|  |  |
| --- | --- |
| Project- | KredSafe |
| Automation QA | Nilesh Shekdar |
| Tool | Selenium |

**Selenium**

**Selenium** is a powerful, open-source **framework used to automate web browsers**. It’s widely used for:

* **Automated testing** of websites and web apps
* **Scraping** data (ethically and within legal/ToS limits)
* **Repetitive browser tasks**, like form filling or clicking

## Key Features

| **Feature** | **Description** |
| --- | --- |
| **Cross-Browser** | Works with Chrome, Firefox, Safari, Edge, etc. |
| **Cross-Platform** | Runs on Windows, macOS, Linux |
| **Supports Multiple Languages** | Python, Java, JavaScript, C#, Ruby |
| **WebDriver API** | Controls the browser like a human |
| **Integrates Easily** | Works with tools like TestNG, JUnit, Pytest, Jenkins, Docker |

## Core Components

### 1. **Selenium WebDriver**

* Core tool for **writing browser automation scripts**.
* Supports modern browsers and programming languages.

### 2. **Selenium IDE**

* A **browser extension** (for Chrome and Firefox) that lets you **record and playback** interactions with a website.
* Best for beginners or quick tests.

### 3. **Selenium Grid**

* Runs tests in **parallel** across different browsers/devices/machines.
* Ideal for **large-scale testing**.

### **Selenium WebDriver**

**Selenium WebDriver** is the **core component** of Selenium that lets you **programmatically control a web browser**—just like a human would.

It's like a **robot browser driver** that opens a real browser (like Chrome, Firefox, etc.) and:

* Clicks buttons
* Enters text into fields
* Submits forms
* Scrolls pages
* Extracts data

## How WebDriver Works

When you write a script using Selenium WebDriver:

1. **You write code** (in Python, Java, etc.)
2. **WebDriver opens the browser** (Chrome, Firefox, etc.)
3. Your code sends **commands** to the browser through WebDriver
4. Browser performs the actions and sends responses back

## What Can You Do With WebDriver?

| **Action** | **Code Example** |
| --- | --- |
| Open a URL | driver.get("https://example.com") |
| Find an element | driver.find\_element(By.ID, "login") |
| Click a button | button.click() |
| Fill out a form | input.send\_keys("yourname") |
| Submit a form | form.submit() |
| Take a screenshot | driver.save\_screenshot("page.png") |
|  |  |

## Browser Drivers

WebDriver needs a **browser-specific driver** to work:

| **Browser** | **Driver** |
| --- | --- |
| Chrome | ChromeDriver |
| Firefox | Gecko Driver |
| Edge | EdgeDriver |
| Safari | Built-in on macOS |

## Step-by-Step: Selenium WebDriver in Eclipse (Java)

### Step 1: **Install Java Development Kit (JDK)**

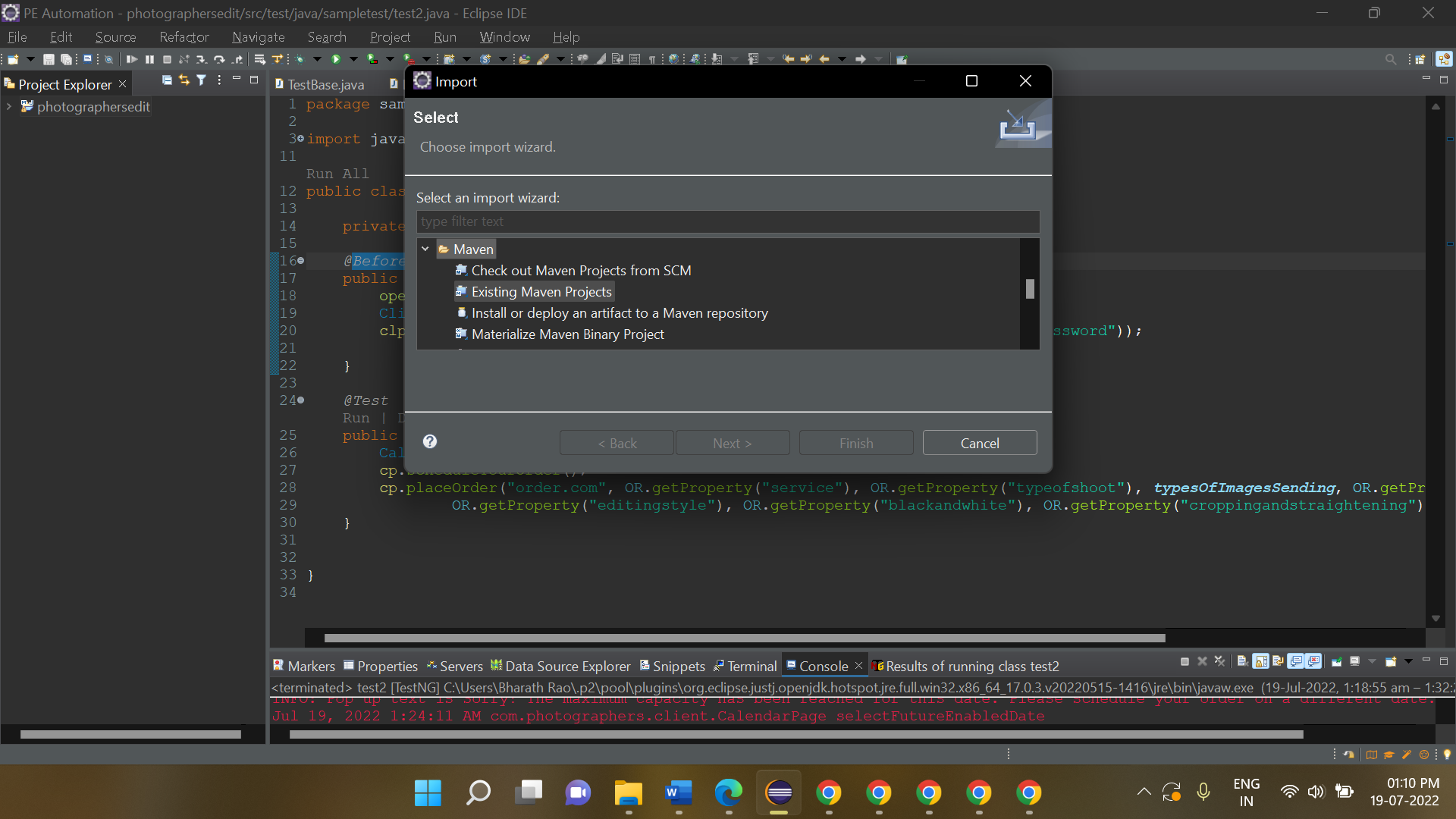
* **Download JDK**: <https://www.oracle.com/java/technologies/javase-downloads.html>
* **Set up JAVA\_HOME** in environment variables
* Check in terminal/cmd:

java -version

javac –version

### Step 2: **Install Eclipse IDE for Java Developers**

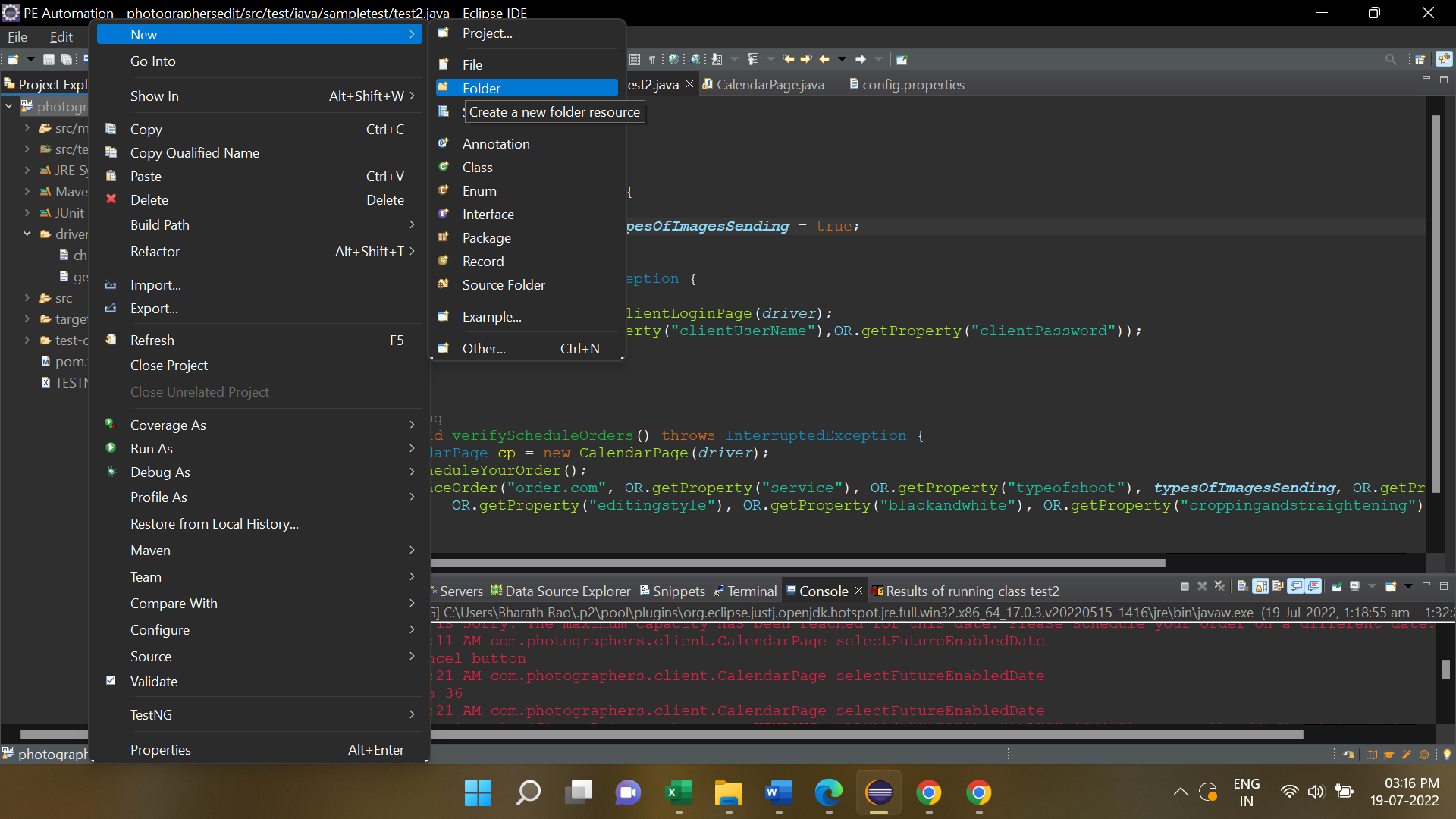
* Download: https://www.eclipse.org/downloads/
* Install and open Eclipse
* Choose a **workspace folder** (any folder where your projects will be saved)
* Navigate to the below URL and download Eclipse IDE 2022‑06.
* [Eclipse Downloads | The Eclipse Foundation](https://www.eclipse.org/downloads/)
* The downloaded exe file or Eclipse installer would like this: eclipse-inst-jre-win64.exe
* Double click on the Eclipse installer and select the following IDE flavour “Eclipse IDE for Enterprise Java and Web Developers” and proceed with the installation.
* Install java version "17.0.3.1"
* Unzip the project file that I have shared.
* In Eclipse IDE, Navigate to File>> Import>> Maven>> Existing Maven Projects. Click Finish



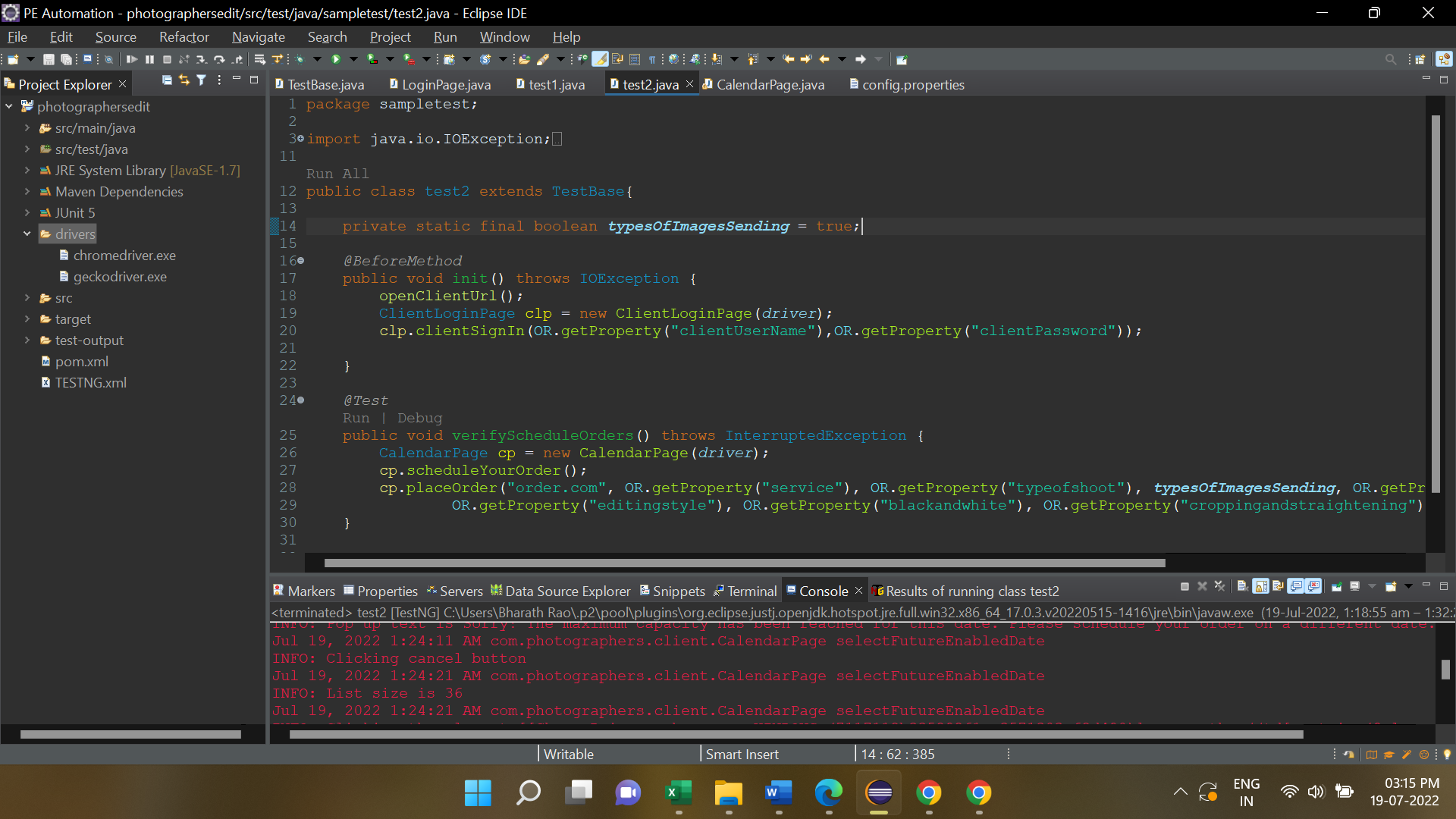
Go to Advanced Settings>> Environment Variables>> System Variables>> Cleck New and add the following:

Variable name CLASSPATH

Variable value C:\Program Files\Java\jdk-17.0.3.1\bin (Add the jdk path from your local machine like it is mentioned)



Find a folder with name “Drivers”, place the drivers in this folder.



### Step 3: **Create a Java Project**

1. Open Eclipse
2. Click **File → New → Java Project**
3. Name it (e.g., SeleniumTest)
4. Click **Finish**

### Step 4: **Add Selenium Library to Project**

1. **Right-click** on your project → **Build Path → Configure Build Path**
2. Go to the **Libraries** tab → click **Add External JARs**
3. Download the **Selenium Java Client**:
   * From: https://www.selenium.dev/downloads/
   * Extract the .zip file
   * Add:
     + All .jar files inside the main folder
     + All .jar files inside the libs folder

Click **Apply and Close**

### Step 5: **Download Browser Driver**

You need a driver to control a browser. Choose one:

| **Browser** | **Driver** | **Link** |
| --- | --- | --- |
| Chrome | ChromeDriver | https://sites.google.com/chromium.org/driver/ |
| Firefox | GeckoDriver | <https://github.com/mozilla/geckodriver/releases> |
| Edge | EdgeDriver | https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/ |

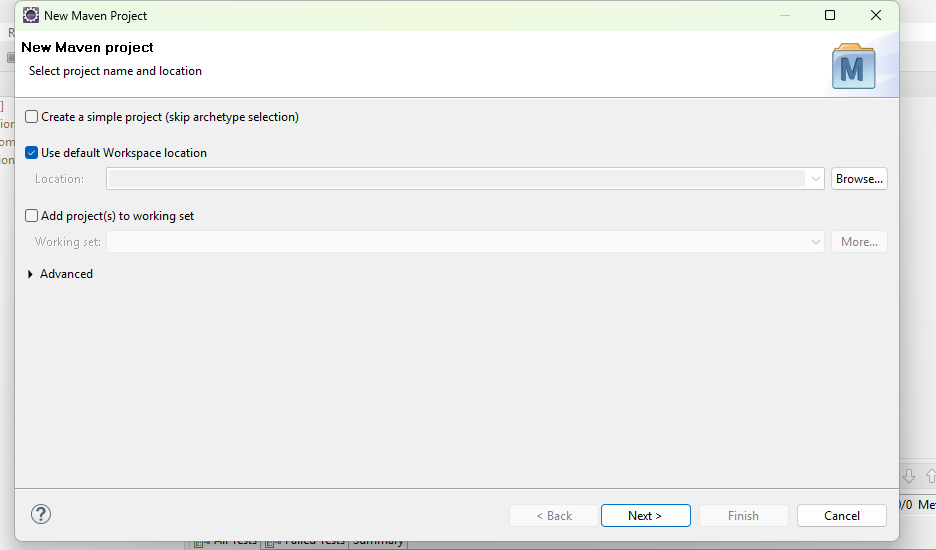
Let Create a Maven Project –

Step to be Followed-

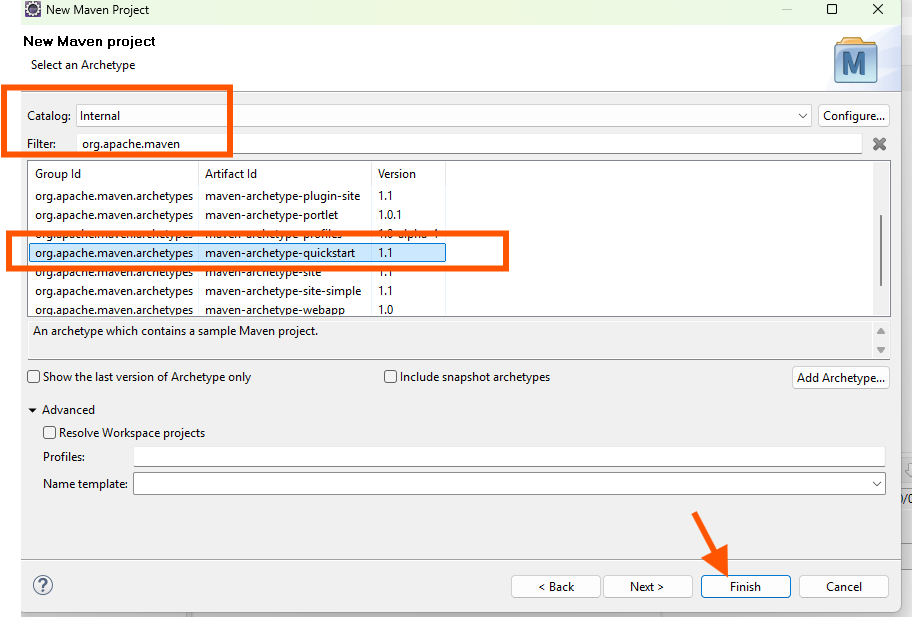
1.Open Eclipse

2.Click on File – New –Maven Project-

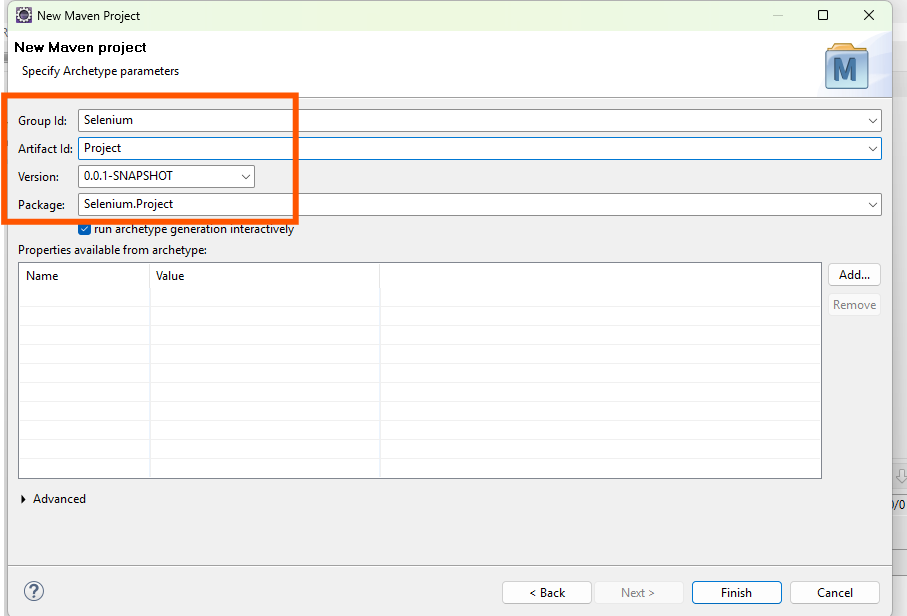
Click on Next



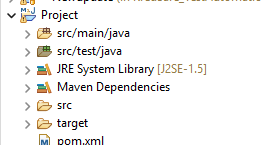
Select and Click on Next



Click on Finish



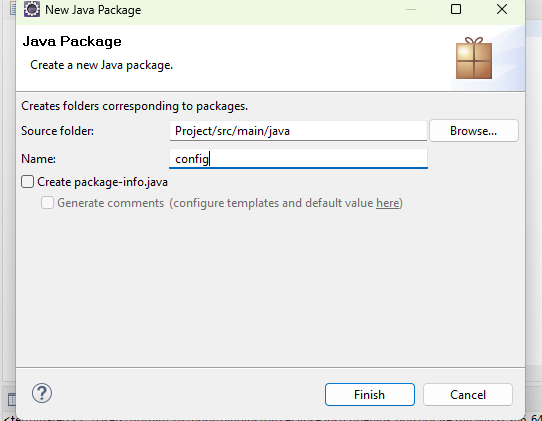
Project Folder created



Step by step add all dependency code-

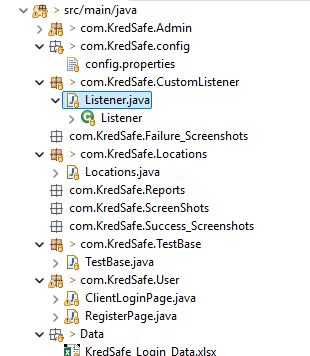
1.Add Config File

Right click on Project name and select New Jave Package



Update other packages as below

Config, Properties add Test Data



Import exiting maven project in Eclipse-

 Open Eclipse

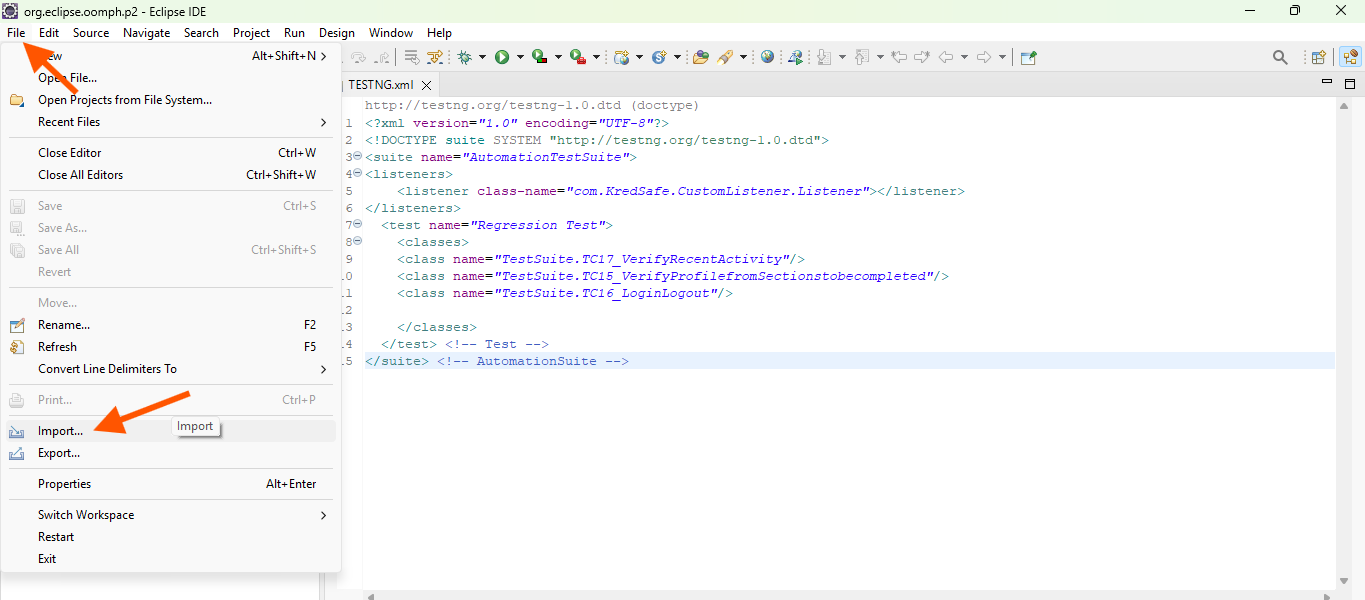
 Select **File** → **Import...**

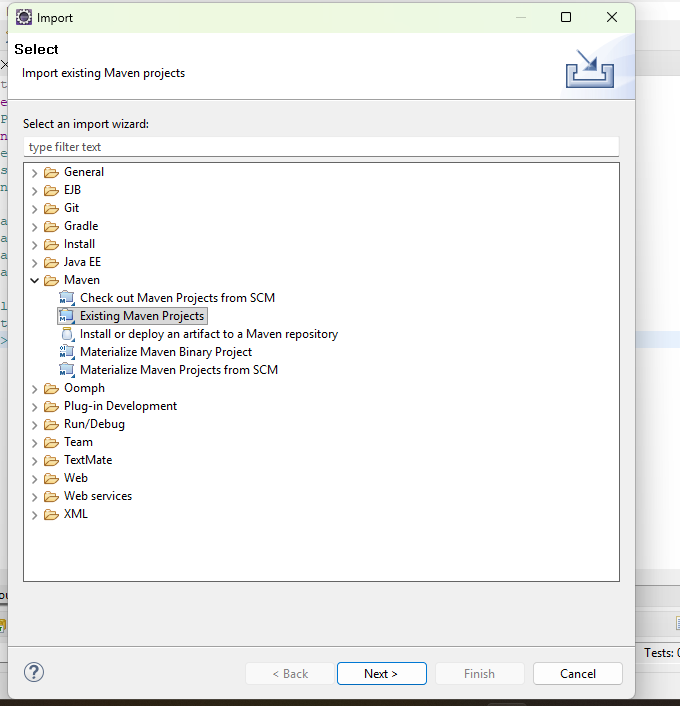
 Choose **Maven** → **Existing Maven Projects**, click **Next**

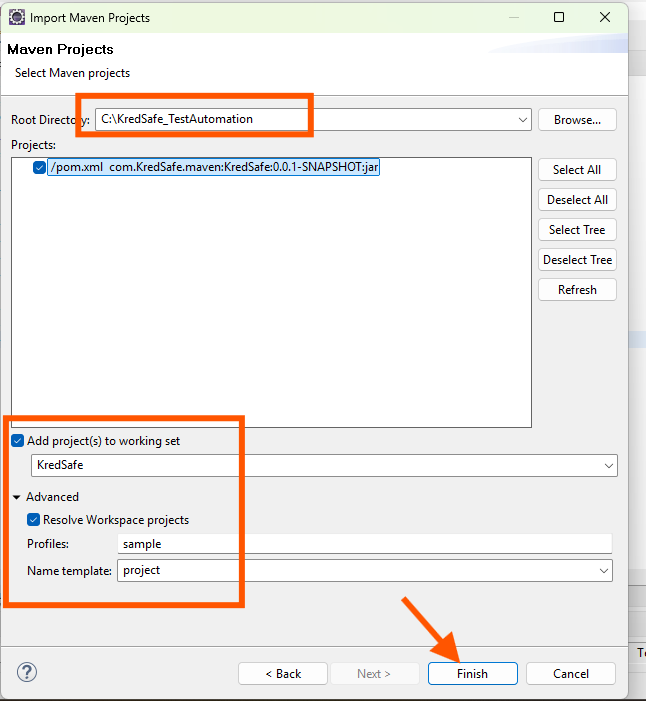
 Browse to the root folder of your Maven project (where pom.xml is)

 Eclipse will detect the project, select it, click **Finish**

 Eclipse downloads dependencies and sets up the project







### Selenium Framework –

A **Selenium Framework** is a structured way to build, organize, and run Selenium test automation scripts efficiently. It improves **code reusability**, **maintainability**, and **scalability** for test projects.

## What Is a Selenium Framework?

A **framework** in Selenium is a set of rules and libraries you build (or use) to streamline your automation process.

### Goals:

* Make test code clean, modular, and reusable.
* Easily manage test data, reports, and logs.
* Integrate with tools like TestNG, Maven, Jenkins, etc.

## Types of Selenium Frameworks

Here are the **most common Selenium framework types**:

### 1. **Modular Framework**

* Breaks the test into **small, independent modules**.
* Reusable code blocks for different parts of the application.

**Pros**: Easy to maintain  
**Cons**: Needs more planning and structure

### 2. **Data-Driven Framework**

* Test data is stored externally (Excel, CSV, DB, JSON).
* Test scripts use the same code but run with different data sets.

**Tools**: Apache POI, JSON, ExcelReader

**Pros**: Run many tests with different inputs  
**Cons**: Data handling can become complex

## ****Steps to Set Up a Data-Driven Framework in Selenium****

Test data is stored externally (Excel/CSV), and test scripts read the data to run tests.

1. **Create a Maven project and add Selenium + Apache dependencies.**
2. **Create a utility class to read data from Excel using Apache POI.**
3. **Store your test data in an Excel file (e.g., testdata.xlsx).**
4. **Create a TestNG test class and fetch data using @DataProvider.**
5. **Pass Excel data to your test methods using parameters.**
6. **Run tests via testng.xml or mvn test to execute with multiple data sets.**

### 3. **Keyword-Driven Framework**

* Keywords (actions like click, enter, verify) are stored in external files.
* Scripts are driven by these keywords.

**Pros**: Non-programmers can contribute  
**Cons**: More setup and parsing logic needed

## ****Steps to Set Up a Keyword-Driven Framework in Selenium****

Actions (like click, type, navigate) are defined as keywords and executed dynamically.

1. **Create a Maven project and add Selenium + Apache dependencies.**
2. **Create an Excel file with columns: TestCaseName, Action, Locator, Value.**
3. **Write a KeywordEngine class to read the Excel and map keywords to WebDriver actions.**
4. **Create a utility class to perform actions like click(), sendKeys(), navigate().**
5. **Write a TestNG test method that calls the KeywordEngine and executes actions from Excel.**
6. **Run tests to execute actions step-by-step based on Excel instructions.**

### 4. **Hybrid Framework**

* Combines Modular + Data-Driven + Keyword-Driven.
* Most commonly used in real-world Selenium projects.

**Pros**: Highly flexible and powerful  
**Cons**: More complex setup

/-----------Application-----------------/

browser = chrome

applicationUrl= https://dev.kredsafe.net/

applicationUrl1= https://stage.kredsafe.net/

/-----------Registration-TC01\_VerifyRegisterUsingOnlineApplication-----------------/

userEmailBoxUrl = https://yopmail.com

firstName = vivek

lastName = Raut

createAccountPassword = Nilesh@2025

createAccountCPassword = Nilesh@2025

mobileNumber = 7600050023

emailID = ts053376@yopmail.com

/------------TC02\_verifyRegisterbyUploadingResume---------------/

resumeEmail = resumetest@yopmail.com

-------------TC03\_verifyProfileDetailsFunctionality----/

emailID1 = swapj@yopmail.com

createAccountPassword1 = Nilesh@2030

dob = 11/11/1994

SSN = 343783424

PhoneNumber = 9864205439

Zipcode = 441601

/------------TC04\_OTP---------------/

emailID11= ts1234@yopmail.com

createAccountPassword11 = Nilesh@2025

/-----------------TC04\_OTPVerification-----------------/

emailID4 = swapj@yopmail.com

createAccountPassword4 = Nilesh@2025

OTPSource = EmailOTP

/-------------------TC05\_VerifyProfilePicture-----TC06\_VerifyChangepassword-----------/

emailID1 = swapj@yopmail.com

createAccountPassword1 = Nilesh@2030

/------------------------ TC09\_ProfileNaviagtion --------/

emailID2 = nh910@yopmail.com

createAccountPassword2 = Nilesh@2025

OTPSource1 = EmailOTP

/----------------------------- TC11\_verifydocumentupload---------------------------/

emailID3= ts17045@yopmail.com

createAccountPassword3 = Nilesh@2025

/-------------------TC12\_Verifynotesprocess--------------------------/

emailID4= swapj@yopmail.com

createAccountPassword4 = Nilesh@2025

/---------------TC13\_ProfileNaviagtionremovetheaddanother----TC14\_MakePrimaryEmail-----------------------------------/

emailID5= snj12@yopmail.com

createAccountPassword5 = Nilesh@2025

/----------------TC15\_VerifyProfilefromSectionstobecompleted---------------------------------------/

emailID6= nh1002@yopmail.com

createAccountPassword6 = Nilesh@2025

/-----------------------TC10\_VerifySubscriptionprocess-------------------------------------/

emailID10= hrd09769@yopmail.com

createAccountPassword10 = Nilesh@2025

/-----------------------TC11\_verifydocumentupload-----------------------------/

emailID12= ts7876001@yopmail.com

createAccountPassword12 = Nilesh@2025

/----------------- TC18\_Restoreprofiledata------------------------------------------------------------------/

emailID18= ts1234@yopmail.com

createAccountPassword18 = Nilesh@2025

/-----------LoginCredentials-----------------/

clientSignIn = pks7@yopmail.com

clientSignInPassword = Nilesh@2025

/------------Admin------------------------/

AdminUsername = priyankacs@yopmail.com

AdminPassword = Tiucs@2025

-----------------------------------------------------------------

Listner .java

package com.KredSafe.CustomListener;

import java.io.File;

import java.io.IOException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.testng.ITestContext;

import org.testng.ITestListener;

import org.testng.ITestResult;

import org.testng.Reporter;

import com.KredSafe.Locations.Locations;

import com.KredSafe.TestBase.TestBase;

public class Listener extends TestBase implements ITestListener {

public void onFinish(ITestContext result) {

log.info("<====Test Finished=====>"+((ITestResult) result).getMethod().getMethodName());

}

public void onStart(ITestContext arg0) {

log.info("The test has started");

}

public void onTestFailedButWithinSuccessPercentage(ITestResult arg0) {

// TODO Auto-generated method stub

}

public void onTestFailure(ITestResult result) {

Calendar calendar=Calendar.getInstance();

SimpleDateFormat formater= new SimpleDateFormat("dd\_MM\_yyyy\_hh\_mm\_ss");

String methodName=result.getName();

if(!result.isSuccess())

{

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

try

{

String reportDirectory=new File(System.getProperty("user.dir")).getAbsolutePath()+Locations.screenshotlocation;

File destFile=new File((String)reportDirectory+"//Failure\_Screenshots//"+methodName+"\_"+formater.format(calendar.getTime())+".png");

FileUtils.copyFile(scrFile, destFile);

//This will help you to link the screenshot in TestNg Report

Reporter.log("<a href='"+destFile.getAbsolutePath()+"'><img src='"+destFile.getAbsolutePath()+"'height='100'width='100'/></a>");

}catch(IOException e)

{

e.printStackTrace();

}

}

}

public void onTestSkipped(ITestResult result) {

log.info("<====Test Skipped=====>"+result.getMethod().getMethodName());

}

public void onTestStart(ITestResult result) {

log.info("<====Test Started Running=====>"+result.getMethod().getMethodName());

}

public void onTestSuccess(ITestResult result) {

Calendar calendar=Calendar.getInstance();

SimpleDateFormat formater= new SimpleDateFormat("dd\_MM\_yyyy\_hh\_mm\_ss");

String methodName=result.getName();

if(result.isSuccess())

{

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

try

{

String reportDirectory=new File(System.getProperty("user.dir")).getAbsolutePath()+Locations.screenshotlocation;

File destFile=new File((String)reportDirectory+"//Success\_Screenshots//"+methodName+"\_"+formater.format(calendar.getTime())+".png");

FileUtils.copyFile(scrFile, destFile);

//This will help you to link the screenshot in TestNg Report

Reporter.log("<a href='"+destFile.getAbsolutePath()+"'><img src='"+destFile.getAbsolutePath()+"'height='100'width='100'/></a>");

}catch(IOException e)

{

e.printStackTrace();

}

}

}

}

----------------------------------------------------------------------------------------------------------------------

Location .Java

**package** com.KredSafe.Locations;

**public** **class** Locations {

**public** **static** String *geckoDriverPath*=System.*getProperty*("user.dir")+"/drivers/geckodriver.exe";

**public** **static** String *chromeDriverPath*=System.*getProperty*("user.dir")+"/drivers/chromedriver.exe";

**public** **static** String *configFilePath*=System.*getProperty*("user.dir") + "//src//main//java//com//KredSafe//Config//config.properties";

**public** **static** String *extentConfigPath*=System.*getProperty*("user.dir")+"//extent-config.xml";

**public** **static** String *extentReportPath*=System.*getProperty*("user.dir")+"//src//main//java//com//KredSafe//Reports//Test\_Execution\_Report-";

**public** **static** String *captureScreenShotPath*="//src//main//java//com//KredSafe//ScreenShots//Screen\_Shot\_Of-";

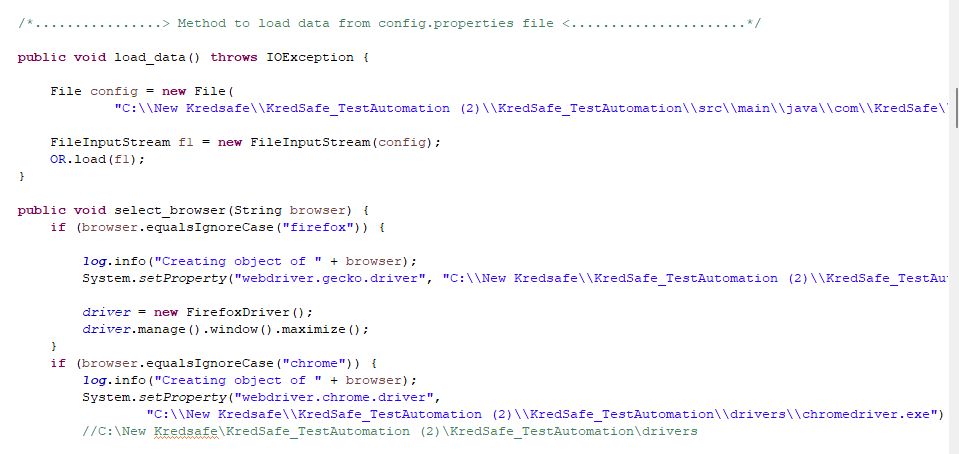
**public** **static** String *screenshotlocation*="//src//main//java//com//KredSafe//";

}

------------------------------------------------------------------------------------

Test Base Java

In Test Base java we have set config url and drivers path



package com.KredSafe.TestBase;

import java.awt.AWTException;

import java.awt.Robot;

import java.awt.Toolkit;

import java.awt.datatransfer.StringSelection;

import java.awt.event.KeyEvent;

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Properties;

import java.util.concurrent.TimeUnit;

import java.util.logging.Logger;

import org.apache.commons.io.FileUtils;

import org.apache.log4j.PropertyConfigurator;

import org.apache.poi.ss.usermodel.DataFormatter;

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;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.ie.InternetExplorerDriver;

import org.openqa.selenium.interactions.Actions;

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

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

import org.testng.ITestResult;

import org.testng.Reporter;

import com.KredSafe.Locations.Locations;

import com.relevantcodes.extentreports.DisplayOrder;

import com.relevantcodes.extentreports.ExtentReports;

import com.relevantcodes.extentreports.ExtentTest;

import com.relevantcodes.extentreports.LogStatus;

import jxl.read.biff.BiffException;

public class TestBase {

public static final Logger log = Logger.getLogger(TestBase.class.getName());

public static WebDriver driver;

public Properties OR = new Properties();

public static ExtentReports extent;

public static ExtentTest test;

public WebDriver getDriver() {

return driver;

}

/\*...................> Static Block to generate Reports <..........................\*/

static

{

Calendar calendar=Calendar.getInstance();

SimpleDateFormat formater= new SimpleDateFormat("dd\_MM\_yyyy\_hh\_mm\_ss");

extent=new ExtentReports(Locations.extentReportPath+formater.format(calendar.getTime())+".html",false, DisplayOrder.NEWEST\_FIRST);

extent

.addSystemInfo("Host Name", "HP Desktop")

.addSystemInfo("Role", "Automation Engineer")

.addSystemInfo("User Name", "Nilesh Shekdar");

extent.loadConfig(new File(Locations.extentConfigPath));

}

/\*...............> Method to initialize browser and to get URL of the page <...............\*/

public void openAdminUrl() throws IOException {

load\_data();

select\_browser(OR.getProperty("browser"));

get\_url(OR.getProperty("adminLoginUrl"));

}

/\*...................> Method to initialize browser and to get URL of the page <..........................\*/

public void init\_agency() throws IOException

{

load\_data();

String log4jConfPath="log4j.properties";

PropertyConfigurator.configure(log4jConfPath);

select\_browser(OR.getProperty("browser"));

get\_url(OR.getProperty("Test\_Url\_Agency"));

}

public void openAppUrl() throws IOException {

load\_data();

select\_browser(OR.getProperty("browser"));

get\_url(OR.getProperty("applicationUrl"));

}

public void openUserEmailUrl() throws IOException {

load\_data();

select\_browser(OR.getProperty("browser"));

get\_url(OR.getProperty("userEmailBoxUrl"));

}

/\*................> Method to load data from config.properties file <......................\*/

public void load\_data() throws IOException {

File config = new File(

"C:\\New Kredsafe\\KredSafe\_TestAutomation (2)\\KredSafe\_TestAutomation\\src\\main\\java\\com\\KredSafe\\config\\config.properties");

FileInputStream f1 = new FileInputStream(config);

OR.load(f1);

}

public void select\_browser(String browser) {

if (browser.equalsIgnoreCase("firefox")) {

log.info("Creating object of " + browser);

System.setProperty("webdriver.gecko.driver", "C:\\New Kredsafe\\KredSafe\_TestAutomation (2)\\KredSafe\_TestAutomation\\drivers/geckodriver.exe");

driver = new FirefoxDriver();

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

}

if (browser.equalsIgnoreCase("chrome")) {

log.info("Creating object of " + browser);

System.setProperty("webdriver.chrome.driver",

"C:\\New Kredsafe\\KredSafe\_TestAutomation (2)\\KredSafe\_TestAutomation\\drivers\\chromedriver.exe");

//C:\New Kredsafe\KredSafe\_TestAutomation (2)\KredSafe\_TestAutomation\drivers

driver = new ChromeDriver();

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

}

if (browser.equalsIgnoreCase("ie")) {

log.info("Creating object of " + browser);

System.setProperty("webdriver.chrome.driver",

System.getProperty("user.dir") + "/drivers/IEDriverServer.exe");

driver = new InternetExplorerDriver();

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

}

}

public void get\_url(String url) {

log.info("Navigating to URL");

driver.get(url);

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

}

public void scrollDown() {

((JavascriptExecutor) driver).executeScript("window.scrollTo(0, document.body.scrollHeight)");

}

public void scrollToElement(WebElement element) {

((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView(false);", element);

}

public static void scrollDownToElement(WebElement element) {

((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView();", element);

}

public void clickElement(WebElement element) {

((JavascriptExecutor) driver).executeScript("window.scrollTo(0," + element.getLocation().y + ")");

element.click();

}

/\*..................> Normal wait method using Thread.sleep method <.......................\*/

public void waitInSeconds(int i) throws InterruptedException{

Thread.sleep(1000\*i);

}

/\*...................> Method to CaptureScreenshot <..........................\*/

public String capture\_screen(String fileName)

{

if (fileName == "")

{

fileName = "blank";

}

File destFile = null;

Calendar calendar = Calendar.getInstance();

SimpleDateFormat formater = new SimpleDateFormat("dd\_MM\_yyyy\_hh\_mm\_ss");

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

try {

String reportDirectory = new File(System.getProperty("user.dir")).getAbsolutePath() + Locations.captureScreenShotPath;

destFile = new File((String) reportDirectory + fileName + "\_" + formater.format(calendar.getTime()) + ".png");

FileUtils.copyFile(scrFile, destFile);

// This will help us to link the screen shot in testNG report

Reporter.log("<a href='" + destFile.getAbsolutePath() + "'> <img src='" + destFile.getAbsolutePath() + "' height='100' width='100'/> </a>");

} catch (IOException e) {

e.printStackTrace();

}

return destFile.toString();

}

/\*...................> Method to close a Browser <..........................\*/

public void closeBrowser() {

wait\_for\_page\_load(15);

driver.close();

log.info("browser closed");

extent.endTest(test);

extent.flush();

}

public static void wait\_for\_page\_load(int time)

{

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

}

/\*...................> Method for generating extent reports <..........................\*/

public void get\_result(ITestResult result) {

if (result.getStatus() == ITestResult.FAILURE) {

test.log(LogStatus.FAIL, result.getThrowable());

String screen = capture\_screen(result.getName());

test.log(LogStatus.FAIL, test.addScreenCapture(screen));

} else if (result.getStatus() == ITestResult.SUCCESS) {

test.log(LogStatus.PASS, "Test passed");

String screen = capture\_screen(result.getName());

test.log(LogStatus.PASS, test.addScreenCapture(screen));

} else if (result.getStatus() == ITestResult.SKIP) {

test.log(LogStatus.SKIP, "Test skipped: " + result.getThrowable());

}

extent.endTest(test);

extent.flush();

//driver.quit();

}

public static void wait\_for\_element\_present(WebElement wb)

{

WebDriverWait wait=new WebDriverWait(driver,40);

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

}

/\*...................> Method to initialize browser and to get URL of the page <..........................\*/

public void init() throws IOException

{

String log4jConfPath="log4j.properties";

PropertyConfigurator.configure(log4jConfPath);

}

public static void mouseClickByLocator(WebElement element) {

JavascriptExecutor executor = (JavascriptExecutor)driver;

executor.executeScript("arguments[0].click();", element);

}

public static void wait\_in\_seconds(int i) throws InterruptedException

{

Thread.sleep(1000\*i);

}

/\*...................> Robot class to upload a File <..........................\*/

public void file\_upload(String path) throws InterruptedException, AWTException

{

Robot r=new Robot();

wait\_in\_seconds(3);

StringSelection s=new StringSelection(path);

Toolkit.getDefaultToolkit().getSystemClipboard().setContents(s, null);

wait\_in\_seconds(4);

log.info("Uploading file");

r.keyPress(KeyEvent.VK\_CONTROL);

r.keyPress(KeyEvent.VK\_V);

r.keyRelease(KeyEvent.VK\_CONTROL);

r.keyRelease(KeyEvent.VK\_V);

wait\_in\_seconds(3);

r.keyPress(KeyEvent.VK\_ENTER);

r.keyRelease(KeyEvent.VK\_ENTER);

wait\_in\_seconds(3);

}

public static void click\_Element\_Using\_JS(WebElement ele)

{

JavascriptExecutor executor = (JavascriptExecutor)driver;

executor.executeScript("arguments[0].click()", ele);

}

public static void click\_Element\_Using\_Actions(WebElement ele)

{

Actions actions = new Actions(driver);

actions.moveToElement(ele).click().perform();

}

public static FileInputStream fis;

public static XSSFWorkbook workbook;

public static XSSFSheet sheet;

private static XSSFCell cell;

public String[][] getData(String excelName, String sheetName) throws BiffException, IOException

{

//System.out.println(System.getProperty("user.dir"));

String excellocation = System.getProperty("user.dir")+"//src//main//java//Data//"+excelName;

System.out.println(excellocation);

return readExcel(excellocation, sheetName);

}

public String[][] readExcel(String excellocation, String sheetName) throws IOException {

// TODO Auto-generated method stub

int rownum = get\_rowcount(excellocation, sheetName);

int columncount = get\_columncount(excellocation, sheetName, 1);

String edata[][] = new String[rownum][columncount];

for (int i = 1; i <= rownum; i++) {

for (int j = 0; j < columncount; j++) {

edata[i - 1][j] = get\_celldata(excellocation, sheetName, i, j);

System.out.print(edata[i - 1][j] + " ");

System.out.print("");

}

}

for (String[] row : edata) {

System.out.print("The excel sheet value are" + row);

}

return edata;

}

private int get\_columncount(String excellocation, String sheetName, int i) throws IOException {

fis = new FileInputStream(excellocation);

workbook = new XSSFWorkbook(fis);

sheet = workbook.getSheet(sheetName);

int rownum = 0;

XSSFRow row = sheet.getRow(rownum);

int columncount = row.getLastCellNum();

return columncount;

}

private int get\_rowcount(String excellocation, String sheetName) throws IOException {

fis = new FileInputStream(excellocation);

workbook = new XSSFWorkbook(fis);

sheet = workbook.getSheet(sheetName);

int rowcount = sheet.getLastRowNum();

return rowcount;

}

public static String get\_celldata(String filepath, String sheetname, int rownum, int column) throws IOException {

fis = new FileInputStream(filepath);

workbook = new XSSFWorkbook(fis);

sheet = workbook.getSheet(sheetname);

XSSFRow row = sheet.getRow(rownum);

cell = row.getCell(column);

String data;

try {

DataFormatter formatter = new DataFormatter();

String celldata = formatter.formatCellValue(cell);

return celldata;

} catch (Exception e) {

data = "";

}

return data;

}

}

-----------------------------------------------------------------------------------------------------

Client Login Page

package com.KredSafe.User;

import java.awt.AWTException;

import java.util.ArrayList;

import java.util.List;

import java.util.Properties;

import javax.xml.datatype.Duration;

import org.apache.log4j.Logger;

import org.openqa.selenium.By;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.Keys;

import org.openqa.selenium.NoSuchElementException;

import org.openqa.selenium.StaleElementReferenceException;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.interactions.Actions;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

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

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

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

import org.testng.AssertJUnit;

import com.KredSafe.TestBase.TestBase;

import edu.emory.mathcs.backport.java.util.concurrent.TimeoutException;

import junit.framework.Assert;

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

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

public class ClientLoginPage extends TestBase {

public static final Logger log = Logger.getLogger(ClientLoginPage.class.getName());

WebDriver driver;

public Properties OR = new Properties();

public ClientLoginPage(WebDriver driver) {

this.driver = driver;

PageFactory.initElements(driver, this);

}

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

WebElement loginField;

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

WebElement passwordField;

@FindBy(xpath = "//button[@class='button btn btn-block']")

WebElement signInButton;

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

WebElement acceptButton;

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

WebElement loginLink;

@FindBy(xpath = "//\*[@id=\"idOTP\"]")

WebElement OTPText1;

@FindBy(xpath = "//\*[@id=\"btnSubmit\"]")

WebElement otpverifybutton;

@FindBy(xpath = "//a[@href='https://dev.kredsafe.net/admin/home']")

WebElement clickHere;

@FindBy(xpath = "//span[@class=\"val\_error\"]")

List<WebElement> incorrectpassword;

@FindBy(xpath = "//\*[@id=\"flashMsgTxt\"]")

WebElement errormsg;

@FindBy(xpath = "//\*[@id=\"zip\"]")

WebElement zipcode;

@FindBy(xpath = "//\*[@id=\"mail\"]/div/div/table/tbody/tr/td/p[2]")

WebElement OTPCode;

@FindBy(xpath = "//\*[@id=\"idOTP\"]")

WebElement idOTP;

@FindBy(xpath = "//\*[@id=\"btnSubmit\"]")

WebElement verifyButton;

public void accessYopmaiforOTP() throws InterruptedException {

((JavascriptExecutor) driver).executeScript("window.open()");

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

driver.switchTo().window(tabs.get(1));

driver.get("https://yopmail.com/");

WebElement emailField = driver.findElement(By.id("login"));

emailField.sendKeys("ts1234@yopmail.com");

WebElement submitYopmail = driver.findElement(By.id("refreshbut"));

submitYopmail.click();

wait\_for\_page\_load(20);

waitInSeconds(10);

driver.switchTo().frame("ifmail");

String OTP = OTPCode.getText();

System.out.printf("Capture OTP code is: " + OTP);

}

public void accessYopmaiforOTP(String username) throws InterruptedException {

((JavascriptExecutor) driver).executeScript("window.open()");

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

driver.switchTo().window(tabs.get(1));

driver.get(" https://yopmail.com/ ");

WebElement emailField = driver.findElement(By.id("login"));

emailField.sendKeys(username);

WebElement submitYopmail = driver.findElement(By.id("refreshbut"));

submitYopmail.click();

wait\_for\_page\_load(20);

waitInSeconds(10);

driver.switchTo().frame("ifmail");

String OTP = OTPCode.getText();

System.out.printf("Capture OTP code is: " + OTP);

String code = OTP.substring(99);

System.out.printf("Extracted OTP code is: " + code);

String otpcode = code.substring(0, 6);

System.out.printf("\nOTP code is:" + otpcode);

ArrayList<String> tab = new ArrayList<String>(driver.getWindowHandles());

System.out.println("Switch to tab...");

driver.switchTo().window(tab.get(0));

idOTP.sendKeys(otpcode);

verifyButton.click();

}

/\*

\* Client Sign In method, pass user name and password from config file Call this

\* method for login to the application This is the most common reused method

\*/

public void clientSignIn(String username, String password) throws InterruptedException {

waitInSeconds(3);

try {

if (loginLink.isDisplayed()) {

loginLink.click();

}

} catch (NoSuchElementException e) {

}

waitInSeconds(3);

log.info("Clicking on accept button");

try {

if (acceptButton.isDisplayed()) {

acceptButton.click();

}

} catch (NoSuchElementException e) {

}

waitInSeconds(3);

loginField.sendKeys(username);

passwordField.sendKeys(password);

wait\_in\_seconds(5);

scrollDownToElement(signInButton);

wait\_in\_seconds(5);

signInButton.click();

wait\_in\_seconds(5);

String get\_currenturl = driver.getCurrentUrl();

System.out.println("Current URL is:" + get\_currenturl);

if (get\_currenturl.contains("https://dev.kredsafe.net/accept-otp")) {

System.out.println("The url is " + get\_currenturl);

accessYopmaiforOTP(username);

}

}

public void Subscriptiontotal() throws InterruptedException, AWTException {

waitInSeconds(10);

driver.get("https://dev.kredsafe.net/user/subscription/dashboard");

wait\_for\_page\_load(10);

scrollDown();

String pageText = driver.findElement(By.tagName("body")).getText();

WebElement subscriptionElement = driver.findElement(By.xpath("//span[@class='pill green']"));

String subscriptionValue = subscriptionElement.getText();

System.out.println("Subscription - " + subscriptionValue);

WebElement formsPacketElement = driver.findElement(By.xpath("//span[@class='pill yellow']"));

String formsPacketValue = formsPacketElement.getText();

System.out.println("Forms and Packet - " + formsPacketValue);

WebElement totalElement = driver.findElement(By.xpath("//span[@class='pill red']"));

String totalValue = totalElement.getText();

System.out.println("Total - " + totalValue);

}

@FindBy(xpath = "//a[contains(@href, 'https://dev.kredsafe.net/user/home')]")

WebElement profile;

@FindBy(xpath = "//p[contains(text(),'Personal Details')]")

WebElement personalDetails;

@FindBy(xpath = "//input[@name='first\_name']")

WebElement firstName;

@FindBy(xpath = "//input[@name='last\_name']")

WebElement lastName;

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

WebElement dob;

@FindBy(xpath = "//input[@val\_error\_msg='SSN is required.']")

WebElement SSN;

@FindBy(xpath = "//input[@ID='phone']")

WebElement phoneNumber;

@FindBy(xpath = "//input[@ID='email']")

WebElement email;

@FindBy(xpath = "//input[@ID='zip']")

WebElement zipCode;

@FindBy(xpath = "//select[@id='country']")

WebElement country;

@FindBy(xpath = "//select[@id='state-list']")

WebElement stateList;

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

WebElement city;

@FindBy(xpath = "//select[@id='industry\_id']")

WebElement industryType;

//\*[@id="industry\_id"]

@FindBy(xpath = "//select[@id='category']")

WebElement categoryType;

@FindBy(xpath = "//button[@id='id\_frm\_submit']")

WebElement submitButton;

@FindBy(xpath = "//span[@class=\"select2-search select2-search--inline\"]")

WebElement specialitiesSelect;

@FindBy(xpath = "//li[@class=\"select2-results\_\_option select2-results\_\_option--selectable select2-results\_\_option--highlighted\"]")

WebElement selectFromDropDown;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\"]")

WebElement saveAndContinue1;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\"]")

WebElement License;

@FindBy(xpath = "//\*[@id=\"profile\_img\"]")

WebElement Uploadimage;

@FindBy(xpath = "//\*[@id=\"userNameId\"]")

WebElement Profilenameclick;

@FindBy(xpath = "//\*[@id=\"oldpassword\"]")

WebElement oldpw;

@FindBy(xpath = "//\*[@id=\"passwordch\"]")

WebElement newpw;

@FindBy(xpath = "//\*[@id=\"password\_conf\"]")

WebElement newpwc;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\"]")

WebElement pwsubmit;

@FindBy(xpath = "//\*[@id=\"loginForm\"]/div[4]/div[2]/a")

WebElement forgotpassword;

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

WebElement emailidf;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\"]")

WebElement submitemail;

@FindBy(xpath = "//a[contains(@href, 'https://dev.kredsafe.net/user/change-password')]")

WebElement Changepasswordlink;

@FindBy(xpath = " //\*[@id=\"passwordChangeAlertModal\"]/div/div/div[2]/button\r\n" + "")

WebElement popok;

@FindBy(xpath = "/html/body/div[3]/nav/ul[1]/li[2]/span/img")

WebElement menuclick;

@FindBy(xpath = "//li[contains(@class,'pendingItemLink')]//a")

WebElement pendinglink;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div/form/div/div[1]/div[2]/div/div/div/div[3]/ul/li[1]/a")

WebElement pending;

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[6]/a")

WebElement formclick;

@FindBy(xpath = " /html/body/div[3]/aside/div/div[2]/nav/ul/li[7]/a/p\r\n" + "")

WebElement Packetclick;

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

WebElement Addtocart;

@FindBy(xpath = "//button[@title='Add to cart']")

WebElement PacketAddtocart;

@FindBy(xpath = "//\*[@id=\"example-select-all\"]")

WebElement boxclick;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[1]/div/div/div[2]/div/div/div[1]/a")

WebElement Profilestep;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[1]/div/div/div[2]/div/div/div[1]/a")

WebElement Profilesteplink;

@FindBy(xpath = "//a[@class='remove\_me' and @rel='1']")

WebElement Closebutton;

@FindBy(xpath = "/html/body/div[12]/div/div/div[3]/button[2]/span")

WebElement YesOk;

@FindBy(xpath = "/html/body/div[5]/div/div/div[3]/button[2]/span")

WebElement YesOk1;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\"]")

WebElement Submitp;

@FindBy(xpath = "//\*[@id=\"statement\"]")

WebElement sectioncompleted;

@FindBy(xpath = "/html/body/div[3]/nav/ul[1]/li[2]/span/img")

WebElement Menu;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div[1]/div/div[2]/button/span/i")

WebElement Archive;

@FindBy(xpath = "//input[@id=\"checkAll\"]")

WebElement ArchiveAll;

@FindBy(xpath = "//\*[@id=\"documents\"]/tbody/tr[2]/td[1]//input")

WebElement Archivechk;

@FindBy(xpath = "//\*[@id=\"documents\"]/tbody/tr[4]/td[1]//input")

WebElement Archivechk1;

@FindBy(xpath = "//\*[@id=\"documents\_filter\"]/label/input")

WebElement SearchArchive;

@FindBy(xpath = "/html/body/div[3]/div[4]/div/div/div[2]/a/span")

WebElement Restore;

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a/p")

WebElement Profileclick;

@FindBy(xpath = " /html/body/div[3]/div[1]/section[2]/div/div/div[2]/div/div/div/form/div/div[2]/ul/li[1]/label/a\r\n"

+ "")

WebElement Eductionlink;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div[2]/div/div/div/form/div/div[2]/ul/li[2]/label/a")

WebElement Internship;

@FindBy(xpath = "//\*[@id=\"userSignOutId\"]/span")

WebElement Signout;

@FindBy(xpath = "//\*[@id=\"activities\"]/tbody/tr")

List<WebElement> listofactivity;

@FindBy(xpath = "//\*[@id=\"activities\"]/tbody/tr[" + "]")

WebElement Lastactivityelement;

@FindBy(xpath = "//\*[@id=\"passwordChangeAlertModal\"]/div/div/div[2]/button")

WebElement OKpopup;

@FindBy(xpath = "//\*[@id=\"email2\"]")

WebElement PrimarayEmail;

@FindBy(xpath = "//input[@type='radio' and @value='email2']")

WebElement MakePrimarayEmail;

@FindBy(xpath = "/html/body/div[5]/div/div/div[3]/button[2]")

WebElement YesEmail;

@FindBy(xpath = " /html/body/div[3]/div[1]/section[1]/div/div/div[2]/div/div/div[2]/div/a/button\r\n" + "")

WebElement Boardingbutton;

@FindBy(xpath = "//\*[@id=\"passwordChangeAlertModal\"]/div/div/div[2]/button\r\n" + "")

WebElement OK;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div[2]/div/div/div/form/div/p/a\r\n" + "")

WebElement Clear;

public void personalDetails() throws InterruptedException {

waitInSeconds(10);

click\_Element\_Using\_JS(profile);

click\_Element\_Using\_JS(personalDetails);

wait\_in\_seconds(5);

dob.clear();

dob.sendKeys("11/11/1990");

SSN.sendKeys("405633499");

zipCode.sendKeys("441601");

waitInSeconds(5);

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

wait\_in\_seconds(3);

// scrollDownToElement(saveAndContinue);

Select state = new Select(stateList);

state.selectByVisibleText("Arizona");

waitInSeconds(5);

city.clear();

city.sendKeys("Thanda");

Select indusrty = new Select(industryType);

indusrty.getOptions().get(16).click();

Select category = new Select(categoryType);

//category.getOptions().get(2).click();

//specialitiesSelect.click();

waitInSeconds(10);

//wait\_for\_element\_present(selectFromDropDown);

//selectFromDropDown.click();

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

//Clear.click();

int list = listofactivity.size();

WebElement inputField = driver.findElement(By.xpath("//\*[@id=\"activities\"]/tbody/tr[" + (list) + "]"));

scrollToElement(inputField);

}

public void Profilepicture() throws InterruptedException, AWTException {

waitInSeconds(10);

click\_Element\_Using\_JS(profile);

click\_Element\_Using\_JS(personalDetails);

wait\_in\_seconds(5);

//click\_Element\_Using\_JS(Uploadimage);

WebElement upload = driver.findElement(By.xpath("//\*[@id=\"profile\_img\"]"));

Actions act = new Actions(driver);

act.moveToElement(upload).click().perform();

wait\_in\_seconds(5);

file\_upload("C:\\Users\\Admin\\Desktop\\Tesst Data\\surecafetest.jpg");

}

public void Profilenaviagtion() throws InterruptedException, AWTException {

wait\_in\_seconds(10);

//stepComplete.click();

//below code for List of activity count is

wait\_in\_seconds(10);

wait\_for\_page\_load(10);

int list = listofactivity.size();

WebElement inputField = driver.findElement(By.xpath("//\*[@id=\"activities\"]/tbody/tr[" + (list) + "]"));

System.out.println("List of activity count is " + list);

scrollToElement(inputField);

}

@FindBy(xpath = "//\*[@id=\"flashMsgTxt\"]")

WebElement alertElement;

public void RestoreValidation() throws InterruptedException, AWTException, TimeoutException {

waitInSeconds(10);

driver.get("https://dev.kredsafe.net/user/education");

wait\_for\_page\_load(10);

wait\_for\_element\_present(Archive);

wait\_in\_seconds(10);

click\_Element\_Using\_JS(Archive);

wait\_in\_seconds(10);

String errormessage = alertElement.getText();

System.out.print("The error message is " + errormessage);

if (errormessage.equals("No archived record found.")) {

System.out.print("The actual error message for validation is " + errormessage);

//AssertJUnit.assertEquals(alertElement, errormessage);

}

}

public void RestoreIntership() throws InterruptedException, AWTException {

waitInSeconds(10);

wait\_for\_element\_present(Profilesteplink);

Profilesteplink.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(Closebutton);

Closebutton.click();

YesOk.click();

wait\_in\_seconds(10);

scrollDownToElement(Submitp);

click\_Element\_Using\_JS(Submitp);

wait\_in\_seconds(10);

wait\_for\_element\_present(Closebutton);

Closebutton.click();

wait\_in\_seconds(5);

YesOk1.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(Archive);

click\_Element\_Using\_JS(Archive);

wait\_in\_seconds(10);

wait\_in\_seconds(10);

wait\_for\_element\_present(Archivechk);

click\_Element\_Using\_JS(Archivechk);

wait\_in\_seconds(10);

wait\_for\_element\_present(Archivechk1);

click\_Element\_Using\_JS(Archivechk1);

scrollDownToElement(Restore);

wait\_in\_seconds(10);

click\_Element\_Using\_JS(Restore);

wait\_in\_seconds(10);

wait\_for\_element\_present(Archive);

click\_Element\_Using\_JS(Archive);

}

public void Steptonecompleted() throws InterruptedException, AWTException {

// Code for profile navigation from step tp completed page.

waitInSeconds(10);

wait\_for\_element\_present(Menu);

Menu.click();

wait\_for\_element\_present(Profileclick);

Profileclick.click();

wait\_in\_seconds(10);

scrollToElement(sectioncompleted);

wait\_for\_element\_present(sectioncompleted);

click\_Element\_Using\_JS(sectioncompleted);

// Code for profile navigation from step tp completed page.

wait\_for\_page\_load(10);

wait\_for\_element\_present(Internship);

click\_Element\_Using\_JS(Internship);

wait\_for\_page\_load(10);

wait\_for\_element\_present(Closebutton);

Closebutton.click();

YesOk1.click();

wait\_in\_seconds(10);

scrollDownToElement(Submitp);

click\_Element\_Using\_JS(Submitp);

wait\_in\_seconds(10);

wait\_for\_element\_present(Closebutton);

Closebutton.click();

YesOk.click();

wait\_in\_seconds(10);

scrollDownToElement(Submitp);

click\_Element\_Using\_JS(Submitp);

wait\_in\_seconds(10);

Clear.click();

}

public void PrimaryEmail() throws InterruptedException, AWTException {

wait\_for\_page\_load(10);

wait\_for\_element\_present(profile);

click\_Element\_Using\_JS(profile);

wait\_in\_seconds(10);

click\_Element\_Using\_JS(personalDetails);

wait\_for\_element\_present(PrimarayEmail);

scrollToElement(PrimarayEmail);

PrimarayEmail.sendKeys("ts7876001@yopmail.com");

wait\_for\_element\_present(MakePrimarayEmail);

click\_Element\_Using\_JS(MakePrimarayEmail);

wait\_in\_seconds(10);

wait\_for\_element\_present(YesEmail);

click\_Element\_Using\_JS(YesEmail);

}

public void Changepassword() throws InterruptedException, AWTException {

Profilenameclick.click();

//wait\_for\_element\_present(Changepasswordlink);

//click\_Element\_Using\_JS(Changepasswordlink);

Changepasswordlink.click();

waitInSeconds(10);

oldpw.sendKeys("Nilesh@2025");

newpw.sendKeys("Nilesh@2030");

newpwc.sendKeys("Nilesh@2030");

waitInSeconds(10);

pwsubmit.click();

//click\_Element\_Using\_JS(pwsubmit);

}

@FindBy(xpath = "//\*[@id=\"facility\_name\"]")

WebElement facilityname;

@FindBy(xpath = "//\*[@id=\"contact\_person\"]")

WebElement contactperson;

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

WebElement emailr;

@FindBy(xpath = "//\*[@id=\"phone\"]")

WebElement phoiner;

@FindBy(xpath = "//textarea[@name=\"address\"]")

WebElement Address;

@FindBy(xpath = "//\*[@id=\"submit\_form\"]")

WebElement frsubmit;

@FindBy(xpath = "//\*[@id=\"template\_ducument\"]")

WebElement uplaodform;

public void formrequest() throws InterruptedException, AWTException {

wait\_for\_page\_load(10);

//click\_Element\_Using\_JS(profile);

//wait\_in\_seconds(10);

driver.navigate().to("https://dev.kredsafe.net/user/es\_forms/request-form");

WebElement upload = driver.findElement(By.xpath("//\*[@id=\"template\_ducument\"]"));

Actions act = new Actions(driver);

act.moveToElement(upload).click().perform();

wait\_in\_seconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\Dice\_Resume\_CV\_Akash\_Patel.pdf");

wait\_in\_seconds(5);

facilityname.sendKeys("Robert");

wait\_in\_seconds(5);

contactperson.sendKeys("Damc");

wait\_in\_seconds(5);

emailr.sendKeys("formtest@yopmail.com");

wait\_in\_seconds(5);

phoiner.sendKeys("1233217654");

wait\_in\_seconds(5);

Address.sendKeys("New Yors US");

wait\_in\_seconds(5);

scrollDownToElement(frsubmit);

frsubmit.click();

wait\_in\_seconds(5);

scrollDown();

}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[8]/a/p")

WebElement Subscription;

@FindBy(xpath = "//\*[@id=\"subBoxInfoClose\"]/section/div/div/div[3]/div/div/div/div/div/ul/li[3]/a")

WebElement calcelltion;

@FindBy(xpath = "//i[@class='fa fa-bell-o' and @title='follow up']")

WebElement bellicon;

@FindBy(xpath = "//textarea[@id='description']")

WebElement folloupmsg;

@FindBy(xpath = "//\*[@id=\"submitFollowUpForm\"]")

WebElement submitmsg;

@FindBy(xpath = "//\*[@id=\"cancellationRequest\"]/tbody/tr/td[6]/a[2]/i")

WebElement comm;

public void Followup() throws InterruptedException, AWTException {

wait\_for\_page\_load(10);

//click\_Element\_Using\_JS(profile);

//wait\_in\_seconds(10);

driver.navigate().to("https://dev.kredsafe.net/user/subscription/dashboard");

//wait\_for\_element\_present(Subscription);

// Subscription.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(calcelltion);

calcelltion.click();

waitInSeconds(5);

scrollDown();

scrollDown();

scrollDown();

scrollDown();

scrollDown();

//wait\_for\_element\_present(bellicon);

WebDriverWait wait = new WebDriverWait(driver, 10);

WebElement bellIcon = wait.until(ExpectedConditions.elementToBeClickable(By.xpath("//i[@title='follow up']")));

bellIcon.click();

wait\_in\_seconds(5);

folloupmsg.sendKeys("Reminder Message new");

wait\_in\_seconds(5);

wait\_for\_element\_present(submitmsg);

submitmsg.click();

wait\_in\_seconds(10);

scrollDown();

scrollDown();

scrollDown();

scrollDown();

scrollDown();

wait\_for\_element\_present(comm);

comm.click();

}

public void LoginLogout() throws InterruptedException, AWTException {

wait\_for\_page\_load(10);

Profilenameclick.click();

waitInSeconds(10);

wait\_for\_element\_present(Signout);

click\_Element\_Using\_JS(Signout);

}

public void Forgotpassword() throws InterruptedException, AWTException {

scrollDown();

click\_Element\_Using\_JS(forgotpassword);

wait\_in\_seconds(5);

wait\_for\_element\_present(submitemail);

emailidf.sendKeys("swapj@yopmail.com");

waitInSeconds(5);

submitemail.click();

}

public void Formpayment() throws InterruptedException, AWTException {

wait\_in\_seconds(10);

menuclick.click();

waitInSeconds(10);

Packetclick.click();

// driver.navigate().refresh();

waitInSeconds(10);

boxclick.click();

// formclick.click();

wait\_in\_seconds(10);

// boxclick.click();

scrollDownToElement(PacketAddtocart);

wait\_in\_seconds(10);

click\_Element\_Using\_JS(PacketAddtocart);

// PacketAddtocart.click();

}

@SuppressWarnings("deprecation")

public void Formpaymentvalidation() throws InterruptedException, AWTException {

wait\_in\_seconds(10);

menuclick.click();

waitInSeconds(10);

Packetclick.click();

String errormessage = errormsg.getText();

System.out.print("The error message is " + errormessage);

if (errormessage.equals("Please complete your profile before accessing the forms and packets area.")) {

System.out.print("The actual error message for validation is " + errormessage);

Assert.assertEquals("Please complete your profile before accessing the forms and packets area.",

errormessage);

}

}

@FindBy(xpath = "//li[contains(@class,'pendingItemLink')]//a")

List<WebElement> pendinglinkactivity;

@FindBy(xpath = "//\*[@id=\"pendingContUl\"]/li[5]/a")

List<WebElement> pendinglinkactivity5;

@FindBy(xpath = " //\*[@id=\"pendingContUl\"]/li[3]/a\r\n" + "")

List<WebElement> pendinglinkactivity3;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div[1]/div/div[1]/h1")

WebElement workexppage;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div/form/div/div[1]/div[2]/div/div/div/div[3]/ul/li[1]/a")

WebElement pendingtab1;

@FindBy(xpath = "//\*[@id=\"pendingActivityTab\"]")

WebElement pendingtab;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div[1]/div/div[1]/h1")

WebElement edupage;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div[1]/div/div[1]/h1")

WebElement intership;

@FindBy(xpath = " /html/body/div[3]/aside/div/div[2]/nav/ul/li[1]/a\r\n" + "")

WebElement Hoemclick;

@FindBy(xpath = "//h1[@class='card-title']")

WebElement sectionTitle;

@FindBy(xpath = "//\*[@id=\"pendingContUl\"]/li[9]/a")

WebElement PacketTitle;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div/form/div/div[1]/div[2]/div/div/div/div[3]/ul/li[8]/a")

WebElement formTitle;

@FindBy(xpath = "//h1[@class='card-title cardTitleRes']")

WebElement cardtitleSES;

@FindBy(xpath = " /html/body/div[3]/nav/ul[1]/li[2]/span/img\r\n" + "")

WebElement menuclick1;

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a/p")

WebElement profileclick;

@FindBy(xpath = "//\*[@id=\"work\_experiences\"]")

WebElement work;

@FindBy(xpath = "//\*[@id=\"company\_name\"]")

WebElement cname;

@FindBy(xpath = "//\*[@id=\"job\_title\"]")

WebElement jname;

@FindBy(xpath = "//\*[@id=\"city\"]")

WebElement cityname;

@FindBy(xpath = " /html/body/div[3]/div[1]/section[1]/div/div/div[2]/div/div/div[1]/a\r\n" + "")

WebElement link;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\"]/span")

WebElement submit;

@FindBy(xpath = "//\*[@id=\"institution\_name\"]")

WebElement intname;

@FindBy(xpath = "//\*[@id=\"subjectStudy\"]")

WebElement subject;

@FindBy(xpath = "//\*[@id=\"city\"]")

WebElement citye;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[1]/div/input")

WebElement intsname;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[2]/div/input")

WebElement sstudy;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[5]/div/input")

WebElement cityp;

@FindBy(xpath = "//\*[@id=\"job\_title\"]")

WebElement tjob;

@FindBy(xpath = "//\*[@id=\"employer\"]")

WebElement iemr;

@FindBy(xpath = "//\*[@id=\"city\"]")

WebElement icity;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[1]/div/input")

WebElement fjobt;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[2]/div/input")

WebElement fsemp;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[5]/div/input")

WebElement fscity;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[1]/div/input")

WebElement rsjobt;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[2]/div/input")

WebElement rssemp;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[5]/div/input")

WebElement rscity;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[1]/div/input")

WebElement certificate;

@FindBy(xpath = "//\*[@id=\"certification\_date\"]")

WebElement certificates;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[1]/div/input")

WebElement certificatename;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[1]/div/input")

WebElement Boardcertificate;

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[2]/div/input")

WebElement Authrity;

@FindBy(xpath = "//\*[@id=\"certification\_date\_0\"]")

WebElement cdate;

@FindBy(xpath = "//\*[@id=\"expiration\_date\_0\"]")

WebElement Edate;

@FindBy(xpath = "/html/body/div[3]/div[1]/section[2]/div/div/div[2]/div/div/div/form/div/p/a")

WebElement clickhere;

@FindBy(xpath = "//\*[@id=\"first\_name\"]")

WebElement fname;

@FindBy(xpath = "//\*[@id=\"last\_name\"]")

WebElement lname;

@FindBy(xpath = "//\*[@id=\"datepicker\"]")

WebElement DOB;

@FindBy(xpath = "//\*[@id=\"ssn-field\"]")

WebElement SSNp;

@FindBy(xpath = " //\*[@id=\"phone\"]")

WebElement mobile;

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

WebElement Email;

@FindBy(xpath = "//\*[@id=\"city\"]")

WebElement Cityp;

public void Profileflow() throws InterruptedException, AWTException {

wait\_in\_seconds(10);

link.click();

menuclick.click();

profileclick.click();

wait\_in\_seconds(10);

fname.sendKeys("Harry");

lname.sendKeys("Sam");

//DOB.sendKeys("07/23/1992");

//SSNp.sendKeys("506634417");

//mobile.sendKeys("4567765498");

//Email.sendKeys("harrysam@yopmail.com");

Cityp.sendKeys("NY");

wait\_for\_element\_present(submit);

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

wait\_in\_seconds(10);

cname.sendKeys("Test");

rssemp.sendKeys("Test");

cityname.sendKeys("NY");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

wait\_for\_element\_present(intname);

intname.sendKeys("Test");

subject.sendKeys("Test");

citye.sendKeys("NY");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

// Professional

wait\_in\_seconds(10);

intsname.sendKeys("Test");

sstudy.sendKeys("Test");

cityp.sendKeys("NY");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

// internship

wait\_in\_seconds(10);

tjob.sendKeys("Test");

iemr.sendKeys("Test");

icity.sendKeys("NY");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

// resisdency

wait\_in\_seconds(10);

rsjobt.sendKeys("Test");

rssemp.sendKeys("Test");

rscity.sendKeys("NY");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

// fellowship

wait\_in\_seconds(10);

fjobt.sendKeys("Test");

fsemp.sendKeys("Test");

fscity.sendKeys("NY");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

// certificate

wait\_in\_seconds(10);

certificate.sendKeys("Test");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

// Boardcertificate

wait\_in\_seconds(10);

Boardcertificate.sendKeys("Test");

Authrity.sendKeys("Test");

cdate.sendKeys("04/04/2025");

//wait\_for\_element\_present(Edate);

//Edate.sendKeys("04/30/2025");

wait\_in\_seconds(10);

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

wait\_in\_seconds(10);

wait\_for\_element\_present(clickhere);

click\_Element\_Using\_JS(clickhere);

}

@SuppressWarnings("deprecation")

public void Pendingactivity() throws InterruptedException, AWTException {

wait\_in\_seconds(5);

wait\_for\_element\_present(pendingtab);

click\_Element\_Using\_JS(pendingtab);

wait\_in\_seconds(5);

for (WebElement element : pendinglinkactivity) {

String activityname = element.getText();

System.out.print("The Activity Name is " + activityname);

if (activityname.contains("Overview")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

driver.close();

// switch to parent

driver.switchTo().window(tab.get(0));

continue;

} else if (activityname.contains("Work Experience")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(sectionTitle);

String sectionname = sectionTitle.getText();

System.out.print("The section name is " + activityname);

waitInSeconds(5);

Assert.assertEquals(activityname, sectionname);

System.out.print("The validationsection name is " + sectionname);

driver.close();

// switch to parent

driver.switchTo().window(tab.get(0));

// continue;

} else if (activityname.contains("Education")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(sectionTitle);

String sectionname = sectionTitle.getText();

System.out.print("The section name is " + activityname);

waitInSeconds(5);

Assert.assertEquals(activityname, sectionname);

System.out.print("The validationsection name is " + sectionname);

driver.close();

// switch to parent

driver.switchTo().window(tab.get(0));

// continue;

}

else if (activityname.contains("Internship")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(sectionTitle);

String sectionname = sectionTitle.getText();

System.out.print("The section name is " + activityname);

waitInSeconds(5);

Assert.assertEquals(activityname, sectionname);

System.out.print("The validationsection name is " + sectionname);

driver.close();

// switch to parent

driver.switchTo().window(tab.get(0));

// continue;

} else if (activityname.contains("Residency")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(sectionTitle);

String sectionname = sectionTitle.getText();

System.out.print("The section name is " + activityname);

waitInSeconds(5);

Assert.assertEquals(activityname, sectionname);

System.out.print("The validationsection name is " + sectionname);

driver.close();

// switch to parent

driver.switchTo().window(tab.get(0));

// continue;

} else if (activityname.contains("Fellowship")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(sectionTitle);

String sectionname = sectionTitle.getText();

System.out.print("The section name is " + activityname);

waitInSeconds(5);

Assert.assertEquals(activityname, sectionname);

System.out.print("The validationsection name is " + sectionname);

driver.close();

// switch to parent

driver.switchTo().window(tab.get(0));

// continue;

} else if (activityname.contains("State Licenses or Certificates")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(cardtitleSES);

String sectionname = cardtitleSES.getText();

System.out.print("The section name is " + activityname);

waitInSeconds(5);

Assert.assertEquals(activityname, sectionname);

System.out.print("The validationsection name is " + sectionname);

driver.close();

// switch to parent

driver.switchTo().window(tab.get(0));

// continue;

} else if (activityname.contains("Board Certification / Eligibility")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(sectionTitle);

String sectionname = sectionTitle.getText();

System.out.print("The section name is " + activityname);

waitInSeconds(5);

Assert.assertEquals(activityname, sectionname);

System.out.print("The validationsection name is " + sectionname);

driver.close(); // switch to parent

driver.switchTo().window(tab.get(0));

// continue;

} else if (activityname.contains("Pending Forms")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(errormsg);

String errormessage = errormsg.getText();

System.out.print("The error message is " + errormessage);

if (errormessage.equals("Please complete your profile before accessing the forms and packets area.")) {

System.out.print("The actual error message for validation is " + errormessage);

Assert.assertEquals("Please complete your profile before accessing the forms and packets area.",

errormessage);

// switch to parent

driver.close();

driver.switchTo().window(tab.get(0));

// continue;

}

} else if (activityname.contains("Pending Packets")) {

String keyPress = Keys.CONTROL.toString() + Keys.SHIFT.toString() + Keys.ENTER.toString();

element.sendKeys(keyPress);

ArrayList<String> tab = new ArrayList<>(driver.getWindowHandles());

driver.switchTo().window(tab.get(1));

wait\_for\_element\_present(errormsg);

String pendingpacket = errormsg.getText();

System.out.print("The error message is " + pendingpacket);

if (pendingpacket.equals("Please complete your profile before accessing the forms and packets area.")) {

System.out.print("The actual error message for validation is " + pendingpacket);

Assert.assertEquals("Please complete your profile before accessing the forms and packets area.",

pendingpacket);

// switch to parent

driver.close();

driver.switchTo().window(tab.get(0));

// continue;

}

} else {

System.out.print("No pending activities");

// continue;

}

}

}

@FindBy(xpath = "//input[@id='company\_name']")

WebElement companyName;

@FindBy(xpath = "//input[@id='job\_title']")

WebElement jonTitle;

@FindBy(xpath = "//select[@id='state-list\_0']")

WebElement stateList0;

@FindBy(xpath = "//input[@id='currently\_working\_status']")

WebElement yesRadioButton;

@FindBy(xpath = "//select[@id='start\_month\_0']")

WebElement startMonth;

@FindBy(xpath = "//select[@id='we\_start\_year\_identity\_0']")

WebElement startYear;

public void workExperience() throws InterruptedException {

wait\_in\_seconds(3);

companyName.clear();

companyName.sendKeys("TIU");

jonTitle.clear();

jonTitle.sendKeys("QE");

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

city.clear();

city.sendKeys("Thanda");

wait\_in\_seconds(3);

yesRadioButton.click();

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

}

@FindBy(xpath = "//input[@id='institution\_name']")

WebElement institution\_name;

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

WebElement subjectStudy;

@FindBy(xpath = "//input[@name='education[0][currently\_studying]' and@ value = '1']")

WebElement yesRadioButtonEducation;

public void education() throws InterruptedException {

institution\_name.sendKeys("High School");

subjectStudy.sendKeys("EEE");

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

city.clear();

city.sendKeys("Thanda");

wait\_in\_seconds(10);

scrollDownToElement(yesRadioButtonEducation);

yesRadioButtonEducation.click();

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

}

@FindBy(xpath = "//input[@name='education[0][institution\_name]']")

WebElement institutionName;

@FindBy(xpath = "//input[@name='education[0][stream]']")

WebElement subjectofStudy;

@FindBy(xpath = "//input[@name='education[0][city]']")

WebElement city1;

public void professionalEducation() throws InterruptedException {

institutionName.sendKeys("TIU");

subjectofStudy.sendKeys("sos");

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

city1.sendKeys("Thanda");

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

}

@FindBy(xpath = "//input[@name='internship[0][job\_title]']")

WebElement jobTtitle;

@FindBy(xpath = "//input[@name='internship[0][employer]']")

WebElement employer;

@FindBy(xpath = "//input[@name='internship[0][city]']")

WebElement internshipCity;

@FindBy(xpath = "//select[@id='end\_month\_0']")

WebElement endMonth;

@FindBy(xpath = "//select[@id='we\_end\_year\_identity\_0']")

WebElement endYear;

public void internship() throws InterruptedException {

jobTtitle.sendKeys("Lead");

employer.sendKeys("TIU");

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

internshipCity.sendKeys("Thanda");

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

Select endmonth = new Select(endMonth);

endmonth.selectByVisibleText("Jan");

Select endyear = new Select(endYear);

endyear.selectByVisibleText("2019");

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

}

@FindBy(xpath = "//input[@name='residency[0][job\_title]']")

WebElement jobTtitleResidence;

@FindBy(xpath = "//input[@name='residency[0][employer]']")

WebElement employerResidence;

@FindBy(xpath = "//input[@name='residency[0][city]']")

WebElement residenceCity;

public void residency() throws InterruptedException {

jobTtitleResidence.sendKeys("QWERTY");

employerResidence.sendKeys("ASD");

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

residenceCity.sendKeys("Thanda");

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

Select endmonth = new Select(endMonth);

endmonth.selectByVisibleText("Jan");

Select endyear = new Select(endYear);

endyear.selectByVisibleText("2019");

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

}

@FindBy(xpath = "//input[@name='fellowship[0][job\_title]']")

WebElement jobTtitlefellowship;

@FindBy(xpath = "//input[@name='fellowship[0][employer]']")

WebElement employerfellowship;

@FindBy(xpath = "//input[@name='fellowship[0][city]']")

WebElement fellowshipCity;

public void fellowship() throws InterruptedException {

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

}

@FindBy(xpath = "//input[@name='certificate[0][certification\_name]']")

WebElement certification\_name;

@FindBy(xpath = "//input[@name='certificate[0][certification\_date]']")

WebElement certification\_date;

@FindBy(xpath = "//select[@id='authority\_<?php echo e(0); ?>']")

WebElement stateAuthority;

public void licensesOrCertificates() throws InterruptedException {

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(10);

wait\_for\_page\_load(10);

scrollDown();

saveAndContinue1.click();

}

@FindBy(xpath = "//a[contains(@href, 'https://dev.kredsafe.net/user/documents')]")

WebElement documentsSection1;

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

WebElement uploadFile;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '10']")

WebElement documentTypeother;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '1']")

WebElement documentTypeDEA;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '2']")

WebElement documentTypeResidency;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '3']")

WebElement documentTypeStateControlledSubstance;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '4']")

WebElement documentTypeMedicalLicence;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '5']")

WebElement documentTypeMedicalDegree;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '6']")

WebElement documentTypeIntershipcertificate;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '7']")

WebElement documentTypeFellowshipcertificate;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '8']")

WebElement documentTypeECFMG;

@FindBy(xpath = "//input[@name='document\_type' and@ value = '9']")

WebElement documentTypeBoardcertificate;

@FindBy(xpath = "//button[@id='submitComment']")

WebElement docSubmitcomment;

@FindBy(xpath = "//button[@id='docSubmit']")

WebElement docSubmit;

@FindBy(xpath = "//a[contains(@href, 'https://dev.kredsafe.net/uploadedFile/MTM5MQ==/112.pdf') and @title = 'View document']")

WebElement viewDocument;

@FindBy(xpath = "//a[contains(@href, '#') and @title = 'View/Add Note']")

WebElement addNote;

public void documentsSection() throws InterruptedException, AWTException {

waitInSeconds(10);

click\_Element\_Using\_JS(documentsSection1);

wait\_in\_seconds(5);

uploadFile.click();

waitInSeconds(10);

wait\_for\_element\_present(documentTypeDEA);

click\_Element\_Using\_JS(documentTypeDEA); // documentTypeECFMG.click();

wait\_in\_seconds(5);

WebElement upload = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act = new Actions(driver);

act.moveToElement(upload).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\Anil\_Kumar\_Salesforce\_Admin.docx");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

documentTypeResidency.click();

wait\_in\_seconds(5);

WebElement upload1 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act1 = new Actions(driver);

act1.moveToElement(upload1).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\DerekSmith(1).doc");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

// wait\_for\_element\_present(documentTypeStateControlledSubstance);

click\_Element\_Using\_JS(documentTypeStateControlledSubstance);

// documentTypeStateControlledSubstance.click();

wait\_in\_seconds(5);

WebElement upload2 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act2 = new Actions(driver);

act2.moveToElement(upload2).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\Dice\_Resume\_CV\_Akash\_Patel.pdf");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

documentTypeMedicalLicence.click();

wait\_in\_seconds(5);

WebElement upload3 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act3 = new Actions(driver);

act3.moveToElement(upload3).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\DerekSmith.odt");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

documentTypeMedicalDegree.click();

wait\_in\_seconds(5);

WebElement upload4 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act4 = new Actions(driver);

act4.moveToElement(upload4).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\surecafetest.jpeg");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

documentTypeIntershipcertificate.click();

wait\_in\_seconds(5);

WebElement upload5 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act5 = new Actions(driver);

act5.moveToElement(upload5).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\surecafetest.jpg");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

documentTypeFellowshipcertificate.click();

wait\_in\_seconds(5);

WebElement upload6 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act6 = new Actions(driver);

act6.moveToElement(upload6).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\surecafetest.png");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

documentTypeECFMG.click();

wait\_in\_seconds(5);

WebElement upload7 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act7 = new Actions(driver);

act7.moveToElement(upload7).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\surecafetest.gif");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

documentTypeBoardcertificate.click();

wait\_in\_seconds(5);

WebElement upload8 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act8 = new Actions(driver);

act8.moveToElement(upload8).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\PavanKalyan-Data Engineer.docx");

waitInSeconds(10);

docSubmit.click();

waitInSeconds(10);

click\_Element\_Using\_JS(uploadFile);

waitInSeconds(10);

documentTypeother.click();

wait\_in\_seconds(5);

WebElement upload9 = driver.findElement(By.xpath("//input[@class='fileUpload inputOtherBrd']"));

Actions act9 = new Actions(driver);

act9.moveToElement(upload9).click().perform();

waitInSeconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\Harideep.docx");

waitInSeconds(10);

docSubmit.click();

}

@FindBy(xpath = "//a[contains(@href, 'https://dev.kredsafe.net/user/notes')]")

WebElement notesSection;

@FindBy(xpath = "//\*[@id=\"noteList\"]/tbody/tr/td[6]/a/i")

WebElement notesSection1;

@FindBy(xpath = "//a[contains(@href, 'https://dev.kredsafe.net/user/comments/view/User/MTAyNw==') and @title = 'View Note']")

WebElement editMessage;

@FindBy(xpath = "//\*[@id=\"message\_info\"]")

WebElement enterMessages;

@FindBy(xpath = "//\*[@id=\"submitComment\"]")

WebElement Submitnote;

public void notesSection() throws InterruptedException, AWTException {

waitInSeconds(10);

click\_Element\_Using\_JS(notesSection);

wait\_for\_element\_present(notesSection1);

click\_Element\_Using\_JS(notesSection1);

// notesSection1.click();

wait\_for\_element\_present(enterMessages);

scrollDownToElement(enterMessages);

enterMessages.sendKeys("Test Notes");

wait\_in\_seconds(10);

scrollToElement(Submitnote);

click\_Element\_Using\_JS(Submitnote);

// Submitnote.click();

}

public void verifyCompleteSection() throws InterruptedException, AWTException {

String get\_currenturl = driver.getCurrentUrl();

System.out.println("Current URL is:" + get\_currenturl);

System.out.println("URL matching --> Part executed");

for (int i = 0; i <= 7; i++) {

if (!get\_currenturl.contains("https://dev.kredsafe.net/user/statement")) {

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

}

}

wait\_for\_element\_present(clickHere);

wait\_in\_seconds(10);

clickHere.click();

wait\_in\_seconds(10);

Addtocart.click();

}

public void Formpayment1() {

menuclick.click();

formclick.click();

}

public void Changepassword(String oldpassword, String newpassword, String confirmpassword, String errorpassword)

throws InterruptedException {

Profilenameclick.click();

click\_Element\_Using\_JS(Changepasswordlink);

//Changepasswordlink.click();

oldpw.sendKeys(oldpassword);

newpw.sendKeys(newpassword);

newpwc.sendKeys(confirmpassword);

waitInSeconds(10);

pwsubmit.click();

wait\_in\_seconds(10);

for (WebElement e : incorrectpassword) {

String errormessage = e.getText();

System.out.print("The error message is " + errormessage);

if (errormessage.equals("Password does not match with the required criteria.")) {

System.out.print("The actual error message for validation is " + errormessage);

AssertJUnit.assertEquals(errorpassword, errormessage);

}

if (errormessage.equals("Incorrect password.")) {

System.out.print("The actual error message for validation is " + errormessage);

AssertJUnit.assertEquals(errorpassword, errormessage);

}

if (errormessage.equals("Incorrect confirm password.")) {

System.out.print("The actual error message for validation is " + errormessage);

AssertJUnit.assertEquals(errorpassword, errormessage);

}

}

}

public void clientSignInn(String username, String password, String errorpassword) throws InterruptedException {

waitInSeconds(3);

try {

if (loginLink.isDisplayed()) {

loginLink.click();

}

} catch (NoSuchElementException e) {

}

waitInSeconds(3);

log.info("Clicking on accept button");

try {

if (acceptButton.isDisplayed()) {

acceptButton.click();

}

} catch (NoSuchElementException e) {

}

waitInSeconds(3);

loginField.sendKeys(username);

passwordField.sendKeys(password);

wait\_in\_seconds(10);

scrollDownToElement(signInButton);

wait\_in\_seconds(5);

signInButton.click();

wait\_in\_seconds(10);

for (WebElement e : incorrectpassword) {

String errormessage = e.getText();

System.out.print("The error message is " + errormessage);

if (errormessage.equals("E-mail address or password is incorrect.")) {

System.out.print("The actual error message for validation is " + errormessage);

AssertJUnit.assertEquals(errorpassword, errormessage);

}

String get\_currenturl = driver.getCurrentUrl();

if (get\_currenturl.contains("https://dev.kredsafe.net/accept-otp")) {

System.out.print("The url is " + get\_currenturl);

AssertJUnit.assertEquals(errorpassword, get\_currenturl);

}

}

}

@FindBy(xpath = "//\*[@id=\"notification-bell\"]/a/i")

WebElement belllicon;

@FindBy(xpath = "//\*[@id=\"notification-bell\"]/div/a[1]")

WebElement packet;

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[6]/a/p")

WebElement form;

@SuppressWarnings("deprecation")

public void Verifypaymentnotification() throws InterruptedException {

//belllicon.click();

click\_Element\_Using\_JS(belllicon);

waitInSeconds(10);

//wait\_for\_element\_present(form);

click\_Element\_Using\_JS(form);

wait\_for\_element\_present(errormsg);

String errormessage = errormsg.getText();

System.out.print("The error message is " + errormessage);

if (errormessage.equals("Please complete your profile before accessing the forms and packets area.")) {

System.out.print("The actual error message for validation is " + errormessage);

Assert.assertEquals("Please complete your profile before accessing the forms and packets area.",

errormessage);

}

}

@FindBy(xpath = "//\*[@class=\"stepComplete\"]")

WebElement stepComplete;

@FindBy(xpath = "//\*[@class=\"card-title\"]")

WebElement cardTitle;

@FindBy(xpath = "//span[@style='display: block;']")

List<WebElement> mandatoryFields;

@FindBy(xpath = "//select[@class='autoupdate country']")

WebElement peContryDropdown;

@FindBy(xpath = "//\*[@class=\"card-title cardTitleRes\"]")

WebElement certResCardTitle;

@FindBy(xpath = "//input[@name='certificate[0][authority]']")

WebElement certificateAuthority;

//@FindBy(xpath = "//input[@name='workexperience[0][currently\_working\_status]' and @value=\"1\"]")

@FindBy(xpath = "//\*[@id=\"currently\_working\_status\"]")

WebElement workingYesButton;

//@FindBy(xpath = "//input[@name='education[0][currently\_studying]' and @value=\"1\"]")

@FindBy(xpath = "//\*[@id=\"work\_experience\_items\"]/div[1]/div/div[2]/div/div[6]/div/label[1]/input[1]")

WebElement eduYesButton;

@FindBy(xpath = "//\*[@name=\"workexperience[0][company\_name]\"]")

WebElement cwname;

@FindBy(xpath = "//\*[@name=\"workexperience[0][job\_title]\"]")

WebElement title;

@FindBy(xpath = "//button[@id=\"id\_frm\_submit\"]")

WebElement save;

public void stepsToComplete(String SSNNumber) throws InterruptedException {

wait\_for\_element\_present(stepComplete);

String stepcompletion = stepComplete.getText();

System.out.println("The steps to complete on the Dashboard on login is " + stepcompletion);

if (stepcompletion.equals("Congratulations! You have completed your on-boarding process.")) {

System.out.println("No steps to be completed as the The step complete message is " + stepcompletion);

}

int number = 0;

if (stepcompletion.contains("step(s) to complete")) {

String stepsToComplete = stepComplete.getText();

System.out.println("The number of steps to complete is " + stepsToComplete);

String stepsNumber = stepsToComplete.substring(9);

System.out.println("Extracted steps completion message is: " + stepsNumber);

String extNumber = stepsNumber.substring(0, 1);

number = Integer.parseInt(extNumber);

System.out.println("The steps number extracted is::" + number);

waitInSeconds(5);

for (int i = 0; i < number; i++) {

try {

if (stepComplete.isDisplayed()){

System.out.println("Clicking on step complete link");

stepComplete.click();

}

} catch (NoSuchElementException e) {

}

String certCardTitle = null;

String sectionName = null;

try {

if (certResCardTitle.isDisplayed()) {

certCardTitle = certResCardTitle.getText();

}

} catch (NoSuchElementException e) {

}

try {

if (cardTitle.isDisplayed()) {

sectionName = cardTitle.getText();

}

} catch (NoSuchElementException e) {

}

if (number >= 1) {

for (int j = 0; j < number; j++) {

wait\_for\_page\_load(10);

try {

if (certResCardTitle.isDisplayed()) {

certCardTitle = certResCardTitle.getText();

System.out.println("The section title is " +certCardTitle);

}

} catch (NoSuchElementException e) {

}

try {

if (cardTitle.isDisplayed()) {

sectionName = cardTitle.getText();

System.out.println("The section name is " +sectionName);

}

} catch (NoSuchElementException e) {

}

if (sectionName != null && sectionName.contains("Personal Details")) {

wait\_for\_element\_present(submit);

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Date of Birth is required.":

System.out.println("The Date of birth is ");

dob.clear();

dob.sendKeys("11/11/1990");

break;

case "SSN is required.":

System.out.println("The SSN number");

SSNp.clear();

SSNp.sendKeys(SSNNumber);

SSNp.sendKeys(Keys.TAB);

wait\_in\_seconds(15);

break;

case "Zip Code is required.":

System.out.println("The Zip code field is required for Personal details section");

zipcode.sendKeys("07086");

break;

case "Country is required.":

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The State is required for Personal details section");

wait\_for\_element\_present(stateList);

Select state = new Select(stateList);

state.selectByValue("AL");

break;

case "The city field is required.":

System.out.println("The city field is required for Personal details section");

city.sendKeys("NY");

break;

case "Specialties are required":

System.out.println("Specialties are required for Personal details section");

Select indusrty = new Select(industryType);

indusrty.selectByValue("255");

Select category = new Select(categoryType);

category.selectByValue("Allied Health");

waitInSeconds(5);

scrollDown();

scrollToElement(specialitiesSelect);

specialitiesSelect.click();

click\_Element\_Using\_JS(selectFromDropDown);

//selectFromDropDown.click();

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

waitInSeconds(10);

break;

default:

break;

}

}

}

else if (sectionName != null && sectionName.contains("Work Experience")) {

wait\_for\_element\_present(submit);

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Company Name is required.":

System.out.println("The Company name is::United States");

cwname.sendKeys("Tiu");

title.sendKeys("Sr Doctor");

break;

case "Job Title is required.":

title.sendKeys("Sr Doctor");

break;

case "Country is required.":

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The Country name in Work experience::United States");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByValue("AL");

break;

case "City / Region is required.":

System.out.println("City / Region is required in work experience.");

city.sendKeys("NY");

wait\_in\_seconds(5);

wait\_for\_element\_present(workingYesButton);

click\_Element\_Using\_JS(workingYesButton);

//workingYesButton.click();

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

waitInSeconds(10);

break;

default:

break;

}

}

}

else if (sectionName != null && sectionName.equals("Education")) {

wait\_for\_element\_present(submit);

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Institution Name is required.":

System.out.println("The Institution Name is");

intname.sendKeys("Test");

subject.sendKeys("Test");

break;

case "Country is required.":

System.out.println("The Country name is::United States");

wait\_for\_element\_present(country);

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The State name is required i Educatio section");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.println("City / Region is required.");

city.sendKeys("NY");

wait\_in\_seconds(10);

click\_Element\_Using\_JS(eduYesButton);

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

waitInSeconds(10);

break;

default:

break;

}

}

} else if (sectionName != null && sectionName.contains("Professional Education")) {

wait\_for\_element\_present(submit);

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Institution Name is required.":

intsname.sendKeys("Test");

break;

case "Country is required.":

institutionName.sendKeys("TIU");

subjectofStudy.sendKeys("sos");

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(peContryDropdown);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The PE selection of state name is::Arizona");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.println("City / Region is required.");

city1.sendKeys("Thanda");

wait\_in\_seconds(5);

wait\_for\_element\_present(eduYesButton);

click\_Element\_Using\_JS(eduYesButton);

//eduYesButton.click();

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

click\_Element\_Using\_JS(saveAndContinue1);

// saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

} else if (sectionName != null && sectionName.contains("Internship")) {

wait\_for\_element\_present(submit);

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Country is required.":

jobTtitle.sendKeys("Lead");

employer.sendKeys("TIU");

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(peContryDropdown);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The Internship selection of state name is::Arizona");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.println("City / Region is required in internship.");

internshipCity.sendKeys("Thanda");

wait\_in\_seconds(3);

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

Select endmonth = new Select(endMonth);

endmonth.selectByVisibleText("Jan");

Select endyear = new Select(endYear);

endyear.selectByVisibleText("2019");

scrollDownToElement(saveAndContinue1);

click\_Element\_Using\_JS(saveAndContinue1);

//saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

} else if (sectionName != null && sectionName.contains("Residency")) {

wait\_for\_element\_present(submit);

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Job title is required.":

rsjobt.sendKeys("AIIMS");

break;

case "Country is required.":

jobTtitleResidence.sendKeys("QWERTY");

employerResidence.sendKeys("ASD");

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(peContryDropdown);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The Residency selection of state name is::Arizona");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.println("City / Region is required.");

residenceCity.sendKeys("Thanda");

wait\_in\_seconds(3);

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

Select endmonth = new Select(endMonth);

endmonth.selectByVisibleText("Jan");

Select endyear = new Select(endYear);

endyear.selectByVisibleText("2019");

scrollDownToElement(saveAndContinue1);

click\_Element\_Using\_JS(saveAndContinue1);

//saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

} else if (sectionName != null && sectionName.contains("Fellowship")) {

wait\_for\_element\_present(submit);

scrollDown();

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Employer is required.":

fsemp.sendKeys("Test");

break;

case "Country is required.":

jobTtitlefellowship.sendKeys("QWERTY");

employerfellowship.sendKeys("ASD");

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(peContryDropdown);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The Fellowship selection of state name is::Arizona");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.printf("City / Region is required.");

fellowshipCity.sendKeys("Thanda");

wait\_in\_seconds(3);

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

Select endmonth = new Select(endMonth);

endmonth.selectByVisibleText("Jan");

Select endyear = new Select(endYear);

endyear.selectByVisibleText("2019");

scrollDownToElement(saveAndContinue1);

click\_Element\_Using\_JS(saveAndContinue1);

//saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

}

else if (certCardTitle != null && certCardTitle.equals("State Licenses or Certificates")) {

System.out.println("There are 2 or 1 steps to complete");

wait\_for\_element\_present(saveAndContinue1);

scrollDownToElement(submit);

wait\_in\_seconds(10);

scrollDown();

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

scrollDownToElement(e);

switch (redField) {

case "Certification Name is required.":

System.out.println("The State Certification Name is required");

wait\_for\_element\_present(certification\_name);

certification\_name.sendKeys("asd");

break;

case "State Authority is required.":

Select selectStateAuthority = new Select(stateAuthority);

selectStateAuthority.selectByValue("AL");

certification\_date.sendKeys("02/02/2025");

scrollDown();

wait\_in\_seconds(5);

scrollDown();

scrollToElement(saveAndContinue1);

click\_Element\_Using\_JS(saveAndContinue1);

wait\_for\_page\_load(10);

//saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

}

else if (sectionName != null && sectionName.equals("Board Certification / Eligibility")) {

System.out.printf("There is 1 steps to complete");

wait\_for\_element\_present(submit);

scrollDownToElement(submit);

wait\_in\_seconds(10);

scrollDown();

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

scrollDownToElement(e);

switch (redField) {

case "Certificate Name is required.":

System.out.printf("The Board Certificate Name is required");

wait\_for\_element\_present(certification\_name);

certification\_name.sendKeys("asd");

break;

case "Authority is required.":

certificateAuthority.sendKeys("ASDF");

//certification\_date.sendKeys("02/02/2025");

break;

case "Date of Certification is required.":

certification\_date.sendKeys("02/02/2025");

scrollDown();

wait\_in\_seconds(5);

click\_Element\_Using\_JS(save);

wait\_in\_seconds(5);

clickhere.click();

break;

default:

break;

}

}

}

}

} else if (number <= 2) {

if (certCardTitle != null && certCardTitle.equals("State Licenses or Certificates")) {

System.out.println("There are 2 or 1 steps to complete");

wait\_for\_element\_present(saveAndContinue1);

scrollDownToElement(submit);

wait\_in\_seconds(10);

wait\_for\_page\_load(10);

scrollDown();

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

scrollDownToElement(e);

switch (redField) {

case "Certification Name is required.":

System.out.println("The Certification Name is required");

wait\_for\_element\_present(certification\_name);

certification\_name.sendKeys("asd");

break;

case "State Authority is required.":

Select selectStateAuthority = new Select(stateAuthority);

selectStateAuthority.selectByValue("AL");

break;

case "Date of Certification is required.":

certification\_date.sendKeys("02/02/2025");

scrollDown();

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

wait\_in\_seconds(5);

break;

default:

break;

}

}

} else if (sectionName.equals("Board Certification / Eligibility")) {

wait\_for\_page\_load(10);

System.out.println("There is 1 steps to complete");

wait\_for\_element\_present(submit);

scrollDownToElement(submit);

wait\_in\_seconds(10);

scrollDown();

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

scrollDownToElement(e);

switch (redField) {

case "Certificate Name is required.":

System.out.printf("The Certificate Name is required");

wait\_for\_element\_present(certification\_name);

certification\_name.sendKeys("asd");

break;

case "Authority is required.":

certificateAuthority.sendKeys("ASDF");

break;

case "Date of Certification is required.":

cdate.sendKeys("05/05/2025");

scrollDown();

// scrollToElement(save);

save.click();

wait\_for\_page\_load(5);

System.out.println("This is the Board Certificate Section");

//saveAndContinue1.click();

//wait\_for\_element\_present(clickhere);

click\_Element\_Using\_JS(clickhere);

clickhere.click();

break;

default:

break;

}

}

}

}

}

}

}

public void stepsToComplete1(String SSNNumber) throws InterruptedException {

wait\_for\_element\_present(stepComplete);

String stepcompletion = stepComplete.getText();

System.out.println("The steps to complete on the Dashboard on login is " + stepcompletion);

if (stepcompletion.equals("Congratulations! You have completed your on-boarding process.")) {

System.out.println("No steps to be completed as the The step complete message is " + stepcompletion);

}

int number = 0;

if (stepcompletion.contains("step(s) to complete")) {

String stepsToComplete = stepComplete.getText();

System.out.println("The number of steps to complete is " + stepsToComplete);

String stepsNumber = stepsToComplete.substring(9);

System.out.println("Extracted steps completion message is: " + stepsNumber);

String extNumber = stepsNumber.substring(0, 1);

number = Integer.parseInt(extNumber);

System.out.println("The steps number extracted is::" + number);

waitInSeconds(5);

for (int i = 0; i < number; i++) {

try {

if (stepComplete.isDisplayed()){

System.out.println("Clicking on step complete link");

stepComplete.click();

}

} catch (NoSuchElementException e) {

}

String certCardTitle = null;

String sectionName = null;

try {

if (certResCardTitle.isDisplayed()) {

certCardTitle = certResCardTitle.getText();

}

} catch (NoSuchElementException e) {

}

try {

if (cardTitle.isDisplayed()) {

sectionName = cardTitle.getText();

}

} catch (NoSuchElementException e) {

}

if (number >= 1) {

for (int j = 0; j < number; j++) {

wait\_for\_page\_load(10);

try {

if (certResCardTitle.isDisplayed()) {

certCardTitle = certResCardTitle.getText();

System.out.println("The section title is " +certCardTitle);

}

} catch (NoSuchElementException e) {

}

try {

if (cardTitle.isDisplayed()) {

sectionName = cardTitle.getText();

System.out.println("The section name is " +sectionName);

}

} catch (NoSuchElementException e) {

}

if (sectionName != null && sectionName.contains("Personal Details")) {

wait\_for\_element\_present(submit);

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Date of Birth is required.":

System.out.println("The Date of birth is ");

dob.clear();

dob.sendKeys("11/11/1990");

break;

case "SSN is required.":

System.out.println("The SSN number");

SSNp.clear();

SSNp.sendKeys(SSNNumber);

SSNp.sendKeys(Keys.TAB);

wait\_in\_seconds(15);

break;

case "Zip Code is required.":

System.out.println("The Zip code field is required for Personal details section");

zipcode.sendKeys("07086");

break;

case "Country is required.":

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The State is required for Personal details section");

wait\_for\_element\_present(stateList);

Select state = new Select(stateList);

state.selectByValue("AL");

break;

case "The city field is required.":

System.out.println("The city field is required for Personal details section");

city.sendKeys("NY");

break;

case "Specialties are required":

System.out.println("Specialties are required for Personal details section");

Select indusrty = new Select(industryType);

indusrty.selectByValue("255");

Select category = new Select(categoryType);

category.selectByValue("Allied Health");

waitInSeconds(5);

scrollDown();

scrollToElement(specialitiesSelect);

specialitiesSelect.click();

click\_Element\_Using\_JS(selectFromDropDown);

//selectFromDropDown.click();

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

waitInSeconds(10);

break;

default:

break;

}

}

}

else if (sectionName != null && sectionName.contains("Work Experience")) {

wait\_for\_element\_present(submit);

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Company Name is required.":

System.out.println("The Company name is::United States");

cwname.sendKeys("Tiu");

title.sendKeys("Sr Doctor");

break;

case "Job Title is required.":

title.sendKeys("Sr Doctor");

break;

case "Country is required.":

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The Country name in Work experience::United States");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByValue("AL");

break;

case "City / Region is required.":

System.out.println("City / Region is required in work experience.");

city.sendKeys("NY");

wait\_in\_seconds(5);

wait\_for\_element\_present(workingYesButton);

click\_Element\_Using\_JS(workingYesButton);

//workingYesButton.click();

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

waitInSeconds(10);

break;

default:

break;

}

}

}

else if (sectionName != null && sectionName.equals("Education")) {

wait\_for\_element\_present(submit);

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Institution Name is required.":

System.out.println("The Institution Name is");

intname.sendKeys("Test");

subject.sendKeys("Test");

break;

case "Country is required.":

System.out.println("The Country name is::United States");

wait\_for\_element\_present(country);

Select countryDropdown = new Select(country);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The State name is required i Educatio section");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.println("City / Region is required.");

city.sendKeys("NY");

wait\_in\_seconds(10);

click\_Element\_Using\_JS(eduYesButton);

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

waitInSeconds(10);

break;

default:

break;

}

}

} else if (sectionName != null && sectionName.contains("Professional Education")) {

wait\_for\_element\_present(submit);

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Institution Name is required.":

intsname.sendKeys("Test");

break;

case "Country is required.":

institutionName.sendKeys("TIU");

subjectofStudy.sendKeys("sos");

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(peContryDropdown);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The PE selection of state name is::Arizona");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.println("City / Region is required.");

city1.sendKeys("Thanda");

wait\_in\_seconds(5);

wait\_for\_element\_present(eduYesButton);

click\_Element\_Using\_JS(eduYesButton);

//eduYesButton.click();

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

click\_Element\_Using\_JS(saveAndContinue1);

// saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

} else if (sectionName != null && sectionName.contains("Internship")) {

wait\_for\_element\_present(submit);

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Country is required.":

jobTtitle.sendKeys("Lead");

employer.sendKeys("TIU");

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(peContryDropdown);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The Internship selection of state name is::Arizona");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.println("City / Region is required in internship.");

internshipCity.sendKeys("Thanda");

wait\_in\_seconds(3);

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

Select endmonth = new Select(endMonth);

endmonth.selectByVisibleText("Jan");

Select endyear = new Select(endYear);

endyear.selectByVisibleText("2019");

scrollDownToElement(saveAndContinue1);

click\_Element\_Using\_JS(saveAndContinue1);

//saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

} else if (sectionName != null && sectionName.contains("Residency")) {

wait\_for\_element\_present(submit);

scrollDown();

scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Job title is required.":

rsjobt.sendKeys("AIIMS");

break;

case "Country is required.":

jobTtitleResidence.sendKeys("QWERTY");

employerResidence.sendKeys("ASD");

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(peContryDropdown);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The Residency selection of state name is::Arizona");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.println("City / Region is required.");

residenceCity.sendKeys("Thanda");

wait\_in\_seconds(3);

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

Select endmonth = new Select(endMonth);

endmonth.selectByVisibleText("Jan");

Select endyear = new Select(endYear);

endyear.selectByVisibleText("2019");

scrollDownToElement(saveAndContinue1);

click\_Element\_Using\_JS(saveAndContinue1);

//saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

} else if (sectionName != null && sectionName.contains("Fellowship")) {

wait\_for\_element\_present(submit);

scrollDown();

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

switch (redField) {

case "Employer is required.":

fsemp.sendKeys("Test");

break;

case "Country is required.":

jobTtitlefellowship.sendKeys("QWERTY");

employerfellowship.sendKeys("ASD");

System.out.println("The Country name is::United States");

Select countryDropdown = new Select(peContryDropdown);

countryDropdown.selectByVisibleText("United States");

break;

case "State is required.":

System.out.println("The Fellowship selection of state name is::Arizona");

wait\_for\_element\_present(stateList0);

Select state = new Select(stateList0);

state.selectByVisibleText("Arizona");

break;

case "City / Region is required.":

System.out.printf("City / Region is required.");

fellowshipCity.sendKeys("Thanda");

wait\_in\_seconds(3);

Select month = new Select(startMonth);

month.selectByVisibleText("Jan");

Select year = new Select(startYear);

year.selectByVisibleText("2010");

Select endmonth = new Select(endMonth);

endmonth.selectByVisibleText("Jan");

Select endyear = new Select(endYear);

endyear.selectByVisibleText("2019");

scrollDownToElement(saveAndContinue1);

click\_Element\_Using\_JS(saveAndContinue1);

//saveAndContinue1.click();

wait\_in\_seconds(10);

break;

default:

break;

}

}

}

else if (certCardTitle != null && certCardTitle.equals("State Licenses or Certificates")) {

System.out.println("Completing 'State Licenses or Certificates' step");

// wait\_for\_element\_present(submit);

//scrollDownToElement(submit);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

try {

// scrollDownToElement(e);

String redField = e.getText().trim();

switch (redField) {

case "Certification Name is required.":

System.out.println("Filling Certification Name");

wait\_for\_element\_present(certification\_name);

certification\_name.sendKeys("asd");

break;

case "State Authority is required.":

System.out.println("Selecting State Authority and entering date");

Select selectStateAuthority = new Select(stateAuthority);

selectStateAuthority.selectByValue("AL");

certification\_date.sendKeys("02/02/2025");

wait\_in\_seconds(5);

scrollToElement(saveAndContinue1);

click\_Element\_Using\_JS(saveAndContinue1);

wait\_for\_page\_load(10);

break;

default:

// You may log unhandled fields if needed

break;

}

} catch (StaleElementReferenceException staleEx) {

System.out.println("Stale element encountered: " + staleEx.getMessage());

// Re-fetch elements if needed

}

}

}

else if (sectionName != null && sectionName.equals("Board Certification / Eligibility")) {

System.out.println("Completing 'Board Certification / Eligibility' step");

// wait\_for\_element\_present(submit);

wait\_in\_seconds(5);

// scrollDownToElement(submit);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

try {

//scrollDownToElement(e);

// String redField = e.getText().trim();

// for (WebElement e : mandatoryFields) {

String redField = e.getText();

// switch (redField) {

switch (redField) {

case "Certificate Name is required.":

System.out.println("Filling Certificate Name");

wait\_for\_element\_present(certification\_name);

certification\_name.sendKeys("asd");

break;

case "Authority is required.":

System.out.println("Filling Authority");

certificateAuthority.sendKeys("ASDF");

break;

case "Date of Certification is required.":

System.out.println("Filling Certification Date and saving");

certification\_date.sendKeys("02/02/2025");

wait\_in\_seconds(5);

click\_Element\_Using\_JS(saveAndContinue1);

wait\_in\_seconds(5);

clickhere.click();

break;

default:

break;

}

} catch (StaleElementReferenceException staleEx) {

System.out.println("Stale element encountered: " + staleEx.getMessage());

// Optional: Re-fetch elements if needed

}

}

}

}

}

}/\* else if (number <= 2) {

if (certCardTitle != null && certCardTitle.equals("State Licenses or Certificates")) {

System.out.println("There are 2 or 1 steps to complete");

wait\_for\_element\_present(saveAndContinue1);

scrollDownToElement(submit);

wait\_in\_seconds(10);

wait\_for\_page\_load(10);

scrollDown();

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText();

scrollDownToElement(e);

switch (redField) {

case "Certification Name is required.":

System.out.println("The Certification Name is required");

wait\_for\_element\_present(certification\_name);

certification\_name.sendKeys("asd");

break;

case "State Authority is required.":

Select selectStateAuthority = new Select(stateAuthority);

selectStateAuthority.selectByValue("AL");

break;

case "Date of Certification is required.":

certification\_date.sendKeys("02/02/2025");

scrollDown();

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

wait\_in\_seconds(5);

break;

default:

break;

}

}

} else if ((sectionName != null && sectionName.contains("Board Certification")) ||

(certCardTitle != null && certCardTitle.contains("Board Certification"))) {

System.out.println("Handling Board Certification / Eligibility section");

wait\_for\_element\_present(submit);

scrollDownToElement(submit);

waitInSeconds(5);

click\_Element\_Using\_JS(submit);

for (WebElement e : mandatoryFields) {

String redField = e.getText().trim();

scrollDownToElement(e);

switch (redField) {

case "Certificate Name is required.":

System.out.println("Entering Certificate Name");

wait\_for\_element\_present(certification\_name);

certification\_name.clear();

certification\_name.sendKeys("Board Certified - Example");

break;

case "Authority is required.":

System.out.println("Entering Authority");

wait\_for\_element\_present(certificateAuthority);

certificateAuthority.clear();

certificateAuthority.sendKeys("Medical Board Authority");

break;

case "Date of Certification is required.":

System.out.println("Entering Certification Date");

wait\_for\_element\_present(certification\_date);

certification\_date.clear();

certification\_date.sendKeys("02/02/2025");

break;

default:

System.out.println("Unrecognized required field: " + redField);

break;

}

}

scrollDownToElement(save);

waitInSeconds(2);

click\_Element\_Using\_JS(save);

wait\_for\_page\_load(10);

try {

if (clickhere.isDisplayed()) {

wait\_for\_element\_present(clickhere);

click\_Element\_Using\_JS(clickhere);

System.out.println("Clicked 'Click Here' after saving");

}

} catch (NoSuchElementException e) {

System.out.println("ClickHere element not found.");

}

}\*/

}

}

}

Register Page Java

package com.KredSafe.User;

import java.awt.AWTException;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import java.util.NoSuchElementException;

import java.util.Properties;

import javax.mail.Folder;

import javax.mail.Message;

import javax.mail.MessagingException;

import javax.mail.Multipart;

import javax.mail.NoSuchProviderException;

import javax.mail.Part;

import javax.mail.PasswordAuthentication;

import javax.mail.Session;

import javax.mail.Store;

import javax.mail.internet.MimeMessage;

import javax.mail.search.SearchTerm;

import org.apache.log4j.Logger;

import org.openqa.selenium.By;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.interactions.Actions;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

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

import com.KredSafe.TestBase.TestBase;

import junit.framework.Assert;

public class RegisterPage extends TestBase {

public static final Logger log = Logger.getLogger(RegisterPage.class.getName());

WebDriver driver;

public RegisterPage(WebDriver driver) {

this.driver = driver;

PageFactory.initElements(driver, this);

}

@FindBy(xpath = "//a[@href='https://dev.kredsafe.net/users/register/options']")

WebElement registerHereLink;

@FindBy(xpath = "//a[@href='https://dev.kredsafe.net/guest/verify']")

WebElement registerUsingOnlineApplication;

@FindBy(xpath = "//input[@class='emailInputText']")

WebElement emailIDInput;

@FindBy(xpath = "//input[@id='search-mobile']")

WebElement yopmilInput;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\_reg\"]")

WebElement registerSubmit;

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

WebElement yopmailRegisterNowButton;

@FindBy(xpath = "//\*[@id=\"idOTP\"]")

WebElement idOTP;

@FindBy(xpath = "//\*[@id=\"btnSubmit\"]")

WebElement verifyButton;

public void registerUser(String emailID) throws IOException, InterruptedException, MessagingException {

scrollDown();

scrollDownToElement(registerHereLink);

wait\_in\_seconds(5);

try {

if (registerHereLink.isDisplayed()) {

System.out.printf("Clicking on the register link ");

registerHereLink.click();

}

} catch (NoSuchElementException e) {

}

wait\_in\_seconds(10);

scrollDownToElement(registerUsingOnlineApplication);

registerUsingOnlineApplication.click();

emailIDInput.sendKeys(emailID);

wait\_in\_seconds(10);

scrollDownToElement(registerSubmit);

registerSubmit.click();

}

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

WebElement emailField;

@FindBy(xpath = "//span[text()='Next']")

WebElement nextButton;

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

WebElement passwordField;

@FindBy(xpath = "//\*[@id=\\\":1v\\\"]")

WebElement emailKredSafe;

@FindBy(xpath = "//tbody/tr/td[1]/a[contains(@href,'https://dev.kredsafe.net/register')]")

WebElement registerButon;

@SuppressWarnings("resource")

public static void accessGmail(String emailID, String emailpassword) throws InterruptedException {

// Set up ChromeDriver using WebDriverManager

WebDriver driver;

System.setProperty("webdriver.chrome.driver", "D:/KredSafe (2)/KredSafe/drivers/chromedriver.exe");

driver = new ChromeDriver();

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

driver.get("https://mail.google.com/");

WebElement emailField = driver.findElement(By.id("identifierId"));

emailField.sendKeys(emailID);

WebElement nextButton = driver.findElement(By.xpath("//span[text()='Next']"));

nextButton.click();

Thread.sleep(10000);

WebElement passwordField = driver.findElement(By.name("Passwd"));

passwordField.sendKeys(emailpassword);

nextButton = driver.findElement(By.xpath("//span[text()='Next']"));

nextButton.click();

wait\_for\_page\_load(20);

// Pause for manual action

System.out.println("Please perform manual action and press Enter to continue...");

new java.util.Scanner(System.in).nextLine();

Thread.sleep(30000);

/\* driver.quit() ; \*/

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

System.out.println("Switch to tab...");

driver.switchTo().window(tabs.get(1));

}

@SuppressWarnings("resource")

public void accessYopmail() throws InterruptedException {

// Set up ChromeDriver using WebDriverManager

WebDriver driver;

System.setProperty("webdriver.chrome.driver", "D:/KredSafe (2)/KredSafe/drivers/chromedriver.exe");

driver = new ChromeDriver();

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

driver.get("https://yopmail.com/");

wait\_for\_page\_load(20);

WebElement emailField = driver.findElement(By.id("login"));

emailField.sendKeys("hcheeti@yopmail.com");

WebElement submitYopmail = driver.findElement(By.id("refreshbut"));

submitYopmail.click();

wait\_for\_page\_load(20);

WebElement rgYopmailButton;

System.out.println("Try catch block...");

JavascriptExecutor executor = (JavascriptExecutor) driver;

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

rgYopmailButton = driver.findElement(By.xpath("//\*[@title= 'Register Now']")));

wait\_for\_element\_present(rgYopmailButton);

boolean dis;

try {

if (dis = rgYopmailButton.isDisplayed()) {

System.out.println("The element is " + dis);

rgYopmailButton.click();

}

} catch (NoSuchElementException e) {

rgYopmailButton.click();

}

// Pause for manual action System.out.println("Please perform manual action and

// press Enter to continue..."); new java.util.Scanner(System.in).nextLine();

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

System.out.println("Switch to tab...");

driver.switchTo().window(tabs.get(1));

WebElement FirstName = driver.findElement(By.xpath("//\*[@id=\\\"first\_name\\\"]"));

FirstName.sendKeys("Harideep");

}

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

WebElement yopmailField;

@FindBy(xpath = "<input type=\"text\" id=\"webmail\" onkeyup=\"try { kp(event); } catch (ex) {}\">")

WebElement OTP;

@FindBy(xpath = "//\*[@id=\"refreshbut\"]/button")

WebElement submitYopmail;

@FindBy(xpath = "//a[contains(@href, 'https://dev.kredsafe.net/register') and @title='Register Now']")

WebElement registerYopmailButton;

public void accessYopmailForOnlineRegister(String email) throws InterruptedException, IOException {

waitInSeconds(10);

yopmailField.sendKeys(email);

waitInSeconds(5);

submitYopmail.click();

waitInSeconds(5);

// wait\_for\_element\_present(registerYopmailButton);

waitInSeconds(5);

waitInSeconds(5);

driver.switchTo().frame("ifmail");

mouseClickByLocator(registerYopmailButton);

System.out.println("Try catch block...");

boolean dis;

try {

if (dis = registerYopmailButton.isDisplayed()) {

System.out.println("The element is " + dis);

wait\_in\_seconds(10);

wait\_for\_element\_present(registerYopmailButton);

scrollToElement(registerYopmailButton);

click\_Element\_Using\_JS(registerYopmailButton);

}

} catch (NoSuchElementException e) {

wait\_in\_seconds(10);

wait\_for\_element\_present(registerYopmailButton);

wait\_for\_element\_present(registerYopmailButton);

registerYopmailButton.click();

}

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

System.out.println("Switch to tab...");

driver.switchTo().window(tabs.get(1));

}

@FindBy(xpath = "//\*[@id='first\_name']")

WebElement FirstName;

@FindBy(xpath = "//\*[@id='last\_name']")

WebElement LasName;

@FindBy(xpath = "//\*[@id='passwordL']")

WebElement Password;

@FindBy(xpath = "//\*[@id='password\_conf']")

WebElement ConfirmPassword;

@FindBy(xpath = "//\*[@id='mobilenumber']")

WebElement MobileNumber;

@FindBy(xpath = "//\*[@id='zip']")

WebElement Zipcode;

@FindBy(xpath = "//\*[@id='id\_frm\_submit']")

WebElement accoutRegisterButton;

public void createAnAccount(String firstName, String lastName, String createAccountPassword,

String createAccountCPassword, String mobileNumber) {

wait\_for\_element\_present(FirstName);

FirstName.sendKeys(firstName);

LasName.sendKeys(lastName);

Password.sendKeys(createAccountPassword);

ConfirmPassword.sendKeys(createAccountCPassword);

MobileNumber.sendKeys(mobileNumber);

Zipcode.sendKeys("111111");

accoutRegisterButton.click();

}

@FindBy(xpath = "/html/body/div[1]/div[1]/div/div/div[2]/div/div/div[2]/a")

WebElement accoutRegisterUploadButton;

@FindBy(xpath = "//\*[@id=\"resume\_file\_upload\"]")

WebElement uploadResumeButton;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\_resume\"]")

WebElement uploadResumeSubmitButton;

@FindBy(xpath = "//button[@id=\"id\_frm\_submit\_resume\_data\"]")

WebElement submitButton;

@FindBy(xpath = "//input[@id=\"zip\"]")

WebElement zipCode;

@FindBy(xpath = "//span[@class=\"select2-search select2-search--inline\"]")

WebElement specialitiesSelect;

@FindBy(xpath = "//ul[@class=\"select2-results\_\_options\"]")

WebElement specialtiesDropDown;

@FindBy(xpath = "//\*[@id=\"select2-industry\_specialitiesR-results\"]/li[1]")

WebElement selectFromDropDown;

public void registerUserByUploadingResume() throws InterruptedException, AWTException {

wait\_for\_page\_load(5);

scrollToElement(registerHereLink);

click\_Element\_Using\_JS(registerHereLink);

wait\_in\_seconds(10);

accoutRegisterUploadButton.click();

wait\_in\_seconds(3);

boolean dis = uploadResumeButton.isDisplayed();

log.info("The element is dislayed " + dis);

driver.navigate().refresh();

wait\_in\_seconds(10);

WebElement upload = driver.findElement(By.xpath("//\*[@id='resume\_file\_upload']"));

Actions act = new Actions(driver);

act.moveToElement(upload).click().perform();

wait\_in\_seconds(5);

file\_upload("C:\\Users\\Admin\\Downloads\\DerekSmith(1).doc");

uploadResumeSubmitButton.click();

wait\_for\_page\_load(10);

// zipCode.sendKeys("123456");

// specialitiesSelect.click();

// selectFromDropDown.click();

wait\_in\_seconds(5);

scrollToElement(submitButton);

scrollDownToElement(submitButton);

click\_Element\_Using\_JS(submitButton);

}

@FindBy(xpath = "//a[contains(@href, 'https://dev.kredsafe.net/password/reset/') and @title='Set Password']")

WebElement setPasswordYopmail;

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

WebElement passwordL;

@FindBy(xpath = "//input[@id=\"password\_conf\"]")

WebElement password\_conf;

@FindBy(xpath = "//button[@type=\"submit\"]")

WebElement passwordSubmit;

@FindBy(xpath = "//a[contains(@href, '#') and @id='generatePassword']")

WebElement generatePassword;

@FindBy(xpath = "//span[@class='fa fa-fw field-icon toggle-password fa-eye-slash']")

WebElement viewPassword;

public void setPasswordYopmail(String resumeEmail) throws InterruptedException, IOException {

waitInSeconds(10);

yopmailField.sendKeys(resumeEmail);

waitInSeconds(5);

submitYopmail.click();

waitInSeconds(5);

driver.switchTo().frame("ifmail");

// mouseClickByLocator(setPasswordYopmail);

setPasswordYopmail.click();

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

System.out.println("Switch to tab...");

driver.switchTo().window(tabs.get(1));

wait\_for\_page\_load(10);

wait\_for\_element\_present(passwordL);

passwordL.sendKeys("Nilesh@2025");

password\_conf.sendKeys("Nilesh@2025");

String gPassword = passwordL.getText();

System.out.println("The gpassword is :" + gPassword);

wait\_in\_seconds(10);

scrollDownToElement(passwordSubmit);

scrollToElement(passwordSubmit);

wait\_in\_seconds(10);

passwordSubmit.click();

}

@FindBy(xpath = "(//button[contains(text(), \"Subscribe\")])[3]")

WebElement subscribeButton;

@FindBy(xpath = "//button[@id='home-tab\_1']")

WebElement creditCard;

@FindBy(xpath = "//button[@id='home-tab\_2']")

WebElement Paypal;

@FindBy(xpath = "//\*[@id=\"confirm\_pay\_btn\"]")

///html/body/div[4]/div[1]/section/div/div/div[2]/div/div/div/div[2]/div/form/div/div[2]/button

WebElement confirmAndPayButton;

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

WebElement cardNumber;

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

WebElement cardExpiry;

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

WebElement cardCVC;

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

WebElement billingName;

@FindBy(xpath = "//input[@id='billingAddressLine1']")

WebElement billingAddress;

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

WebElement billingPostalCode;

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

WebElement billingCity;

@FindBy(xpath = "//select[@id='billingAdministrativeArea']")

WebElement billingState;

@FindBy(xpath = "//\*[@id=\"billing-address-autocomplete-results\"]/div[1]/button/div/svg/g/rect")

WebElement cancelBillingAddress;

@FindBy(xpath = "//div[@class='SubmitButton-IconContainer']")

WebElement paySubmitButton;

@FindBy(xpath = "//a[@href='https://dev.kredsafe.net/admin/home']")

WebElement clickHere;

@FindBy(xpath = "//\*[@id=\"id\_frm\_submit\"]")

WebElement saveAndContinue1;

@FindBy(xpath = "//\*[@id=\"noId\"]")

WebElement No;

@FindBy(xpath = "//a[@href='#tab\_std\_res\_30']")

WebElement oneyear;

@FindBy(xpath = "//input[@name='installments' and @value='2']")

WebElement sixmonth;

@FindBy(xpath = "//input[@type='radio' and @name='installments' and @value='1']")

WebElement twelvemonth;

@SuppressWarnings("deprecation")

public void subscriptionPlans() throws InterruptedException, IOException {

try {

wait\_for\_page\_load(10);

String get\_currenturl = driver.getCurrentUrl();

System.out.println("Current URL is:" + get\_currenturl);

System.out.println("URL matching --> Part executed");

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscription/cart")) {

wait\_for\_element\_present(creditCard);

// wait\_in\_seconds(10);

creditCard.click();

wait\_in\_seconds(10);

wait\_for\_element\_present(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

// confirmAndPayButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("ct");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

wait\_in\_seconds(10);

paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

try {

waitInSeconds(10);

String get\_currenturl = driver.getCurrentUrl();

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscriptions")) {

wait\_for\_element\_present(subscribeButton);

// scrollDown();

scrollDownToElement(subscribeButton);

// scrollToElement(subscribeButton);

// wait\_for\_element\_present(subscribeButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(subscribeButton);

// subscribeButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(creditCard);

wait\_in\_seconds(20);

click\_Element\_Using\_JS(creditCard);

// creditCard.click();

waitInSeconds(5);

wait\_for\_element\_present(confirmAndPayButton);

scrollDownToElement(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("Nagpur");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

scrollDown();

scrollDownToElement(paySubmitButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(paySubmitButton);

// paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

}

public void subscriptionPlansStandard() throws InterruptedException, IOException {

try {

wait\_for\_page\_load(10);

String get\_currenturl = driver.getCurrentUrl();

System.out.println("Current URL is:" + get\_currenturl);

System.out.println("URL matching --> Part executed");

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscription/cart")) {

wait\_for\_element\_present(creditCard);

// wait\_in\_seconds(10);

creditCard.click();

wait\_in\_seconds(10);

wait\_for\_element\_present(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

// confirmAndPayButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("ct");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

wait\_in\_seconds(10);

paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

try {

waitInSeconds(10);

String get\_currenturl = driver.getCurrentUrl();

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscriptions")) {

wait\_for\_element\_present(oneyear);

click\_Element\_Using\_JS(oneyear);

wait\_for\_element\_present(sixmonth);

click\_Element\_Using\_JS(sixmonth);

//wait\_for\_element\_present(twelvemonth);

//click\_Element\_Using\_JS(twelvemonth);

wait\_for\_element\_present(subscribeButton);

// scrollDown();

scrollDownToElement(subscribeButton);

// scrollToElement(subscribeButton);

// wait\_for\_element\_present(subscribeButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(subscribeButton);

// subscribeButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(creditCard);

wait\_in\_seconds(20);

click\_Element\_Using\_JS(creditCard);

// creditCard.click();

waitInSeconds(5);

wait\_for\_element\_present(confirmAndPayButton);

scrollDownToElement(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("Nagpur");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

scrollDown();

scrollDownToElement(paySubmitButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(paySubmitButton);

// paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

}

@FindBy(xpath = "//a[@href='#tab\_premium\_res\_50']")

WebElement poneyear;

@FindBy(xpath = "//input[@type='radio' and @name='prem\_installments' and @data-month='6' and @value='2']")

WebElement psixmonth;

@FindBy(xpath = "//input[contains(@class, 'prem\_radio\_12') and contains(@class, 'prem\_radio\_50') and @value='1']")

WebElement ptwelvwmonth;

@FindBy(xpath = "//a[@href='#tab\_premium\_res\_48']")

WebElement pthreeyear;

@FindBy(xpath = "//input[@type='radio' and @name='prem\_installments' and @data-month='6' and @value='6']")

WebElement pthreesixmonthr;

@FindBy(xpath = "//input[@type='radio' and @name='prem\_installments' and @data-month='1' and @value='3']")

WebElement pthreetwelvwmonth;

@FindBy(xpath = "//input[@name='prem\_installments' and @data-month='3' and @value='1']")

WebElement pthreefullpay;

@FindBy(xpath = "//button[@id='subscribe5']")

WebElement psb;

//button[@id='subscribe5']

//input[@name='prem\_installments' and @data-month='3' and @value='1']

//input[@name='prem\_installments' and @data-month='1' and @value='3']

public void subscriptionPlanspremium() throws InterruptedException, IOException {

try {

wait\_for\_page\_load(10);

String get\_currenturl = driver.getCurrentUrl();

System.out.println("Current URL is:" + get\_currenturl);

System.out.println("URL matching --> Part executed");

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscription/cart")) {

wait\_for\_element\_present(creditCard);

// wait\_in\_seconds(10);

creditCard.click();

wait\_in\_seconds(10);

wait\_for\_element\_present(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

// confirmAndPayButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("ct");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

wait\_in\_seconds(10);

paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

try {

waitInSeconds(10);

String get\_currenturl = driver.getCurrentUrl();

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscriptions")) {

//for 1 Year

wait\_for\_element\_present(poneyear);

click\_Element\_Using\_JS(poneyear);

wait\_for\_element\_present (ptwelvwmonth);

click\_Element\_Using\_JS(ptwelvwmonth);

//wait\_for\_element\_present(psixmonth);

//click\_Element\_Using\_JS(psixmonth);

//for 3 Year

//wait\_for\_element\_present(pthreeyear);

//click\_Element\_Using\_JS(pthreeyear);

///wait\_for\_element\_present (pthreesixmonthr);

//click\_Element\_Using\_JS(pthreesixmonthr);

wait\_in\_seconds(5);

//wait\_for\_element\_present (pthreetwelvwmonth);

//click\_Element\_Using\_JS(pthreetwelvwmonth);

//wait\_for\_element\_present (pthreefullpay);

//click\_Element\_Using\_JS(pthreefullpay);

wait\_for\_element\_present(psb);

// scrollDown();

scrollDownToElement(psb);

// scrollToElement(subscribeButton);

// wait\_for\_element\_present(subscribeButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(psb);

// subscribeButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(creditCard);

wait\_in\_seconds(20);

click\_Element\_Using\_JS(creditCard);

// creditCard.click();

waitInSeconds(5);

wait\_for\_element\_present(confirmAndPayButton);

scrollDownToElement(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("Nagpur");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

scrollDown();

scrollDownToElement(paySubmitButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(paySubmitButton);

// paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

}

public void FormPlans() throws InterruptedException, IOException {

wait\_for\_page\_load(10);

wait\_for\_element\_present(confirmAndPayButton);

wait\_in\_seconds(5);

scrollToElement(confirmAndPayButton);

//scrollDown();

click\_Element\_Using\_JS(confirmAndPayButton);

waitInSeconds(10);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("Nagpur");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

wait\_for\_element\_present(paySubmitButton);

wait\_in\_seconds(5);

scrollDown();

click\_Element\_Using\_JS(paySubmitButton);

}

public static String readGmail(String USER\_NAME, String PASSWORD)

throws MessagingException, IOException, InterruptedException {

RegisterPage gmailHelper = new RegisterPage(userName, password);

String keyword = "Daily Status - KredSafe Automation Testing";

@SuppressWarnings("deprecation")

Date mailSendTime = new Date("Oct 21, 2024, 10:42 PM");

System.out.println("Mail send time : " + mailSendTime.toString());

List<Message> inboxMessageList = gmailHelper.searchEmail(0, "INBOX", keyword);

Message firstMessage = inboxMessageList.get(0);

String body = gmailHelper.getText(true, firstMessage);

System.out.println("Message Subject : " + firstMessage.getSubject());

System.out.println("Message Content : " + body);

return body;

}

private static String imapHost = "imap.gmail.com";

private static String imapPort = "993";

private static String userName = null;

private static String password = null;

/\*\*

\* Constructor

\*

\* @param userName

\* @param password

\* @return

\*/

public RegisterPage(String userName, String password) {

RegisterPage.userName = userName;

RegisterPage.password = password;

}

/\*\*

\* Set IMAP session

\*/

public Session setIMAPSession() {

// IMAP settings

Properties properties = new Properties();

// server setting

properties.put("mail.imap.host", imapHost);

properties.put("mail.imap.port", imapPort);

// SSL setting

properties.setProperty("mail.imap.socketFactory.class", "javax.net.ssl.SSLSocketFactory");

properties.setProperty("mail.imap.socketFactory.fallback", "false");

properties.setProperty("mail.imap.socketFactory.port", String.valueOf(imapPort));

Session session = Session.getDefaultInstance(properties, new javax.mail.Authenticator() {

protected PasswordAuthentication getPasswordAuthentication() {

return new PasswordAuthentication("hcheeti@tiuconsulting.com", "sTGV@8777\*");// Specify the Username and

// // the PassWord

}

});

return session;

}

/\*\*

\* Searches for e-mail messages containing the specified keyword in Subject

\* field.

\*

\* @param no of seconds to wait before checking for mails

\* @param folderName (e.g. INBOX)

\* @param keyword Search with this Keyword in subject line

\* @param Date object after which emails to be searched. Null for any date

\* @throws InterruptedException

\*/

public List<Message> searchEmail(int noOfSecToWait, String folderName, final String keyword)

throws InterruptedException {

Session session = setIMAPSession();

List<Message> returnList = new ArrayList<>();

try {

Thread.sleep(noOfSecToWait \* 1000);

System.out.println("Connects to Message Store");

Store store = session.getStore("imap");

store.connect(userName, password);

System.out.println("Opens folder : " + folderName);

Folder folderInbox = store.getFolder(folderName);

folderInbox.open(Folder.READ\_ONLY);

System.out.println("Creates search condition for Folder : " + folderName

+ ". Searching message with Subject line : " + keyword);

SearchTerm searchCondition = new SearchTerm() {

private static final long serialVersionUID = 1L;

@Override

public boolean match(Message message) {

try {

if (message.getSubject().contains(keyword)) {

return true;

}

} catch (MessagingException ex) {

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

ex.printStackTrace();

}

return false;

}

};

System.out.println("Performs search through the folder : " + folderName);

Message[] foundMessages = folderInbox.search(searchCondition);

System.out.println("Number of messages found : " + foundMessages.length);

for (int i = 0; i < foundMessages.length; i++) {

Message message = foundMessages[i];

/\* assuming you retrieved 'message' from your folder object \*/

Message copyOfMessage = new MimeMessage((MimeMessage) message);

returnList.add(copyOfMessage);

}

System.out.println("Disconnecting...");

folderInbox.close(false);

store.close();

} catch (NoSuchProviderException ex) {

System.out.println("No provider found.\n" + ex.getMessage());

ex.printStackTrace();

} catch (MessagingException ex) {

System.out.println("Could not connect to the message store.\n" + ex.getMessage());

ex.printStackTrace();

}

return returnList;

}

/\*\*

\* Return the primary text content of the message part.

\*/

public String getText(boolean textIsHtml, Part p) throws MessagingException, IOException {

if (p.isMimeType("text/\*")) {

String s = (String) p.getContent();

textIsHtml = p.isMimeType("text/html");

return s;

}

if (p.isMimeType("multipart/alternative")) {

System.out.println("refer html text over plain text");

// prefer html text over plain text

Multipart mp = (Multipart) p.getContent();

String text = null;

for (int i = 0; i < mp.getCount(); i++) {

Part bp = mp.getBodyPart(i);

if (bp.isMimeType("text/plain")) {

if (text == null)

text = getText(textIsHtml, bp);

continue;

} else if (bp.isMimeType("text/html")) {

String s = getText(textIsHtml, bp);

if (s != null)

return s;

} else {

return getText(textIsHtml, bp);

}

}

return text;

} else if (p.isMimeType("multipart/\*")) {

Multipart mp = (Multipart) p.getContent();

for (int i = 0; i < mp.getCount(); i++) {

String s = getText(textIsHtml, mp.getBodyPart(i));

if (s != null)

return s;

}

}

return null;

}

@FindBy(xpath = "//\*[@id=\"mail\"]/div/div/table/tbody/tr/td/p[2]")

WebElement OTPCode;

public void accessYopmaiforOTP() throws InterruptedException {

((JavascriptExecutor) driver).executeScript("window.open()");

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

driver.switchTo().window(tabs.get(1));

driver.get("https://yopmail.com/");

WebElement emailField = driver.findElement(By.id("login"));

//clp.clientSignIn(OR.getProperty("emailID4")));

emailField.sendKeys("ts1234@yopmail.com");

WebElement submitYopmail = driver.findElement(By.id("refreshbut"));

submitYopmail.click();

wait\_for\_page\_load(20);

waitInSeconds(10);

driver.switchTo().frame("ifmail");

String OTP = OTPCode.getText();

System.out.printf("Capture OTP code is: " + OTP);

}

@SuppressWarnings("resource")

public void verifyOTPFunctionality(String source, String clientSignIn) throws InterruptedException {

switch (source) {

case "EmailOTP":

((JavascriptExecutor) driver).executeScript("window.open()");

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

driver.switchTo().window(tabs.get(1));

driver.get("https://yopmail.com/");

WebElement emailField = driver.findElement(By.id("login"));

emailField.sendKeys("ts1234@yopmail.com");

WebElement submitYopmail = driver.findElement(By.id("refreshbut"));

submitYopmail.click();

wait\_for\_page\_load(20);

waitInSeconds(10);

driver.switchTo().frame("ifmail");

String OTP = OTPCode.getText();

System.out.printf("Capture OTP code is: " + OTP);

String code = OTP.substring(99);

System.out.printf("Extracted OTP code is: " + code);

String otpcode = code.substring(0, 6);

System.out.printf("\nOTP code is:" + otpcode);

ArrayList<String> tab = new ArrayList<String>(driver.getWindowHandles());

System.out.println("Switch to tab...");

driver.switchTo().window(tab.get(0));

idOTP.sendKeys(otpcode);

verifyButton.click();

break;

case "SMSOTP":

waitInSeconds(10);

// Pause for manual action

System.out.println(

"Please check the Registered Mobile number for the OTP received, enter the OTP from the phone SMS and press Enter to continue...");

new java.util.Scanner(System.in).nextLine();

System.out.println("Clicking on verify button");

verifyButton.click();

break;

default:

break;

}

}

public void verifyCompleteSection() throws InterruptedException, AWTException {

String get\_currenturl = driver.getCurrentUrl();

System.out.println("Current URL is:" + get\_currenturl);

System.out.println("URL matching --> Part executed");

for (int i = 0; i <= 10; i++) {

if (!get\_currenturl.contains("https://dev.kredsafe.net/user/statement")) {

scrollDownToElement(saveAndContinue1);

wait\_in\_seconds(5);

scrollDown();

saveAndContinue1.click();

}

}

}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[8]/a/i")

WebElement Subscription;

@FindBy(xpath = "//\*[@id=\"subBoxInfoClose\"]/section/div/div/div[2]/div/div/div/div[2]/div[9]/a")

WebElement Upgradeplan;

@FindBy(xpath = "//\*[@id=\"upgradePlanTable\"]/tbody/tr[2]/td[7]/a/i")

WebElement Upgrade;

//@FindBy(xpath = " /html/body/div[3]/div[1]/section/div/div/div[2]/div/div/div/div[2]/div[11]/div/div[2]/div/table/tbody/tr/td[6]/a\r\n"

//+ "")

//WebElement Upgrade;

@SuppressWarnings("deprecation")

public void subscriptionPlans1() throws InterruptedException, IOException {

try {

wait\_for\_page\_load(10);

String get\_currenturl = driver.getCurrentUrl();

System.out.println("Current URL is:" + get\_currenturl);

System.out.println("URL matching --> Part executed");

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscription/cart")) {

wait\_for\_element\_present(creditCard);

// wait\_in\_seconds(10);

creditCard.click();

wait\_in\_seconds(10);

wait\_for\_element\_present(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

// confirmAndPayButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("ct");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

wait\_in\_seconds(10);

paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

try {

waitInSeconds(10);

String get\_currenturl = driver.getCurrentUrl();

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscriptions")) {

wait\_for\_element\_present(subscribeButton);

// scrollDown();

scrollDownToElement(subscribeButton);

// scrollToElement(subscribeButton);

// wait\_for\_element\_present(subscribeButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(subscribeButton);

// subscribeButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(creditCard);

wait\_in\_seconds(20);

click\_Element\_Using\_JS(creditCard);

// creditCard.click();

waitInSeconds(5);

wait\_for\_element\_present(confirmAndPayButton);

scrollDownToElement(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("Nagpur");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

scrollDown();

scrollDownToElement(paySubmitButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(paySubmitButton);

// paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

try {

waitInSeconds(10);

String get\_currenturl = driver.getCurrentUrl();

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscription/upgrade/cart")) {

wait\_for\_page\_load(10);

wait\_for\_element\_present(creditCard);

wait\_in\_seconds(20);

click\_Element\_Using\_JS(creditCard);

// creditCard.click();

waitInSeconds(10);

wait\_for\_element\_present(confirmAndPayButton);

scrollDownToElement(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("Nagpur");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

scrollDown();

scrollDownToElement(paySubmitButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(paySubmitButton);

// paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

}

public void Renewsubscription() throws InterruptedException, IOException {

wait\_for\_page\_load(10);

wait\_for\_element\_present(Subscription);

click\_Element\_Using\_JS(Subscription);

wait\_for\_element\_present(Upgradeplan);

click\_Element\_Using\_JS(Upgradeplan);

scrollToElement(Upgrade);

wait\_for\_element\_present(Upgrade);

click\_Element\_Using\_JS(Upgrade);

}

public void Renewplan() throws InterruptedException, IOException {

try {

wait\_for\_page\_load(10);

String get\_currenturl = driver.getCurrentUrl();

System.out.println("Current URL is:" + get\_currenturl);

System.out.println("URL matching --> Part executed");

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscription/cart")) {

wait\_for\_element\_present(creditCard);

// wait\_in\_seconds(10);

creditCard.click();

wait\_in\_seconds(10);

wait\_for\_element\_present(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

// confirmAndPayButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("ct");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

wait\_in\_seconds(10);

paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

try {

waitInSeconds(10);

String get\_currenturl = driver.getCurrentUrl();

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscriptions")) {

wait\_for\_element\_present(subscribeButton);

// scrollDown();

scrollDownToElement(subscribeButton);

// scrollToElement(subscribeButton);

// wait\_for\_element\_present(subscribeButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(subscribeButton);

// subscribeButton.click();

wait\_for\_page\_load(10);

wait\_for\_element\_present(creditCard);

wait\_in\_seconds(20);

click\_Element\_Using\_JS(creditCard);

// creditCard.click();

waitInSeconds(5);

wait\_for\_element\_present(confirmAndPayButton);

scrollDownToElement(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("Nagpur");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

scrollDown();

scrollDownToElement(paySubmitButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(paySubmitButton);

// paySubmitButton.click();

wait\_for\_page\_load(10);

String get\_url = driver.getCurrentUrl();

Assert.assertEquals("https://dev.kredsafe.net/user/subscription/dashboard", get\_url);

System.out.println("Current URL is:" + get\_url);

System.out.println("URL matching --> Part executed");

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

try {

waitInSeconds(10);

String get\_currenturl = driver.getCurrentUrl();

if (get\_currenturl.equals("https://dev.kredsafe.net/user/subscription/upgrade/cart")) {

wait\_for\_page\_load(10);

// wait\_for\_element\_present(creditCard);

// wait\_in\_seconds(20);

// click\_Element\_Using\_JS(creditCard);

// creditCard.click();No

waitInSeconds(10);

//scrollDownToElement(No);

//click\_Element\_Using\_JS(No);

wait\_for\_element\_present(confirmAndPayButton);

scrollDownToElement(confirmAndPayButton);

click\_Element\_Using\_JS(confirmAndPayButton);

wait\_for\_element\_present(cardNumber);

cardNumber.sendKeys("4111111111111111");

cardExpiry.sendKeys("09/25");

cardCVC.sendKeys("111");

billingName.sendKeys("hcheeti");

billingAddress.sendKeys("Nagpur");

billingCity.sendKeys("Nagpur");

billingPostalCode.sendKeys("441108");

Select state = new Select(billingState);

state.getOptions().get(2).click();

scrollDown();

scrollDownToElement(paySubmitButton);

wait\_in\_seconds(5);

click\_Element\_Using\_JS(paySubmitButton);

}

} catch (NoSuchElementException ex) {

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

ex.printStackTrace();

}

}

}

Test Suite

package TestSuite;

import java.io.IOException;

import java.lang.reflect.Method;

import javax.mail.MessagingException;

import org.apache.log4j.Logger;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC01\_VerifyRegisterUsingOnlineApplication extends TestBase {

public static final Logger log = Logger.getLogger(TC01\_VerifyRegisterUsingOnlineApplication.class.getName());

/\*

\* @BeforeClass public static void startTest() {

\*

\* test = extent.startTest("TC01\_VerifyRegisterUsingOnlineApplication");

\*

\* }

\*/

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void registerUserUsingOnlineApplication() throws InterruptedException, IOException, MessagingException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

RegisterPage rp = new RegisterPage(driver);

rp.registerUser(OR.getProperty("emailID"));

}

@Test(priority = 2, enabled = true)

public void registerUsingYopmail() throws InterruptedException, IOException, MessagingException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openUserEmailUrl();

RegisterPage rp = new RegisterPage(driver);

rp.accessYopmailForOnlineRegister(OR.getProperty("emailID"));

rp.createAnAccount(OR.getProperty("firstName"), OR.getProperty("lastName"),

OR.getProperty("createAccountPassword"), OR.getProperty("createAccountCPassword"),

OR.getProperty("mobileNumber"));

rp.subscriptionPlans();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC02\_verifyRegisterbyUploadingResume extends TestBase {

public static final Logger log = Logger.getLogger(TC02\_verifyRegisterbyUploadingResume.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC02\_verifyRegisterbyUploadingResume");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void registerUsersUsingResume() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

RegisterPage rp = new RegisterPage(driver);

rp.registerUserByUploadingResume();

}

@Test(priority = 1, enabled = true)

public void setPasswordForResumeRegistration() throws IOException, InterruptedException, AWTException {

openUserEmailUrl();

RegisterPage rp = new RegisterPage(driver);

rp.setPasswordYopmail(OR.getProperty("resumeEmail"));

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------------------------------------- package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC03\_verifyProfileDetailsFunctionality extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC03\_verifyProfileDetailsFunctionality");

///}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying Profile Details Functionality===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID"), OR.getProperty("createAccountPassword"));

}

@Test(priority = 2, enabled = true)

public void verifyProfileDetails() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying Profile Details Functionality===========> ");

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

rp.subscriptionPlans();

clp.personalDetails();

}

@Test(priority = 3, enabled = false)

public void verifyDocumentsSection() throws IOException, InterruptedException, AWTException {

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

rp.subscriptionPlans();

clp.documentsSection();

clp.notesSection();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

----------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

import jxl.read.biff.BiffException;

public class TC04\_OTP extends TestBase {

public static final Logger log = Logger.getLogger(TC04\_OTP.class.getName());

private static final String By = null;

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC04\_OTP");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying OTP Functionality===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID11"), OR.getProperty("createAccountPassword11"));

//clp.clientSignIn("ts1234@yopmail.com", "Nilesh@2030");

RegisterPage rp = new RegisterPage(driver);

rp.accessYopmaiforOTP();

rp.verifyOTPFunctionality(OR.getProperty("OTPSource"), null);

}

@DataProvider(name = "exceldata") String[][] dataprovidefunc() throws

BiffException, IOException {

String[][] obj = getData("KredSafe\_Login\_Data.xlsx", "Sheet1");

return obj;

}

@Test(dataProvider = "exceldata", priority = 2, enabled = true)

public void verifyLoginFunctionality(String username, String password,String errormessage) throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying Login Functionality===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignInn(username, password,errormessage);

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC05\_VerifyProfilePicture extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC03\_verifyProfileDetailsFunctionality");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

//clp.clientSignIn(OR.getProperty("emailID4"), OR.getProperty("createAccountPassword4"));

clp.clientSignIn("ats51@yopmail.com", "surecafe");

wait\_for\_page\_load(5);

waitInSeconds(10);

clp.Profilepicture();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

------------------------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC06\_VerifyChangepassword extends TestBase {

public static final Logger log = Logger.getLogger(TC06\_VerifyChangepassword.class.getName());

//@BeforeClass

//public static void startTest() {

/// test = extent.startTest("TC06\_VerifyChangepassword");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID1"), OR.getProperty("createAccountPassword1"));

wait\_for\_page\_load(5);

waitInSeconds(10);

clp.Changepassword();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

---------------------------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC07\_VerifyForgotpassword extends TestBase {

public static final Logger log = Logger.getLogger(TC07\_VerifyForgotpassword.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC07\_VerifyForgotpassword");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.Forgotpassword();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC08\_VerifyFormpayment extends TestBase {

public static final Logger log = Logger.getLogger(TC08\_VerifyFormpayment.class.getName());

private static final String By = null;

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC08\_VerifyFormpayment");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID5"), OR.getProperty("createAccountPassword5"));

// clp.clientSignIn("nh910@yopmail.com", "Nilesh@2025");

RegisterPage rp = new RegisterPage(driver);

waitInSeconds(10);

ClientLoginPage clp1 = new ClientLoginPage(driver);

clp1.Formpayment();

RegisterPage clp2 = new RegisterPage(driver);

clp2.FormPlans();

}

@AfterMethod public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC09\_ProfileNaviagtion extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

// public static void startTest() {

// test = extent.startTest("TC09\_ProfileNavigation");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID3"), OR.getProperty("createAccountPassword3"));

RegisterPage clp2 = new RegisterPage(driver);

wait\_for\_page\_load(5);

waitInSeconds(10);

//clp2.accessYopmaiforOTP();

//clp2.verifyOTPFunctionality(OR.getProperty("OTPSource"),OR.getProperty("clientSignIn"));

clp.Profilenaviagtion();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

-------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC10\_VerifySubscriptionprocess extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

///public static void startTest() {

// test = extent.startTest("TC10\_VerifySubscriptionprocess");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@FindBy(xpath = "//\*[@id=\"userNameId\"]")

WebElement Profilenameclick;

@FindBy(xpath = "/html/body/div[3]/nav/ul[2]/li[5]/ul/li[1]/p/a\r\n"

+ "")

WebElement Changepasswordlink;

@FindBy(xpath = " //\*[@id=\"passwordChangeAlertModal\"]/div/div/div[2]/button\r\n"

+ "")

WebElement popok;

//\*[@id="passwordChangeAlertModal"]/div/div/div[2]/button

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

//clp.clientSignIn("samt123@yopmail.com", "Nilesh@2025");

clp.clientSignIn(OR.getProperty("emailID10"), OR.getProperty("createAccountPassword10"));

wait\_for\_page\_load(5);

waitInSeconds(10);

rp.subscriptionPlans();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC11\_verifydocumentupload extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC11\_verifydocumentupload");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID12"), OR.getProperty("createAccountPassword12"));

}

@Test(priority = 2, enabled = false)

public void verifyProfileDetails() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

rp.subscriptionPlans();

clp.personalDetails();

//clp.verifyCompleteSection();

}

@Test(priority = 3, enabled = true)

public void verifyDocumentsSection() throws IOException, InterruptedException, AWTException {

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

//rp.subscriptionPlans();

clp.documentsSection();

//clp.notesSection();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC12\_Verifynotesprocess extends TestBase {

public static final Logger log = Logger.getLogger(TC12\_Verifynotesprocess.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC12\_Verifynotesprocess");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID4"), OR.getProperty("createAccountPassword4"));

}

@Test(priority = 2, enabled = false)

public void verifyProfileDetails() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

rp.subscriptionPlans();

clp.personalDetails();

//clp.verifyCompleteSection();

}

@Test(priority = 3, enabled = true)

public void verifyDocumentsSection() throws IOException, InterruptedException, AWTException {

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

//rp.subscriptionPlans();

//clp.documentsSection();

clp.notesSection();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

------------------------------------------------------------------------------------------------------------------------ package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC13\_ProfileNaviagtionremovetheaddanother extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

@BeforeClass

public static void startTest() {

test = extent.startTest("TC13\_ProfileNaviagtionremovetheaddanother");

}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID"), OR.getProperty("createAccountPassword"));

//RegisterPage clp2 = new RegisterPage(driver);

wait\_for\_page\_load(5);

waitInSeconds(10);

//clp2.accessYopmaiforOTP();

//clp2.verifyOTPFunctionality(OR.getProperty("OTPSource"),OR.getProperty("clientSignIn"));

clp.Profilenaviagtion();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

---------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC14\_MakePrimaryEmail extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC14\_MakePrimaryEmail ");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID4"), OR.getProperty("createAccountPassword4"));

//RegisterPage clp2 = new RegisterPage(driver);

wait\_for\_page\_load(5);

waitInSeconds(10);

// clp2.accessYopmaiforOTP();

// clp2.verifyOTPFunctionality(OR.getProperty("OTPSource"),OR.getProperty("clientSignIn"));

clp.PrimaryEmail();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

-------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC15\_VerifyProfilefromSectionstobecompleted extends TestBase {

public static final Logger log = Logger.getLogger(TC15\_VerifyProfilefromSectionstobecompleted.class.getName());

// @BeforeClass

//public static void startTest() {

// test = extent.startTest("TC15\_VerifyProfilefromSectionstobecompleted");

// }

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID4"), OR.getProperty("createAccountPassword4"));

}

@Test(priority = 2, enabled = true)

public void verifyProfileDetails() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

ClientLoginPage clp = new ClientLoginPage(driver);

//RegisterPage rp = new RegisterPage(driver);

//rp.subscriptionPlans();

clp.Steptonecompleted();

//clp.verifyCompleteSection();

}

@Test(priority = 3, enabled = false)

public void verifyDocumentsSection() throws IOException, InterruptedException, AWTException {

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

rp.subscriptionPlans();

clp.documentsSection();

clp.notesSection();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC16\_LoginLogout extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC16\_LoginLogout");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID4"), OR.getProperty("createAccountPassword4"));

//RegisterPage clp2 = new RegisterPage(driver);

wait\_for\_page\_load(5);

waitInSeconds(10);

//clp2.accessYopmaiforOTP();

//clp2.verifyOTPFunctionality(OR.getProperty("OTPSource"),OR.getProperty("clientSignIn"));

clp.LoginLogout();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

----------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC17\_VerifyRecentActivity extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC17\_VerifyRecentActivity");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID3"), OR.getProperty("createAccountPassword3"));

//clp.clientSignIn("nh910@yopmail.com", "Nilesh@2025");

//clp.clientSignIn(OR.getProperty("emailID"), OR.getProperty("createAccountPassword"));

//RegisterPage clp2 = new RegisterPage(driver);

wait\_for\_page\_load(5);

waitInSeconds(10);

//clp2.accessYopmaiforOTP();

//clp2.verifyOTPFunctionality(OR.getProperty("OTPSource"),OR.getProperty("clientSignIn"));

clp.Profilenaviagtion();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------------------------------------------

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC18\_Restoreprofiledata extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC18\_Restoreprofiledata");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID18"), OR.getProperty("createAccountPassword18"));

//RegisterPage clp2 = new RegisterPage(driver);

wait\_for\_page\_load(5);

waitInSeconds(10);

//clp2.accessYopmaiforOTP();

//clp2.verifyOTPFunctionality(OR.getProperty("OTPSource"),OR.getProperty("clientSignIn"));

clp.RestoreIntership();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

import jxl.read.biff.BiffException;

public class TC19\_ChangepasswordbyExcel extends TestBase {

public static final Logger log = Logger.getLogger(TC19\_ChangepasswordbyExcel.class.getName());

private static final String By = null;

//@BeforeClass

// public static void startTest() {

// test = extent.startTest("TC19\_ChangepasswordbyExcel");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn("ts1234@yopmail.com", "Nilesh@2025");

RegisterPage rp = new RegisterPage(driver);

//rp.accessYopmaiforOTP();

//rp.verifyOTPFunctionality(OR.getProperty("OTPSource"), null);

}

@DataProvider(name = "exceldata") String[][] dataprovidefunc() throws

BiffException, IOException {

String[][] obj = getData("KredSafe\_Login\_Data.xlsx", "Sheet2");

return obj;

}

@Test(dataProvider = "exceldata", priority = 2, enabled = true)

public void verifyLoginFunctionality(String oldpassword , String newpassword,String confirmpassword,String errormessage) throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying Login Functionality===========> ");

//openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.Changepassword(oldpassword,newpassword,confirmpassword,errormessage);

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC20\_VerifyRenewSubscriptionprocess extends TestBase {

public static final Logger log = Logger.getLogger(TC20\_VerifyRenewSubscriptionprocess.class.getName());

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC20\_VerifyRenewSubscriptionprocess");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@FindBy(xpath = "//\*[@id=\"userNameId\"]")

WebElement Profilenameclick;

@FindBy(xpath = "/html/body/div[3]/nav/ul[2]/li[5]/ul/li[1]/p/a\r\n"

+ "")

WebElement Changepasswordlink;

@FindBy(xpath = " //\*[@id=\"passwordChangeAlertModal\"]/div/div/div[2]/button\r\n"

+ "")

WebElement popok;

//\*[@id="passwordChangeAlertModal"]/div/div/div[2]/button

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

clp.clientSignIn("ts17045@yopmail.com", "Nilesh@2025");

//clp.clientSignIn(OR.getProperty("emailID18"), OR.getProperty("createAccountPassword18"));

wait\_for\_page\_load(5);

waitInSeconds(10);

rp.Renewsubscription();

rp.Renewplan();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC21\_VerifyFormpaymentwhenprofilenotcompleted extends TestBase {

public static final Logger log = Logger.getLogger(TC04\_OTP.class.getName());

private static final String By = null;

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC21\_VerifyFormpaymentwhenprofilenotcompleted");

//}

@FindBy(xpath = "/html/body/div[3]/nav/ul[1]/li[2]/span/img")

WebElement menuclick;

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[6]/a")

WebElement formclick;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

// clp.clientSignIn("nh910@yopmail.com", "Nilesh@2025");

clp.clientSignIn(OR.getProperty("emailID3"), OR.getProperty("createAccountPassword3"));

RegisterPage rp = new RegisterPage(driver);

//rp.accessYopmaiforOTP();

//rp.verifyOTPFunctionality(OR.getProperty("OTPSource"), null);

waitInSeconds(10);

ClientLoginPage clp1 = new ClientLoginPage(driver);

clp1.Formpaymentvalidation();

//RegisterPage clp2 = new RegisterPage(driver);

// clp2.FormPlans();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC22\_VerifyUpgradeSubscriptionprocess extends TestBase {

public static final Logger log = Logger.getLogger(TC03\_verifyProfileDetailsFunctionality.class.getName());

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC22\_VerifyUpgradeSubscriptionprocess");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@FindBy(xpath = "//\*[@id=\"userNameId\"]")

WebElement Profilenameclick;

@FindBy(xpath = "/html/body/div[3]/nav/ul[2]/li[5]/ul/li[1]/p/a\r\n"

+ "")

WebElement Changepasswordlink;

@FindBy(xpath = " //\*[@id=\"passwordChangeAlertModal\"]/div/div/div[2]/button\r\n"

+ "")

WebElement popok;

//\*[@id="passwordChangeAlertModal"]/div/div/div[2]/button

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

clp.clientSignIn(OR.getProperty("emailID2"), OR.getProperty("createAccountPassword2"));

wait\_for\_page\_load(5);

waitInSeconds(10);

rp.Renewsubscription();

rp.subscriptionPlans();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC23\_VerifyPendingActivity extends TestBase {

public static final Logger log = Logger.getLogger(TC04\_OTP.class.getName());

private static final String By = null;

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC23\_VerifyPendingActivity");

//}

@FindBy(xpath = "/html/body/div[3]/nav/ul[1]/li[2]/span/img")

WebElement menuclick;

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[6]/a")

WebElement formclick;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID3"), OR.getProperty("createAccountPassword3"));

//clp.clientSignIn("nh910@yopmail.com", "Nilesh@2025");

RegisterPage rp = new RegisterPage(driver);

//rp.accessYopmaiforOTP();

//rp.verifyOTPFunctionality(OR.getProperty("OTPSource"), null);

waitInSeconds(10);

ClientLoginPage clp1 = new ClientLoginPage(driver);

clp1.Pendingactivity();

//RegisterPage clp2 = new RegisterPage(driver);

// clp2.FormPlans();

}

@AfterMethod public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC24\_VerifyFormpaymentwhenprofilenotcompletedfromnotification extends TestBase {

public static final Logger log = Logger.getLogger(TC04\_OTP.class.getName());

private static final String By = null;

//@BeforeClass

//public static void startTest() {

///test = extent.startTest(" TC24\_VerifyFormpaymentwhenprofilenotcompletedfromnotification");

//}

@FindBy(xpath = "/html/body/div[3]/nav/ul[1]/li[2]/span/img")

WebElement menuclick;

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[6]/a")

WebElement formclick;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn(OR.getProperty("emailID3"), OR.getProperty("createAccountPassword3"));

//clp.clientSignIn("nh910@yopmail.com", "Nilesh@2025");

RegisterPage rp = new RegisterPage(driver);

//rp.accessYopmaiforOTP();

//rp.verifyOTPFunctionality(OR.getProperty("OTPSource"), null);

waitInSeconds(10);

ClientLoginPage clp1 = new ClientLoginPage(driver);

clp1.Verifypaymentnotification();

//RegisterPage clp2 = new RegisterPage(driver);

// clp2.FormPlans();

}

@AfterMethod public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC25\_VerifyProfileflow extends TestBase {

public static final Logger log = Logger.getLogger(TC25\_VerifyProfileflow.class.getName());

private static final String By = null;

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC25\_VerifyProfileflow");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn("ts10@yopmail.com", "Nilesh@2025");

//RegisterPage rp = new RegisterPage(driver);

//rp.accessYopmaiforOTP();

//rp.verifyOTPFunctionality(OR.getProperty("OTPSource"), null);

waitInSeconds(10);

ClientLoginPage clp1 = new ClientLoginPage(driver);

clp1.Profileflow();

//RegisterPage clp2 = new RegisterPage(driver);

// clp2.FormPlans();

}

@AfterMethod public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC26\_VerifyStepstoComplete extends TestBase {

public static final Logger log = Logger.getLogger(TC26\_VerifyStepstoComplete.class.getName());

private static final String By = null;

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC26\_VerifyStepstoComplete");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void stepsToComplete() throws IOException, InterruptedException, AWTException {

log.info("<=========== Started Verifying stepsToComplete ===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn("ts053376@yopmail.com", "Nilesh@2025");

clp.stepsToComplete("173556789");

}

@Test(priority = 2, enabled = true)

public void stepsToCompleteAll() throws IOException, InterruptedException, AWTException {

log.info("<=========== Started Verifying stepsToComplete ===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn("ts053376@yopmail.com", "Nilesh@2025");

clp.stepsToComplete("173556789");

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC27\_Profileflow extends TestBase {

public static final Logger log = Logger.getLogger(TC27\_Profileflow .class.getName());

private static final String By = null;

//@BeforeClass

//public static void startTest() {

//test = extent.startTest("TC27\_Profileflow ");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void stepsToComplete() throws IOException, InterruptedException, AWTException {

log.info("<=========== Started Verifying stepsToComplete ===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn("hrd786@yopmail.com", "Nilesh@2025");

clp.stepsToComplete1("178956789");

}

@Test(priority = 2, enabled = false)

public void stepsToCompleteAll() throws IOException, InterruptedException, AWTException {

log.info("<=========== Started Verifying stepsToComplete ===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

clp.clientSignIn("hrd786@yopmail.com", "Nilesh@2025");

clp.stepsToComplete1("178956789");

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC28\_Verifyfollowupnotes extends TestBase {

public static final Logger log = Logger.getLogger(TC28\_Verifyfollowupnotes.class.getName());

//@BeforeClass

//public static void startTest() {

/// test = extent.startTest("TC06\_VerifyChangepassword");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void Followup() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

//clp.clientSignIn(OR.getProperty("emailID1"), OR.getProperty("createAccountPassword1"));

clp.clientSignIn("hrd786@yopmail.com", "Nilesh@2025");

wait\_for\_page\_load(5);

waitInSeconds(10);

clp.Followup();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC29\_Verifyformrequest extends TestBase {

public static final Logger log = Logger.getLogger(TC29\_Verifyformrequest.class.getName());

//@BeforeClass

//public static void startTest() {

/// test = extent.startTest("TC06\_VerifyChangepassword");

//}

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void Formrequest() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

//clp.clientSignIn(OR.getProperty("emailID1"), OR.getProperty("createAccountPassword1"));

clp.clientSignIn("hrd786@yopmail.com", "Nilesh@2025");

wait\_for\_page\_load(5);

waitInSeconds(10);

clp.formrequest();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC30\_VerifySubscriptionprocessStandard extends TestBase {

public static final Logger log = Logger.getLogger(TC30\_VerifySubscriptionprocessStandard.class.getName());

//@BeforeClass

///public static void startTest() {

// test = extent.startTest("TC10\_VerifySubscriptionprocess");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@FindBy(xpath = "//\*[@id=\"userNameId\"]")

WebElement Profilenameclick;

@FindBy(xpath = "/html/body/div[3]/nav/ul[2]/li[5]/ul/li[1]/p/a\r\n"

+ "")

WebElement Changepasswordlink;

@FindBy(xpath = " //\*[@id=\"passwordChangeAlertModal\"]/div/div/div[2]/button\r\n"

+ "")

WebElement popok;

//\*[@id="passwordChangeAlertModal"]/div/div/div[2]/button

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

clp.clientSignIn("rajb07@yopmail.com", "Nilesh@2025");

//clp.clientSignIn(OR.getProperty("emailID10"), OR.getProperty("createAccountPassword10"));

wait\_for\_page\_load(5);

waitInSeconds(10);

rp.subscriptionPlansStandard();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

public class TC31\_subscriptionPlanspremium extends TestBase {

public static final Logger log = Logger.getLogger( TC31\_subscriptionPlanspremium.class.getName());

//@BeforeClass

///public static void startTest() {

// test = extent.startTest("TC10\_VerifySubscriptionprocess");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@FindBy(xpath = "//\*[@id=\"userNameId\"]")

WebElement Profilenameclick;

@FindBy(xpath = "/html/body/div[3]/nav/ul[2]/li[5]/ul/li[1]/p/a\r\n"

+ "")

WebElement Changepasswordlink;

@FindBy(xpath = " //\*[@id=\"passwordChangeAlertModal\"]/div/div/div[2]/button\r\n"

+ "")

WebElement popok;

//\*[@id="passwordChangeAlertModal"]/div/div/div[2]/button

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void clientLogin() throws IOException, InterruptedException, AWTException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

RegisterPage rp = new RegisterPage(driver);

clp.clientSignIn("hr9@yopmail.com", "Nilesh@2025");

//clp.clientSignIn(OR.getProperty("emailID10"), OR.getProperty("createAccountPassword10"));

wait\_for\_page\_load(5);

waitInSeconds(10);

rp.subscriptionPlanspremium();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

import edu.emory.mathcs.backport.java.util.concurrent.TimeoutException;

public class TC32\_VerifyRestorevalidation extends TestBase {

public static final Logger log = Logger.getLogger(TC32\_VerifyRestorevalidation.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC18\_Restoreprofiledata");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void RestoreValidation() throws IOException, InterruptedException, AWTException, TimeoutException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

//clp.clientSignIn(OR.getProperty("emailID18"), OR.getProperty("createAccountPassword18"));

clp.clientSignIn("sep9@yopmail.com", "Nilesh@2025");

//RegisterPage clp2 = new RegisterPage(driver);

wait\_for\_page\_load(5);

waitInSeconds(10);

//clp2.accessYopmaiforOTP();

//clp2.verifyOTPFunctionality(OR.getProperty("OTPSource"),OR.getProperty("clientSignIn"));

clp.RestoreValidation();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

package TestSuite;

import java.awt.AWTException;

import java.io.IOException;

import java.lang.reflect.Method;

import org.apache.log4j.Logger;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.KredSafe.TestBase.TestBase;

import com.KredSafe.User.ClientLoginPage;

import com.KredSafe.User.RegisterPage;

import com.relevantcodes.extentreports.LogStatus;

import edu.emory.mathcs.backport.java.util.concurrent.TimeoutException;

public class TC33\_SubscriptionTotal extends TestBase {

public static final Logger log = Logger.getLogger(TC33\_SubscriptionTotal.class.getName());

//@BeforeClass

//public static void startTest() {

// test = extent.startTest("TC18\_Restoreprofiledata");

//}

@FindBy(xpath = "/html/body/div[3]/aside/div/div[2]/nav/ul/li[2]/a")

WebElement personalDetails;

@BeforeMethod

public void setup(Method result) throws IOException {

test = extent.startTest(result.getName());

test.log(LogStatus.INFO, result.getName() + " test Started");

init();

}

@Test(priority = 1, enabled = true)

public void SubsctionToatl() throws IOException, InterruptedException, AWTException, TimeoutException {

log.info("<===========Started Verifying RegisterUsingOnlineApplication===========> ");

openAppUrl();

ClientLoginPage clp = new ClientLoginPage(driver);

//clp.clientSignIn(OR.getProperty("emailID18"), OR.getProperty("createAccountPassword18"));

clp.clientSignIn("sep9@yopmail.com", "Nilesh@2025");

//RegisterPage clp2 = new RegisterPage(driver);

wait\_for\_page\_load(5);

waitInSeconds(10);

//clp2.accessYopmaiforOTP();

//clp2.verifyOTPFunctionality(OR.getProperty("OTPSource"),OR.getProperty("clientSignIn"));

clp.Subscriptiontotal();

}

@AfterMethod

public void afterMethod(ITestResult result) {

get\_result(result);

}

}

--------------------------------------------------------------------------------------

Test Suite

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

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

<suite name=*"AutomationTestSuite"*>

<listeners>

<listener class-name=*"com.KredSafe.CustomListener.Listener"*></listener>

</listeners>

<test name=*"Regression Test"*>

<classes>

<class name=*"TestSuite.TC17\_VerifyRecentActivity"*/>

<class name=*"TestSuite.TC15\_VerifyProfilefromSectionstobecompleted"*/>

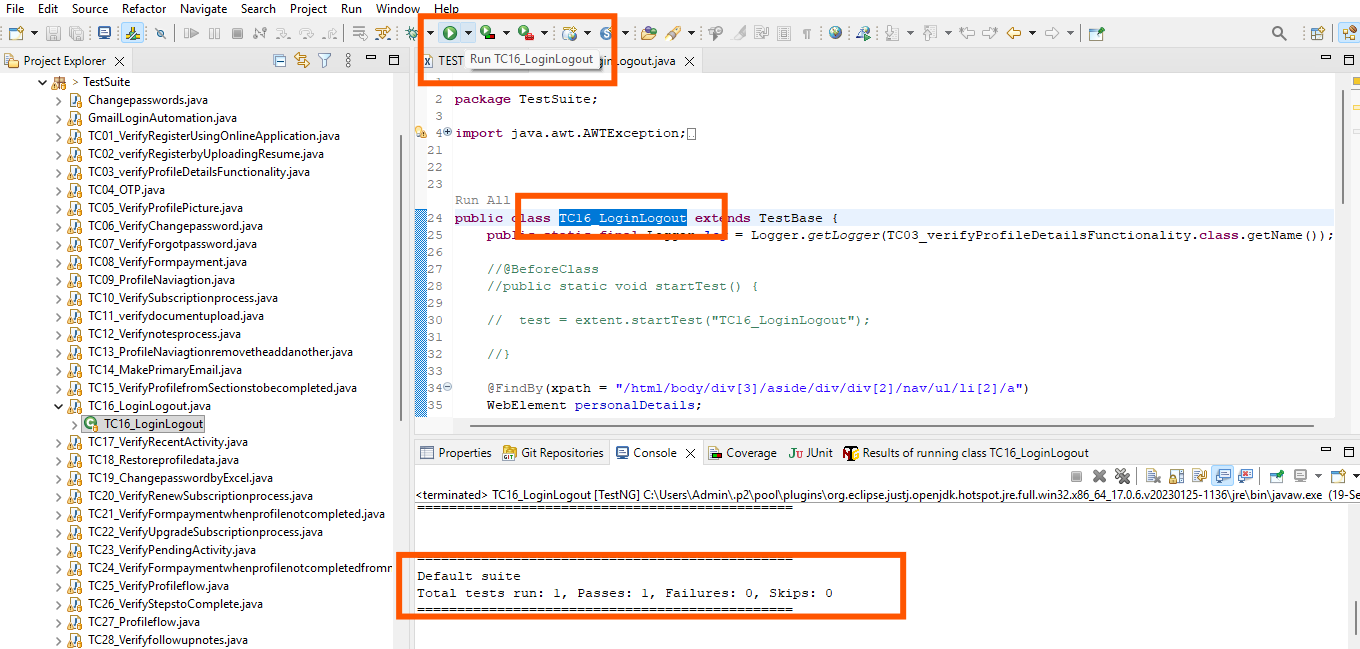
<class name=*"TestSuite.TC16\_LoginLogout"*/>

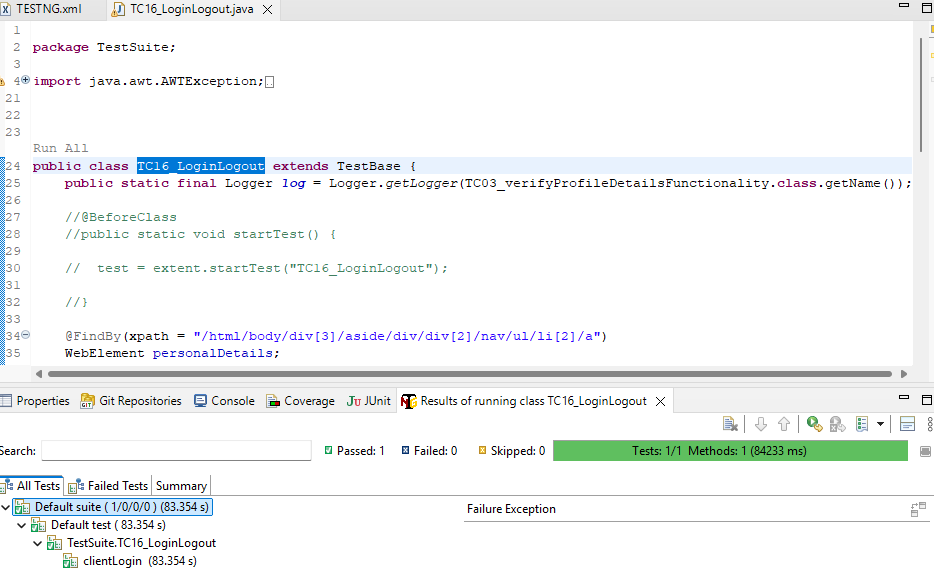
</classes>

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

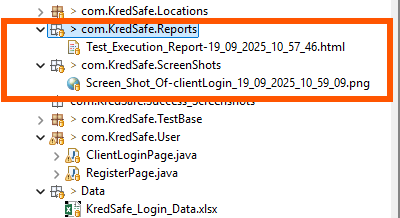
</suite> <!-- AutomationSuite -->

Run Login Logout Test





We can see Report and success screen shot of test result.



-----------------------------------------------------------------------------------------------------------------------

Step-by-Step: Set Up **Extent Reports** with **Selenium in Eclipse**

### What We Need

* **Java**
* **Eclipse IDE**
* **Selenium WebDriver**
* **ExtentReports Library (v5.x or v4.x)**

## Step 1: ****Download and Add ExtentReports Library****

### Option 1: Using Maven (Recommended)

If you are using **Maven** in Eclipse:

1. Right-click project → **Configure → Convert to Maven Project**
2. Open pom.xml and add:
3. <?xml version="1.0" encoding="UTF-8"?>
4. <project xmlns="http://maven.apache.org/POM/4.0.0"
5. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6. xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
7. <modelVersion>4.0.0</modelVersion>
8. <groupId>com.KredSafe.maven</groupId>
9. <artifactId>KredSafe</artifactId>
10. <version>0.0.1-SNAPSHOT</version>
11. <properties>
12. <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
13. <maven.compiler.source>1.7</maven.compiler.source>
14. <maven.compiler.target>1.7</maven.compiler.target>
15. </properties>
16. <dependencies>
17. <!--
18. https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
19. <dependency>
20. <groupId>org.seleniumhq.selenium</groupId>
21. <artifactId>selenium-java</artifactId>
22. <version>4.2.1</version>
23. </dependency>
24. <dependency>
25. <groupId>org.junit.jupiter</groupId>
26. <artifactId>junit-jupiter-api</artifactId>
27. <version>5.7.0</version>
28. <scope>test</scope>
29. </dependency>
30. <!-- https://mvnrepository.com/artifact/org.testng/testng -->
31. <dependency>
32. <groupId>org.testng</groupId>
33. <artifactId>testng</artifactId>
34. <version>7.4.0</version>
35. <scope>test</scope>
36. </dependency>
37. <dependency>
38. <groupId>org.slf4j</groupId>
39. <artifactId>slf4j-api</artifactId>
40. <version>1.7.5</version>
41. </dependency>
42. <dependency>
43. <groupId>org.slf4j</groupId>
44. <artifactId>slf4j-log4j12</artifactId>
45. <version>1.7.5</version>
46. </dependency>
47. <dependency>
48. <groupId>com.relevantcodes</groupId>
49. <artifactId>extentreports</artifactId>
50. <version>2.40.2</version>
51. </dependency>
52. <dependency>
53. <groupId>org.seleniumhq.selenium</groupId>
54. <artifactId>selenium-java</artifactId>
55. <version>3.4.0</version>
56. </dependency>
57. <!--
58. https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-server -->
59. <dependency>
60. <groupId>org.seleniumhq.selenium</groupId>
61. <artifactId>selenium-server</artifactId>
62. <version>3.4.0</version>
63. </dependency>
64. <!-- https://mvnrepository.com/artifact/net.sourceforge.jexcelapi/jxl -->
65. <dependency>
66. <groupId>net.sourceforge.jexcelapi</groupId>
67. <artifactId>jxl</artifactId>
68. <version>2.6.12</version>
69. </dependency>
70. <!-- https://mvnrepository.com/artifact/org.testng/testng -->
71. <dependency>
72. <groupId>org.testng</groupId>
73. <artifactId>testng</artifactId>
74. <version>7.10.2</version>
75. <scope>test</scope>
76. </dependency>
77. <!-- https://mvnrepository.com/artifact/log4j/log4j -->
78. <dependency>
79. <groupId>log4j</groupId>
80. <artifactId>log4j</artifactId>
81. <version>1.2.17</version>
82. </dependency>
83. <!-- https://mvnrepository.com/artifact/org.apache.poi/poi -->
84. <dependency>
85. <groupId>org.apache.poi</groupId>
86. <artifactId>poi</artifactId>
87. <version>3.15</version>
88. </dependency>
89. <!-- https://mvnrepository.com/artifact/org.apache.poi/poi-ooxml -->
90. <dependency>
91. <groupId>org.apache.poi</groupId>
92. <artifactId>poi-ooxml</artifactId>
93. <version>3.15</version>
94. </dependency>
95. <dependency>
96. <groupId>com.relevantcodes</groupId>
97. <artifactId>extentreports</artifactId>
98. <version>2.41.2</version>
99. </dependency>
100. <dependency>
101. <groupId>com.aventstack</groupId>
102. <artifactId>extentreports</artifactId>
103. <version>3.0.7</version>
104. </dependency>
105. <dependency>
106. <groupId>org.apache.maven.plugins</groupId>
107. <artifactId>maven-surefire-plugin</artifactId>
108. <version>2.20.1</version>
109. <type>maven-plugin</type>
110. </dependency>
111. <dependency>
112. <groupId>org.apache.maven.plugins</groupId>
113. <artifactId>maven-compiler-plugin</artifactId>
114. <version>3.7.0</version>
115. <type>maven-plugin</type>
116. </dependency>
117. <dependency>
118. <groupId>javax.mail</groupId>
119. <artifactId>javax.mail-api</artifactId>
120. <version>1.5.4</version>
121. </dependency>
122. <dependency>
123. <groupId>com.sun.mail</groupId>
124. <artifactId>javax.mail</artifactId>
125. <version>1.6.2</version>
126. </dependency>
127. <dependency>
128. <groupId>org.springframework.data</groupId>
129. <artifactId>spring-data-jpa</artifactId>
130. <version>2.7.10</version>
131. </dependency>
132. </dependencies>
133. <build>
134. <pluginManagement>
135. <!-- lock down plugins versions to avoid using Maven defaults (may
136. be moved to parent pom) -->
137. <plugins>
138. <!-- clean lifecycle, see
139. https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean\_Lifecycle -->
140. <plugin>
141. <artifactId>maven-clean-plugin</artifactId>
142. <version>3.1.0</version>
143. </plugin>
144. <!-- default lifecycle, jar packaging: see
145. https://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_jar\_packaging -->
146. <plugin>
147. <artifactId>maven-resources-plugin</artifactId>
148. <version>3.0.2</version>
149. </plugin>
150. <plugin>
151. <artifactId>maven-compiler-plugin</artifactId>
152. <version>3.8.0</version>
153. </plugin>
154. <plugin>
155. <artifactId>maven-surefire-plugin</artifactId>
156. <version>2.22.1</version>
157. </plugin>
158. <plugin>
159. <artifactId>maven-jar-plugin</artifactId>
160. <version>3.0.2</version>
161. </plugin>
162. <plugin>
163. <artifactId>maven-install-plugin</artifactId>
164. <version>2.5.2</version>
165. </plugin>
166. <plugin>
167. <artifactId>maven-deploy-plugin</artifactId>
168. <version>2.8.2</version>
169. </plugin>
170. <!-- site lifecycle, see
171. https://maven.apache.org/ref/current/maven-core/lifecycles.html#site\_Lifecycle -->
172. <plugin>
173. <artifactId>maven-site-plugin</artifactId>
174. <version>3.7.1</version>
175. </plugin>
176. <plugin>
177. <artifactId>maven-project-info-reports-plugin</artifactId>
178. <version>3.0.0</version>
179. </plugin>
180. </plugins>
181. </pluginManagement>
182. </build>
183. </project>

Then **Save** the file and Maven will download the jars.

### Option 2: Manual JARs (Non-Maven)

If you're not using Maven:

1. Go to: <https://github.com/extent-framework/extentreports-java>
2. Download the JAR files (for **v5**, click [Releases](https://github.com/extent-framework/extentreports-java/releases))
3. Add the JARs to your project:
   * Right-click Project → **Build Path → Configure Build Path → Add External JARs**

Step 2

**Add in TestBase Java**

/\*......... Static Block to generate Reports ....................\*/

**static**

{

Calendar calendar=Calendar.*getInstance*();

SimpleDateFormat formater= **new** SimpleDateFormat("dd\_MM\_yyyy\_hh\_mm\_ss");

*extent*=**new** ExtentReports(Locations.*extentReportPath*+formater.format(calendar.getTime())+".html",**false**, DisplayOrder.***NEWEST\_FIRST***);

*extent*

.addSystemInfo("Host Name", "HP Desktop")

.addSystemInfo("Role", "Automation Engineer")

.addSystemInfo("User Name", "Nilesh Shekdar");

*extent*.loadConfig(**new** File(Locations.*extentConfigPath*));

}

------------------------------------

**Add om Location Java**

**public** **class** Locations {

**public** **static** String *geckoDriverPath*=System.*getProperty*("user.dir")+"/drivers/geckodriver.exe";

**public** **static** String *chromeDriverPath*=System.*getProperty*("user.dir")+"/drivers/chromedriver.exe";

**public** **static** String *configFilePath*=System.*getProperty*("user.dir") + "//src//main//java//com//KredSafe//Config//config.properties";

**public** **static** String *extentConfigPath*=System.*getProperty*("user.dir")+"//extent-config.xml";

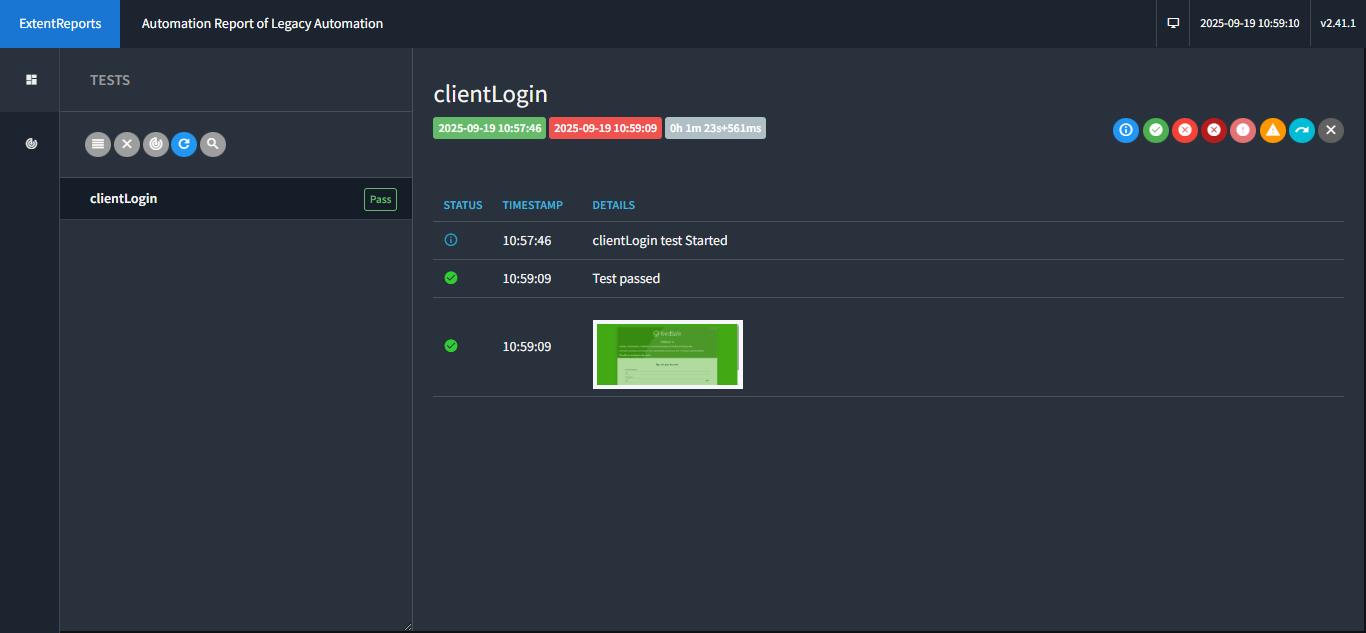
**public** **static** String *extentReportPath*=System.*getProperty*("user.dir")+"//src//main//java//com//KredSafe//Reports//Test\_Execution\_Report-";

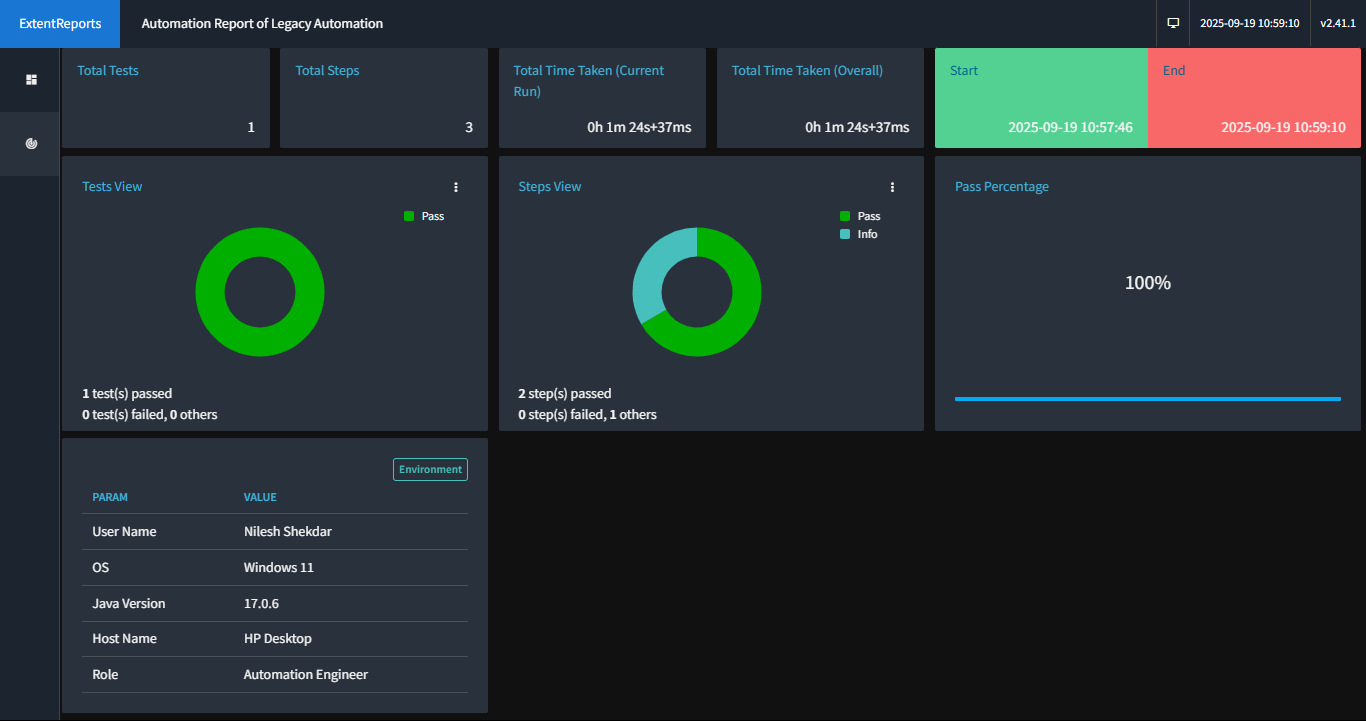
**public** **static** String *captureScreenShotPath*="//src//main//java//com//KredSafe//ScreenShots//Screen\_Shot\_Of-";

**public** **static** String *screenshotlocation*="//src//main//java//com//KredSafe//";

}

---------------------------------------------------------------  
  
Report open in browser





How to Debug Selenium Errors in Eclipse (Java)

### 1. **Read the Stack Trace**

* Eclipse will show the error stack trace in the **Console** window.
* Read from the top: The **Exception type** and the **line number** in your code are key.
* Double-click the line number to jump directly to the code

### 2. **Use Breakpoints & Debug Mode**

* Set breakpoints by **double-clicking in the left margin** next to a line number.
* Right-click your class → Debug As → Java Application.
* Use **Step Over (F6)** and **Step Into (F5)** to move through your code.
* Use the **Variables** and **Expressions** view to inspect values in real-time.

### 3. **Log Messages to Console**

* Use System.out.println("Check Point 1"); or a logger to understand flow.

System.out.println("Trying to click login button");

### 4. **Take Screenshots on Failure**

Helpful to capture the browser state when something goes wrong.

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

FileUtils.copyFile(screenshot, new File("error.png"));

### 5. **Use Implicit or Explicit Waits**

Avoid errors due to timing by using waits properly.

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("username")));

## Common Selenium Errors in Eclipse (Java)

| **Error / Exception** | **Cause** | **How to Fix** |
| --- | --- | --- |
| NoSuchElementException | Element not found in DOM | - Check locator - Use waits |
| ElementNotInteractableException | Element is present but cannot be interacted with | - Wait for visibility - Check if hidden or disabled |
| StaleElementReferenceException | Element is no longer attached to the DOM | - Re-locate the element before interacting |
| TimeoutException | Condition not met in given wait time | - Increase timeout - Verify locator |
| WebDriverException | Driver issue, browser crash, etc. | - Check driver version - Ensure browser is installed correctly |
| SessionNotCreatedException | Mismatch between browser and driver versions | - Update WebDriver and browser |
| InvalidSelectorException | Invalid XPath or CSS selector syntax | - Fix selector string |
| ElementClickInterceptedException | Another element (e.g., modal) is blocking the click | - Use Actions.moveToElement() - Close overlay if present |

## Pro Tips for Selenium Debugging in Eclipse

* Keep your **locators in one place** (e.g., Page Object Model) for easier debugging.
* Always test your **XPath/CSS selectors** in Chrome DevTools before using.
* Keep Selenium, WebDriver, and browser **up-to-date**.
* Use **try-catch blocks** to gracefully handle exceptions and log them.

try {

WebElement loginButton = driver.findElement(By.id("login"));

loginButton.click();

} catch (NoSuchElementException e) {

System.out.println("Login button not found!");

e.printStackTrace();

}

## CI/CD for Selenium

* **CI (Continuous Integration)**: Automatically runs Selenium tests every time code is pushed (e.g., to GitHub).
* **CD (Continuous Deployment/Delivery)**: Optionally deploys the app if tests pass (can be for frontend/backend).

In the context of Selenium:

* Selenium tests run automatically on code changes (CI).
* If tests pass, the app can be deployed (CD).

## Tools We Can Use

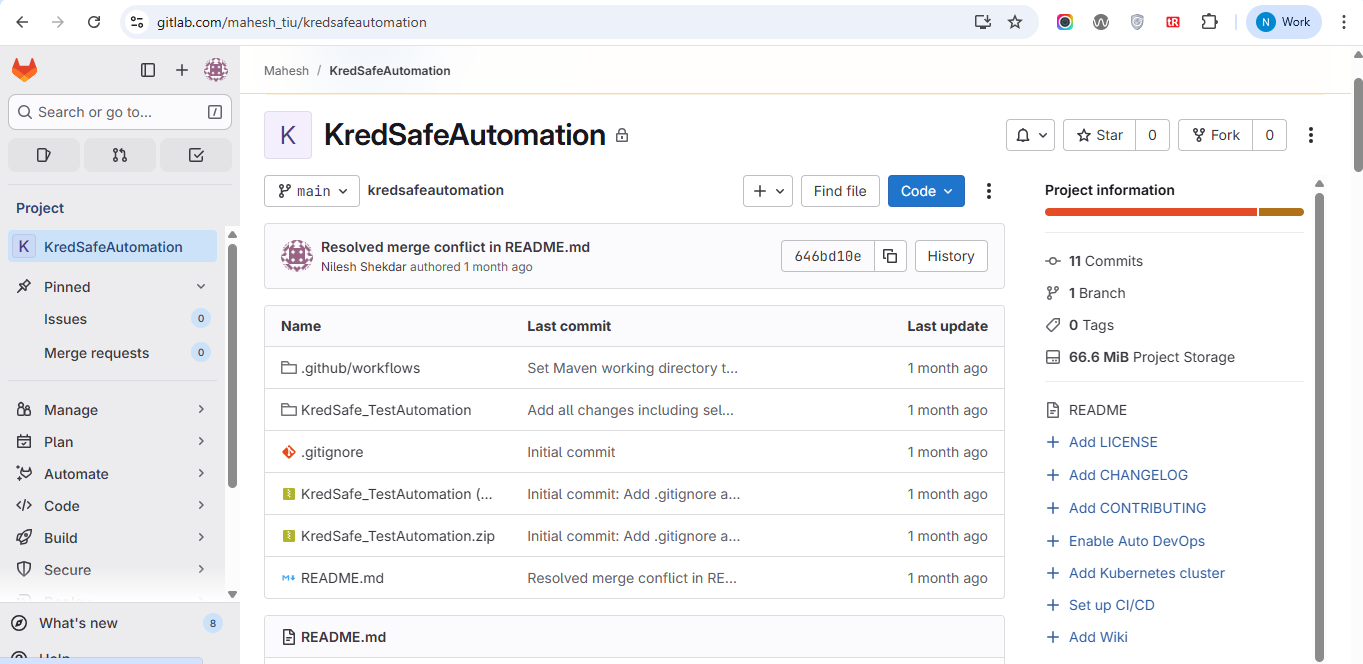
| **Component** | **Tool Example** |
| --- | --- |
| Code Repo | GitHub, GitLab, Bitbucket |
| CI/CD Runner | Jenkins, GitHub Actions, GitLab CI, Azure DevOps |
| Test Framework | JUnit, TestNG (Java) / PyTest (Python) |
| Browser Driver | ChromeDriver, GeckoDriver |
| Headless Browsers | Chrome Headless, Firefox Headless |
| Reporting | Allure, Extent Reports |
| Grid (Optional) | Selenium Grid, Dockerized Browsers |

Step-by-Step: Set Up GitLab CI/CD for Selenium

### 1. **Create a GitLab Repository**

* Go to [gitlab.com](https://gitlab.com/)
* Create a new repository (public or private)
* Clone it locally and add your Selenium project

https://nshekdar@gitlab.com/mahesh\_tiu/kredsafeautomation.git



Flow gitlab/workflows/ci.yml

ci.yml

name: Java CI

on:

push:

branches: [ master, main ]

pull\_request:

branches: [ master, main ]

jobs:

build:

runs-on: windows-latest

steps:

- name: Checkout repository

uses: actions/checkout@v3

- name: Set up JDK 8

uses: actions/setup-java@v3

with:

java-version: '8' # changed from '1.7'

distribution: 'temurin'

- name: Build with Maven

working-directory: KredSafe\_TestAutomation

run: mvn clean install

- name: Run tests

working-directory: KredSafe\_TestAutomation

run: mvn test

-------------------------------------------------------------

git init

git remote add origin https://nshekdar@gitlab.com/mahesh\_tiu/kredsafeautomation.git

git pull origin main

git pull origin main --allow-unrelated-histories

git commit -m "Initial commit"

git push -u origin main

## Line-by-Line Explanation

### 1. git init

* **What it does**: Initializes a new, empty **Git repository** in your current folder.
* This creates a .git directory, making the folder a Git project.

### 2. git remote add origin https://nshekdar@gitlab.com/mahesh\_tiu/kredsafeautomation.git

* **What it does**: Adds a **remote repository** named origin.
* This is the **GitLab repo URL** where you want to push/pull your code.
* origin is just a name (conventionally used for the main remote).

### 3. git pull origin main

* **What it does**: Tries to **pull code from the main branch** of the remote (origin) into your local project.
* This may **fail** if:
  + The local repo is new/empty.
  + There are no shared histories.

### 4. git pull origin main --allow-unrelated-histories

* **What it does**: **Forces a pull** even when the local and remote repositories have **unrelated histories**.
* Useful if:
  + You initialized a local repo (git init) and want to merge it with an existing remote that has commits.
* This avoids Git refusing to merge due to unrelated histories.

### 5. git commit -m "Initial commit"

* **What it does**: Commits your current staged files with the message "Initial commit".
* ⚠️ This will only work **if you have staged files** using git add . before this. If not, nothing will be committed.

### 6. git push -u origin main

* **What it does**: Pushes your local main branch to the remote (origin).
* The -u flag sets the **upstream** so that future git push or git pull commands can be run without specifying origin main.

## 🧠 In Summary

| **Command** | **Purpose** |
| --- | --- |
| git init | Start Git in current folder |
| git remote add | Link local folder to remote GitLab repo |
| git pull origin main | Fetch remote code |
| --allow-unrelated-histories | Allow merge of unrelated projects |
| git commit -m | Save local changes with a message |
| git push -u | Upload your code and set upstream |

## Advantages of Selenium with Eclipse

* **Open-source and free**  
  Both Selenium and Eclipse are free to use, making them ideal for budget-friendly automation solutions.
* **Java integration**  
  Eclipse is one of the best IDEs for Java, and Selenium WebDriver works very well with Java.
* **Plugin support**  
  Eclipse supports many helpful plugins like **Maven**, **TestNG**, **Git**, and **reporting tools**, which streamline test automation workflows.
* **Testing framework integration**  
  Seamlessly integrates with popular testing frameworks like **JUnit**, **TestNG**, and build tools like **Maven**.
* **Built-in debugging tools**  
  Features like **step-through debugging**, **breakpoints**, and **variable inspection** make debugging Selenium tests much easier.
* **Project management**  
  Eclipse helps manage **large-scale test projects** efficiently and supports best practices like the **Page Object Model (POM)**.
* **CI/CD compatibility**  
  Works well with **Git**, **Jenkins**, and **GitLab**, making it easy to integrate Selenium tests into CI/CD pipelines.
* **Auto code-completion and syntax checking**  
  Speeds up development and reduces errors with features like **intelligent code suggestions** and **real-time syntax validation**.

## Disadvantages of Selenium with Eclipse

* Steep learning curve for beginners, especially if new to Java and Eclipse.
* Manual setup required for Selenium libraries, WebDriver binaries, and testing frameworks.
* Eclipse can become slow or unstable with large projects or many plugins installed.
* No built-in browser; requires external browsers and drivers for testing.
* Managing test reports, browser drivers, and headless execution for CI/CD can be complex.
* No visual test recorder; all tests must be coded manually.
* Frequent updates needed for browser drivers to match browser versions.
* Heavy reliance on plugins like TestNG and Maven, which can cause configuration issues if not managed well.

**Recommended For**

* Testers and developers comfortable with **Java programming**.
* Teams building **robust and maintainable test automation frameworks** (e.g., using Page Object Model).
* Projects that require integration with **JUnit**, **TestNG**, and **Maven**.
* Automation engineers wanting to use **advanced debugging and project management** features.
* Teams planning to integrate tests into **CI/CD pipelines** with tools like GitLab, Jenkins, or Git.
* Users who want a **free, open-source** solution with a powerful IDE.

**Not Recommended For**

* Beginners with **no programming experience** or unfamiliar with Java.
* Testers looking for **no-code or low-code** test automation tools.
* Small or quick one-off testing tasks where lightweight tools like **Selenium IDE** or **Python scripts** might be easier.
* Users who want **faster, simpler setup** without manual configuration.
* Projects that need **visual test recording or codeless automation**.
* Users who experience performance issues with Eclipse or want a more modern IDE experience (might prefer IntelliJ IDEA or VS Code).

## Conclusion: Selenium with Eclipse

Using **Selenium with Eclipse** is a powerful combination for building professional, Java-based test automation frameworks. It is widely used in enterprise environments due to its flexibility, plugin ecosystem, and strong support for debugging and CI/CD integration.

However, it also comes with a **steeper learning curve**, and may not be the best choice for beginners or small-scale testing needs.

### In Summary:

* **Best for**: Experienced testers and developers working on Java-based automation with long-term goals and CI/CD integration.
* **Not ideal for**: Beginners, quick testing tasks, or teams looking for low-code/codeless solutions.

If you're comfortable with Java and looking for full control over your test framework, **Selenium with Eclipse is a solid choice**. For simpler or faster setups, alternatives like **Selenium with Python**, **Playwright**, or **Selenium IDE** might be more suitable.