Project Name: World Clock

Project documentation:

Code implementation:

DriverSetUp.java

package WorldClock;

import java.util.Scanner;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.edge.EdgeDriver;

public class DriverSetUp {

// Variable of type WebDriver

public static WebDriver driver;

// Method to Handle the Driver of different types

public static WebDriver getWebDriver(String url) {

System.out.println("Choose the browser below");

System.out.println("1.Chrome");

System.out.println("2.Edge");

System.out.println("Enter the option number: ");

Scanner sc=new Scanner(System.in);

byte b=sc.nextByte();

if(b==1) {

// Creating the Object of the ChromeDriver

driver=new ChromeDriver();

driver.get(url);

}else if(b==2) {

// Creating the Object of the EdgeDriver

driver=new EdgeDriver();

driver.get(url);

}

sc.close();

// Returning the Driver

return driver;

} }

Clock.java

package WorldClock;

import java.time.Duration;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

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

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

public class Clock {

public WebDriver driver;

public WebDriverWait wait;

Clock (WebDriver driver){

this.driver=driver;

PageFactory.initElements(driver, this);

}

// \*\*\*\*\* User Info \*\*\*\*\*\*

public void clickOnUserIcon() throws InterruptedException {

wait = new WebDriverWait(driver, Duration.ofSeconds(30));

// Selecting xpath of the div tag containing img tag

WebElement user= wait.until(ExpectedConditions.visibilityOfElementLocated((By.id("O365\_MainLink\_MePhoto"))));

// Waiting Until The Setting Icon is visible

wait.until(ExpectedConditions.visibilityOfElementLocated((By.xpath("//\*[@id='O365\_MainLink\_Settings']/div"))));

Thread.sleep(3000);

user.click();

}

// @FindBy(xpath="//\*[@id='mectrl\_headerPicture']")

@FindBy(id ="mectrl\_headerPicture")

WebElement clickback;

public void clickBackUser() {

clickback.click();

}

@FindBy(xpath="//\*[@id='mectrl\_currentAccount\_primary']")

WebElement username;

public String getUserName() {

return username.getText();

}

@FindBy(xpath="//\*[@id='mectrl\_currentAccount\_secondary']")

WebElement userId;

public String getUserId() {

return userId.getText();

}

// \*\*\*\*\*\*\* World Clock \*\*\*\*\*\*\*

// Validating the World Clock

@FindBy(xpath="//\*[@id='CaptionElementView' and contains(text(),'World Clock')]")

WebElement Title\_clock;

public String checkClockBox(){

String present=Title\_clock.getText();

return present;

}

//===================== Bangalore, India (IST) =============================

@FindBy(xpath="(//\*[@data-automation-id=\"clock-card-location\"])[1]")

WebElement indiaLocation;

@FindBy(xpath="//\*[@id=\"vpc\_WebPart.WorldClockWebPart.internal.60655e4a-73c8-49d0-9571-c762791557af\"]/div/div/div[2]/div/div/div/div/div/div/div[1]/div/div/div/div[2]/div[1]")

WebElement indiaTime;

@FindBy(xpath="//\*[@id=\"vpc\_WebPart.WorldClockWebPart.internal.60655e4a-73c8-49d0-9571-c762791557af\"]/div/div/div[2]/div/div/div/div/div/div/div[1]/div/div/div/div[2]/div[2]/div[2]")

WebElement India\_Day\_date;

public String checkIndiaClock() {

String IndiaTitle=indiaLocation.getText();

return IndiaTitle;

}

public String checkIndiaTime() {

return indiaTime.getText();

}

public String chechIndiaDate() {

return India\_Day\_date.getText();

}

//========================================== London ===================================

@FindBy(xpath="(//\*[@data-automation-id='clock-card-location'])[2]")

WebElement londonLocation;

@FindBy(xpath="//\*[@id=\"vpc\_WebPart.WorldClockWebPart.internal.60655e4a-73c8-49d0-9571-c762791557af\"]/div/div/div[2]/div/div/div/div/div/div/div[2]/div/div/div/div[2]/div[1]/span[1]")

WebElement LondonTime;

@FindBy(xpath="//\*[@id=\"vpc\_WebPart.WorldClockWebPart.internal.60655e4a-73c8-49d0-9571-c762791557af\"]/div/div/div[2]/div/div/div/div/div/div/div[2]/div/div/div/div[2]/div[2]/div[2]")

WebElement lon\_date;

@FindBy(xpath="//\*[@id=\"vpc\_WebPart.WorldClockWebPart.internal.60655e4a-73c8-49d0-9571-c762791557af\"]/div/div/div[2]/div/div/div/div/div/div/div[2]/div/div/div/div[2]/div[2]/div[1]")

WebElement London\_India\_Gap;

public String checkLondonClock() {

String lontitle=londonLocation.getText();

return lontitle;

}

public String checkLondonTime() {

String lontime = LondonTime.getText();

return lontime;

}

public String checkLondonDate() {

String londate = lon\_date.getText();

return londate;

}

public String checkLondonGapTime() {

String longap=London\_India\_Gap.getText();

return longap;

}

//============================================= NewYork =====================================

@FindBy(xpath="(//\*[@data-automation-id='clock-card-location'])[3]")

WebElement NewYorkLocation;

@FindBy(xpath="//\*[@id=\"vpc\_WebPart.WorldClockWebPart.internal.60655e4a-73c8-49d0-9571-c762791557af\"]/div/div/div[2]/div/div/div/div/div/div/div[3]/div/div/div/div[2]/div[1]/span[1]")

WebElement NewYorkTime;

@FindBy(xpath="//\*[@id=\"vpc\_WebPart.WorldClockWebPart.internal.60655e4a-73c8-49d0-9571-c762791557af\"]/div/div/div[2]/div/div/div/div/div/div/div[3]/div/div/div/div[2]/div[2]/div[2]")

WebElement ny\_date;

@FindBy(xpath="//\*[@id=\"vpc\_WebPart.WorldClockWebPart.internal.60655e4a-73c8-49d0-9571-c762791557af\"]/div/div/div[2]/div/div/div/div/div/div/div[3]/div/div/div/div[2]/div[2]/div[1]")

WebElement NewYorkIndiaGap;

public String newYorkClock() {

String nytitle=NewYorkLocation.getText();

return nytitle;

}

public String checkNewYorkTime() {

String nytime = NewYorkTime.getText();

return nytime;

}

public String checkNewYorkDate() {

String nydate = ny\_date.getText();

return nydate;

}

public String checkNewYorkIndiaGapTime() {

String nygap=NewYorkIndiaGap.getText();

return nygap;

}

}

OneCognizantApps.java

package WorldClock;

import java.io.IOException;

import java.util.List;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class OneCognizantApps {

public WebDriver driver;

public ExcelUtils eo=new ExcelUtils();

OneCognizantApps(WebDriver driver) {

this.driver=driver;

PageFactory.initElements(driver,this);

}

// @FindBy (xpath="(//div[@id='QuicklinksItemTitle'])[5]") WebElement ele;

@FindBy (xpath="//div[@id='QuicklinksItemTitle' and contains(text(),\"OneCognizant\")]") WebElement ele;

@FindBy (xpath="//div[@class='viewAllHotAppsBtn']") WebElement ele2;

@FindBy (xpath="//div[@class ='aZHolder']/div") List<WebElement> ele3;

public void oneCognizantClick()

{

ele.click();

}

public void hotAppsScroll()

{

JavascriptExecutor js = (JavascriptExecutor)driver;

js.executeScript("arguments[0].scrollIntoView();",ele2);

}

public void hotAppsViewAllClick() {

JavascriptExecutor js = (JavascriptExecutor)driver;

js.executeScript("arguments[0].click();", ele2);

}

@FindBy (xpath="//\*[@id='div\_appFilteredList']/div/div") List<WebElement> apps;

@FindBy(xpath="//\*[@id='div\_appFilteredList']/div/div/div/div[@class='appStoreAppText valign-wrapper']/div")

List<WebElement> app;

public void printAppsName() throws IOException, InterruptedException {

eo.OneCognizantData(app);

}

}

ExcelUtils.java

package WorldClock;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.util.List;

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.WebElement;

public class ExcelUtils {

public void TestExcelData(String str, int row) throws IOException, InterruptedException {

String filePath = System.getProperty("user.dir") + "\\target\\OutputDetails.xlsx";

XSSFWorkbook workbook;

XSSFSheet sheet;

File file = new File(filePath);

if (file.exists()) {

FileInputStream fis = new FileInputStream(file);

workbook = new XSSFWorkbook(fis);

sheet = workbook.getSheetAt(0);

fis.close();

} else {

workbook = new XSSFWorkbook();

sheet = workbook.createSheet("Sheet1");

}

XSSFRow currentRow = sheet.createRow(row);

currentRow.createCell(0).setCellValue(str);

FileOutputStream fos = new FileOutputStream(file);

workbook.write(fos);

workbook.close();

fos.close();

}

public void OneCognizantData(List<WebElement> apps) throws IOException, InterruptedException {

Thread.sleep(3000);

FileOutputStream file=new FileOutputStream(System.getProperty("user.dir")+"\\target\\OneCognizantDetails.xlsx");

XSSFWorkbook workbook=new XSSFWorkbook();

XSSFSheet sheet=workbook.createSheet();

for(int r=0;r<apps.size();r++)

{

XSSFRow currentrow=sheet.createRow(r);

String value=apps.get(r).getText();

System.out.println(value);

currentrow.createCell(0).setCellValue(value);

}

workbook.write(file);

workbook.close();

file.close();

}

}

TestMain.java

package WorldClock;

import java.io.File;

import java.io.IOException;

import java.text.SimpleDateFormat;

import java.time.Duration;

import java.time.LocalDate;

import java.time.format.DateTimeFormatter;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.List;

import java.util.Random;

import java.util.Set;

import java.util.TimeZone;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.By;

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.support.ui.ExpectedConditions;

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

import org.testng.Assert;

import org.testng.annotations.AfterSuite;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.Listeners;

import org.testng.annotations.Test;

import WorldClock.Clock;

import WorldClock.DriverSetUp;

import WorldClock.ExcelUtils;

import WorldClock.OneCognizantApps;

@Listeners(WorldClock.ExtentReportManager.class)

//Here Listeners Annotation take ExtentReportManager as a paremeter This means that the ExtentReportManager class will listen to the events of the test methods in the WorldClock class, allowing it to generate reports based on those events.

public class TestMain {

public static WebDriver driver;

public static Clock wc;

public static String webtime;

public static OneCognizantApps one;

public static ExcelUtils ex=new ExcelUtils();

public static void screenshot(String name) {

TakesScreenshot ss = ((TakesScreenshot) driver);

File src=ss.getScreenshotAs(OutputType.FILE);

try {

File target=new File("./reports/Screenshot/"+name+".png");

FileUtils.copyFile(src, target);

} catch (IOException e) {

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

}

}

@BeforeSuite

public void openPage() {

driver=DriverSetUp.getWebDriver("https://be.cognizant.com/");

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

driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(30));

driver.manage().deleteAllCookies();

System.out.println("OpenPage is Successfully Passed");

}

@Test(priority=1)

public void userInfo() throws InterruptedException, IOException {

wc=new Clock(driver);

wc.clickOnUserIcon();

Thread.sleep(3000);

TestMain.screenshot("userProfile");

// Printing User Data

System.out.println("---- 1.UserData ----");

System.out.println(wc.getUserName());

System.out.println(wc.getUserId());

ex.TestExcelData("1.UserData", 0);

ex.TestExcelData(wc.getUserName(), 1);

ex.TestExcelData(wc.getUserId(), 2);

wc.clickBackUser();

// Scrolling Down By targeting the(SeeAll) using xpath

WebElement seeAll=driver.findElement(By.xpath("//\*[@id=\"spPageCanvasContent\"]/div/div/div/div/div/div/div[2]/div/div[1]"));

JavascriptExecutor js=(JavascriptExecutor)driver;

js.executeScript("arguments[0].scrollIntoView();",seeAll);

System.out.println("UserInfo is Successfully Captured");

ex.TestExcelData("UserInfo is Successfully Captured", 3);

}

@Test(priority = 2)

public static void Test\_WorldClock\_Title() throws IOException, InterruptedException {

wc=new Clock(driver);

System.out.println("---- 2.WorldClock ----");

ex.TestExcelData("2.WorldClock", 4);

System.out.println("Captured Text From WebPage:"+wc.checkClockBox().toLowerCase());

ex.TestExcelData(wc.checkClockBox().toLowerCase(), 5);

Assert.assertEquals(wc.checkClockBox().toLowerCase(), "world clock");

TestMain.screenshot("worldclock");

System.out.println("Test\_WorldClock\_Title is Successfully Passed");

ex.TestExcelData("Test\_WorldClock\_Title is Successfully Passed",6);

}

//======================== India Validation ============================

@Test(priority = 3)

void bangloreTimeZone() throws IOException, InterruptedException {

// IST

Date d = new Date();

SimpleDateFormat t=new SimpleDateFormat("z");

String a=(t.format(d)).toUpperCase();

System.out.println("System TimeZone:"+a);

ex.TestExcelData("System TimeZone:"+a, 7);

wc=new Clock(driver);

String TimeZone=wc.checkIndiaClock();

int start = TimeZone.indexOf("(") + 1;

int end = TimeZone.indexOf(")");

String result = (TimeZone.substring(start, end)).toUpperCase();

System.out.println("WebPage TimeZone:"+result);

ex.TestExcelData("WebPage TimeZone:"+result, 8);

Assert.assertEquals(result, a);

System.out.println("bangloreTimeZone is Successfully Passed");

ex.TestExcelData("bangloreTimeZone is Successfully Passed", 9);

}

@Test(priority = 4)

void bangaloreWatchTitle() throws IOException, InterruptedException {

System.out.println(wc.checkIndiaClock());

Assert.assertEquals(wc.checkIndiaClock().toLowerCase(), "bangalore, india (ist)");

System.out.println("bangaloreWatchTitle is Successfully Passed");

ex.TestExcelData(wc.checkIndiaClock(), 10);

ex.TestExcelData("bangaloreWatchTitle is Successfully Passed", 11);

}

@Test(priority = 5)

public static void bangaloreTime() throws IOException, InterruptedException {

// Getting the date

Date currentTime = new Date();

// Formatting the in hours:minutes:(pm/am)

SimpleDateFormat timeformat=new SimpleDateFormat("h:mma");

String formatedtime=timeformat.format(currentTime);

System.out.println("Time of System:"+formatedtime.toLowerCase());

webtime=wc.checkIndiaTime();

System.out.println("Time of WebPage:"+webtime.toLowerCase());

Assert.assertEquals(webtime.toLowerCase(), formatedtime.toLowerCase());

System.out.println("bangaloreTime is Successfully Passed");

ex.TestExcelData("Time of System:"+formatedtime.toLowerCase(),12);

ex.TestExcelData("Time of WebPage:"+webtime.toLowerCase(),13);

ex.TestExcelData("bangaloreTime is Successfully Passed",12);

}

@Test(priority = 6)

public static void bangaloreDate() throws IOException, InterruptedException {

// Getting the local date -> yyyy-mm-dd

LocalDate currentSysDate=LocalDate.now();

// Formatting the date in the desired pattern

DateTimeFormatter date\_formatter=DateTimeFormatter.ofPattern("EEEE, M/d/yyyy");

// Converting the date as per the format

String formattedDate=currentSysDate.format(date\_formatter);

System.out.println("Date of System:"+formattedDate.toLowerCase());

String webdate=wc.chechIndiaDate();

System.out.println("Date of WebPage:"+webdate.toLowerCase());

Assert.assertEquals(webdate.toLowerCase(), formattedDate.toLowerCase());

System.out.println("bangaloreDate is Successfully Passed");

ex.TestExcelData("Date of System:"+formattedDate.toLowerCase(), 13);

ex.TestExcelData("Date of WebPage:"+webdate.toLowerCase(), 14);

ex.TestExcelData("bangaloreDate is Successfully Passed", 15);

}

// ======================== London Validation ============================

@Test(priority = 7)

void londonWatchTitle() {

Assert.assertEquals(wc.checkLondonClock().toLowerCase(), "london, uk (bst)");

}

@Test(priority = 8)

void londonTime() throws IOException, InterruptedException {

wc=new Clock(driver);

TimeZone.setDefault(TimeZone.getTimeZone("Europe/London"));

SimpleDateFormat time = new SimpleDateFormat("h:mm");

Date Stime = new Date();

String time\_lon = time.format(Stime);

System.out.println("London System time:"+wc.checkLondonTime());

System.out.println("London Webpage time:"+time\_lon);

Assert.assertEquals(time\_lon,wc.checkLondonTime());

System.out.println("londonTime is Successfully Passed");

ex.TestExcelData("London System time:"+wc.checkLondonTime(),16);

ex.TestExcelData("London Webpage time:"+time\_lon,17);

ex.TestExcelData("londonTime is Successfully Passed",18);

}

@Test(priority = 9)

void londonDate() throws IOException, InterruptedException {

TimeZone.setDefault(TimeZone.getTimeZone("Europe/London"));

SimpleDateFormat date = new SimpleDateFormat("EEEE, M/d/yyyy");

Date date\_ = new Date();

String date\_lon = date.format(date\_);

System.out.println("London Webpage date:"+wc.checkLondonDate());

System.out.println("London System date:"+date\_lon);

Assert.assertEquals(date\_lon,wc.checkLondonDate());

System.out.println("londonDate is Successfully Passed");

ex.TestExcelData("London Webpage date:"+wc.checkLondonDate(), 19);

ex.TestExcelData("London System date:"+date\_lon, 20);

ex.TestExcelData("londonDate is Successfully Passed", 21);

}

@Test(priority = 10)

void londonTimeGap() throws IOException, InterruptedException {

TimeZone bangloreTimeZone = TimeZone.getTimeZone("Asia/Kolkata");

TimeZone LondonTimeZone = TimeZone.getTimeZone("Europe/London");

Calendar cal = Calendar.getInstance();

int hoursDifference = (bangloreTimeZone.getRawOffset() - LondonTimeZone.getRawOffset()) / (60 \* 60 \* 1000);

if (LondonTimeZone.inDaylightTime(cal.getTime())) {

hoursDifference--;

}

int minutesDifference = (bangloreTimeZone.getRawOffset() - LondonTimeZone.getRawOffset()) / (60 \* 1000) % 60;

String lonbanggap = hoursDifference + "h " + minutesDifference + "m " + "behind";

System.out.println("London System Time-Gap:" + lonbanggap);

System.out.println("London Webpage Time-Gap:" + wc.checkLondonGapTime());

Assert.assertEquals(lonbanggap, wc.checkLondonGapTime());

System.out.println("londonTimeGap is Successfully Passed");

ex.TestExcelData("London System Time-Gap:" + lonbanggap, 22);

ex.TestExcelData("London Webpage Time-Gap:" + wc.checkLondonGapTime(), 23);

ex.TestExcelData("londonTimeGap is Successfully Passed", 24);

}

// ======================== NewYork Validation ============================

@Test (priority = 11)

void NewYorkWatchTitle() throws IOException, InterruptedException {

Assert.assertEquals(wc.newYorkClock().toLowerCase(), "new york, ny (est)");

System.out.println("NewYorkWatchTitle is Successfully Passed");

ex.TestExcelData("NewYorkWatchTitle is Successfully Passed", 25);

}

@Test(priority = 12)

void NewYorkTime() throws IOException, InterruptedException {

TimeZone.setDefault(TimeZone.getTimeZone("America/New\_York"));

SimpleDateFormat time = new SimpleDateFormat("h:mm");

Date time\_ = new Date();

String time\_NY = time.format(time\_);

System.out.println("NewYork System time:"+wc.checkNewYorkTime());

System.out.println("NewYork WebPage time:"+time\_NY);

Assert.assertEquals(time\_NY,wc.checkNewYorkTime());

System.out.println("NewYorkTime is Successfully Passed");

ex.TestExcelData("NewYork System time:"+wc.checkNewYorkTime(), 26);

ex.TestExcelData("NewYork WebPage time:"+time\_NY, 27);

ex.TestExcelData("NewYorkTime is Successfully Passed", 28);

}

@Test (priority = 13)

void NewYorkDate() throws IOException, InterruptedException {

TimeZone.setDefault(TimeZone.getTimeZone("America/New\_York"));

SimpleDateFormat date = new SimpleDateFormat("EEEE, M/d/yyyy");

Date date\_ = new Date();

String date\_NY = date.format(date\_);

System.out.println("NewYork WebPage Date:"+wc.checkNewYorkDate());

System.out.println("NewYork System Date:"+date\_NY);

Assert.assertEquals(date\_NY,wc.checkNewYorkDate());

System.out.println("NewYorkDate is Successfully Passed");

ex.TestExcelData("NewYork WebPage Date:"+wc.checkNewYorkDate(), 29);

ex.TestExcelData("NewYork System Date:"+date\_NY, 30);

ex.TestExcelData("NewYorkDate is Successfully Passed", 31);

}

@Test (priority = 14)

void NewYorkTimeGap() throws IOException, InterruptedException {

TimeZone bangloreTimeZone = TimeZone.getTimeZone("Asia/Kolkata");

TimeZone newYorkTimeZone = TimeZone.getTimeZone("America/New\_York");

long currentTime = System.currentTimeMillis();

int hoursDifference = (bangloreTimeZone.getOffset(currentTime) - newYorkTimeZone.getOffset(currentTime)) / (60 \* 60 \* 1000);

int minutesDifference = Math.abs((bangloreTimeZone.getOffset(currentTime) - newYorkTimeZone.getOffset(currentTime)) / (60 \* 1000) % 60);

String NYbanggap = hoursDifference + "h " + minutesDifference + "m " + "behind";

System.out.println("NewYork System Time-Gap"+NYbanggap);

System.out.println("NewYork WebPage Time-Gap"+wc.checkNewYorkIndiaGapTime());

Assert.assertEquals(NYbanggap, wc.checkNewYorkIndiaGapTime());

TimeZone.setDefault(TimeZone.getTimeZone("Asia/Kolkata"));

System.out.println("NewYorkTimeGap is Successfully Passed");

ex.TestExcelData("NewYork System Time-Gap"+NYbanggap,32);

ex.TestExcelData("NewYork WebPage Time-Gap"+wc.checkNewYorkIndiaGapTime(),33);

ex.TestExcelData("NewYorkTimeGap is Successfully Passed",34);

}

// ===================== One Cognizant ===========================

@Test(priority=15)

public void scroll() {

WebElement appAndTool=driver.findElement(By.xpath("(//\*[@id=\"CaptionElementView\"])[2]"));

JavascriptExecutor j= (JavascriptExecutor)driver;

j.executeScript("arguments[0].scrollIntoView();",appAndTool);

TestMain.screenshot("onecognizant");

}

@Test(priority=16)

public void viewApps() throws InterruptedException{

//view all apps

one=new OneCognizantApps(driver);

one.oneCognizantClick();

Set<String> windowid = driver.getWindowHandles();

List<String> windowsidList = new ArrayList<String> (windowid);

for (int i =0;i<windowsidList.size();i++) {

String title=driver.switchTo().window(windowsidList.get(i)).getTitle();

if (title.equals("OneCognizant")) {

break;

}

}

one.hotAppsScroll();

Thread.sleep(2000);

TestMain.screenshot("hotsapps");

one.hotAppsViewAllClick();

}

@Test(priority=17)

public void appsAlphabet() throws IOException, InterruptedException {

//disabled alphabet

one=new OneCognizantApps(driver);

int i=60;

List<String> str= new ArrayList<String>();

for(WebElement e:one.ele3) {

if(e.getAttribute("role")!=null) {

continue;

}

else

str.add(e.getText());

}

for(String s: str) {

System.out.println(s+" is disabled");

ex.TestExcelData(s+" is disabled", i);

i++;

}

}

@Test(priority=18)

void randomAlphabet() throws InterruptedException, IOException {

Random rand=new Random();

List<WebElement> list = one.ele3;

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

while(true) {

int n=rand.nextInt(list.size()); // 0 to (size of list) generate random numbers

if(list.get(n).equals("X")) {

continue;

}else if(list.get(n).equals("Y")) {

continue;

}else {

// Waiting to implicit wait load

driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(60));

// Create a new WebDriverWait instance

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(30));

// Wait until the element is clickable

WebElement element = wait.until(ExpectedConditions.elementToBeClickable(list.get(n)));

// Click the element

element.click();

Thread.sleep(2000);

TestMain.screenshot("randomclick");

System.out.println("\*\*\*\*\*\* Apps of Random Click \*\*\*\*\*\*\*");

one.printAppsName();

break;

}

}

}

@AfterSuite

void closeBrowser() throws InterruptedException {

Thread.sleep(1000);

driver.quit();

}

}

ExtentReportManager.java

package WorldClock;

import org.testng.ITestContext;

import org.testng.ITestListener;

import org.testng.ITestResult;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import com.aventstack.extentreports.reporter.configuration.Theme;

public class ExtentReportManager implements ITestListener

{

public ExtentSparkReporter sparkReporter; // UI of the report

public ExtentReports extent; //populate common info on the report

public ExtentTest test; // creating test case entries in the report and update status of the test methods

public void onStart(ITestContext context) {

sparkReporter=new ExtentSparkReporter(System.getProperty("user.dir")+ "/reports/myReport.html");//specify location of the report

sparkReporter.config().setDocumentTitle("Ankit Automation Report"); // TiTle of report

sparkReporter.config().setReportName("Functional Testing"); // name of the report

sparkReporter.config().setTheme(Theme.DARK);

extent=new ExtentReports();

extent.attachReporter(sparkReporter);

extent.setSystemInfo("Computer Name","localhost");

extent.setSystemInfo("Tester Name","Ankit Tiwari");

extent.setSystemInfo("Environment"," https://be.cognizant.com");

extent.setSystemInfo("os","Windows11");

extent.setSystemInfo("Browser name","Chrome,Edge");

}

public void onTestSuccess(ITestResult result) {

test = extent.createTest(result.getName()); // create a new entry in the report

test.log(Status.PASS, "Test case PASSED is:" + result.getName()); // update status p/f/s

}

public void onTestFailure(ITestResult result) {

test = extent.createTest(result.getName());

test.log(Status.FAIL, "Test case FAILED is:" + result.getName());

test.log(Status.FAIL, "Test Case FAILED cause is: " + result.getThrowable());

}

public void onTestSkipped(ITestResult result) {

test = extent.createTest(result.getName());

test.log(Status.SKIP, "Test case SKIPPED is:" + result.getName());

}

public void onFinish(ITestContext context) {

extent.flush();

}

}

Screenshots:

TestOutput1:

A screenshot of a computer

Description automatically generated

TestOutput2:

A screenshot of a computer

Description automatically generated

TestOutput3:

A screenshot of a computer

Description automatically generated

TestOutput4:

A screenshot of a computer

Description automatically generated

TestOutput5:

A screenshot of a computer

Description automatically generated

Console Output:

[RemoteTestNG] detected TestNG version 7.9.0

SLF4J(W): No SLF4J providers were found.

SLF4J(W): Defaulting to no-operation (NOP) logger implementation

SLF4J(W): See https://www.slf4j.org/codes.html#noProviders for further details.

Choose the browser below

1.Chrome

2.Edge

Enter the option number:

1

May 02, 2024 10:31:14 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch

WARNING: Unable to find an exact match for CDP version 123, returning the closest version; found: 122; Please update to a Selenium version that supports CDP version 123

OpenPage is Successfully Passed

---- 1.UserData ----

Kumar, Piyush (Contractor)

2317696@cognizant.com

ERROR StatusLogger Log4j2 could not find a logging implementation. Please add log4j-core to the classpath. Using SimpleLogger to log to the console...

UserInfo is Successfully Captured

---- 2.WorldClock ----

Captured Text From WebPage:world clock

Test\_WorldClock\_Title is Successfully Passed

System TimeZone:IST

WebPage TimeZone:IST

bangloreTimeZone is Successfully Passed

Bangalore, India (IST)

bangaloreWatchTitle is Successfully Passed

Time of System:10:31am

Time of WebPage:10:31am

bangaloreTime is Successfully Passed

Date of System:thursday, 5/2/2024

Date of WebPage:thursday, 5/2/2024

bangaloreDate is Successfully Passed

London System time:6:01

London Webpage time:6:01

londonTime is Successfully Passed

London Webpage date:Thursday, 5/2/2024

London System date:Thursday, 5/2/2024

londonDate is Successfully Passed

London System Time-Gap:4h 30m behind

London Webpage Time-Gap:4h 30m behind

londonTimeGap is Successfully Passed

NewYorkWatchTitle is Successfully Passed

NewYork System time:1:01

NewYork WebPage time:1:01

NewYorkTime is Successfully Passed

NewYork WebPage Date:Thursday, 5/2/2024

NewYork System Date:Thursday, 5/2/2024

NewYorkDate is Successfully Passed

NewYork System Time-Gap9h 30m behind

NewYork WebPage Time-Gap9h 30m behind

NewYorkTimeGap is Successfully Passed

X is disabled

Y is disabled

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\* Apps of Random Click \*\*\*\*\*\*\*

Clearance Automation

Client Visit Experience

Customer Holidays

Chire(Oracle Taleo)

Contract Concourse

Corporate Card

Corporate Services Dashboard

CAPPS(Ariba)

CBOM

Cognizant Legislative Solution

CDS

CATS(Spire)

Catalyst

Cognizant Cheers

CS PMO Dashboard

Contractor Dashboard

CWR 360

Cognizant 360

CS GRC

CogCOM

Cognizant Advantage

Customer Experience

CADET

Careersite Analytics(Phenom)

Client Conversations

CAT PMC

CWorkspace

ConnectMe

CInsights

Cognizant for Good

CoStar

Cisco DNAC

CIO Asset Management

Career Site

Cohesity Helios

Cisco Intersight

Cisco APIC

Cisco RTMT

Cognizant Discounts

Cvent

CSS Analytics

ClarizenOne

Cloud Academy

CRecruit

Codebase Hungary

Corrigo

Compliance and Controls Management

CertificateAuthority

Cloud Orchestration

Cisco Umbrella

Cloud Apps Defender

Canada employer site

Cisco Threat Grid

CDL Opportunity Tracker

Checkmarx SAST - TriZetto-TPS

Cribl Data Stream Processing

Cofense Vision and Triage

Cisco AnyConnect for SFDC project

Citrix-Web Application Firewall- HS

CPES Survey

Cognizant Clinical Data Insights

CloudHedge OmniDeq\_cmp

Cognizant discounts - NZ

Cardinus Health Working

Cisco Meraki

Cofense Vision

Cofense PhishMe

Corporate Security Incident-Ticket

Cognizant Health Challenge

CAST AIP

CoreLogic Governance Dashboard

Cognizant Wireless Infrastructure

Cognizant On

Clinical Systems Connector

CS Risk Metrics Platform

Cognizant University

PASSED: WorldClock.TestMain.bangaloreDate

PASSED: WorldClock.TestMain.bangaloreTime

PASSED: WorldClock.TestMain.Test\_WorldClock\_Title

PASSED: WorldClock.TestMain.viewApps

PASSED: WorldClock.TestMain.londonWatchTitle

PASSED: WorldClock.TestMain.scroll

PASSED: WorldClock.TestMain.userInfo

PASSED: WorldClock.TestMain.bangloreTimeZone

PASSED: WorldClock.TestMain.londonTime

PASSED: WorldClock.TestMain.NewYorkWatchTitle

PASSED: WorldClock.TestMain.NewYorkDate

PASSED: WorldClock.TestMain.londonTimeGap

PASSED: WorldClock.TestMain.NewYorkTimeGap

PASSED: WorldClock.TestMain.NewYorkTime

PASSED: WorldClock.TestMain.bangaloreWatchTitle

PASSED: WorldClock.TestMain.londonDate

PASSED: WorldClock.TestMain.appsAlphabet

PASSED: WorldClock.TestMain.randomAlphabet

===============================================

Default test

Tests run: 18, Failures: 0, Skips: 0

===============================================

===============================================

Default suite

Total tests run: 18, Passes: 18, Failures: 0, Skips: 0

===============================================