginreq()---///");

Thread.*sleep*(1000);

objloginreq.enteruser(3);

Thread.*sleep*(1000);

objloginreq.enterpswd(3);

objloginreq.log();

Thread.*sleep*(2000);

}

//Create new request

@Test

**public** **void** newreq() **throws** InterruptedException{

New\_Request objreq=PageFactory.*initElements*(driver, New\_Request.**class**);

System.***out***.println("\n" +"///--- actionclass.newreq() ---///");

Thread.*sleep*(5000);

objreq.raise\_Req();

}

// Check visibility and functionality on view request page

@Test

**public** **void** viewreq() **throws** InterruptedException

{

View\_request objview=PageFactory.*initElements*(driver, View\_request.**class**);

System.***out***.println("\n" +"///--- actionclass.viewreq() ---///");

objview.unassigned();

objview.inprogress();

objview.onhold();

objview.completed();

objview.closed();

objview.cancelled();

objview.deadline\_TBD();

}

//Check visibility and functionality on request details page

@Test

**public** **void** reqdetails() **throws** InterruptedException

{

Request\_details objdetails=PageFactory.*initElements*(driver, Request\_details.**class**);

System.***out***.println("\n" + "///--- actionclass.reqdetails() ---///");

objdetails.unassignedreq();

//objdetails.inprogressreq();

/\*objdetails.onholdreq();

objdetails.cancelreq();

objdetails.TBDreq();\*/

objdetails.completedreq();

objdetails.closedreq();

}

//Check visibility and functionality on request deliverable page

@Test

**public** **void** Request\_deliverable() **throws** InterruptedException, AWTException

{

Req\_deliverables objdeli=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

System.***out***.println("\n" +"///--- actionclass.Request\_deliverable() ---///");

/\*objdeli.upload();

objdeli.addcomment();

objdeli.download();

objdeli.deletedeliverable();\*/

objdeli.deliverable\_action();

/\*objdeli.actionunassigned1();

objdeli.actioninprogress();

objdeli.actiononhold();

objdeli.actioncompleted();

objdeli.actionclosed();

objdeli.actionTBD();

objdeli.actioncancelled();\*/

}

//Check visibility and functionality on Fill Timesheet page

@Test

**public** **void** timesheet1() **throws** InterruptedException, AWTException

{

System.***out***.println("\n" + "///--- actionclass.timesheet1() ---///");

**if**(driver.findElements(By.*linkText*("Timesheet")).size()>0)

{

FillTimesheet objfill=PageFactory.*initElements*(driver, FillTimesheet.**class**);

objfill.fill\_timesheet();

//objfill.fill\_activity();

//objfill.reset();

ViewTimesheet objview\_timesheet=PageFactory.*initElements*(driver, ViewTimesheet.**class**);

objview\_timesheet.view\_timesheet();

}

**else**

{

System.***out***.println("\n" + "Timesheet module not present");

}

}

//Check visibility of reports

@Test

**public** **void** Reports\_visibility() **throws** InterruptedException

{

System.***out***.println("\n" + "///actionclass.Reports() ---///");

Reports objrep=PageFactory.*initElements*(driver, Reports.**class**);

objrep.reports\_TM();

}

//Ckeck visibility and functionaility of View feedback module

@Test

**public** **void** feedback() **throws** InterruptedException

{

System.***out***.println("\n" + "///---actionclass.feedback() ---///");

**boolean** feedbackpresent = driver.findElements(By.*linkText*("Feedback")).size()>0;

**if**(feedbackpresent)

{

Viewfeedback objfeedback=PageFactory.*initElements*(driver, Viewfeedback.**class**);

objfeedback.View\_feedback1();

Fillfeedback objfillback=PageFactory.*initElements*(driver, Fillfeedback.**class**);

objfillback.fill\_feedback1();

}

**else**

{

System.***out***.println("Feedback module is not present");

}

}

@Test

**public** **void** logout() **throws** InterruptedException

{

*log*.info("Executing logout operation");

Loginclass objlogout=PageFactory.*initElements*(driver,Loginclass.**class**);

objlogout.logout();

Thread.*sleep*(2000);

}

//Take screenshot and Close browser after test is completed

@Test

**public** **void** closetest() **throws** IOException

{

*log*.info("Taking screenshot and closing window");

File scrFile=((TakesScreenshot)driver).getScreenshotAs(OutputType.***FILE***);

FileUtils.*copyFile*(scrFile, **new** File("C://Users//chaman.preet//Desktop//screenshotreq.png"));

driver.quit();

}

}

**Excelutils.java**

**package** rmspackage;

**import** java.io.FileInputStream;

**import** java.io.IOException;

**import** org.testng.annotations.Test;

**import** org.apache.poi.xssf.usermodel.XSSFCell;

**import** org.apache.poi.xssf.usermodel.XSSFRow;

**import** org.apache.poi.xssf.usermodel.XSSFSheet;

**import** org.apache.poi.xssf.usermodel.XSSFWorkbook;

**public** **class** ExcelUtils {

String path\_testdata= "C:\\RMS\_workspace\\First\_RMSProject\\";

String File\_testdata="RMS data.xlsx";

**public** **static** XSSFWorkbook *work*;

**public** **static** XSSFSheet *sheet*;

**public** **static** XSSFCell *cell*;

**public** **static** FileInputStream *fs*;

**public** **static** **void** setexcel(String path,String sheetname) **throws** IOException {

**try**{

*fs*= **new** FileInputStream(path);

*work*=**new** XSSFWorkbook(*fs*);

*sheet*= *work*.getSheet(sheetname);

}

**catch**(Exception e)

{

**throw**(e);

}

}

**public** **static** String getCellData(**int** RowNum, **int** ColNum)

{

**try**{

*cell*=*sheet*.getRow(RowNum).getCell(ColNum);

String cellvalue;

/\*switch (cell.getCellType()) {

case XSSFCell.CELL\_TYPE\_NUMERIC:

System.out.println(cell.getRawValue());

cellvalue=cell.getRawValue();

break;

case XSSFCell.CELL\_TYPE\_BOOLEAN:

System.out.println(cell.getBooleanCellValue());

cellvalue=cell.getBooleanCellValue();

break;

case XSSFCell.CELL\_TYPE\_STRING:

System.out.println(cell.getStringCellValue());

cellvalue=cell.getStringCellValue();

break;

default:

System.out.println(cell.getRawValue());

cellvalue=cell.getRawValue();

} \*/

cellvalue=*cell*.getStringCellValue();

**return** cellvalue;

/\* for(int i=0;i<=10;i++)

{

XSSFRow allrow = sheet.getRow(i);

for(int j=0;j<=1;j++)

{

XSSFCell allcell = allrow.getCell(j);

System.out.println("All row values are " +allcell);

}

//sheet.getRow(3).getCell(0);

} \*/

}

**catch**(Exception e)

{

**throw**(e);

}

}

}

/\* FileOutputStream ofs=new FileOutputStream(path\_testdata);

work.write(ofs);

fs.close();

work.close();

}

}

}\*/

**Fillfeedback.java**

**package** rmspackage;

**import** java.util.Set;

**import** java.util.concurrent.TimeUnit;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebDriverException;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.support.FindBy;

**import** org.openqa.selenium.support.PageFactory;

**import** org.testng.Assert;

**import** org.testng.annotations.Test;

**public** **class** Fillfeedback {

WebDriver driver;

@FindBy(linkText="Feedback")

WebElement Feedbacklink;

@FindBy(linkText="Pending feedback")

WebElement pendingfeedlink;

@FindBy(id="FillFeedbackDialogue")

WebElement fillicon;

@FindBy(xpath=".//\*[@id='innerTitleHeader']/div[1]/h1")

WebElement Filltitle;

@FindBy(id="radButtonListQuestion1\_0")

WebElement ques1select;

@FindBy(id="radButtonListQuestion2\_1")

WebElement ques2select;

@FindBy(id="radButtonListQuestion3\_2")

WebElement ques3select;

@FindBy(id="radButtonListQuestion4\_3")

WebElement ques4select;

@FindBy(id="radButtonListQuestion5\_4")

WebElement ques5select;

@FindBy(id="radButtonListQuestion6\_5")

WebElement ques6select;

@FindBy(id="radButtonListQuestion7\_0")

WebElement ques7select;

@FindBy(id="radButtonListQuestion8\_1")

WebElement ques8select;

@FindBy(id="radButtonListQuestion9\_3")

WebElement ques9select;

@FindBy(id="radButtonListQuestion10\_4")

WebElement ques10select;

@FindBy(id="txtCommentField")

WebElement feedcomment;

@FindBy(id="btnSubmitFeedback")

WebElement submitfeedback;

@FindBy(xpath=".//\*[@id='divAcknowledgement']/div/div/div/h1")

WebElement feedbackacknow;

//constructor

**public** Fillfeedback(WebDriver driver)

{

**this**.driver=driver;

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

}

//Open pending feedback if present

**public** **void** open\_pendingfeedback()

{

**boolean** feedbackpresent = driver.findElements(By.*linkText*("Feedback")).size()>0;

**if**(feedbackpresent)

{

Feedbacklink.click();

}

**else**

{

System.***out***.println("Feedback module is not present");

}

**boolean** pendingfeedbackpresent = driver.findElements(By.*linkText*("Pending feedback")).size()>0;

**if**(pendingfeedbackpresent)

{

pendingfeedlink.click();

}

**else**

{

System.***out***.println("Pending Feedback module is not present");

}

}

//Open feedback form if available

**public** **void** open\_feedbackform() **throws** InterruptedException

{

**boolean** feedbackformpresent = driver.findElements(By.*id*("FillFeedbackDialogue")).size()>0;

**if**(feedbackformpresent)

{

fillicon.click();

//Set<String> handle=driver.getWindowHandles();

**for**(String handle:driver.getWindowHandles())

{

driver.switchTo().window(handle);

}

Thread.*sleep*(4000);

//driver.switchTo().window("");

String Fillfeedbacktitle = Filltitle.getText();

Assert.*assertEquals*(Fillfeedbacktitle, "Fill feedback","Fillfeedback page not opened");

fill\_feedbackform();

}

**else**

{

System.***out***.println("No feedback is available in pending feedback");

}

}

//Fill feedback form

**public** **void** fill\_feedbackform() **throws** InterruptedException

{

String windowbefore=driver.getWindowHandle();

driver.manage().timeouts().implicitlyWait(8, TimeUnit.***SECONDS***);

ques1select.click();

ques2select.click();

ques3select.click();

ques4select.click();

ques5select.click();

ques6select.click();

ques7select.click();

ques8select.click();

ques9select.click();

ques10select.click();

feedcomment.sendKeys("Enter comments for feedback");

//submitfeedback.click();

String acknowledgment=feedbackacknow.getText();

Assert.*assertEquals*(acknowledgment, "Your feedback has been submitted");

Thread.*sleep*(1000);

driver.close();

driver.switchTo().window(windowbefore);

}

//Perform functionality of fill feedback module

**public** **void** fill\_feedback1() **throws** InterruptedException {

System.***out***.println("---\*\*\* Fillfeedback.fill\_feedback1() ---\*\*\*");

Feedbacklink.click();

**boolean** pendingfeedbackpresent = driver.findElements(By.*linkText*("Pending feedback")).size()>0;

**if**(pendingfeedbackpresent)

{

pendingfeedlink.click();

Feedbacklink.click();

// open\_pendingfeedback();

open\_feedbackform();

}

**else**

{

System.***out***.println("Pending Feedback module is not present");

}

}

}

**Filltimesheet.java**

**package** rmspackage;

**import** java.util.concurrent.TimeUnit;

**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.Select;

**import** org.testng.Assert;

**import** org.testng.annotations.Test;

**import** rmspackage.ExcelUtils;

**public** **class** FillTimesheet {

WebDriver driver;

@FindBy(linkText="Timesheet")

WebElement timesheet;

@FindBy(linkText="Fill timesheet")

WebElement Fill;

@FindBy(xpath="//input[contains(@id,'\_txtRequestIds')]")

WebElement Req\_id;

@FindBy(id="dateTimesheetDate")

WebElement timesheet\_date;

@FindBy(xpath=".//\*[@id='ctl00\_SPWebPartManager1\_g\_9cee05a3\_a7fe\_4289\_a52a\_79f82a342526\_tblData']/tbody/tr[4]/td[2]/div/div/div/div[1]/div[2]/table/tbody/tr[1]/td[1]/div")

WebElement calenderdate;

@FindBy(xpath="//table[@id='ctl00\_SPWebPartManager1\_g\_9cee05a3\_a7fe\_4289\_a52a\_79f82a342526\_tblData']/tbody/tr[4]/td[2]/div/div/div/div[1]/div[2]/table/tbody/tr[2]/td[1]")

WebElement calenderact;

@FindBy(id="btnGo")

WebElement Idgo;

@FindBy(xpath="//input[contains(@id,'\_txtHours')]")

WebElement hours;

@FindBy(xpath="//textarea[contains(@id,'\_txtComments')]")

WebElement timesheet\_comment;

@FindBy(xpath="//input[contains(@id,'\_btnSubmit')]")

WebElement timesheet\_submit;

@FindBy(xpath="//input[contains(@id,'\_radActivity')]")

WebElement Timesheet\_radio;

@FindBy(xpath="//select[contains(@id,'\_ddlTimesheetActivity')]")

WebElement act\_dropdown;

@FindBy(xpath="//input[contains(@id,'btnViewTimesheet')]")

WebElement Viewlink;

@FindBy(xpath="//input[contains(@id,'\_btnAddTimesheet')]")

WebElement Fillnewlink;

@FindBy(xpath="//input[contains(@id,'\_btnReset')]")

WebElement Resettimesheet;

@FindBy(xpath="//span[contains(@id,'\_rangeValidateHours')][@class='errorMsg']")

WebElement valimsg;

//Constructor

**public** FillTimesheet(WebDriver driver) {

**this**.driver=driver;

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

}

//Check If fill timesheet is present or not.

**public** **void** fill\_availability() **throws** InterruptedException

{

**boolean** fillpresent = driver.findElements(By.*linkText*("Fill timesheet")).size()>0;

**if**(fillpresent)

{

Fill.click();

Thread.*sleep*(2000);

//fill\_activity();

// reset();

Thread.*sleep*(1000);

fill\_reqid();

Thread.*sleep*(3000);

//String date=ExcelUtils.getCellData(21, 2);

// timesheet\_date.sendKeys(date);

//fill\_date();

//calenderdate.click();

// System.out.println("get text in time" +get);

Thread.*sleep*(1000);

fill\_hours();

Thread.*sleep*(1000);

fill\_comment();

Thread.*sleep*(1000);

timesheet\_submit.click();

Thread.*sleep*(2000);

String timetitle=driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeader']/div[1]/h1")).getText();

//System.out.println("text is" +timetitle);

Assert.*assertEquals*(timetitle,"Timesheet acknowledgement","Title not matches");

Viewlink.click();

}

**else**{

System.***out***.println("Fill Timesheet module is not present");

}

}

//Fill Request ID and click Go

**public** **void** fill\_reqid()

{

String ID=ExcelUtils.*getCellData*(21, 2);

Req\_id.sendKeys(ID);

Idgo.click();

}

//Fill timesheet date

**public** **void** fill\_date() **throws** InterruptedException

{

timesheet\_date.click();

Thread.*sleep*(2000);

driver.findElement(By.*className*("xdsoft\_today\_button")).click();

Thread.*sleep*(2000);

}

//Fill hours

**public** **void** fill\_hours() **throws** InterruptedException

{

String morehour=ExcelUtils.*getCellData*(23, 3);

hours.sendKeys(morehour);

timesheet\_comment.click();

Thread.*sleep*(1000);

Assert.*assertEquals*(valimsg.getText(),"Timesheet hours should be between 0.1-24","No validation message");

hours.clear();

String hour=ExcelUtils.*getCellData*(23, 2);

hours.sendKeys(hour);

}

//Fill timesheet comment

**public** **void** fill\_comment() **throws** InterruptedException

{

String comment=ExcelUtils.*getCellData*(24, 2);

timesheet\_comment.sendKeys(comment);

}

//Fill timesheet by Request

**public** **void** fill\_timesheet() **throws** InterruptedException

{

System.***out***.println("---\*\*\* FillTimesheet.fill\_timesheet() \*\*\*---");

Thread.*sleep*(1000);

timesheet.click();

fill\_availability();

Thread.*sleep*(1000);

}

//Fill timesheet by Activity

**public** **void** fill\_activity() **throws** InterruptedException

{

System.***out***.println("---\*\*\* FillTimesheet.fill\_activity() \*\*\*---");

Thread.*sleep*(1000);

Timesheet\_radio.click();

Thread.*sleep*(1000);

Select sel=**new** Select(act\_dropdown);

String act=ExcelUtils.*getCellData*(25, 2);

sel.selectByVisibleText(act);

Thread.*sleep*(1000);

//fill\_date();

//calenderact.click();

//Thread.sleep(1000);

fill\_hours();

Thread.*sleep*(1000);

fill\_comment();

Thread.*sleep*(1000);

timesheet\_submit.click();

Thread.*sleep*(3000);

//String timetitle=driver.findElement(By.xpath("//div/div/div/div/div/h1")).getText();

String timetitle=driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeader']/div[1]/h1")).getText();

Assert.*assertEquals*(timetitle, "Timesheet acknowledgement","Title matches");

Fillnewlink.click();

}

//Check Reset functionality

**public** **void** reset() **throws** InterruptedException

{

System.***out***.println("---\*\*\* FillTimesheet.reset() \*\*\*---");

Thread.*sleep*(1000);

timesheet.click();

Thread.*sleep*(1000);

Fill.click();

Thread.*sleep*(1000);

fill\_reqid();

Thread.*sleep*(2000);

Resettimesheet.click();

Thread.*sleep*(2000);

String id\_present = Req\_id.getText();

Assert.*assertEquals*(id\_present, "", "Request Id is not blank");

}

}

**Loginclass.java**

**package** rmspackage;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.support.FindBy;

**import** org.openqa.selenium.support.PageFactory;

**import** rmspackage.ExcelUtils;

**public** **class** Loginclass {

WebDriver driver;

@FindBy(id="ctl00\_PlaceHolderMain\_signInControl\_UserName")

WebElement username;

@FindBy(id="ctl00\_PlaceHolderMain\_signInControl\_password")

WebElement password;

@FindBy(id="ctl00\_PlaceHolderMain\_signInControl\_login")

WebElement login;

@FindBy(id="ctl00\_HeaderControl1\_lnkRightDD")

WebElement useredit;

@FindBy(id="ctl00\_HeaderControl1\_lnkSignOut")

WebElement logoutlink;

**public** Loginclass(WebDriver driver) {

**this**.driver=driver;

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

}

//Open URL

**public** **void** logintourl()

{

String URL=ExcelUtils.*getCellData*(1, 2);

System.***out***.println("URL is " +URL);

driver.get(URL);

}

// Methods to login to application

**public** **void** enteruser(**int** i) **throws** InterruptedException

{

String user=ExcelUtils.*getCellData*(2, i);

System.***out***.println("Username is " +user);

username.click();

username.sendKeys(user);

}

**public** **void** enterpswd(**int** i) **throws** InterruptedException

{

String pswd=ExcelUtils.*getCellData*(3, i);

System.***out***.println("Password is " +pswd);

password.click();

password.sendKeys(pswd);

}

**public** **void** log()

{

login.click();

}

**public** **void** logout() **throws** InterruptedException

{

Thread.*sleep*(1000);

useredit.click();

Thread.*sleep*(1000);

logoutlink.click();

}

}

**NewRequest.java**

**package** rmspackage;

**import** java.awt.Robot;

**import** java.awt.Toolkit;

**import** java.awt.datatransfer.StringSelection;

**import** java.awt.event.KeyEvent;

**import** java.util.List;

**import** java.util.concurrent.TimeUnit;

**import** junit.framework.~~Assert~~;

**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.WebDriverWait;

**public** **class** New\_Request {

WebDriver driver;

@FindBy(linkText="New request")

WebElement Requestlink;

@FindBy(xpath="//a[contains(@id,'ddlRegion\_Arrow')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_ddlRequestFormType\_Arrow")

WebElement region;

@FindAll({@FindBy(tagName="li")})

List<WebElement> regionvalue;

@FindBy(xpath="//a[contains(@id,'ddlCoverageGroup\_Arrow')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_68cb04f8\_161f\_4e9a\_8842\_a39c56f735c7\_ddlCoverageGroup\_Arrow")

WebElement coverage;

@FindAll({@FindBy(tagName="li")})

List<WebElement> covvalue;

@FindBy(className="rcbItem")

WebElement covclass;

@FindBy(xpath="//input[contains(@id,'ddlRequestor\_Input')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_ddlRequestor\_Input")

WebElement Banker;

@FindBy(xpath="//input[contains(@id,'\_listBoxCoOwner\_Input')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_listBoxCoOwner\_Input")

WebElement coowner;

@FindBy(xpath="//input[contains(@id,'\_txtRequestTitle')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_txtRequestTitle")

WebElement Title;

@FindBy(xpath="//input[contains(@id,'\_txtClientProjectCode')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_txtClientProjectCode")

WebElement code;

@FindBy(xpath="//a[contains(@id,'\_ddlRequestPriority\_Arrow')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_ddlRequestPriority\_Arrow")

WebElement priority;

@FindAll({@FindBy(className="rcbItem")})

List<WebElement> priclass;

@FindBy(id="dateExpectedDeliveryDateTime")

WebElement originaldate;

@FindBy(id="timeExpectedDeliveryDateTime")

WebElement originaltime;

@FindBy(xpath="//a[contains(@id,'\_ddlMemberFirm\_Arrow')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_68cb04f8\_161f\_4e9a\_8842\_a39c56f735c7\_ddlMemberFirm\_Input")

WebElement Reqcoverage;

//By requesttype=By.id("ctl00\_SPWebPartManager1\_g\_68cb04f8\_161f\_4e9a\_8842\_a39c56f735c7\_ddlRequestType\_Input");

@FindBy(xpath="//a[contains(@id,'\_ddlRequestType\_Arrow')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_ddlRequestType\_Arrow")

WebElement requesttype;

@FindBy(className="rcbItem")

WebElement typeclass;

@FindBy(xpath="//a[contains(@id,'\_ddlRequestSubType\_Arrow')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_ddlRequestSubType\_Arrow")

WebElement subtype;

@FindBy(className ="rcbItem")

WebElement subclass;

@FindBy(id="txtVolume")

WebElement volume;

@FindBy(id="btnRequestMultipleVolume")

WebElement Add;

@FindBy(xpath="//iframe[contains(@id,'\_richTextRequestDescription\_contentIframe')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b65c754d\_38c4\_4ab6\_8735\_e954ebdb9dda\_richTextRequestDescription\_contentIframe")

WebElement desframe;

@FindBy(tagName="body")

WebElement desc;

@FindBy(xpath="//input[contains(@id,'\_RequestSupportingDocument\_radUpSupportingDocumentfile0')]")

WebElement upload;

@FindBy(id="btnReq")

WebElement submit;

@FindBy(xpath="//input[contains(@id,'\_btnViewRequests')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_f850089b\_cd99\_4089\_ae99\_4f81201e95e4\_btnViewRequests")

WebElement viewreq;

String path="C:\\Users\\chaman.preet\\Desktop\\Projects without test cases (Autosaved).xlsx";

**public** New\_Request(WebDriver driver) {

**this**.driver=driver;

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

}

**public** **void** clicklink()

{

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

Requestlink.click();

}

**public** **void** selectregion(**int** i)

{

region.click();

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

//regvalue=ExcelUtils.getCellData(5, 2);

regionvalue.get(i).click();

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**public** **void** selectcoverage(**int** i) **throws** InterruptedException

{

coverage.click();

Thread.*sleep*(1000);

covvalue.get(i).click();

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

//covclass.click();

}

**public** **void** selectbanker() **throws** InterruptedException

{

String bank=ExcelUtils.*getCellData*(7, 2);

Banker.sendKeys(bank);

Thread.*sleep*(1000);

driver.findElement(By.*className*("racList")).click();

}

**public** **void** selectcoowner() **throws** InterruptedException

{

String coo=ExcelUtils.*getCellData*(19, 2);

coowner.sendKeys(coo);

Thread.*sleep*(1000);

driver.findElement(By.*className*("racList")).click();

}

**public** **void** selecttitle()

{

String titl=(String) ExcelUtils.*getCellData*(8, 2);

Title.sendKeys(titl);

}

**public** **void** selectcode()

{

String co=ExcelUtils.*getCellData*(9, 2);

code.sendKeys(co);

}

**public** **void** selectpriority() **throws** InterruptedException

{

priority.click();

Thread.*sleep*(2000);

//driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);

priclass.get(2).click();

}

**public** **void** date()

{

String dt=ExcelUtils.*getCellData*(11, 2);

originaldate.sendKeys(dt);

driver.manage().timeouts().implicitlyWait(15, TimeUnit.***SECONDS***);

originaldate.click();

}

**public** **void** time()

{

String tm=ExcelUtils.*getCellData*(12, 2);

originaltime.sendKeys(tm);

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

originaltime.click();

}

**public** **void** selectreqcov()

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

Reqcoverage.click();

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

covvalue.get(1).click();

}

**public** **void** selecttype() **throws** InterruptedException

{

requesttype.click();

Thread.*sleep*(2000);

// driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

typeclass.click();

}

**public** **void** selectsubtype() **throws** InterruptedException

{

subtype.click();

Thread.*sleep*(2000);

subclass.click();

}

**public** **void** selectvol()

{

String vol=ExcelUtils.*getCellData*(16, 2);

volume.sendKeys(vol);

}

/\*public void clickadd()

{

Add.click();

}\*/

**public** **void** selectframe()

{

driver.switchTo().frame(desframe);

}

**public** **void** enterdesc()

{

String body=ExcelUtils.*getCellData*(17, 2);

desc.sendKeys(body);

driver.switchTo().defaultContent();

}

**public** **void** uploadfile() **throws** InterruptedException

{

StringSelection ss=**new** StringSelection(path);

Toolkit.*getDefaultToolkit*().getSystemClipboard().setContents(ss, **null**);

Thread.*sleep*(1000);

upload.click();

/\*JavascriptExecutor js = (JavascriptExecutor)driver;

js.executeScript("arguments[0].click();", upload);

Thread.sleep(3000);\*/

System.***out***.println("New\_Request.uploadfile()");

**try**{

Robot r=**new** Robot();

r.keyPress(KeyEvent.***VK\_CONTROL***);

r.keyPress(KeyEvent.***VK\_V***);

r.keyRelease(KeyEvent.***VK\_V***);

r.keyRelease(KeyEvent.***VK\_CONTROL***);

r.delay(1000);

r.keyPress(KeyEvent.***VK\_ENTER***);

r.keyRelease(KeyEvent.***VK\_ENTER***);

r.delay(1000);

}

**catch**(Exception e)

{

e.printStackTrace();

}

}

**public** **void** submitreq()

{

submit.click();

//JavascriptExecutor js = (JavascriptExecutor)driver;

// js.executeScript("arguments[0].click();", submit);

}

**public** **void** reqack()

{

String str = driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeader']/div[1]/h1")).getText();

System.***out***.println("title is " +str);

~~Assert~~.*assertEquals*("Request acknowledgement",str);

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

viewreq.click();

}

**public** **void** submit\_req() **throws** InterruptedException

{

clicklink();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

String logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

System.***out***.println("Logname is" +logname);

selectregion(1);

Thread.*sleep*(3000);

**switch**(logname)

{

**case** "Analyst Demo2":

selectcoverage(2);

**break**;

**case** "Teammanager Demo2":

selectcoverage(4);

**break**;

**case** "Gatekeeper Demo2":

selectcoverage(2);

**break**;

}

Thread.*sleep*(2000);

selectbanker();

Thread.*sleep*(2000);

selectcoowner();

Thread.*sleep*(1000);

selecttitle();

Thread.*sleep*(1000);

selectcode();

Thread.*sleep*(1000);

selectpriority();

Thread.*sleep*(1000);

date();

Thread.*sleep*(1000);

time();

Thread.*sleep*(1000);

selectreqcov();

Thread.*sleep*(1000);

selecttype();

Thread.*sleep*(2000);

selectsubtype();

Thread.*sleep*(2000);

selectvol();

Thread.*sleep*(2000);

selectframe();

enterdesc();

Thread.*sleep*(3000);

uploadfile();

Thread.*sleep*(1000);

submitreq();

Thread.*sleep*(3000);

reqack();

}

**public** **void** submit\_req\_requestor() **throws** InterruptedException

{

clicklink();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

selectregion(1);

Thread.*sleep*(3000);

selectcoverage(1);

Thread.*sleep*(3000);

//selectcoowner();

Thread.*sleep*(1000);

selecttitle();

Thread.*sleep*(1000);

selectcode();

Thread.*sleep*(1000);

selectpriority();

Thread.*sleep*(1000);

date();

// objreq.date("06/15/2016");

Thread.*sleep*(1000);

time();

Thread.*sleep*(1000);

selectreqcov();

Thread.*sleep*(1000);

selecttype();

Thread.*sleep*(3000);

selectsubtype();

Thread.*sleep*(1000);

selectvol();

Thread.*sleep*(1000);

selectframe();

enterdesc();

Thread.*sleep*(3000);

uploadfile();

Thread.*sleep*(1000);

submitreq();

Thread.*sleep*(3000);

reqack();

}

**public** **void** raise\_Req() **throws** InterruptedException

{

String logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

**switch**(logname)

{

**case** "Requestor Demo2":

submit\_req\_requestor();

**break**;

**default**:

submit\_req();

**break**;

}

}

}

**Reports.java**

**package** rmspackage;

**import** java.util.List;

**import** org.openqa.selenium.By;

**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.testng.annotations.Test;

**public** **class** Reports {

WebDriver driver;

@FindBy(linkText="Reports")

WebElement clickreports;

@FindBy(linkText="My reports")

WebElement myreportlink;

@FindAll({@FindBy(linkText="Request details")})

List<WebElement> reqdetails;

@FindAll({@FindBy(linkText="Large Request details")})

List<WebElement> lareqdetails;

@FindAll({@FindBy(linkText="Request multiple volume")})

List<WebElement> Reqmultiple;

@FindBy(xpath="//li[contains(@id,'\_ctl02\_liCategory')]")

WebElement feedbacklink;

@FindAll({@FindBy(linkText="Feedback request")})

List<WebElement> Feedbackreq;

@FindAll({@FindBy(linkText="Request type")})

List<WebElement> reqtype;

@FindBy(xpath="//li[contains(@id,'\_ctl03\_liCategory')]")

WebElement timelink;

@FindAll({@FindBy(linkText="My timesheet report")})

List<WebElement> mytimesheetrep;

@FindAll({@FindBy(linkText="Timesheet report")})

List<WebElement> timesheetrep;

@FindBy(xpath="//li[contains(@id,'\_ctl04\_liCategory')]")

WebElement userlink;

@FindAll({@FindBy(linkText="User details")})

List<WebElement> userdetails;

@FindAll({@FindBy(linkText="User mappings")})

List<WebElement> usermapping;

@FindBy(xpath="//li[contains(@id,'\_ctl05\_liCategory')]")

WebElement SLAlink;

@FindAll({@FindBy(linkText="SLA report")})

List<WebElement> SLAreport;

@FindBy(xpath="//li[contains(@id,'\_ctl06\_liCategory')]")

WebElement auditlink;

@FindAll({@FindBy(linkText="Tracking report")})

List<WebElement> trackingreport;

@FindBy(xpath="//li[contains(@id,'\_ctl07\_liCategory')]")

WebElement knowlink;

@FindAll({@FindBy(linkText="Knowledge document details")})

List<WebElement> knowdoc;

@FindAll({@FindBy(xpath="//a[contains(@id,'lnkReport')]")})

//@FindAll({@FindBy(xpath="//a[contains(@class,'toolTipClass')][@href='javascript:void()']")})

List<WebElement> countreports;

//constructor

**public** Reports(WebDriver driver) {

**this**.driver=driver;

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

}

//Check visibility of request details report

**public** **void** Req\_detail()

{

**boolean** reqpresent=reqdetails.size()>0;

// boolean reqpresent = driver.findElements(By.linkText("Requests details")).size()>0;

**if**(reqpresent)

{

// System.out.println("Request details report is PRESENT");

}

**else**

{

System.***out***.println("Request details report is not present");

}

}

//Check visibility of large request details report

**public** **void** Large\_Req\_detail()

{

**boolean** largereqpresent =lareqdetails.size()>0;

**if**(largereqpresent)

{

// System.out.println("Large Request details report is PRESENT");

}

**else**

{

System.***out***.println("Large Request details report is not present");

}

}

//Check visibility of request multiple volume report

**public** **void** Req\_Multiple\_vol()

{

**boolean** reqmultiplepresent =Reqmultiple.size()>0;

**if**(reqmultiplepresent)

{

//System.out.println("Request multiple volume report is PRESENT");

}

**else**

{

System.***out***.println("Request multiple volume report is not present");

}

}

//Check visibility of feedbck request report

**public** **void** Feedback\_req()

{

**boolean** feedbackreqpresent =Feedbackreq.size()>0;

**if**(feedbackreqpresent)

{

// System.out.println("Feedback Request report is PRESENT");

}

**else**

{

System.***out***.println("Feedback Request report is not present");

}

}

//Check visibility of request type report

**public** **void** Request\_type()

{

**boolean** reqtypepresent =reqtype.size()>0;

**if**(reqtypepresent)

{

//System.out.println("Request type report is PRESENT");

}

**else**

{

System.***out***.println("Request type report is not present");

}

}

//Check visibility of timesheet report

**public** **void** timesheet\_report()

{

**boolean** timesheetreppresent =timesheetrep.size()>0;

**if**(timesheetreppresent)

{

// System.out.println("Timesheet report is PRESENT");

}

**else**

{

System.***out***.println("Timesheet report is not present");

}

}

//Check visibility of timesheet report

**public** **void** my\_timesheet\_report()

{

**boolean** timesheetreppresent =mytimesheetrep.size()>0;

**if**(timesheetreppresent)

{

// System.out.println(" My Timesheet report is PRESENT");

}

**else**

{

System.***out***.println("Timesheet report is not present");

}

}

//Check visibility of user details report

**public** **void** user\_details()

{

**boolean** userdetpresent =userdetails.size()>0;

**if**(userdetpresent)

{

//System.out.println("User details report is PRESENT");

}

**else**

{

System.***out***.println("User details report is not present");

}

}

//Check visibility of user mapping report

**public** **void** user\_mappings()

{

**boolean** mappingpresent =usermapping.size()>0;

**if**(mappingpresent)

{

// System.out.println("User mapping report is PRESENT");

}

**else**

{

System.***out***.println("User mapping report is not present");

}

}

//Check visibility of SLA report

**public** **void** SLA\_Report()

{

**boolean** SLApresent =SLAreport.size()>0;

**if**(SLApresent)

{

// System.out.println("SLA report is PRESENT");

}

**else**

{

System.***out***.println("SLA report is not present");

}

}

//Check visibility of tracking report

**public** **void** tracking\_Report()

{

**boolean** trackingpresent =trackingreport.size()>0;

**if**(trackingpresent)

{

//System.out.println("Tracking report is PRESENT");

}

**else**

{

System.***out***.println("Tracking report is not present");

}

}

//Check visibility of Knowledge document details report

**public** **void** know\_doc\_Report()

{

**boolean** knowedgepresent =knowdoc.size()>0;

**if**(knowedgepresent)

{

// System.out.println("Knowledge document details report is PRESENT");

}

**else**

{

System.***out***.println("Knowledge document details report is not present");

}

}

//Check reports visibility for Team Manager

**public** **void** reports\_TM() **throws** InterruptedException

{

System.***out***.println("---\*\*\* Reports.reports\_TM() \*\*\*---");

**if**(driver.findElements(By.*linkText*("Reports")).size()>0)

{

clickreports.click();

Thread.*sleep*(1000);

**if**(driver.findElements(By.*linkText*("My reports")).size()>0)

{

myreportlink.click();

Thread.*sleep*(2000);

String logname = driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

System.***out***.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

Req\_detail();

Large\_Req\_detail();

Req\_Multiple\_vol();

feedbacklink.click();

Thread.*sleep*(1000);

Feedback\_req();

Request\_type();

timelink.click();

Thread.*sleep*(1000);

timesheet\_report();

userlink.click();

Thread.*sleep*(1000);

user\_details();

user\_mappings();

SLAlink.click();

Thread.*sleep*(1000);

SLA\_Report();

auditlink.click();

Thread.*sleep*(1000);

tracking\_Report();

knowlink.click();

Thread.*sleep*(1000);

know\_doc\_Report();

**if**(countreports.size()!=11)

{

System.***out***.println("INCORRECT reports displaying for Team manager");

}

**else**

{

System.***out***.println("Correct reports displaying for Team manager");

}

**break**;

**case** "Analyst Demo2":

Req\_detail();

feedbacklink.click();

Thread.*sleep*(1000);

my\_timesheet\_report();

**if**(countreports.size()!=2)

{

System.***out***.println("INCORRECT reports displaying for Analyst");

}

**else**

{

System.***out***.println("Correct reports displaying for Analyst");

}

**break**;

}

}

**else**

{

System.***out***.println("\n" + "My Reports is not present");

}

}

**else**

{

System.***out***.println("\n" + "Reports module not present");

}

}

}

**Req\_deliverables.java**

**package** rmspackage;

**import** java.awt.AWTException;

**import** java.awt.Robot;

**import** java.awt.Toolkit;

**import** java.awt.datatransfer.StringSelection;

**import** java.awt.event.KeyEvent;

**import** java.util.List;

**import** java.util.concurrent.TimeUnit;

**import** rmspackage.Request\_details;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.firefox.FirefoxProfile;

**import** org.openqa.selenium.support.FindAll;

**import** org.openqa.selenium.support.FindBy;

**import** org.openqa.selenium.support.PageFactory;

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

**import** org.testng.Assert;

**public** **class** Req\_deliverables {

WebDriver driver;

Request\_details objdetails;

@FindBy(linkText="View requests")

WebElement View;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl00\_lnkButtonRequestStatus")

WebElement unassignlink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl01\_lnkButtonRequestStatus")

WebElement inprogresslink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl02\_lnkButtonRequestStatus")

WebElement onholdlink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl03\_lnkButtonRequestStatus")

WebElement cancellink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl04\_lnkButtonRequestStatus")

WebElement completelink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl05\_lnkButtonRequestStatus")

WebElement closelink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl06\_lnkButtonRequestStatus")

WebElement TBDlink;

/\* @FindBy(id="lnkFilterpane")

WebElement Filter;

@FindBy(id="ctl00\_SPWebPartManager1\_RequestFilterProvider\_rptrFilters\_ctl02\_lst\_\_Arrow")

WebElement Status;

@FindAll({@FindBy(className="rcbCheckBox")})

List<WebElement> checkstatus;

@FindBy(id="btnFilter")

WebElement Apply;\*/

@FindBy(linkText="Deliverables")

WebElement deliclick;

@FindBy(xpath="//Select[contains(@id,'\_ddlDeliverableType')]")

WebElement type;

@FindBy(xpath="//input[contains(@id,'\_radUpDeliverableDocumentfile0')]")

WebElement choose;

@FindBy(tagName="iframe")

WebElement frametag;

@FindBy(xpath="//iframe[contains(@id,'\_rptrDeliverables\_ctl00\_richTextMultiLineTextBox1\_contentIframe')]")

WebElement descframe;

@FindBy(tagName="body")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_b9b66502\_20fe\_447d\_ada5\_7decc0bdf3a3\_richTextMultiLineTextBox1ContentHiddenTextarea")

WebElement delcomment;

@FindBy(id="btnUpload")

WebElement delsubmit;

@FindBy(xpath="//input[contains(@id,'\_rptrDeliverables\_ctl00\_btnAddComment')]")

WebElement addsubmit;

@FindBy(id="btnDownloadAllFiles")

WebElement downbtn;

@FindBy(xpath="//a[contains(@id,'\_imgDeleteDeliverableSet')]")

WebElement Deletedel;

@FindAll({@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_ddlRecordsPerPage")})

List<WebElement> requestexist;

@FindAll({@FindBy(xpath="//a[starts-with(@class,'icon')][contains(@class,'toolTipClass')]")})

//@FindAll({@FindBy(xpath="//a[contains(@class,'toolTipClass')][@href='javascript:void()']")})

List<WebElement> countelements;

String path1="C:\\Users\\chaman.preet\\Downloads\\TimesheetReport (9) (1) (2).xlsx";

String logname;

**public** **static** String *downloadPath* = "C:\\Users\\chaman.preet\\Desktop\\download";

**public** Req\_deliverables(WebDriver driver)

{

**this**.driver=driver;

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

}

**public** **void** opendeliverables() **throws** InterruptedException

{

Thread.*sleep*(1000);

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

Thread.*sleep*(3000);

deliclick.click();

}

**public** **void** upload() **throws** InterruptedException, AWTException {

/\*View.click();

Thread.sleep(3000);

inprogresslink.click();

Thread.sleep(3000);

opendeliverables();

Thread.sleep(1000);\*/

Select selecttype=**new** Select(type);

selecttype.selectByVisibleText("Interim");

Thread.*sleep*(1000);

choose.click();

Thread.*sleep*(1000);

StringSelection ss=**new** StringSelection(path1);

Toolkit.*getDefaultToolkit*().getSystemClipboard().setContents(ss, **null**);

Robot rob=**new** Robot();

rob.keyPress(KeyEvent.***VK\_CONTROL***);

rob.keyPress(KeyEvent.***VK\_V***);

rob.keyRelease(KeyEvent.***VK\_CONTROL***);

rob.keyRelease(KeyEvent.***VK\_V***);

Thread.*sleep*(1000);

rob.keyPress(KeyEvent.***VK\_ENTER***);

rob.keyRelease(KeyEvent.***VK\_ENTER***);

Thread.*sleep*(1000);

driver.switchTo().frame(frametag);

delcomment.sendKeys("Add comment while uploading deliverables");

driver.switchTo().defaultContent();

Thread.*sleep*(1000);

delsubmit.click();

}

**public** **void** addcomment() **throws** InterruptedException

{

//opendeliverables();

Thread.*sleep*(1000);

//driver.switchTo().frame(3);

driver.switchTo().frame(descframe);

delcomment.sendKeys("Add comment in comment section");

driver.switchTo().defaultContent();

addsubmit.click();

}

**public** **void** deletedeliverable() **throws** InterruptedException, AWTException

{

opendeliverables();

Thread.*sleep*(1000);

Deletedel.click();

Thread.*sleep*(1000);

Robot rob=**new** Robot();

rob.keyPress(KeyEvent.***VK\_ENTER***);

rob.keyRelease(KeyEvent.***VK\_ENTER***);

rob.delay(1000);

}

**public** FirefoxProfile download() **throws** InterruptedException, AWTException

{

/\*View.click();

Thread.sleep(3000);

inprogresslink.click();

Thread.sleep(2000);

opendeliverables();\*/

downbtn.click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

FirefoxProfile profile = **new** FirefoxProfile();

profile.setPreference("browser.download.folderList", 2);

profile.setPreference("browser.download.manager.showWhenStarting", **false**);

profile.setPreference("browser.download.dir", *downloadPath*);

profile.setPreference("browser.helperApps.neverAsk.openFile",

"text/csv,application/x-msexcel,application/excel,application/x-excel,application/vnd.ms-excel,image/png,image/jpeg,text/html,text/plain,application/msword,application/xml");

profile.setPreference("browser.helperApps.neverAsk.saveToDisk",

"text/csv,application/x-msexcel,application/excel,application/x-excel,application/vnd.ms-excel,image/png,image/jpeg,text/html,text/plain,application/msword,application/xml");

profile.setPreference("browser.helperApps.alwaysAsk.force", **false**);

profile.setPreference("browser.download.manager.alertOnEXEOpen", **false**);

profile.setPreference("browser.download.manager.focusWhenStarting", **false**);

profile.setPreference("browser.download.manager.useWindow", **false**);

profile.setPreference("browser.download.manager.showAlertOnComplete", **false**);

profile.setPreference("browser.download.manager.closeWhenDone", **false**);

Thread.*sleep*(1000);

Robot rob=**new** Robot();

rob.keyPress(KeyEvent.***VK\_DOWN***);

rob.keyRelease(KeyEvent.***VK\_DOWN***);

rob.keyPress(KeyEvent.***VK\_TAB***);

rob.keyRelease(KeyEvent.***VK\_TAB***);

rob.keyPress(KeyEvent.***VK\_TAB***);

rob.keyRelease(KeyEvent.***VK\_TAB***);

rob.keyPress(KeyEvent.***VK\_TAB***);

rob.keyRelease(KeyEvent.***VK\_TAB***);

Thread.*sleep*(1000);

rob.keyPress(KeyEvent.***VK\_ENTER***);

rob.keyRelease(KeyEvent.***VK\_ENTER***);

Thread.*sleep*(1000);

driver.navigate().back();

**return** profile;

}

//Perform all actions on deliverables

**public** **void** deliverable\_action() **throws** InterruptedException, AWTException

{

View.click();

Thread.*sleep*(3000);

inprogresslink.click();

Thread.*sleep*(3000);

opendeliverables();

Thread.*sleep*(1000);

**if**(driver.findElements(By.*xpath*("//div[contains(@id,'\_noData')][class='noData']")).size()>0)

{

System.***out***.println("Deliverables cannot be viewed");

}

**else**

{

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2": **case** "Analyst Demo2":

upload();

addcomment();

deletedeliverable();

download();

**break**;

**case** "Gatekeeper Demo2":

addcomment();

download();

**break**;

**case** "Requestor Demo2":

addcomment();

download();

**break**;

}

}

}

// To verify if assign analyst is present.

**public** **void** assign\_analyst\_details1() **throws** InterruptedException {

String detailtitle = driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeaderRequestDetail']/div[1]/h1")).getText();

Assert.*assertEquals*(detailtitle, "Deliverables");

**boolean** assignpresent=driver.findElements(By.*id*("lnkAssign")).size()>0;

**if**(assignpresent)

{

// System.out.println("Assign analyst action button is PRESENT");

}

**else**

{

System.***out***.println("Assign analyst action button is not present");

}

}

// To verify if add supporting document is present.

**public** **void** addsupporting\_details1()

{

**boolean** addpresent=driver.findElements(By.*id*("lnkAddSupportingDocument")).size()>0;

**if**(addpresent)

{

//System.out.println("Add supporting document action button is PRESENT");

}

**else**

{

System.***out***.println("Add supporting document action button is not present.");

}

}

// To verify if edit deadline is present.

**public** **void** editdead\_details1()

{

**boolean** editpresent=driver.findElements(By.*id*("lnkEditRequestDeadline")).size()>0;

**if**(editpresent)

{

//System.out.println("Edit deadline is PRESENT");

}

**else**{

System.***out***.println("Edit deadline is not present");

}

}

//To verify if edit request coverage is present.

**public** **void** editcoverage\_details1()

{

**boolean** covpresent=driver.findElements(By.*id*("lnkEditCoverageGroup")).size()>0;

**if**(covpresent)

{

//System.out.println("Edit coverage is PRESENT");

}

**else**

{

System.***out***.println("Edit coverage is not present");

}

}

//To verify if fill timesheet is present.

**public** **void** filltimesheet\_details1()

{

**boolean** timesheetpresent=driver.findElements(By.*id*("lnkFillTimesheet")).size()>0;

**if**(timesheetpresent)

{

// System.out.println("Fill timesheet is PRESENT");

}

**else**{

System.***out***.println("Fill timesheet is not present");

}

}

//To verify if Edit request is present.

**public** **void** Editrequest\_details1()

{

**boolean** requestpresent=driver.findElements(By.*id*("lnkEdit")).size()>0;

**if**(requestpresent)

{

// System.out.println("Edit request action button is PRESENT");

}

**else**

{

System.***out***.println("Edit request action button is not present");

}

}

//To verify if hold request is present.

**public** **void** holdreq\_details1()

{

**boolean** holdpresent=driver.findElements(By.*id*("lnkHold")).size()>0;

**if**(holdpresent)

{

//System.out.println("Hold action button is PRESENT");

}

**else**

{

System.***out***.println("Hold action button is not present");

}

}

//To verify if unhold request is present

**public** **void** unholdreq\_details1()

{

**boolean** unholdpresent=driver.findElements(By.*id*("lnkUnHold")).size()>0;

**if**(unholdpresent)

{

// System.out.println("Hold action button is PRESENT");

}

**else**

{

System.***out***.println("Hold action button is not available");

}

}

//To verify if cancel request is present.

**public** **void** cancelreq\_details1()

{

**boolean** cancelpresent=driver.findElements(By.*id*("lnkCancelRequest")).size()>0;

**if**(cancelpresent)

{

//System.out.println("Cancel action button is PRESENT");

}

**else**

{

System.***out***.println("Cancel action button is not available");

}

}

//To verify if version history is present.

**public** **void** requesthistory\_details1() **throws** InterruptedException

{

**boolean** historypresent=driver.findElements(By.*id*("lnkVersionHistory")).size()>0;

**if**(historypresent)

{

// System.out.println("Version history is PRESENT");

}

**else**

{

System.***out***.println("Version history is not present");

}

}

//To verify if recurring request is present.

**public** **void** recurring\_details1()

{

**boolean** recurpresent=driver.findElements(By.*id*("lnkRecurringRequest")).size()>0;

**if**(recurpresent)

{

// System.out.println("Recurring request is PRESENT");

}

**else**

{

System.***out***.println("Recurring request is not present");

}

}

//To verify if set as large request is present.

**public** **void** setaslarge1()

{

// if(largereq.isDisplayed())

**boolean** largepresent = driver.findElements(By.*id*("lnkSetAsLargeRequest")).size()>0;

**if**(largepresent)

{

// System.out.println("Set as large request is PRESENT");

}

**else**

{

System.***out***.println("Set as large request is not present");

}

}

//To verify if request comments is present.

**public** **void** req\_comments1()

{

**boolean** commentpresent=driver.findElements(By.*id*("lnkRequestComments")).size()>0;

**if**(commentpresent)

{

// System.out.println("req comment is PRESENT");

}

**else**

{

System.***out***.println("req comment is not present");

}

}

//To verify if deadline TBD is present.

**public** **void** deadline\_TBD1()

{

**boolean** TBDpresent=driver.findElements(By.*id*("lnkPostponeRequest")).size()>0;

**if**(TBDpresent)

{

// System.out.println("deadline TBD is PRESENT");

}

**else**

{

System.***out***.println("deadline TBD is not present");

}

}

//To verify if deadline confirmed is present.

**public** **void** deadline\_confirmed1()

{

**boolean** confirmpresent=driver.findElements(By.*id*("lnkResumeRequest")).size()>0;

**if**(confirmpresent)

{

// System.out.println("deadline confirmed is PRESENT");

}

**else**

{

System.***out***.println("deadline confirmed is not present");

}

}

//To verify if reactivate is present.

**public** **void** reactivatereq1() **throws** InterruptedException {

String detailtitle = driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeaderRequestDetail']/div[1]/h1")).getText();

Assert.*assertEquals*(detailtitle, "Request detail");

**boolean** reactivatepresent=driver.findElements(By.*id*("lnkReactivateRequest")).size()>0;

**if**(reactivatepresent)

{

// System.out.println("Reactivate action button is PRESENT");

}

**else**

{

System.***out***.println("Reactivate action button is not present");

}

}

//To verify if view feedback is present

**public** **void** view\_feedback1() **throws** InterruptedException

{

**boolean** viewpresent=driver.findElements(By.*id*("lnkViewFeedback")).size()>0;

**if**(viewpresent)

{

// System.out.println("View feedback button is PRESENT");

}

**else**

{

System.***out***.println("View feedback button is not available");

}

}

//To verify if fill feedback is present

**public** **void** fill\_feedback1() **throws** InterruptedException

{

**boolean** fillpresent=driver.findElements(By.*id*("lnkFillFeedback")).size()>0;

**if**(fillpresent)

{

// System.out.println("View feedback button is PRESENT");

}

**else**

{

System.***out***.println("Fill feedback button is not available");

}

}

//Perform action buttons for unassigned request

**public** **void** actionunassigned1() **throws** InterruptedException

{

System.***out***.println("\n" +"Req\_deliverables.actionunassigned()");

View.click();

Thread.*sleep*(3000);

unassignlink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

opendeliverables();

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=12) && (countelements.size()!=13))

{

System.***out***.println("INCORRECT action buttons displaying for unassigned request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for unassigned request");

}

assign\_analyst\_details1();

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

filltimesheet\_details1();

Editrequest\_details1();

holdreq\_details1();

cancelreq\_details1();

requesthistory\_details1();

recurring\_details1();

setaslarge1();

req\_comments1();

deadline\_TBD1();

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=9)

{

System.***out***.println("INCORRECT action buttons displaying for unassigned request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for unassigned request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

Editrequest\_details1();

requesthistory\_details1();

req\_comments1();

holdreq\_details1();

deadline\_TBD1();

cancelreq\_details1();

**break**;

**case** "Analyst Demo2":

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for unassigned request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for unassigned request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

filltimesheet\_details1();

Editrequest\_details1();

requesthistory\_details1();

req\_comments1();

assign\_analyst\_details1();

**break**;

}

}

**else**

{

System.***out***.println("No unassigned request available" + "\n");

}

}

**public** **void** actioninprogress() **throws** InterruptedException

{

System.***out***.println("\n" +"Req\_deliverables.actioninprogress()");

View.click();

Thread.*sleep*(3000);

inprogresslink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

opendeliverables();

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=12) && (countelements.size()!=13))

{

System.***out***.println("INCORRECT action buttons displaying for inprogress request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for inprogress request");

}

assign\_analyst\_details1();

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

filltimesheet\_details1();

Editrequest\_details1();

holdreq\_details1();

cancelreq\_details1();

requesthistory\_details1();

recurring\_details1();

setaslarge1();

req\_comments1();

deadline\_TBD1();

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

System.***out***.println("Element count is" +countelements.size());

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for inprogress request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for inprogress request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

Editrequest\_details1();

holdreq\_details1();

requesthistory\_details1();

req\_comments1();

deadline\_TBD1();

**break**;

**case** "Analyst Demo2":

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for inprogress request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for inprogress request");

}

assign\_analyst\_details1();

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

filltimesheet\_details1();

Editrequest\_details1();

requesthistory\_details1();

req\_comments1();

**break**;

}

}

**else**

{

System.***out***.println("No inprogress request available" + "\n");

}

}

**public** **void** actiononhold() **throws** InterruptedException

{

System.***out***.println("\n" + "Req\_deliverables.actiononhold()");

View.click();

Thread.*sleep*(3000);

onholdlink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

opendeliverables();

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=9) && (countelements.size()!=10))

{

System.***out***.println("INCORRECT action buttons displaying for onhold request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for onhold request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

filltimesheet\_details1();

Editrequest\_details1();

cancelreq\_details1();

requesthistory\_details1();

setaslarge1();

req\_comments1();

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for onhold request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for onhold request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

Editrequest\_details1();

requesthistory\_details1();

req\_comments1();

cancelreq\_details1();

unholdreq\_details1();

**break**;

**case** "Analyst Demo2":

**if**(countelements.size()!=7)

{

System.***out***.println("INCORRECT action buttons displaying for onhold request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for onhold request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

filltimesheet\_details1();

Editrequest\_details1();

requesthistory\_details1();

req\_comments1();

**break**;

}

}

**else**

{

System.***out***.println("No onhold request available" + "\n");

}

}

**public** **void** actioncompleted() **throws** InterruptedException

{

System.***out***.println("\n" + "Req\_deliverables.actioncompleted()");

View.click();

Thread.*sleep*(3000);

completelink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

opendeliverables();

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=5) && (countelements.size()!=6))

{

System.***out***.println("INCORRECT action buttons displaying for complete request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for complete request");

}

filltimesheet\_details1();

requesthistory\_details1();

recurring\_details1();

setaslarge1();

req\_comments1();

reactivatereq1();

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=4)

{

System.***out***.println("INCORRECT action buttons displaying for complete request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for complete request");

}

requesthistory\_details1();

req\_comments1();

fill\_feedback1();

reactivatereq1();

**break**;

**case** "Analyst Demo2" :

**if**(countelements.size()!=4)

{

System.***out***.println("INCORRECT action buttons displaying for complete request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for complete request");

}

requesthistory\_details1();

req\_comments1();

filltimesheet\_details1();

reactivatereq1();

Thread.*sleep*(1000);

}

}

**else**

{

System.***out***.println("No complete request available" + "\n");

}

}

**public** **void** actionclosed() **throws** InterruptedException

{

System.***out***.println("\n" + "Req\_deliverables.actionclosed()");

View.click();

Thread.*sleep*(3000);

closelink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

opendeliverables();

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**(countelements.size()!=5)

{

System.***out***.println("INCORRECT action buttons displaying for closed request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for closed request");

}

filltimesheet\_details1();

requesthistory\_details1();

recurring\_details1();

req\_comments1();

view\_feedback1();

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

System.***out***.println("Element count is" +countelements.size());

**if**(countelements.size()!=3)

{

System.***out***.println("INCORRECT action buttons displaying for closed request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for closed request");

}

requesthistory\_details1();

req\_comments1();

view\_feedback1();

**break**;

**case** "Analyst Demo2" :

**if**(countelements.size()!=4)

{

System.***out***.println("INCORRECT action buttons displaying for closed request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for closed request");

}

requesthistory\_details1();

req\_comments1();

filltimesheet\_details1();

view\_feedback1();

Thread.*sleep*(1000);

**break**;

}

}

**else**

{

System.***out***.println("No closed request available" + "\n");

}

}

**public** **void** actionTBD() **throws** InterruptedException

{

System.***out***.println("\n" + "Req\_deliverables.actionTBD()");

View.click();

Thread.*sleep*(3000);

TBDlink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

opendeliverables();

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=10) && (countelements.size()!=9))

{

System.***out***.println("INCORRECT action buttons displaying for deadline TBD request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for deadline TBD request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

filltimesheet\_details1();

Editrequest\_details1();

cancelreq\_details1();

requesthistory\_details1();

setaslarge1();

req\_comments1();

deadline\_confirmed1();

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for deadline TBD request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for deadline TBD request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

Editrequest\_details1();

cancelreq\_details1();

requesthistory\_details1();

req\_comments1();

deadline\_confirmed1();

**break**;

**case** "Analyst Demo2" :

**if**(countelements.size()!=7)

{

System.***out***.println("INCORRECT action buttons displaying for deadline TBD request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for deadline TBD request");

}

addsupporting\_details1();

editdead\_details1();

editcoverage\_details1();

Editrequest\_details1();

requesthistory\_details1();

req\_comments1();

filltimesheet\_details1();

}

}

**else**

{

System.***out***.println("No TBD request available" + "\n");

}

}

**public** **void** actioncancelled() **throws** InterruptedException

{

System.***out***.println("\n" + "Req\_deliverables.actioncancelled()");

View.click();

Thread.*sleep*(3000);

cancellink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

opendeliverables();

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**(countelements.size()!=3)

{

System.***out***.println("INCORRECT action buttons displaying for cancelled request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for cancelled request");

}

filltimesheet\_details1();

requesthistory\_details1();

req\_comments1();

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=2)

{

System.***out***.println("INCORRECT action buttons displaying for cancelled request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for cancelled request");

}

requesthistory\_details1();

req\_comments1();

**break**;

**case** "Analyst Demo2" :

**if**(countelements.size()!=3)

{

System.***out***.println("INCORRECT action buttons displaying for cancelled request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for cancelled request");

}

requesthistory\_details1();

req\_comments1();

filltimesheet\_details1();

}

}

**else**

{

System.***out***.println("No cancelled request available" + "\n");

}

}

}

**Request\_details.java**

**package** rmspackage;

**import** java.awt.Robot;

**import** java.awt.Toolkit;

**import** java.awt.datatransfer.StringSelection;

**import** java.awt.event.KeyEvent;

**import** java.util.List;

**import** java.util.concurrent.TimeUnit;

**import** rmspackage.Req\_deliverables;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.Dimension;

**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;

**import** org.testng.Assert;

**public** **class** Request\_details {

WebDriver driver;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl00\_lnkButtonRequestStatus")

WebElement unassignlink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl01\_lnkButtonRequestStatus")

WebElement inprogresslink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl02\_lnkButtonRequestStatus")

WebElement onholdlink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl03\_lnkButtonRequestStatus")

WebElement cancellink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl04\_lnkButtonRequestStatus")

WebElement completelink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl05\_lnkButtonRequestStatus")

WebElement closelink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl06\_lnkButtonRequestStatus")

WebElement TBDlink;

@FindBy(linkText="View requests")

WebElement View;

/\* @FindBy(id="lnkFilterpane")

WebElement Filter;

@FindBy(id="ctl00\_SPWebPartManager1\_RequestFilterProvider\_rptrFilters\_ctl02\_lst\_\_Arrow")

WebElement Status;

@FindAll({@FindBy(className="rcbCheckBox")})

List<WebElement> checkstatus;

@FindBy(id="btnFilter")

WebElement Apply;\*/

@FindBy(id="lnkAssign")

WebElement Assignreq;

@FindBy(xpath="//input[contains(@id,'\_ddlAnalysts\_Input')]")

// @FindBy(id="ctl00\_SPWebPartManager1\_g\_4de00c5e\_f4bc\_4ae2\_84a9\_f8fccf0ab154\_ddlAnalysts\_Input")

WebElement Team;

@FindBy(id="btnAssignAnalyst")

WebElement submitanalyst;

@FindBy(id="lnkAddSupportingDocument")

WebElement Addsupport;

@FindBy(className="ms-dlgFrame")

WebElement supportframe;

@FindAll({@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_ddlRecordsPerPage")})

List<WebElement> requestexist;

//@FindBy(id="DlgFrameed9443c0-f070-4e62-8b0f-c98aefc4cda8")

// WebElement supportframe;

@FindBy(id="txtRequestDocComments")

WebElement supportcomment;

@FindBy(xpath="//input[contains(@id,'\_radUpSupportingDocumentfile0')]")

// @FindBy(id="ctl00\_SPWebPartManager1\_g\_5499d7fe\_762c\_4d81\_a2b7\_da2102ea58fb\_SupportingDocument\_radUpSupportingDocumentfile0")

WebElement supportupload;

@FindBy(id="btnSupportingDocument")

WebElement supportsubmit;

@FindBy (id="lnkEditRequestDeadline")

WebElement reqdeadline;

@FindBy(id="dateRequestDeadline")

WebElement enterdate;

@FindBy(className="ms-dlgFrame")

WebElement framedeadline;

@FindBy(xpath=".//\*[@id='divEditRequestDeadline']/table/tbody/tr[1]/td[2]/table/tbody/tr/td/div[1]/div/div[1]/div[2]/table/tbody/tr[5]/td[7]/div")

WebElement dateframe;

@FindBy(id="timeRequestDeadline")

WebElement entertime;

@FindBy(xpath="//textarea[contains(@id,'\_txtComment')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_8bc97765\_7a09\_45e1\_a126\_f1bcd8da8c8b\_txtComment")

WebElement editdesc;

@FindBy(id="btnSaveRequestDeadline")

WebElement submitdeadline;

@FindBy(id="lnkEditCoverageGroup")

WebElement addcov;

@FindBy(xpath="//a[contains(@id,'\_ddlMemberFirmKey\_Arrow')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_1fe24510\_7104\_4c9e\_82d2\_34e2a18e851e\_ddlMemberFirmKey\_Arrow")

WebElement covarrow;

/\*@FindBy(xpath="//textarea[contains(@id,'\_txtComment')]")

@FindBy(id="ctl00\_SPWebPartManager1\_g\_1fe24510\_7104\_4c9e\_82d2\_34e2a18e851e\_txtComment")

WebElement covcomment;\*/

@FindBy(id="btnSaveRequestCoverageGroup")

WebElement covsubmit;

@FindBy(id="lnkFillTimesheet")

WebElement timesheet;

@FindBy(id="lnkEdit")

WebElement edit;

@FindBy(xpath="//a[contains(@id,'\_ddlRequestPriority\_Arrow')]")

// @FindBy(id="ctl00\_SPWebPartManager1\_g\_68cb04f8\_161f\_4e9a\_8842\_a39c56f735c7\_ddlRequestPriority\_Arrow")

WebElement priority1;

@FindAll({@FindBy(className="rcbItem")})

List<WebElement> priclass1;

@FindBy(id="btnReq")

WebElement updatereq;

@FindBy(id="lnkHold")

WebElement holdclick;

@FindBy(xpath="//textarea[contains(@id,'\_txtHoldComment')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_9674e814\_c15f\_45e2\_8a18\_0adc8103db7c\_txtHoldComment")

WebElement holdcomment;

@FindBy(id="btnRequestHold")

WebElement holdsubmit;

@FindBy(id="lnkUnHold")

WebElement unholdclick;

@FindBy(xpath="//textarea[contains(@id,'\_txtUnholdComment')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_32e85aea\_6b75\_490a\_a9d5\_f5c0a88885c2\_txtUnholdComment")

WebElement unholdcomment;

@FindBy(id="btnRequestUnhold")

WebElement unholdbtn;

@FindBy(id="lnkCancelRequest")

WebElement cancel;

/\*@FindBy(id="ctl00\_SPWebPartManager1\_g\_9674e814\_c15f\_45e2\_8a18\_0adc8103db7c\_txtHoldComment")

WebElement cancel\_comment;\*/

@FindBy(id="btnRequestHold")

WebElement clickcancel;

@FindBy(id="lnkVersionHistory")

WebElement history;

@FindBy(id="lnkRecurringRequest")

WebElement recurring;

@FindBy(id="startDatePicker")

WebElement startdt;

@FindBy(xpath=".//\*[@id='RecurringRequest']/table/tbody/tr[1]/td[2]/div/div/div[1]/div[2]/table/tbody/tr[5]/td[2]/div")

WebElement choosestart;

@FindBy(id="endDatePicker")

WebElement enddt;

@FindBy(xpath=".//\*[@id='RecurringRequest']/table/tbody/tr[2]/td[2]/div/div/div[1]/div[2]/table/tbody/tr[5]/td[5]/div")

WebElement chooseend;

@FindBy(id="ddlFrequency")

WebElement Frequency;

@FindBy(id="txtDeadline")

WebElement recurdeadline;

@FindBy(id="btnRecurringRequest")

WebElement recursubmit;

@FindBy(id="lnkSetAsLargeRequest")

WebElement largereq;

@FindBy(xpath="//input[contains(@id,'\_txtRequestor')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_d831e079\_5257\_47cc\_9264\_3d0abf87348c\_txtRequestor")

WebElement Emailto;

@FindBy(id="btnSaveLargeRequest")

WebElement Largesubmit;

@FindBy(id="lnkRequestComments")

WebElement reqcomment;

@FindBy(className="ms-dlgCloseBtn")

WebElement closecomment;

@FindBy(id="lnkPostponeRequest")

WebElement TBD;

@FindBy(id="dateRequestDeadline")

WebElement deadlineTBD;

@FindBy(id="timeRequestDeadline")

WebElement timeTBD;

/\*@FindBy(id="ctl00\_SPWebPartManager1\_g\_6c19edff\_a865\_4bd0\_a803\_d28d563a62e0\_txtComment")

WebElement TBDcomment;\*/

@FindBy(id="btnPostponeRequest")

WebElement SubmitTBD;

@FindBy(id="lnkResumeRequest")

WebElement confimedlink;

@FindBy(id="lnkReactivateRequest")

WebElement reactivatelink;

@FindBy(id="dateProposedDeliveryDateTime")

WebElement reactivatedate;

@FindBy(id="timeProposedDeliveryDateTime")

WebElement reactivatetime;

@FindBy(id="btnReactivateRequest")

WebElement reactivatebtn;

@FindBy(id="lnkViewFeedback")

WebElement viewfeedlink;

@FindBy(id="lnkFillFeedback")

WebElement fillfeed;

@FindBy(xpath=".//\*[@id='innerTitleHeader']/div[1]/h1")

WebElement Filltitle;

@FindAll({@FindBy(xpath="//a[starts-with(@class,'icon')][contains(@class,'toolTipClass')]")})

//@FindAll({@FindBy(xpath="//a[contains(@class,'toolTipClass')]")})

List<WebElement> countelements;

String logname;

String path="C:\\Users\\chaman.preet\\Desktop\\04 PIA Requirement V 2 3.docx";

**public** Request\_details(WebDriver driver)

{

**this**.driver=driver;

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

}

//To verify and perform action if assign analyst is present.

**public** **void** assign\_analyst\_details() **throws** InterruptedException {

String detailtitle = driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeaderRequestDetail']/div[1]/h1")).getText();

Assert.*assertEquals*(detailtitle, "Request detail");

**boolean** assignpresent=driver.findElements(By.*id*("lnkAssign")).size()>0;

**if**(assignpresent)

{

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

Assignreq.click();

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

//driver.findElement(By.className("racTextToken")).sendKeys("Kamaljeet Verma");

driver.switchTo().frame(framedeadline);

Thread.*sleep*(2000);

//driver.findElement(By.className("racTextToken")).clear();

Thread.*sleep*(2000);

Team.sendKeys("tea");

Thread.*sleep*(1000);

driver.findElement(By.*className*("racList")).click();

// driver.findElements(By.tagName("td")).get(1).click();

// driver.findElement(By.xpath(".//\*[@id='aspnetForm']/div[1]/div/ul/li[1]")).click();

//driver.findElement(By.cssSelector("td")).click();

submitanalyst.click();

Thread.*sleep*(1000);

Assert.*assertEquals*(detailtitle, "Request detail","Title not matching");

String req2=driver.findElement(By.*xpath*("//span[contains(@id,'\_rptrPrimaryDetail\_ctl02\_ValueLabel1')]")).getText();

System.***out***.println("Status of request is " +req2);

Assert.*assertEquals*(req2, "In-progress");

}

**else**

{

System.***out***.println("Assign analyst action button is not present");

}

}

//To verify if assign analyst is present.

**public** **void** assign\_analyst\_details2() **throws** InterruptedException {

String detailtitle = driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeaderRequestDetail']/div[1]/h1")).getText();

Assert.*assertEquals*(detailtitle, "Request detail");

**boolean** assignpresent=driver.findElements(By.*id*("lnkAssign")).size()>0;

**if**(assignpresent)

{

//System.out.println("Assign analyst action button is PRESENT");

}

**else**

{

System.***out***.println("Assign analyst action button is not present");

}

}

// To verify and perform action if add supporting document is present.

**public** **void** addsupporting\_details() **throws** InterruptedException

{

String detailtitle = driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeaderRequestDetail']/div[1]/h1")).getText();

Assert.*assertEquals*(detailtitle, "Request detail");

**boolean** addpresent=driver.findElements(By.*id*("lnkAddSupportingDocument")).size()>0;

**if**(addpresent)

{

Addsupport.click();

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

driver.switchTo().frame(supportframe);

supportcomment.sendKeys("This is to add support document");

supportupload.click();

StringSelection ss=**new** StringSelection(path);

Toolkit.*getDefaultToolkit*().getSystemClipboard().setContents(ss, **null**);

upload2();

Thread.*sleep*(1000);

supportsubmit.click();

driver.switchTo().defaultContent();

Thread.*sleep*(2000);

}

**else**

{

System.***out***.println("Add supporting document action button is not present.");

}

}

**public** **void** upload2()

{

**try**

{

Robot rr=**new** Robot();

rr.keyPress(KeyEvent.***VK\_CONTROL***);

rr.keyPress(KeyEvent.***VK\_V***);

rr.keyRelease(KeyEvent.***VK\_CONTROL***);

rr.keyRelease(KeyEvent.***VK\_V***);

rr.delay(2000);

rr.keyPress(KeyEvent.***VK\_ENTER***);

rr.keyRelease(KeyEvent.***VK\_ENTER***);

rr.delay(1000);

}

**catch**(Exception e)

{

e.printStackTrace();

}

}

// To verify and perform action if edit deadline is present.

**public** **void** editdead\_details() **throws** InterruptedException

{

**boolean** editpresent=driver.findElements(By.*id*("lnkEditRequestDeadline")).size()>0;

**if**(editpresent)

//if(driver.findElement(By.xpath("//a[contains(@href,'javascript:void()')]")).isDisplayed())

{

**try** {

Thread.*sleep*(3000);

} **catch** (InterruptedException e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

}

reqdeadline.click();

driver.manage().timeouts().implicitlyWait(35,TimeUnit.***SECONDS***);

driver.switchTo().frame(framedeadline);

enterdate.click();

Thread.*sleep*(2000);

driver.findElement(By.*className*("xdsoft\_today\_button")).click();

Thread.*sleep*(2000);

dateframe.click();

entertime.sendKeys("05:00");

editdesc.sendKeys("edit deadline");

submitdeadline.click();

}

**else**{

System.***out***.println("Edit deadline is not present");

}

}

//To verify and perform action if edit request coverage is present.

**public** **void** editcoverage\_details()

{

**boolean** covpresent=driver.findElements(By.*id*("lnkEditCoverageGroup")).size()>0;

**if**(covpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

addcov.click();

**try** {

Thread.*sleep*(2000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

driver.switchTo().frame(framedeadline);

covarrow.click();

driver.findElement(By.*tagName*("li")).click();

editdesc.sendKeys("Edit request coverage");

covsubmit.click();

}

**else**

{

System.***out***.println("Edit coverage is not present");

}

}

//To verify and perform action if fill timesheet is present.

**public** **void** filltimesheet\_details() **throws** InterruptedException

{

**boolean** timesheetpresent=driver.findElements(By.*id*("lnkFillTimesheet")).size()>0;

**if**(timesheetpresent)

{

Thread.*sleep*(1000);

timesheet.click();

String heading = driver.findElement(By.*id*("innerTitleHeader")).getText();

Assert.*assertEquals*(heading, "Fill timesheet");

View.click();

Thread.*sleep*(2000);

}

**else**{

System.***out***.println("Fill timesheet is not present");

}

}

//To verify and perform action if Edit request is present.

**public** **void** Editrequest\_details()

{

**boolean** requestpresent=driver.findElements(By.*id*("lnkEdit")).size()>0;

**if**(requestpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

edit.click();

WebDriverWait wait = **new** WebDriverWait(driver, 10);

wait.until(ExpectedConditions.*presenceOfElementLocated*(By.*xpath*("//input[contains(@id,'\_txtClientProjectCode')]")));

String newrequest = driver.findElement(By.*id*("innerTitleHeader")).getText();

Assert.*assertEquals*(newrequest, "New request");

priority1.click();

wait.until(ExpectedConditions.*visibilityOfElementLocated*(By.*className*("rcbItem")));

priclass1.get(2).click();

updatereq.click();

}

**else**

{

System.***out***.println("Edit request action button is not present");

}

}

//To verify and perform action if hold request is present.

**public** **void** holdreq\_details()

{

**boolean** holdpresent=driver.findElements(By.*id*("lnkHold")).size()>0;

**if**(holdpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

holdclick.click();

driver.manage().timeouts().implicitlyWait(15, TimeUnit.***SECONDS***);

driver.switchTo().frame(framedeadline);

holdcomment.sendKeys("Request is put on hold");

holdsubmit.click();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

String req=driver.findElement(By.*xpath*("//span[contains(@id,'\_rptrPrimaryDetail\_ctl02\_ValueLabel1')]")).getText();

System.***out***.println("\n" + "Status of request is " +req);

Assert.*assertEquals*(req, "On hold");

}

**else**

{

System.***out***.println("Hold action button is not available");

}

}

//To verify and perform action if unhold request is present

**public** **void** unholdreq\_details()

{

**boolean** unholdpresent=driver.findElements(By.*id*("lnkUnHold")).size()>0;

**if**(unholdpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

unholdclick.click();

driver.switchTo().frame(framedeadline);

driver.manage().timeouts().implicitlyWait(15, TimeUnit.***SECONDS***);

unholdcomment.sendKeys("Unhold request");

unholdbtn.click();

String req=driver.findElement(By.*xpath*("//span[contains(@id,'\_rptrPrimaryDetail\_ctl02\_ValueLabel1')]")).getText();

System.***out***.println("\n" + "Status of request is " +req);

Assert.*assertEquals*(req, "Unassigned");

}

**else**

{

System.***out***.println("Hold action button is not available");

}

}

//To verify and perform action if cancel request is present.

**public** **void** cancelreq\_details()

{

**boolean** cancelpresent=driver.findElements(By.*id*("lnkCancelRequest")).size()>0;

**if**(cancelpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

cancel.click();

driver.manage().timeouts().implicitlyWait(5, TimeUnit.***SECONDS***);

driver.switchTo().frame(framedeadline);

holdcomment.sendKeys("this is cancelling request from details");

clickcancel.click();

String req=driver.findElement(By.*xpath*("//span[contains(@id,'\_rptrPrimaryDetail\_ctl02\_ValueLabel1')]")).getText();

System.***out***.println("\n" +"Status of request is " +req);

Assert.*assertEquals*(req, "Cancelled");

}

**else**

{

System.***out***.println("Cancel action button is not available");

}

}

//To verify and perform action if version history is present.

**public** **void** requesthistory\_details() **throws** InterruptedException

{

**boolean** historypresent=driver.findElements(By.*id*("lnkVersionHistory")).size()>0;

**if**(historypresent)

{

Thread.*sleep*(1000);

history.click();

Thread.*sleep*(1000);

String version=driver.findElement(By.*xpath*(".//\*[@id='dialogTitleSpan']")).getAttribute("title");

Assert.*assertEquals*(version, "Version History", "Title is not matching");

Thread.*sleep*(2000);

driver.findElement(By.*xpath*("//a[contains(@id,'DlgClose')]")).click();

Thread.*sleep*(1000);

}

**else**

{

System.***out***.println("Version history is not present");

}

}

//To verify and perform action if recurring request is present.

**public** **void** recurring\_details() **throws** InterruptedException

{

**boolean** recurpresent=driver.findElements(By.*id*("lnkRecurringRequest")).size()>0;

**if**(recurpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

recurring.click();

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

}

driver.switchTo().frame(framedeadline);

startdt.click();

Thread.*sleep*(2000);

choosestart.click();

Thread.*sleep*(1000);

enddt.click();

Thread.*sleep*(2000);

chooseend.click();

Select myselect1=**new** Select(Frequency);

myselect1.selectByValue("Daily");

recurdeadline.sendKeys("2");

recursubmit.click();

}

**else**

{

System.***out***.println("Recurring request is not present");

}

}

//To verify and perform action if set as large request is present.

**public** **void** setaslarge()

{

// if(largereq.isDisplayed())

**boolean** largepresent = driver.findElements(By.*id*("lnkSetAsLargeRequest")).size()>0;

**if**(largepresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

largereq.click();

driver.switchTo().frame(framedeadline);

Emailto.clear();

Emailto.sendKeys("chaman.preet@evalueserve.com");

Largesubmit.click();

}

**else**

{

System.***out***.println("Set as large request is not present");

}

}

//To verify and perform action if request comments is present.

**public** **void** req\_comments()

{

**boolean** commentpresent=driver.findElements(By.*id*("lnkRequestComments")).size()>0;

**if**(commentpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

reqcomment.click();

//driver.switchTo().frame(framedeadline);

driver.manage().timeouts().implicitlyWait(5, TimeUnit.***SECONDS***);

closecomment.click();

}

**else**

{

System.***out***.println("req comment is not present");

}

}

//To verify and perform action if deadlineTBD is present.

**public** **void** deadline\_TBD()

{

**boolean** TBDpresent=driver.findElements(By.*id*("lnkPostponeRequest")).size()>0;

**if**(TBDpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

TBD.click();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

driver.switchTo().frame(framedeadline);

deadlineTBD.clear();

deadlineTBD.sendKeys("20-Apr-16");

timeTBD.clear();

timeTBD.sendKeys("20:00");

editdesc.sendKeys("Add comment to change request to TBD");

SubmitTBD.click();

String req1=driver.findElement(By.*xpath*("//span[contains(@id,'\_rptrPrimaryDetail\_ctl02\_ValueLabel1')]")).getText();

System.***out***.println("\n" + "Status of request is " +req1);

Assert.*assertEquals*(req1, "Deadline TBD");

}

**else**

{

System.***out***.println("deadline TBD is not present");

}

}

//To verify and perform action if deadline confirmed is present.

**public** **void** deadline\_confirmed()

{

**boolean** confirmpresent=driver.findElements(By.*id*("lnkResumeRequest")).size()>0;

**if**(confirmpresent)

{

**try** {

Thread.*sleep*(1000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

confimedlink.click();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

driver.switchTo().frame(framedeadline);

driver.manage().timeouts().implicitlyWait(15, TimeUnit.***SECONDS***);

unholdcomment.sendKeys("Unhold request");

unholdbtn.click();

String req3=driver.findElement(By.*xpath*("//span[contains(@id,'\_rptrPrimaryDetail\_ctl02\_ValueLabel1')]")).getText();

System.***out***.println("\n" + "Status of request is " +req3);

Assert.*assertEquals*(req3, "Unassigned");

}

**else**

{

System.***out***.println("deadline confirmed is not present");

}

}

//To verify and perform action if reactivate is present.

**public** **void** reactivatereq() **throws** InterruptedException {

String detailtitle = driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeaderRequestDetail']/div[1]/h1")).getText();

Assert.*assertEquals*(detailtitle, "Request detail");

**boolean** reactivatepresent=driver.findElements(By.*id*("lnkReactivateRequest")).size()>0;

**if**(reactivatepresent)

{

reactivatelink.click();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

driver.switchTo().frame(framedeadline);

driver.manage().timeouts().implicitlyWait(15, TimeUnit.***SECONDS***);

reactivatedate.sendKeys("06/29/2016");

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

reactivatetime.sendKeys("16:15");

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

supportcomment.sendKeys("This is to reactivate request");

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

reactivatebtn.click();

String req4=driver.findElement(By.*xpath*("//span[contains(@id,'\_rptrPrimaryDetail\_ctl02\_ValueLabel1')]")).getText();

System.***out***.println("\n" + "Status of request is " +req4);

Assert.*assertEquals*(req4, "In-progress");

}

**else**

{

System.***out***.println("Reactivate action button is not present");

}

}

//To verify and perform action if view feedback is present

**public** **void** view\_feedback() **throws** InterruptedException

{

**boolean** viewpresent=driver.findElements(By.*id*("lnkViewFeedback")).size()>0;

**if**(viewpresent)

{

Thread.*sleep*(1000);

viewfeedlink.click();

String viewfeedbacktitle=driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeader']/div[1]/h1")).getText();

Assert.*assertEquals*(viewfeedbacktitle, "View feedback");

View.click();

Thread.*sleep*(2000);

}

**else**

{

System.***out***.println("View feedback button is not available");

}

}

**public** **void** fill\_feedback() **throws** InterruptedException

{

**boolean** fillpresent=driver.findElements(By.*id*("lnkFillFeedback")).size()>0;

**if**(fillpresent)

{

String windowbefore=driver.getWindowHandle();

Thread.*sleep*(1000);

fillfeed.click();

**for**(String handle:driver.getWindowHandles())

{

driver.switchTo().window(handle);

}

String Fillfeedbacktitle = Filltitle.getText();

Assert.*assertEquals*(Fillfeedbacktitle, "Fill feedback","Fillfeedback page not opened");

driver.switchTo().window(windowbefore);

Thread.*sleep*(2000);

}

**else**

{

System.***out***.println("Fill feedback button is not available");

}

}

//Perform action buttons for unassigned request

**public** **void** unassignedreq() **throws** InterruptedException

{

System.***out***.println("\n" +"\*\*\* Request\_details.unassignedreq() \*\*\*");

View.click();

Thread.*sleep*(3000);

unassignlink.click();

Thread.*sleep*(4000);

**if**(requestexist.size()>0)

{

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=12) && (countelements.size()!=13))

{

System.***out***.println("INCORRECT action buttons displaying for unassigned request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for unassigned request");

}

addsupporting\_details();

Thread.*sleep*(2000);

editdead\_details();

Thread.*sleep*(1000);

editcoverage\_details();

Thread.*sleep*(1000);

filltimesheet\_details();

Thread.*sleep*(3000);

unassignlink.click();

Thread.*sleep*(2000);

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

Editrequest\_details();

Thread.*sleep*(1000);

requesthistory\_details();

Thread.*sleep*(1000);

recurring\_details();

Thread.*sleep*(1000);

setaslarge();

Thread.*sleep*(1000);

req\_comments();

Thread.*sleep*(1000);

holdreq\_details();

Thread.*sleep*(1000);

onholdreq();

Thread.*sleep*(1000);

deadline\_TBD();

Thread.*sleep*(1000);

TBDreq();

assign\_analyst\_details();

Thread.*sleep*(1000);

inprogressreq();

cancelreq\_details();

cancelreq();

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=9)

{

System.***out***.println("INCORRECT action buttons displaying for unassigned request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for unassigned request");

}

addsupporting\_details();

Thread.*sleep*(2000);

editdead\_details();

Thread.*sleep*(1000);

editcoverage\_details();

Thread.*sleep*(1000);

Editrequest\_details();

Thread.*sleep*(1000);

requesthistory\_details();

Thread.*sleep*(1000);

req\_comments();

Thread.*sleep*(1000);

holdreq\_details();

Thread.*sleep*(1000);

onholdreq();

Thread.*sleep*(1000);

deadline\_TBD();

Thread.*sleep*(1000);

TBDreq();

cancelreq\_details();

cancelreq();

**break**;

**case** "Analyst Demo2":

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for unassigned request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for unassigned request");

}

addsupporting\_details();

Thread.*sleep*(2000);

editdead\_details();

Thread.*sleep*(1000);

editcoverage\_details();

Thread.*sleep*(1000);

filltimesheet\_details();

Thread.*sleep*(3000);

unassignlink.click();

Thread.*sleep*(2000);

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

Editrequest\_details();

Thread.*sleep*(1000);

requesthistory\_details();

Thread.*sleep*(1000);

req\_comments();

Thread.*sleep*(1000);

assign\_analyst\_details();

Thread.*sleep*(1000);

inprogressreq();

**break**;

}

}

**else**

{

System.***out***.println("No unassigned request available" + "\n");

}

}

//Perform action buttons for inprogress request

**public** **void** inprogressreq() **throws** InterruptedException

{

System.***out***.println("\n" +"\*\*\* Request\_details.inprogressreq() \*\*\*");

Req\_deliverables objdel=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=12) && (countelements.size()!=13))

{

System.***out***.println("INCORRECT action buttons displaying for inprogress request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for inprogress request");

}

assign\_analyst\_details2();

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.filltimesheet\_details1();

objdel.Editrequest\_details1();

objdel.holdreq\_details1();

objdel.cancelreq\_details1();

objdel.requesthistory\_details1();

objdel.recurring\_details1();

objdel.setaslarge1();

objdel.req\_comments1();

objdel.deadline\_TBD1();

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

View.click();

Thread.*sleep*(3000);

inprogresslink.click();

Thread.*sleep*(3000);

**if**(requestexist.size()>0)

{

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

System.***out***.println("Element count is " +countelements.size());

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for inprogress request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for inprogress request");

}

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.Editrequest\_details1();

objdel.holdreq\_details1();

objdel.requesthistory\_details1();

objdel.req\_comments1();

objdel.deadline\_TBD1();

}

**else**

{

System.***out***.println("No inprogress request available" + "\n");

}

**break**;

**case** "Analyst Demo2":

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for inprogress request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for inprogress request");

}

assign\_analyst\_details2();

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.filltimesheet\_details1();

objdel.Editrequest\_details1();

objdel.requesthistory\_details1();

objdel.req\_comments1();

**break**;

}

}

//Perform action buttons for onhold request

**public** **void** onholdreq() **throws** InterruptedException

{

System.***out***.println("\n" +"\*\*\* Request\_details.onholdreq() \*\*\*");

Req\_deliverables objdel=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=9) && (countelements.size()!=10))

{

System.***out***.println("INCORRECT action buttons displaying for onhold request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for onhold request");

}

assign\_analyst\_details2();

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.filltimesheet\_details1();

objdel.Editrequest\_details1();

objdel.cancelreq\_details1();

objdel.requesthistory\_details1();

objdel.setaslarge1();

objdel.req\_comments1();

unholdreq\_details();

Thread.*sleep*(2000);

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for onhold request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for onhold request");

}

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.Editrequest\_details1();

objdel.requesthistory\_details1();

objdel.req\_comments1();

objdel.cancelreq\_details1();

unholdreq\_details();

Thread.*sleep*(2000);

**break**;

**case** "Analyst Demo2":

View.click();

Thread.*sleep*(3000);

onholdlink.click();

Thread.*sleep*(3000);

**if**(requestexist.size()>0)

{

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

**if**(countelements.size()!=7)

{

System.***out***.println("INCORRECT action buttons displaying for onhold request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for onhold request");

}

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.filltimesheet\_details1();

objdel.Editrequest\_details1();

objdel.requesthistory\_details1();

objdel.req\_comments1();

}

**else**

{

System.***out***.println("No onhold request available" + "\n");

}

**break**;

}

}

//Perform action buttons for completed request

**public** **void** completedreq() **throws** InterruptedException

{

System.***out***.println("\n" +"\*\*\* Request\_details.completedreq() \*\*\*");

Req\_deliverables objdel=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

View.click();

Thread.*sleep*(3000);

completelink.click();

Thread.*sleep*(3000);

**if**(requestexist.size()>0)

{

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=5) && (countelements.size()!=6))

{

System.***out***.println("INCORRECT action buttons displaying for complete request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for complete request");

}

objdel.filltimesheet\_details1();

objdel.requesthistory\_details1();

objdel.req\_comments1();

objdel.recurring\_details1();

objdel.setaslarge1();

reactivatereq();

Thread.*sleep*(1000);

//System.out.println("count is " +countele);

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=4)

{

System.***out***.println("INCORRECT action buttons displaying for complete request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for complete request");

}

objdel.requesthistory\_details1();

objdel.req\_comments1();

fill\_feedback();

reactivatereq();

Thread.*sleep*(1000);

**break**;

**case** "Analyst Demo2" :

**if**(countelements.size()!=4)

{

System.***out***.println("INCORRECT action buttons displaying for complete request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for complete request");

}

objdel.requesthistory\_details1();

objdel.req\_comments1();

objdel.filltimesheet\_details1();

reactivatereq();

Thread.*sleep*(1000);

}

}

**else**

{

System.***out***.println("No completed request available" + "\n");

}

}

//Perform action buttons for closed request

**public** **void** closedreq() **throws** InterruptedException

{

System.***out***.println("\n" +"\*\*\* Request\_details.closedreq() \*\*\*");

View.click();

Thread.*sleep*(3000);

closelink.click();

Thread.*sleep*(3000);

Req\_deliverables objdel=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

**if**(requestexist.size()>0)

{

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**(countelements.size()!=5)

{

System.***out***.println("INCORRECT action buttons displaying for closed request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for closed request");

}

objdel.filltimesheet\_details1();

objdel.requesthistory\_details1();

objdel.recurring\_details1();

objdel.req\_comments1();

view\_feedback();

Thread.*sleep*(1000);

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=3)

{

System.***out***.println("INCORRECT action buttons displaying for closed request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for closed request");

}

objdel.requesthistory\_details1();

objdel.req\_comments1();

view\_feedback();

Thread.*sleep*(1000);

inprogressreq();

**break**;

**case** "Analyst Demo2" :

**if**(countelements.size()!=4)

{

System.***out***.println("INCORRECT action buttons displaying for closed request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for closed request");

}

objdel.requesthistory\_details1();

objdel.req\_comments1();

objdel.filltimesheet\_details1();

view\_feedback();

Thread.*sleep*(1000);

onholdreq();

cancelreq();

TBDreq();

**break**;

}

}

**else**

{

System.***out***.println("No closed request available" + "\n");

}

}

//Perform action buttons for cancelled request

**public** **void** cancelreq() **throws** InterruptedException

{

System.***out***.println("\n" +"\*\*\* Request\_details.cancelreq() \*\*\*");

Req\_deliverables objdel=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**(countelements.size()!=3)

{

System.***out***.println("INCORRECT action buttons displaying for cancelled request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for cancelled request");

}

objdel.filltimesheet\_details1();

objdel.requesthistory\_details1();

objdel.req\_comments1();

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=2)

{

System.***out***.println("INCORRECT action buttons displaying for cancelled request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for cancelled request");

}

objdel.requesthistory\_details1();

objdel.req\_comments1();

**break**;

**case** "Analyst Demo2" :

View.click();

Thread.*sleep*(3000);

cancellink.click();

Thread.*sleep*(3000);

**if**(requestexist.size()>0)

{

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

**if**(countelements.size()!=3)

{

System.***out***.println("INCORRECT action buttons displaying for cancelled request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for cancelled request");

}

objdel.requesthistory\_details1();

objdel.req\_comments1();

objdel.filltimesheet\_details1();

}

**else**

{

System.***out***.println("No cancelled request available" + "\n");

}

**break**;

}

}

//Perform action buttons for TBD request

**public** **void** TBDreq() **throws** InterruptedException

{

System.***out***.println("\*\*\* Request\_details.TBDreq() \*\*\*");

Req\_deliverables objdel=PageFactory.*initElements*(driver, Req\_deliverables.**class**);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

**if**((countelements.size()!=10) && (countelements.size()!=9))

{

System.***out***.println("INCORRECT action buttons displaying for deadline TBD request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for deadline TBD request");

}

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.filltimesheet\_details1();

objdel.Editrequest\_details1();

objdel.cancelreq\_details1();

objdel.requesthistory\_details1();

objdel.setaslarge1();

objdel.req\_comments1();

objdel.deadline\_confirmed1();

deadline\_confirmed();

//Thread.sleep(2000);

**break**;

**case** "Gatekeeper Demo2": **case** "Requestor Demo2":

**if**(countelements.size()!=8)

{

System.***out***.println("INCORRECT action buttons displaying for deadline TBD request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for deadline TBD request");

}

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.Editrequest\_details1();

objdel.cancelreq\_details1();

objdel.requesthistory\_details1();

objdel.req\_comments1();

objdel.deadline\_confirmed1();

deadline\_confirmed();

**break**;

**case** "Analyst Demo2" :

View.click();

Thread.*sleep*(3000);

TBDlink.click();

Thread.*sleep*(3000);

**if**(requestexist.size()>0)

{

driver.findElement(By.*xpath*(".//\*[@id='lblRequestTitle']")).click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

**if**(countelements.size()!=7)

{

System.***out***.println("INCORRECT action buttons displaying for deadline TBD request");

}

**else**

{

System.***out***.println("Correct action buttons displaying for deadline TBD request");

}

objdel.addsupporting\_details1();

objdel.editdead\_details1();

objdel.editcoverage\_details1();

objdel.Editrequest\_details1();

objdel.requesthistory\_details1();

objdel.req\_comments1();

objdel.filltimesheet\_details1();

}

**else**

{

System.***out***.println("No deadline TBD request available" + "\n");

}

**break**;

}

}

}

**View\_request.java**

**package** rmspackage;

**import** java.awt.Toolkit;

**import** java.awt.datatransfer.StringSelection;

**import** java.util.List;

**import** java.util.concurrent.TimeUnit;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.interactions.Actions;

**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.WebDriverWait;

**import** org.testng.Assert;

**import** rmspackage.Request\_details;

**public** **class** View\_request {

WebDriver driver;

@FindBy(linkText="View requests")

WebElement View;

/\* @FindBy(id="lnkFilterpane")

WebElement Filter;

@FindBy(id="ctl00\_SPWebPartManager1\_RequestFilterProvider\_rptrFilters\_ctl02\_lst\_\_Arrow")

WebElement Status;

@FindAll({@FindBy(className="rcbCheckBox")})

List<WebElement> checkstatus;

@FindBy(id="btnFilter")

WebElement Apply; \*/

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl00\_lnkButtonRequestStatus")

WebElement unassignlink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl01\_lnkButtonRequestStatus")

WebElement inprogresslink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl02\_lnkButtonRequestStatus")

WebElement onholdlink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl03\_lnkButtonRequestStatus")

WebElement cancellink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl04\_lnkButtonRequestStatus")

WebElement completelink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl05\_lnkButtonRequestStatus")

WebElement closelink;

@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_rptrRequestStatus\_ctl06\_lnkButtonRequestStatus")

WebElement TBDlink;

@FindAll({@FindBy(id="ctl00\_SPWebPartManager1\_ViewRequestsConsumer\_ddlRecordsPerPage")})

List<WebElement> requestexist;

@FindBy(id="lnkRequestComments")

WebElement reqcomment;

@FindBy(className="ms-dlgCloseBtn")

WebElement closecomment;

@FindBy (id="lnkEditRequestDeadline")

WebElement reqdeadline;

@FindBy(id="dateRequestDeadline")

WebElement enterdate;

@FindBy(className="ms-dlgFrame")

WebElement framedeadline;

@FindBy(xpath="//\*[@id='divEditRequestDeadline']/table/tbody/tr[1]/td[2]/table/tbody/tr/td/div[1]/div/div[1]/div[2]/table/tbody/tr[5]/td[6]/div")

WebElement dateframe;

@FindBy(id="timeRequestDeadline")

WebElement entertime;

@FindBy(xpath="//textarea[contains(@id,'\_txtComment')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_8bc97765\_7a09\_45e1\_a126\_f1bcd8da8c8b\_txtComment")

WebElement editdesc;

@FindBy(id="btnSaveRequestDeadline")

WebElement submitdeadline;

@FindBy(id="lnkFillTimesheet")

WebElement timesheet;

@FindBy(id="lnkEdit")

WebElement edit;

@FindBy(xpath="//a[contains(@id,'\_ddlRequestPriority\_Arrow')]")

//@FindBy(id="ctl00\_SPWebPartManager1\_g\_68cb04f8\_161f\_4e9a\_8842\_a39c56f735c7\_ddlRequestPriority\_Arrow")

WebElement priority1;

@FindAll({@FindBy(className="rcbItem")})

List<WebElement> priclass1;

@FindBy(id="btnReq")

WebElement updatereq;

@FindBy(id="lnkCancelRequest")

WebElement cancel;

@FindBy(className="ms-dlgFrame")

//@FindBy (id="DlgFrameadfc18b5-7c60-4d65-a94b-66fcbd322515")

WebElement cancelframe;

@FindBy(xpath="//textarea[contains(@id,'\_txtHoldComment')]")

WebElement cancel\_comment;

@FindBy(id="btnRequestHold")

WebElement clickcancel;

@FindBy(id="lnkVersionHistory")

WebElement history;

@FindBy(id="lnkViewDeliverables")

WebElement viewdel;

@FindBy(id="lnkFillFeedback")

WebElement fillfeed;

@FindBy(id="lnkViewFeedback")

WebElement viewfeedlink;

@FindBy(xpath=".//\*[@id='innerTitleHeader']/div[1]/h1")

WebElement Filltitle;

@FindBy(id="lnkAssign")

WebElement Assignreq;

@FindBy(xpath="//input[contains(@id,'\_ddlAnalysts\_Input')]")

WebElement Team;

@FindBy(id="btnAssignAnalyst")

WebElement submitanalyst;

@FindBy(xpath=".//\*[@id='tblViewRequest']/tbody/tr[1]/th[10]")

WebElement Actionstitle;

WebElement getstatus;

String Status1;

String logname;

String path="C:\\Users\\chaman.preet\\Desktop\\Selenium file for upload.xlsx";

**public** View\_request(WebDriver driver)

{

**this**.driver=driver;

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

}

//Open view request page and select unassigned request by Filter

**public** **void** viewreq()

{

View.click();

}

// Verify request comment button is present and perform function if present

**public** **void** requestcomment(String abc)

{

**boolean** present1 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkRequestComments']")).size()>0;

**if**(present1)

{

System.***out***.println("Request comment not available for " +abc);

}

**else**{

reqcomment.click();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

closecomment.click();

}

}

//Verify Edit deadline button is present and perform function if present

**public** **void** Edit\_deadline(String abcd) **throws** InterruptedException

{

//String cls=driver.findElement(By.xpath("//a[contains(@href, 'javascript:void()')]")).getTagName();

//System.out.println("class is" +cls);

//WebElement disablededit = driver.findElement(By.xpath("//a[contains(@href,'javascript:void()')]"));

//System.out.println("View\_request.Edit\_deadline "+ "STATUS " +abcd);

**boolean** present = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkEditRequestDeadline']")).size()>0;

**if**(present)

//if(driver.findElement(By.xpath("//a[contains(@href,'javascript:void()')]")).isDisplayed())

{

System.***out***.println("Request edit deadline is not available for " +abcd);

}

**else**{

reqdeadline.click();

driver.manage().timeouts().implicitlyWait(10,TimeUnit.***SECONDS***);

driver.switchTo().frame(framedeadline);

enterdate.click();

Thread.*sleep*(1000);

/\*driver.findElement(By.className("xdsoft\_today\_button")).click();

Thread.sleep(2000);

dateframe.click();\*/

Actions act = **new** Actions(driver);

act.doubleClick(driver.findElement(By.*className*("xdsoft\_today\_button"))).build().perform();

entertime.sendKeys("20:00");

editdesc.sendKeys("edit deadline");

submitdeadline.click();

String str = driver.findElement(By.*className*("titleContainer")).getText();

System.***out***.println("title is " +str);

Assert.*assertEquals*(" Current requests",str);

//return;

}

}

/\*if(driver.findElement(By.xpath("//a[contains(@class,'iconEdtReqTimeline toolTipClass')]")).isDisplayed())

if(driver.findElement(By.xpath("//a[contains(@onclick,'javascript:openModalDialogWithReturnValue')]")).isDisplayed())

if(driver.findElement(By.xpath("//a[contains(@href, 'javascript:void()')]")).isDisplayed())\*/

//Verify fill timesheet button is present and perform function if present

**public** **void** filltimesheet(String fill)

{

**boolean** present3 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkFillTimesheet']")).size()>0;

**if**(present3)

{

System.***out***.println("Fill timesheet is not available for " +fill);

}

**else**

{

timesheet.click();

String heading = driver.findElement(By.*id*("innerTitleHeader")).getText();

Assert.*assertEquals*(heading, "Fill timesheet");

viewreq();

**try** {

Thread.*sleep*(2000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

/\*public void selectbackfilter()

{

clickfilter();

selectstatus();

}\*/

//Verify edit request button is present and perform function if present

**public** **void** Editrequest(String ed)

{

**boolean** present4 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkEdit']")).size()>0;

**if**(present4)

{

System.***out***.println("Edit request is not available for " +ed);

}

**else**

{

edit.click();

WebDriverWait wait = **new** WebDriverWait(driver, 10);

wait.until(ExpectedConditions.*presenceOfElementLocated*(By.*xpath*("//input[contains(@id,'\_txtClientProjectCode')]")));

String newrequest = driver.findElement(By.*id*("innerTitleHeader")).getText();

Assert.*assertEquals*(newrequest, "New request");

priority1.click();

wait.until(ExpectedConditions.*visibilityOfElementLocated*(By.*className*("rcbItem")));

priclass1.get(2).click();

updatereq.click();

}

}

//Verify cancel request button is present and perform function if present

**public** **void** cancelreq(String can) **throws** InterruptedException

{

**boolean** present5 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkCancelRequest']")).size()>0;

**if**(present5)

{

System.***out***.println("Cancel request is not available for " +can);

}

**else**

{

cancel.click();

driver.manage().timeouts().implicitlyWait(5, TimeUnit.***SECONDS***);

driver.switchTo().frame(cancelframe);

cancel\_comment.sendKeys("this is to cancel request");

clickcancel.click();

Thread.*sleep*(2000);

}

}

//Verify version history request button is present and perform function if present

**public** **void** version\_his(String his) **throws** InterruptedException

{

**boolean** present6 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkVersionHistory']")).size()>0;

**if**(present6)

{

System.***out***.println("Version history is not present for" +his);

}

**else**

{

Thread.*sleep*(1000);

history.click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

String version=driver.findElement(By.*xpath*(".//\*[@id='dialogTitleSpan']")).getAttribute("title");

//System.out.println("Dialog box opened is " +version);

Assert.*assertEquals*(version, "Version History", "Title is not matching");

Thread.*sleep*(2000);

driver.findElement(By.*xpath*("//a[contains(@id,'DlgClose')]")).click();

Thread.*sleep*(1000);

}

}

//Verify version history request button is present and perform function if present

**public** **void** view\_deliver(String del) **throws** InterruptedException

{

**boolean** present7 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkViewDeliverables']")).size()>0;

**if**(present7)

{

System.***out***.println("View Deliverables is not present for " +del);

}

**else**

{

Thread.*sleep*(1000);

viewdel.click();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

String detailtitle = driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeaderRequestDetail']/div[1]/h1")).getText();

Assert.*assertEquals*(detailtitle, "Deliverables");

Thread.*sleep*(2000);

viewreq();

Thread.*sleep*(2000);

}

}

//Verify version fill\_feedback is present and perform function if present

**public** **void** fill\_feedback(String feed) **throws** InterruptedException

{

**boolean** present7 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkFillFeedback']")).size()>0;

**if**(present7)

{

System.***out***.println("Fill feedback is not present for " +feed);

}

**else**

{

String windowbefore=driver.getWindowHandle();

Thread.*sleep*(1000);

fillfeed.click();

**for**(String handle:driver.getWindowHandles())

{

driver.switchTo().window(handle);

}

String Fillfeedbacktitle = Filltitle.getText();

Assert.*assertEquals*(Fillfeedbacktitle, "Fill feedback","Fillfeedback page not opened");

driver.switchTo().window(windowbefore);

Thread.*sleep*(2000);

}

}

//Verify version view\_feedback is present and perform function if present

**public** **void** view\_feedback(String feed1) **throws** InterruptedException

{

**boolean** present7 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkViewFeedback']")).size()>0;

**if**(present7)

{

System.***out***.println("View feedback is not present for " +feed1);

}

**else**

{

Thread.*sleep*(1000);

viewfeedlink.click();

String viewfeedbacktitle=driver.findElement(By.*xpath*(".//\*[@id='innerTitleHeader']/div[1]/h1")).getText();

Assert.*assertEquals*(viewfeedbacktitle, "View feedback");

View.click();

Thread.*sleep*(2000);

}

}

// Verify add supporting document is present and perform function if present

**public** **void** addsupporting\_detail(String support1) **throws** InterruptedException

{

{

**boolean** present7 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkAddSupportingDocument']")).size()>0;

**if**(present7)

{

System.***out***.println("View feedback is not present for " +support1);

}

**else**

{

Request\_details objdetails=PageFactory.*initElements*(driver, Request\_details.**class**);

objdetails.Addsupport.click();

Thread.*sleep*(1000);

driver.switchTo().frame(objdetails.supportframe);

objdetails.supportcomment.sendKeys("This is to add support document");

objdetails.supportupload.click();

StringSelection ss=**new** StringSelection(path);

Toolkit.*getDefaultToolkit*().getSystemClipboard().setContents(ss, **null**);

objdetails.upload2();

Thread.*sleep*(1000);

objdetails.supportsubmit.click();

driver.switchTo().defaultContent();

Thread.*sleep*(2000);

}

}

}

// Verify assign analyst is present and perform function if present

**public** **void** assign\_analyst(String assi1) **throws** InterruptedException

{

**boolean** present8 = driver.findElements(By.*xpath*("//a[@href='javascript:void()'][@id='lnkAssign']")).size()>0;

**if**(present8)

{

System.***out***.println("Assign analyst is not present for " +assi1);

}

**else**

{

Status1=driver.findElement(By.*xpath*(".//\*[@id='tblViewRequest']/tbody/tr[2]/td[1]")).getText();

**switch**(Status1)

{

**case** "In-progress":

Assignreq.click();

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

driver.switchTo().frame(framedeadline);

Thread.*sleep*(2000);

submitanalyst.click();

Thread.*sleep*(1000);

**break**;

**default**:

Assignreq.click();

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

driver.switchTo().frame(framedeadline);

Thread.*sleep*(2000);

Team.sendKeys("tea");

Thread.*sleep*(1000);

driver.findElement(By.*className*("racList")).click();

// driver.findElements(By.tagName("td")).get(1).click();

// driver.findElement(By.xpath(".//\*[@id='aspnetForm']/div[1]/div/ul/li[1]")).click();

//driver.findElement(By.cssSelector("td")).click();

submitanalyst.click();

Thread.*sleep*(1000);

**break**;

}

}

}

//Open view request page and perform actions for unassigned request

**public** **void** unassigned() **throws** InterruptedException

{

driver.manage().timeouts().implicitlyWait(3, TimeUnit.***SECONDS***);

System.***out***.println("\n" + "View\_request.unassigned()");

viewreq();

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

Status1=driver.findElement(By.*xpath*(".//\*[@id='tblViewRequest']/tbody/tr[2]/td[1]")).getText();

System.***out***.println("Status for request is " +Status1);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

requestcomment(Status1);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(1000);

filltimesheet(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Editrequest(Status1);

Thread.*sleep*(5000);

unassignlink.click();

//cancelreq(Status1);

**break**;

**case** "Gatekeeper Demo2":

version\_his(Status1);

view\_deliver(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(1000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(1000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Requestor Demo2":

view\_deliver(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(1000);

addsupporting\_detail(Status1);

Thread.*sleep*(2000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(1000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Analyst Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

assign\_analyst(Status1);

Thread.*sleep*(1000);

view\_deliver(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(1000);

filltimesheet(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(1000);

Editrequest(Status1);

Thread.*sleep*(4000);

unassignlink.click();

Thread.*sleep*(1000);

**break**;

}

}

**else**

{

System.***out***.println("No unassigned request available " + "\n");

}

}

//Open view request page and perform actions for inprogress request

**public** **void** inprogress() **throws** InterruptedException

{

System.***out***.println("\n" + "View\_request.inprogress()");

viewreq();

Thread.*sleep*(2000);

/\*Filter.click();

Status.click();

checkstatus.get(0).click();

checkstatus.get(1).click();

Apply.click();\*/

inprogresslink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

Status1=driver.findElement(By.*xpath*(".//\*[@id='tblViewRequest']/tbody/tr[2]/td[1]")).getText();

System.***out***.println("Status for request is " +Status1);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Logged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(4000);

filltimesheet(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(4000);

Editrequest(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

//cancelreq(Status1);

**break**;

**case** "Gatekeeper Demo2":

version\_his(Status1);

view\_deliver(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(4000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Requestor Demo2":

view\_deliver(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(1000);

addsupporting\_detail(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(1000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(1000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Analyst Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

assign\_analyst(Status1);

Thread.*sleep*(3000);

inprogresslink.click();

Thread.*sleep*(2000);

view\_deliver(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(1000);

filltimesheet(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(1000);

Editrequest(Status1);

Thread.*sleep*(4000);

inprogresslink.click();

Thread.*sleep*(1000);

**break**;

}

}

**else**

{

System.***out***.println("No inprogress request available " + "\n");

}

}

//Open view request page and perform actions for onhold request

**public** **void** onhold() **throws** InterruptedException

{

System.***out***.println("\n" + "View\_request.onhold()");

viewreq();

Thread.*sleep*(2000);

onholdlink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

Status1=driver.findElement(By.*xpath*(".//\*[@id='tblViewRequest']/tbody/tr[2]/td[1]")).getText();

System.***out***.println("Status for request is " +Status1);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

onholdlink.click();

Thread.*sleep*(4000);

filltimesheet(Status1);

onholdlink.click();

Thread.*sleep*(4000);

Editrequest(Status1);

Thread.*sleep*(4000);

onholdlink.click();

//cancelreq(Status1);

**break**;

**case** "Gatekeeper Demo2":

version\_his(Status1);

view\_deliver(Status1);

Thread.*sleep*(4000);

onholdlink.click();

Thread.*sleep*(1000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

onholdlink.click();

Thread.*sleep*(4000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Requestor Demo2":

view\_deliver(Status1);

Thread.*sleep*(4000);

onholdlink.click();

Thread.*sleep*(1000);

addsupporting\_detail(Status1);

Thread.*sleep*(2000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

onholdlink.click();

Thread.*sleep*(1000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Analyst Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

assign\_analyst(Status1);

Thread.*sleep*(3000);

onholdlink.click();

Thread.*sleep*(1000);

view\_deliver(Status1);

Thread.*sleep*(4000);

onholdlink.click();

Thread.*sleep*(1000);

filltimesheet(Status1);

Thread.*sleep*(4000);

onholdlink.click();

Thread.*sleep*(1000);

Editrequest(Status1);

Thread.*sleep*(4000);

onholdlink.click();

Thread.*sleep*(1000);

**break**;

}

}

**else**

{

System.***out***.println("No onhold request available " + "\n");

}

}

//Open view request page and perform actions for completed request

**public** **void** completed() **throws** InterruptedException

{

System.***out***.println("\n" + "View\_request.completed()");

viewreq();

Thread.*sleep*(2000);

completelink.click();

Thread.*sleep*(2000);

completelink.click();

**if**(requestexist.size()>0)

{

Status1=driver.findElement(By.*xpath*(".//\*[@id='tblViewRequest']/tbody/tr[2]/td[1]")).getText();

System.***out***.println("Status for request is " +Status1);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

filltimesheet(Status1);

completelink.click();

Thread.*sleep*(4000);

Editrequest(Status1);

Thread.*sleep*(4000);

//cancelreq(Status1);

**break**;

**case** "Gatekeeper Demo2":

version\_his(Status1);

view\_deliver(Status1);

Thread.*sleep*(4000);

completelink.click();

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

completelink.click();

Thread.*sleep*(4000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Requestor Demo2":

view\_deliver(Status1);

Thread.*sleep*(4000);

completelink.click();

Thread.*sleep*(1000);

addsupporting\_detail(Status1);

Thread.*sleep*(2000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

completelink.click();

Thread.*sleep*(1000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Analyst Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

assign\_analyst(Status1);

Thread.*sleep*(4000);

completelink.click();

Thread.*sleep*(1000);

view\_deliver(Status1);

Thread.*sleep*(4000);

completelink.click();

Thread.*sleep*(1000);

filltimesheet(Status1);

Thread.*sleep*(4000);

completelink.click();

Thread.*sleep*(1000);

Editrequest(Status1);

Thread.*sleep*(4000);

completelink.click();

Thread.*sleep*(1000);

**break**;

}

}

**else**

{

System.***out***.println("No completed request available " + "\n");

}

}

//Open view request page and perform actions for closed request

**public** **void** closed() **throws** InterruptedException

{

System.***out***.println("\n" + "View\_request.closed()");

viewreq();

//driver.navigate().refresh();

Thread.*sleep*(1000);

closelink.click();

Thread.*sleep*(4000);

**if**(requestexist.size()>0)

{

Status1=driver.findElement(By.*xpath*(".//\*[@id='tblViewRequest']/tbody/tr[2]/td[1]")).getText();

System.***out***.println("Status for request is " +Status1);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

closelink.click();

filltimesheet(Status1);

Thread.*sleep*(4000);

closelink.click();

Editrequest(Status1);

Thread.*sleep*(4000);

closelink.click();

//cancelreq(Status1);

**break**;

**case** "Gatekeeper Demo2":

version\_his(Status1);

view\_deliver(Status1);

Thread.*sleep*(4000);

closelink.click();

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

closelink.click();

Thread.*sleep*(4000);

view\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Requestor Demo2":

view\_deliver(Status1);

Thread.*sleep*(4000);

closelink.click();

Thread.*sleep*(1000);

addsupporting\_detail(Status1);

Thread.*sleep*(2000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

closelink.click();

Thread.*sleep*(1000);

view\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Analyst Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

assign\_analyst(Status1);

Thread.*sleep*(3000);

closelink.click();

Thread.*sleep*(1000);

view\_deliver(Status1);

Thread.*sleep*(4000);

closelink.click();

Thread.*sleep*(1000);

filltimesheet(Status1);

Thread.*sleep*(4000);

closelink.click();

Thread.*sleep*(1000);

Editrequest(Status1);

Thread.*sleep*(4000);

closelink.click();

Thread.*sleep*(1000);

**break**;

}

}

**else**

{

System.***out***.println("No closed request available " + "\n");

}

}

//open view request page and perform actions for CANCELLED request

**public** **void** cancelled() **throws** InterruptedException

{

System.***out***.println("\n" + "View\_request.cancelled()");

viewreq();

//driver.navigate().refresh();

Thread.*sleep*(1000);

cancellink.click();

Thread.*sleep*(4000);

**if**(requestexist.size()>0)

{

Status1=driver.findElement(By.*xpath*(".//\*[@id='tblViewRequest']/tbody/tr[2]/td[1]")).getText();

System.***out***.println("Status for request is " +Status1);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

cancellink.click();

filltimesheet(Status1);

Thread.*sleep*(4000);

cancellink.click();

Editrequest(Status1);

Thread.*sleep*(4000);

cancellink.click();

//cancelreq(Status1);

**break**;

**case** "Gatekeeper Demo2":

version\_his(Status1);

view\_deliver(Status1);

Thread.*sleep*(4000);

cancellink.click();

Thread.*sleep*(1000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

cancellink.click();

Thread.*sleep*(4000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Requestor Demo2":

view\_deliver(Status1);

Thread.*sleep*(4000);

cancellink.click();

Thread.*sleep*(1000);

addsupporting\_detail(Status1);

Thread.*sleep*(2000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

cancellink.click();

Thread.*sleep*(1000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Analyst Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

assign\_analyst(Status1);

Thread.*sleep*(4000);

cancellink.click();

Thread.*sleep*(1000);

view\_deliver(Status1);

Thread.*sleep*(4000);

cancellink.click();

Thread.*sleep*(1000);

filltimesheet(Status1);

Thread.*sleep*(4000);

cancellink.click();

Thread.*sleep*(1000);

Editrequest(Status1);

Thread.*sleep*(4000);

cancellink.click();

Thread.*sleep*(1000);

**break**;

}

}

**else**

{

System.***out***.println("No cancelled request available " + "\n");

}

}

//Open view request page and perform actions for Deadline TBD request

**public** **void** deadline\_TBD() **throws** InterruptedException

{

System.***out***.println("\n" + "View\_request.deadline\_TBD()");

viewreq();

Thread.*sleep*(1000);

TBDlink.click();

Thread.*sleep*(2000);

**if**(requestexist.size()>0)

{

Status1=driver.findElement(By.*xpath*(".//\*[@id='tblViewRequest']/tbody/tr[2]/td[1]")).getText();

System.***out***.println("Status for request is " +Status1);

logname=driver.findElement(By.*xpath*("//span[@class='userInfo']")).getText();

//System.out.println("Loged in user is " +logname);

**switch**(logname)

{

**case** "Teammanager Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

//Edit\_deadline(Status1);

Thread.*sleep*(4000);

TBDlink.click();

filltimesheet(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(4000);

Editrequest(Status1);

Thread.*sleep*(4000);

TBDlink.click();

//cancelreq(Status1);

**break**;

**case** "Gatekeeper Demo2":

version\_his(Status1);

view\_deliver(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(1000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(4000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Requestor Demo2":

view\_deliver(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(1000);

addsupporting\_detail(Status1);

Thread.*sleep*(2000);

// Edit\_deadline(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(1000);

fill\_feedback(Status1);

//cancelreq(Status1);

**break**;

**case** "Analyst Demo2":

requestcomment(Status1);

Thread.*sleep*(1000);

assign\_analyst(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(1000);

view\_deliver(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(1000);

filltimesheet(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(1000);

Editrequest(Status1);

Thread.*sleep*(4000);

TBDlink.click();

Thread.*sleep*(1000);

**break**;

}

}

**else**

{

System.***out***.println("No deadline TBD request available " + "\n");

}

}

}

**View\_feedback.java**

**package** rmspackage;

**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.testng.Assert;

**import** org.testng.annotations.Test;

**public** **class** Viewfeedback {

WebDriver driver;

@FindBy(linkText="Feedback")

WebElement Feedbacklink;

@FindBy(linkText="View feedback")

WebElement viewfeedlink;

@FindBy(id="ViewFeedbackDialogue")

WebElement viewfeedback;

@FindBy(xpath=".//\*[@id='innerTitleHeader']/div[1]/h1")

WebElement viewfeedtitle;

**public** Viewfeedback(WebDriver driver)

{

**this**.driver=driver;

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

}

//Open View feedback if present

**public** **void** open\_viewfeedback() **throws** InterruptedException {

Feedbacklink.click();

Thread.*sleep*(1000);

**boolean** viewfeedbackpresent = driver.findElements(By.*linkText*("View feedback")).size()>0;

**if**(viewfeedbackpresent)

{

viewfeedlink.click();

click\_feedback();

}

**else**

{

System.***out***.println("View feedback module is not present");

}

}

//Click view feedback form and verify if correct page opens.

**public** **void** click\_feedback() **throws** InterruptedException

{

**boolean** feedbackiconpresent = driver.findElements(By.*id*("ViewFeedbackDialogue")).size()>0;

**if**(feedbackiconpresent)

{

viewfeedback.click();

}

**else**

{

System.***out***.println("No feedback is available in view feedback");

}

Thread.*sleep*(2000);

String feedbacktitle=viewfeedtitle.getText();

Assert.*assertEquals*(feedbacktitle, "View feedback", "View feedback page do not opened");

}

//Perform functionality of view feedback module

**public** **void** View\_feedback1() **throws** InterruptedException

{

System.***out***.println("---\*\*\* Viewfeedback.View\_feedback1() \*\*\*---");

open\_viewfeedback();

//click\_feedback();

}

}

**View\_timesheet.java**

**package** rmspackage;

**import** java.awt.AWTException;

**import** java.awt.Robot;

**import** java.awt.event.KeyEvent;

**import** java.util.concurrent.TimeUnit;

**import** java.util.logging.Logger;

**import** org.openqa.selenium.Alert;

**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.testng.Assert;

**import** org.testng.annotations.Test;

**import** actionpackage.actionclass;

**import** rmspackage.FillTimesheet;

**public** **class** ViewTimesheet {

WebDriver driver;

**static** Logger *log* = Logger.*getLogger*(actionclass.**class**.getName());

@FindBy(linkText="Timesheet")

WebElement timesheet;

@FindBy(linkText="View timesheet")

WebElement viewlink;

@FindBy(linkText="View all timesheet")

WebElement viewalllink;

@FindBy(xpath="//a[@class='iconReqDetail toolTipClass']")

WebElement timesheet\_details;

@FindBy(xpath="//a[contains(@id,'\_lnkBtnDelete')]")

WebElement deletetimesheet;

@FindBy(className="ms-dlgFrame")

WebElement iframe;

@FindBy(xpath="//h1[@id='dialogTitleSpan']")

WebElement detailheading;

@FindBy(className="ms-dlgCloseBtn")

WebElement closedetail;

@FindBy(xpath="//span[contains(@id,'\_LabelPaging')]")

WebElement pagecount;

@FindBy(xpath="//a[contains(@id,'\_lnkBtnEdit')]")

WebElement Edittimesheet;

@FindBy(xpath="//input[contains(@id,'\_radRequestID')]")

WebElement searchbyid;

@FindBy(id="txtRequestID")

WebElement searchtext;

@FindBy(id="btnSubmit")

WebElement submitbtn;

@FindBy(xpath="//input[contains(@id,'\_txtHours')]")

WebElement edittext;

@FindBy(xpath="//a[contains(@id,'\_btnUpdate')]")

WebElement updatebtn;

@FindBy(xpath="//input[contains(@id,'\_btnReset')]")

WebElement Resetbtn;

//constructor

**public** ViewTimesheet(WebDriver driver)

{

**this**.driver=driver;

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

}

//Check If view timesheet is present or not.

**public** **void** view\_availability() **throws** InterruptedException, AWTException

{

**boolean** viewpresent = driver.findElements(By.*linkText*("View timesheet")).size()>0;

**if**(viewpresent)

{

viewlink.click();

Thread.*sleep*(1000);

*log*.info("Checking View timesheet details");

view\_details();

Thread.*sleep*(1000);

*log*.info("Checking delete timesheet");

delete();

}

**else**{

System.***out***.println("View Timesheet module is not present");

}

}

//Check If view all timesheet is present or not.

**public** **void** viewall\_availability() **throws** InterruptedException, AWTException

{

**boolean** viewallpresent = driver.findElements(By.*linkText*("View all timesheet")).size()>0;

**if**(viewallpresent)

{

Thread.*sleep*(1000);

viewalllink.click();

Thread.*sleep*(1000);

*log*.info("Checking View timesheet details");

view\_details();

Thread.*sleep*(1000);

*log*.info("Searching timesheet");

filterbyid();

*log*.info("Editing timesheet");

edit\_timesheet();

Resetbtn.click();

Thread.*sleep*(2000);

*log*.info("Checking delete timesheet");

delete();

}

**else**{

System.***out***.println("View all Timesheet module is not present");

}

}

//Click on timesheet detail and check dialog box opened

**public** **void** view\_details() **throws** InterruptedException

{

timesheet\_details.click();

// driver.switchTo().frame(iframe);

Thread.*sleep*(2000);

String detail = detailheading.getText();

Thread.*sleep*(1000);

Assert.*assertEquals*(detail, "View timesheet detail","Timesheet detail title not matching.");

Thread.*sleep*(1000);

closedetail.click();

}

//Delete timesheet

**public** **void** delete() **throws** InterruptedException, AWTException

{

String count=pagecount.getText();

String countdelete = count.substring(count.lastIndexOf("of ")+3);

System.***out***.println("before delete, count is " +countdelete);

**int** countint = Integer.*parseInt*(countdelete);

**int** afterdelete = countint-1;

String expectedcount = Integer.*toString*(afterdelete);

Thread.*sleep*(1000);

deletetimesheet.click();

Thread.*sleep*(1000);

Alert alr=driver.switchTo().alert();

alr.accept();

Thread.*sleep*(4000);

String count1=pagecount.getText();

//System.out.println("total count is " +count);

//String s[]=count.split("of ");

//String countdelete = s[s.length-1];

String countdelete1 = count1.substring(count.lastIndexOf("of ")+3);

System.***out***.println("After deletion count is " +countdelete1);

//String countdelete = count.substring(count.length()-1);

//String countdelete = count.substring(10, 12);

// Assert.assertEquals(countdelete1, expectedcount, "timesheet count is not matching");

//Assert.assertEquals(countdelete1, afterdelete,"timesheet count is not matching");

}

//Edit timesheet

**public** **void** edit\_timesheet() **throws** InterruptedException

{

Edittimesheet.click();

**boolean** editbutton = driver.findElements(By.*xpath*("//a[@href='javascript:void(0)'][contains(@id,'\_lnkBtnEdit')]")).size()>0;

**if**(editbutton)

{

System.***out***.println("Edit button is disabled");

}

**else**

{

Edittimesheet.click();

Thread.*sleep*(3000);

edittext.clear();

edittext.sendKeys("2");

updatebtn.click();

Thread.*sleep*(1000);

}

}

//To search timesheet by request Id

**public** **void** filterbyid() **throws** InterruptedException

{

searchbyid.click();

Thread.*sleep*(2000);

String reqtext = ExcelUtils.*getCellData*(21, 2);

searchtext.sendKeys(reqtext);

Thread.*sleep*(1000);

submitbtn.click();

Thread.*sleep*(1000);

**boolean** searchresult = driver.findElements(By.*xpath*("//span[contains(@id,'\_lblTimesheetTitle')]")).size()>0;

**if**(searchresult)

{

System.***out***.println("Search result is CORRECT");

}

**else**

{System.***out***.println("Search result is incorrect");

}

}

//Check View timesheet and perform actions

**public** **void** view\_timesheet() **throws** InterruptedException, AWTException

{

System.***out***.println("---\*\*\* ViewTimesheet.view\_timesheet()\*\*\*--- ");

timesheet.click();

Thread.*sleep*(1000);

*log*.info("Executing View all timesheet");

viewall\_availability();

Thread.*sleep*(1000);

*log*.info("Executing View timesheet");

timesheet.click();

Thread.*sleep*(1000);

view\_availability();

}

}

**Testing.xml**

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

<suite name=*"Sample Test Suite"* verbose=*"1"*>

<test name=*"Sample Test"*>

<classes>

<class name=*"actionpackage.actionclass"*>

<methods>

<include name=*"enterurl"*></include>

<include name=*"loginteam"*></include>

<!--include name="newreq"></include-->

<include name=*"viewreq"*></include>

<include name=*"reqdetails"*></include>

<include name=*"Request\_deliverable"*></include>

<include name=*"timesheet1"*></include>

<include name=*"Reports\_visibility"*></include>

<include name=*"feedback"*></include>

<include name=*"logout"*></include>

<include name=*"closetest"*></include>

</methods>

</class>

<!--/classes>

</test>

<test name="Sample Test2">

<classes>

<class name="actionpackage.actionrequestor">

<methods>

<include name="enterurl"></include>

<include name="loginreq"></include>

<include name="newreq"></include>

<include name="viewreq"></include>

<include name="reqdetails"></include>

<include name="Request\_deliverable"></include>

<include name="timesheet1"></include>

<include name="Reports\_visibility"></include>

<include name="feedback"></include>

<include name="logout"></include>

<include name="closetest"></include>

</methods>

</class>

<class name="actionpackage.actiongatekeeper">

<methods>

<include name="enterurl"></include>

<include name="logingate"></include>

<include name="newreq"></include>

<include name="viewreq"></include>

<include name="reqdetails"></include>

<include name="Request\_deliverable"></include>

<include name="timesheet1"></include>

<include name="Reports\_visibility"></include>

<include name="feedback"></include>

<include name="logout"></include>

<include name="closetest"></include>

</methods>

</class>

<class name="actionpackage.actionanalyst">

<methods>

<include name="enterurl"></include>

<include name="loginanalyst"></include>

<include name="newreq"></include>

<include name="viewreq"></include>

<include name="reqdetails"></include>

<include name="Request\_deliverable"></include>

<include name="timesheet1"></include>

<include name="Reports\_visibility"></include>

<include name="feedback"></include>

<include name="logout"></include>

<include name="closetest"></include>

</methods>

</class-->

</classes>

</test>

</suite>