

Capstone Project	Student Name: Vijina P
Human Resource Management Domain Project	

Github repository

The source code has been uploaded to the following git repository;

Url	https://github.com/vijinabalakrishnan/Capstone_Protects/tree/main/Human%20Resource%20Management/OrangeHRM_Automation_Test
Branch	main
Access	Public

Scenario 1:

Automate Orange HRM login and logout with 5 different data sets including valid and invalid data sets.

1. Save the data set in an Excel file
2. Write a script to read data from Excel
3. Prepare a script for Login and logout
4. Perform assertion for valid data set (use Username: Admin and Password: admin123) test case should be passed and invalid data set (other than given data) test case should fail.
5. Capture Screenshot for every login functionality
6. Generate Extent Report for the same.

Ans:

Data Input Excel Sheet: LoginData.xlsx

	A	B
1	Username	Password
2	Admin	admin123
3	admin	wrong123
4	user1	pass1
5	test	12345
6	admin123	Admin
7		

Source code is given below.

Scenario 2:

Create a Page Object Model for two pages

1. Login Page
2. Admin Page

Write an automation script for Login functionality and Admin search feature

Create a Page and Class for the Login Page and automate the functionality of Orange HRM login for Valid Test Data:

- Username: Admin
- Password: admin123.

Ans:

Create a Page and class for Admin where you can prepare 4 important test cases:

1. Create a test case to get all 12 options from the left side menu and print the count which should be 12 from that list click on Admin then the Admin page will open.

Test Scenario

1. Login to OrangeHRM
 2. Get all 12 left side menu items
 3. Print the count of menu items(12)
 4. Click on "Admin" menu link.
-
2. **Create test case for search For Existing Employee search ByUserName(): here send username Admin to username text box and click on the search button and display total record found and refresh page.**

Test Scenario

1. Login to OrangeHRM
 2. Navigate to **Admin** section
 3. Search for user with username Admin
 4. Click **Search**
 5. Display total records found
 6. Refresh the page
-
3. **Create test case for search For Existing Employee search ByUserRole(): here automate dropdown and select Role Admin and click on the search button and display the total record found and refresh the page.**

Test Scenario

1. Open OrangeHRM dashboard (after login)
 2. Navigate to Admin page
 3. Use SearchByUserRole("Admin") method to select user role from dropdown
 4. Click Search
 5. Display total records found
 6. Refresh the page
-
4. **Create test case for search For Existing Employee SearchByUserStatus(): here automate dropdown and select status Enabled or Disabled then click on the search button and display the total record found.**

Test Scenario:

1. Login and open the OrangeHRM dashboard
2. Navigate to Admin section
3. Use SearchByUserStatus ("Enabled") or SearchByUserStatus ("Disabled") to select from the dropdown
4. Click Search
5. Display total records found

Source Code

Pom.xml:

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>OrangeHRM_Automation_Test</groupId>
```

```

<artifactId>OrangeHRM_Automation_Test</artifactId>
<version>0.0.1-SNAPSHOT</version>
<dependencies>
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>4.34.0</version>
    </dependency>
    <dependency>
<groupId>com.aventstack</groupId>
<artifactId>extentreports</artifactId>
<version>5.0.9</version>
</dependency>
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi-ooxml</artifactId>
    <version>5.2.5</version>
</dependency>
</dependencies>
<build>
    <sourceDirectory>src</sourceDirectory>
    <plugins>
        <plugin>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.13.0</version>
            <configuration>
                <release>13</release>
            </configuration>
        </plugin>
    </plugins>
</build>
</project>

```

testng.xml:

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd

```

\src\test\java\utils\ExcelUtils.java

```

package utils;

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

```

```

import java.io.FileInputStream;

public class ExcelUtils {
    public static Object[][] getData(String filePath, String sheetName) throws Exception {
        FileInputStream fis = new FileInputStream(filePath);
        Workbook wb = new XSSFWorkbook(fis);
        Sheet sheet = wb.getSheet(sheetName);
        int rowCount = sheet.getPhysicalNumberOfRows();
        int colCount = sheet.getRow(0).getPhysicalNumberOfCells();

        Object[][] data = new Object[rowCount - 1][colCount];

        for (int i = 1; i < rowCount; i++) {
            Row row = sheet.getRow(i);
            for (int j = 0; j < colCount; j++) {
                data[i - 1][j] = row.getCell(j).toString();
            }
        }
        wb.close();
        return data;
    }
}

```

\\src\\test\\java\\utils\\WaitUtils.java

```

package main.java.utils;
import org.openqa.selenium.*;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

import java.time.Duration;

public class WaitUtils {

    private WebDriver driver;
    private WebDriverWait wait;

    // Constructor
    public WaitUtils(WebDriver driver, int timeoutInSeconds) {
        this.driver = driver;
        this.wait = new WebDriverWait(driver, Duration.ofSeconds(timeoutInSeconds));
    }

    // Wait until element is visible
    public WebElement waitForVisibility(By locator) {
        return wait.until(ExpectedConditions.visibilityOfElementLocated(locator));
    }

    // Wait until element is clickable
    public WebElement waitForClickable(By locator) {
        return wait.until(ExpectedConditions.elementToBeClickable(locator));
    }
}

```

```

// Wait until element is present in the DOM
public WebElement waitForPresence(By locator) {
    return wait.until(ExpectedConditions.presenceOfElementLocated(locator));
}

// Wait until element is invisible
public boolean waitForInvisibility(By locator) {
    return wait.until(ExpectedConditions.invisibilityOfElementLocated(locator));
}

// Wait until text is present in element
public boolean waitForText(By locator, String text) {
    return wait.until(ExpectedConditions.textToBePresentInElementLocated(locator, text));
}

// Wait until alert is present
public Alert waitForAlert() {
    return wait.until(ExpectedConditions.alertIsPresent());
}

public void sleep()
{
    try {
        Thread.sleep(1000);
    } catch (InterruptedException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
}

```

\src\test\java\base\BaseTest.java

```

package test.java.tests;

import com.aventstack.extentreports.ExtentReports;
import com.aventstack.extentreports.ExtentTest;
import com.aventstack.extentreports.reporter.ExtentSparkReporter;
import org.openqa.selenium.WebDriver;
import org.testng.annotations.*;

import main.java.utils.WaitUtils;
import main.java.utils.WebDriverFactory;
import com.aventstack.extentreports.*;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;

import java.io.File;
import java.io.FileOutputStream;
import java.text.SimpleDateFormat;
import java.util.Date;

```

```

public class BaseTest {
    protected WebDriver driver;
    protected static ExtentReports extent;
    protected ExtentTest test;
    protected WaitUtils waitUtil;

    @BeforeSuite
    public void setupExtent() {
        ExtentSparkReporter spark = new ExtentSparkReporter("test-output/ExtentReport.html");
        extent = new ExtentReports();
        extent.attachReporter(spark);
    }

    @AfterSuite
    public void tearDownExtent() {
        extent.flush();
    }

    @BeforeMethod
    public void setupDriver() {
        System.setProperty("webdriver.chrome.driver",
            "D:\\Learning\\Java\\Projects\\path\\to\\chromedriver-win32\\chromedriver-
win32\\chromedriver.exe");
        driver = WebDriverFactory.createDriver("chrome");
        driver.get("https://opensource-demo.orangehrmlive.com");

        waitUtil = new WaitUtils(driver,5);
    }

    @AfterMethod
    public void quitDriver() {
        if(driver != null) {
            driver.quit();
        }
    }

    public String captureScreenshot(String testName) {
        String filePath = "screenshots/" + testName + "_" + timestamp() + ".png";
        try {
            TakesScreenshot ts = (TakesScreenshot) driver;
            byte[] src = ts.getScreenshotAs(OutputType.BYTES);
            FileOutputStream fos = new FileOutputStream(new File(filePath));
            fos.write(src);
            fos.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return filePath;
    }

    private String timestamp() {
        return new SimpleDateFormat("yyyyMMddHHmmss").format(new Date());
    }
}

```

```

}
\src\test\java\pages\DashboardPage.java
package main.java.pages;

import java.util.List;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

import main.java.utils.WaitUtils;

public class DashboardPage {
    private WebDriver driver;
    private WaitUtils waitUtil;
    By dashboardHeader = By.xpath("//h6[text()='Dashboard']");
    By navMenu = By.className("oxd-userdropdown-tab");
    By logoutBtn = By.cssSelector("a[href='/web/index.php/auth/logout']");
    By leftNavMenuBtn = By.xpath("/html/body/div/div[1]/div[1]/aside/nav/div[2]/div/div/button");
    By leftNavMenu = By.className("oxd-main-menu");
    By navMenuItem = By.className("oxd-main-menu-item-wrapper");
    By navMenuItemSpan = By.className("oxd-main-menu-item--name");

    public DashboardPage(WebDriver driver, WaitUtils waitUtil) {
        this.driver = driver;
        this.waitUtil = waitUtil;
    }

    public boolean isDashboardDisplayed() {
        return driver.findElements(dashboardHeader).size() > 0;
    }

    public boolean logoutUser() {
        driver.findElement(navMenu).click();
        waitUtil.sleep();
        waitUtil.waitForVisibility(logoutBtn);
        driver.findElement(logoutBtn).click();
        waitUtil.sleep();
        return true;
    }

    public int getLeftNavMenuCount() {
        driver.findElement(leftNavMenuBtn).click();
        waitUtil.sleep();
        waitUtil.waitForVisibility(leftNavMenu);
        waitUtil.sleep();
        return driver.findElements(navMenuItem).size();
    }

    public void openAdminMenu() {
        List<WebElement> elems = driver.findElements(navMenuItem);
        for (WebElement elem : elems) {

```

```
        if (elem.getText().equals("Admin")) {  
            elem.click();  
            waitUtil.sleep();  
            return;  
        }  
    }  
}
```


\src\test\java\pages\LoginPage.java

```
package main.java.pages;

import org.openqa.selenium.By;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

import main.java.utils.WaitUtils;

public class LoginPage {
    private WebDriver driver;
    private WaitUtils waitUtil;

    By username = By.name("username");
    By password = By.name("password");
    By loginBtn = By.xpath("//button[@type='submit']");
    By dashboardHeader = By.xpath("//h6[text()='Dashboard']");
    private By errorMessage = By.className("api-error");

    public LoginPage(WebDriver driver, WaitUtils waitUtil) {
        this.driver = driver;
        this.waitUtil = waitUtil;
    }

    public boolean isDashboardDisplayed() {
        return driver.findElements(dashboardHeader).size() > 0;
    }

    public void login(String user, String pass) {
        driver.findElement(username).sendKeys(user);
        driver.findElement(password).sendKeys(pass);
        waitUtil.waitForClickable(loginBtn);
        driver.findElement(loginBtn).click();
    }
}
```

\src\test\java\pages\DashboardPage.java

```
package main.java.pages;

import java.util.List;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

import main.java.utils.WaitUtils;

public class DashboardPage {
    private WebDriver driver;
    private WaitUtils waitUtil;
    By dashboardHeader = By.xpath("//h6[text()='Dashboard']");
```

```

By navMenu = By.className("oxd-userdropdown-tab");
By logoutBtn = By.cssSelector("a[href='/web/index.php/auth/logout']");
By leftNavMenuBtn = By.xpath("/html/body/div/div[1]/div[1]/aside/nav/div[2]/div/div/button");
By leftNavMenu = By.className("oxd-main-menu");
By navMenuItem = By.className("oxd-main-menu-item-wrapper");
By navMenuItemSpan = By.className("oxd-main-menu-item--name");

```

```

public DashboardPage(WebDriver driver, WaitUtils waitUtil) {
    this.driver = driver;
    this.waitUtil = waitUtil;
}

```

```

public boolean isDashboardDisplayed() {
    return driver.findElements(dashboardHeader).size() > 0;
}

```

```

public boolean logoutUser() {
    driver.findElement(navMenu).click();
    waitUtil.sleep();
    waitUtil.waitForVisibility(logoutBtn);
    driver.findElement(logoutBtn).click();
    waitUtil.sleep();
    return true;
}

```

```

public int getLeftNavMenuCount() {
    driver.findElement(leftNavMenuBtn).click();
    waitUtil.sleep();
    waitUtil.waitForVisibility(leftNavMenu);
    waitUtil.sleep();
    return driver.findElements(navMenuItem).size();
}

```

```

public void openAdminMenu() {
    List<WebElement> elems = driver.findElements(navMenuItem);
    for (WebElement elem : elems) {
        if (elem.getText().equals("Admin")) {
            elem.click();
            waitUtil.sleep();
            return;
        }
    }
}
}

```

\src\test\java\pages\SearchUserPage.java

```
package main.java.pages;
```

```

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import main.java.utils.WaitUtils;

```

```
import java.util.List;
```

```

public class SearchUserPage {
    private WebDriver driver;
    private WaitUtils waitUtil;

    private By usrenameInput = By.className("oxd-input");
    private By userRoleDropdown = By.xpath("//label[text()='User Role']/../following-sibling::div//div[contains(@class,'select-text')]");
    private By dropdownOptions = By.xpath("//div[@role='listbox']//span");
    private By searchButton = By.xpath("//button[normalize-space()='Search']");
    private By resultRows = By.cssSelector(".oxd-table-body .oxd-table-card");
    private By statusDropdown = By.xpath("//label[text()='Status']/../following-sibling::div//div[contains(@class,'select-text')]");
    private By statusOptions = By.xpath("//div[@role='listbox']//span");

    public SearchUserPage(WebDriver driver, WaitUtils waitUtil) {
        this.driver = driver;
        this.waitUtil = waitUtil;
    }

    public int SearchByUsername(String username) {
        List<WebElement> allInputs = driver.findElements(usrenameInput);
        WebElement[] arrayInputs = new WebElement[allInputs.size()];
        WebElement elem = allInputs.toArray(arrayInputs)[1]; //username input textbox

        elem.click();
        elem.sendKeys(username); //enter input
        driver.findElement(searchButton).click(); //search
        waitUtil.sleep();
        return driver.findElements(resultRows).size();
    }

    public int SearchByUserRole(String roleName) {
        driver.findElement(userRoleDropdown).click();
        waitUtil.waitForVisibility(dropdownOptions);
        for (WebElement option : driver.findElements(dropdownOptions)) {
            if (option.getText().equalsIgnoreCase(roleName)) {
                option.click();
                driver.findElement(searchButton).click(); //search
                waitUtil.sleep();
                return driver.findElements(resultRows).size();
            }
        }
    }

    return 0;
}

public int SearchByUserStatus(String statusText) {
    driver.findElement(statusDropdown).click();
    waitUtil.waitForVisibility(statusOptions);

    for (WebElement option : driver.findElements(statusOptions)) {
        if (option.getText().equalsIgnoreCase(statusText)) {

```

```

        option.click();
        driver.findElement(searchButton).click();//search
            waitUtil.sleep();
            return driver.findElements(resultRows).size();
        }
    }

    return 0;
}

public void refreshPage() {
    driver.navigate().refresh();
}

}

```

\src\test\java\tests\LoginTest.java

```

package test.java.tests;

import com.aventstack.extentreports.Status;

import org.testng.Assert;
import org.testng.annotations.DataProvider;
import org.testng.annotations.Test;
import main.java.pages.*;
import main.java.utils.ExcelUtils;

public class LoginTests extends BaseTest {

    @Test(dataProvider = "loginData")
    public void loginTest(String username, String password) {
        test = extent.createTest("Login Test with user: " + username);
        LoginPage loginPage = new LoginPage(driver, waitUtil);
        loginPage.login(username, password);

        DashboardPage dashboardPage = new DashboardPage(driver, waitUtil);
        String screenshotPath = captureScreenshot("Login_" + username);

        if (username.equals("Admin") && password.equals("admin123")) {
            Assert.assertTrue(dashboardPage.isDashboardDisplayed());
            test.pass("Valid login successful").addScreenCaptureFromPath(screenshotPath);

            //logout
            dashboardPage.logoutUser();
            screenshotPath = captureScreenshot("Logout_" + username);
            test.pass("Logout successful").addScreenCaptureFromPath(screenshotPath);
        } else {
            Assert.assertFalse(dashboardPage.isDashboardDisplayed());
            test.fail("Invalid login.").addScreenCaptureFromPath(screenshotPath);
        }
    }
}

```

```

@DataProvider(name = "loginData")
public Object[][] getLoginData() throws Exception {
    return ExcelUtils.getData("data/LoginData.xlsx", "Sheet1");
}
}

```

\src\test\java\tests\DashboardTests.java

```
package test.java.tests;
```

```

import com.aventstack.extentreports.Status;
import org.testng.Assert;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;
import main.java.pages.*;
import main.java.utils.WaitUtils;

```

```
public class DashboardTests extends BaseTest {
```

```

    @BeforeMethod
    public void login() {
        LoginPage loginPage = new LoginPage(driver, waitUtil);
        loginPage.login("Admin", "admin123");
    }

```

```

    @Test
    public void checkLeftMenuItemsCount() {
        test = extent.createTest("Check left menu items count");
        DashboardPage dashBoardPage = new DashboardPage(driver, waitUtil);

        int cnt = dashBoardPage.getLeftNavMenuCount();
        String screenshotPath = captureScreenshot("Dashboard_Menu_Count");

        Assert.assertEquals(cnt, 12, "Left menu item count was not 12. It was "+cnt+" instead.");
        if(cnt == 12)
        {
            test.log(Status.PASS, "Left menu item count test passed. "+cnt+" menu items found").addScreenCaptureFromPath(screenshotPath);
        }
        else
        {
            test.log(Status.FAIL, "Left menu item count test is not passed. Menu count is: "+cnt+" instead of 12").addScreenCaptureFromPath(screenshotPath);
        }
    }
}

```

\src\test\java\tests\SearchUserTests.java

```
package test.java.tests;
```

```

import com.aventstack.extentreports.Status;
import org.testng.Assert;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

```

```

import main.java.pages.*;

public class SearchUserTests extends BaseTest {

    @BeforeMethod
    public void login() {
        LoginPage loginPage = new LoginPage(driver, waitUtil);
        loginPage.login("Admin", "admin123");
    }

    @Test
    public void searchByUserName() {
        String keyword = "Admin";
        test = extent.createTest("Search for admin user using SearchByUserName()");
        DashboardPage dashBoardPage = new DashboardPage(driver, waitUtil);

        dashBoardPage.openAdminMenu();

        SearchUserPage searchUserPage = new SearchUserPage(driver, waitUtil);
        int cntResults = searchUserPage.SearchByUserName(keyword);
        String screenshotPath = captureScreenshot("Search_Username_" + keyword);

        Assert.assertTrue(cntResults > 0, "Failed: searchByUserName() test failed. No of users found for '" + keyword + "' is " + cntResults);
        if(cntResults > 0)
        {
            test.log(Status.PASS, "searchByUserName() test passed. No of users found for '" + keyword + "' is " + cntResults).addScreenCaptureFromPath(screenshotPath);
        }
        else
        {
            test.log(Status.FAIL, "Failed: searchByUserName() test failed. No of users found for '" + keyword + "' is " + cntResults).addScreenCaptureFromPath(screenshotPath);
        }
        searchUserPage.refreshPage();
        test.log(Status.PASS, "Page refreshed");
    }

    @Test
    public void searchByUserRoleDropdown() {
        String keyword = "Admin";
        test = extent.createTest("Search for admin user using SearchByRole() dropdown");
        DashboardPage dashBoardPage = new DashboardPage(driver, waitUtil);

        dashBoardPage.openAdminMenu();

        SearchUserPage searchUserPage = new SearchUserPage(driver, waitUtil);
        int cntResults = searchUserPage.SearchByUserRole(keyword);
        String screenshotPath = captureScreenshot("Search_User_Role");

        Assert.assertTrue(cntResults > 0, "Failed: SearchByRole() dropdown test failed. No of users found for '" + keyword + "' is " + cntResults);
        if(cntResults > 0)
    
```

```

        {
            test.log(Status.PASS, "SearchByRole() dropdown test passed. No of users found for '"+keyword+"' is
"+cntResults).addScreenCaptureFromPath(screenshotPath);
        }
        else
        {
            test.log(Status.FAIL, "Failed: SearchByRole() dropdown test failed. No of users found for
 '"+keyword+"' is "+cntResults).addScreenCaptureFromPath(screenshotPath);
        }
        searchUserPage.refreshPage();
        test.log(Status.PASS, "Page refreshed");
    }
    @Test
    public void searchByUserStatusDropdown() {
        String keyword = "Enabled";//Enabled status
        test = extent.createTest("Search for admin user using SearchByUserStatus() dropdown");
        DashboardPage dashBoardPage = new DashboardPage(driver, waitUtil);

        dashBoardPage.openAdminMenu();

        SearchUserPage searchUserPage = new SearchUserPage(driver, waitUtil);
        int cntResults = searchUserPage.SearchByUserStatus(keyword);
        String screenshotPath = captureScreenshot("Search_User_Status");

        Assert.assertTrue(cntResults>0,"Failed: SearchByUserStatus() dropdown test failed. No of users found for
 '"+keyword+"' is "+cntResults);
        if(cntResults > 0)
        {
            test.log(Status.PASS, "SearchByUserStatus() dropdown test passed. No of users found for
 '"+keyword+"' is "+cntResults).addScreenCaptureFromPath(screenshotPath);
        }
        else
        {
            test.log(Status.FAIL, "Failed: SearchByUserStatus() dropdown test failed. No of users found for
 '"+keyword+"' is "+cntResults).addScreenCaptureFromPath(screenshotPath);
        }
        searchUserPage.refreshPage();
        test.log(Status.PASS, "Page refreshed");
    }

    @Test
    public void searchByUserStatusDropdownDisabled() {
        String keyword = "Disabled";//Disabled status
        test = extent.createTest("Search for admin user using SearchByUserStatus() dropdown");
        DashboardPage dashBoardPage = new DashboardPage(driver, waitUtil);



























        dashBoardPage.openAdminMenu();

        SearchUserPage searchUserPage = new SearchUserPage(driver, waitUtil);
        int cntResults = searchUserPage.SearchByUserStatus(keyword);
        String screenshotPath = captureScreenshot("Search_User_Status");

```

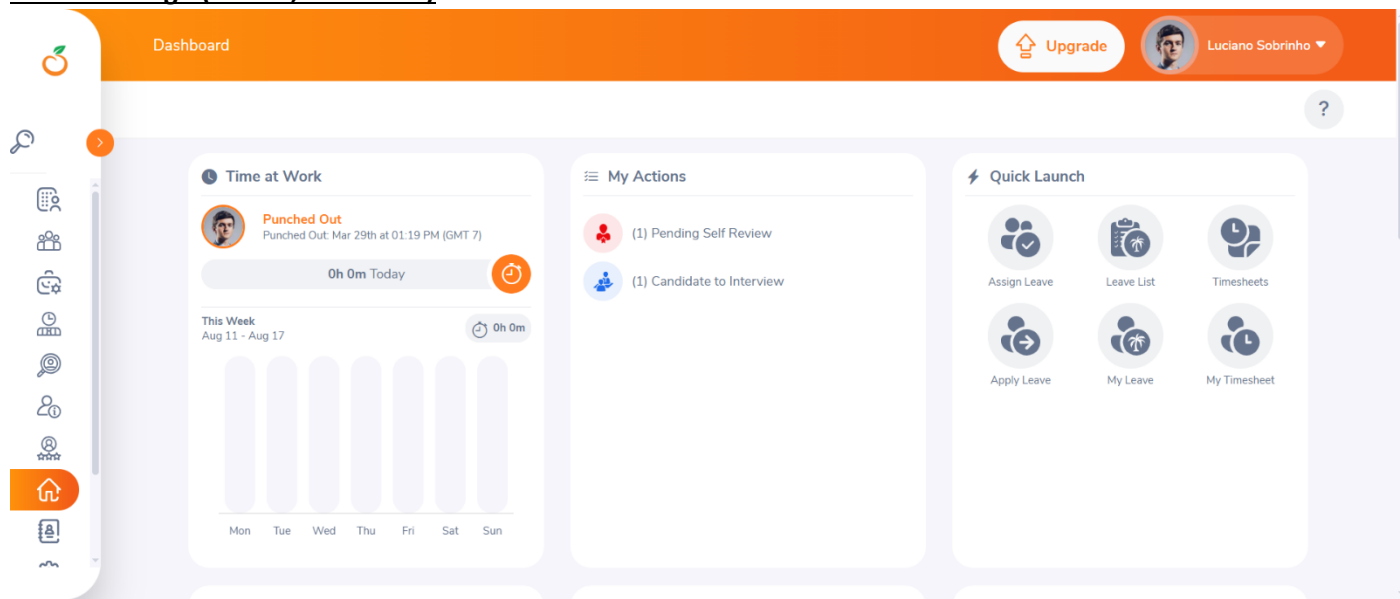
```
Assert.assertTrue(cntResults>0,"Failed: SearchByUserStatus() dropdown test failed. No of users found for
"+keyword+" is "+cntResults);
    if(cntResults > 0)
    {
        test.log(Status.PASS, "SearchByUserStatus() dropdown test passed. No of users found for
"+keyword+" is "+cntResults).addScreenCaptureFromPath(screenshotPath);
    }
    else
    {
        test.log(Status.FAIL, "Failed: SearchByUserStatus() dropdown test failed. No of users found for
"+keyword+" is "+cntResults).addScreenCaptureFromPath(screenshotPath);
    }
    searchUserPage.refreshPage();
    test.log(Status.PASS, "Page refreshed");
}
}
```


Project Structure

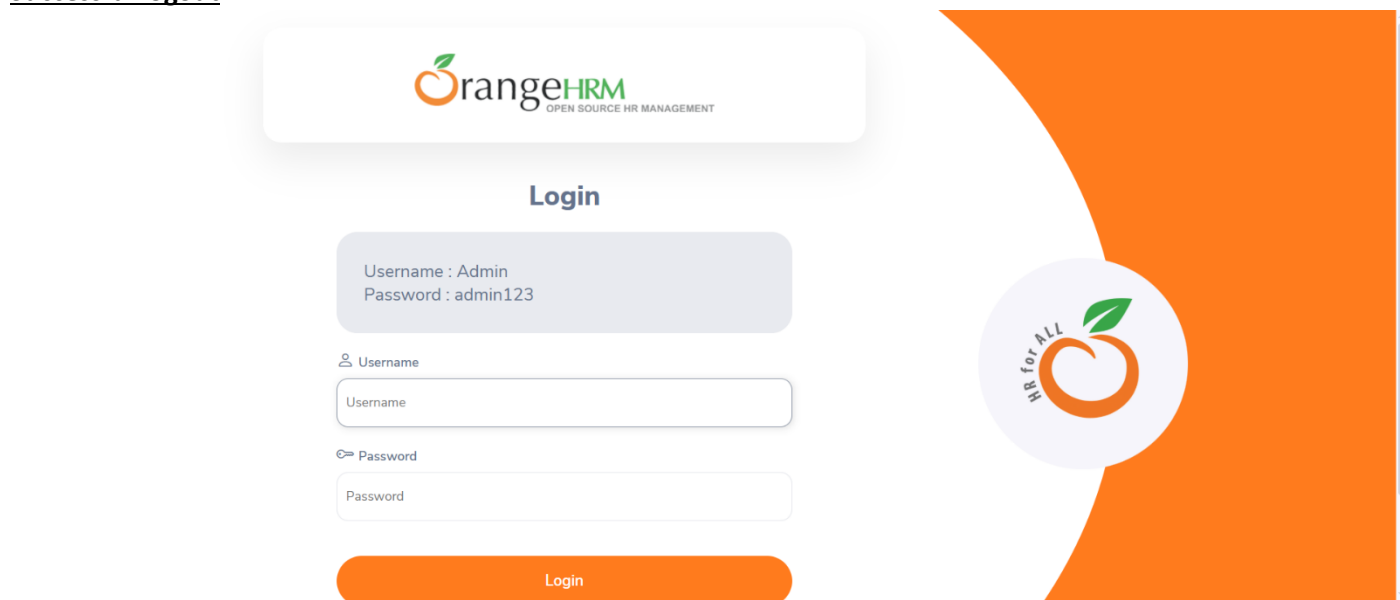
- ▼  OrangeHRM_Automation_Test [Capstone_Projects m
- ▼  src
 - ▼  main.java.pages
 - >  DashboardPage.java
 - >  LoginPage.java
 - >  SearchUserPage.java
 - ▼  main.java.utils
 - >  ExcelUtils.java
 - >  WaitUtils.java
 - >  WebDriverFactory.java
 - >  test.java.listeners
 - ▼  test.java.tests
 - >  BaseTest.java
 - >  DashboardTests.java
 - >  LoginTests.java
 - >  SearchUserTests.java
- >  JRE System Library [JavaSE-13]
- >  Maven Dependencies
- >  TestNG
- >  bin
- >  data
- >  screenshots
- >  target
- >  test-output
- >  pom.xml
- >  testng.xml

Screenshots


Successful login(Admin, admin123)



Successful logout



Menu item count = 12



Search

Admin

PIM

Leave

Time

Recruitment

My Info

Performance

Dashboard

Directory

Admin / User Management

Upgrade

Luciano Sobrinho

User Management

Job

Organization

Qualifications

Nationalities

Corporate Branding

Configuration

?

System Users

Username

User Role

Employee Name

Status

Admin

Type for hints...

-- Select --

Reset

Search

+ Add

(5) Records Found

Opening Admin menu

The screenshot shows the OrangeHRM Admin / User Management interface. The left sidebar contains a search bar and a menu with options: Admin, PIM, Leave, Time, Recruitment, My Info, Performance, Dashboard, and Directory. The main content area is titled 'Admin / User Management' and features a top navigation bar with an 'Upgrade' button and a user profile for 'Luciano Sobrinho'. Below the navigation bar is a tabbed interface with 'User Management' selected. The 'System Users' section includes search filters for Username, User Role, Employee Name, and Status. The 'Status' filter is set to 'Enabled'. A green '+ Add' button is visible, and the text '(11) Records Found' is displayed at the bottom.

This screenshot is identical to the previous one, showing the OrangeHRM Admin / User Management interface. The 'System Users' section has the same search filters and '+ Add' button. However, the text at the bottom now reads '(1) Record Found', indicating a change in the number of records displayed.

Search by username : Admin

This screenshot shows the OrangeHRM Admin / User Management interface with the 'System Users' section. The 'Username' filter is set to 'Admin'. The 'Status' filter is set to '-- Select --'. A green '+ Add' button is visible, and the text '(1) Record Found' is displayed at the bottom.

Search by user status: Disabled

The screenshot shows the OrangeHRM Admin / User Management interface. The left sidebar contains a search bar and a list of menu items: Admin, PIM, Leave, Time, Recruitment, My Info, Performance, Dashboard, and Directory. The main content area is titled "Admin / User Management" and features a top navigation bar with an "Upgrade" button and a user profile for "Luciano Sobrinho". Below the navigation bar, there are tabs for "User Management", "Job", "Organization", "Qualifications", "Nationalities", "Corporate Branding", and "Configuration". The "User Management" tab is active, displaying a "System Users" section. This section includes a search form with fields for "Username", "User Role" (a dropdown menu showing "-- Select --"), "Employee Name" (a text input with a hint "Type for hints..."), and "Status" (a dropdown menu showing "Disabled"). There are "Reset" and "Search" buttons below the search form. Below the search form, there is a "+ Add" button. At the bottom, it indicates "(1) Record Found".

Search by user status: Enabled

The screenshot shows the OrangeHRM Admin / User Management interface, similar to the previous one, but with the "Status" dropdown menu set to "Enabled". The search form fields are the same: "Username", "User Role" (dropdown showing "-- Select --"), "Employee Name" (text input with hint "Type for hints..."), and "Status" (dropdown showing "Enabled"). The "Reset" and "Search" buttons are present. Below the search form, there is a "+ Add" button. At the bottom, it indicates "(11) Records Found".

Search by user role: Admin

System Users

Username

User Role

Admin ▾

Employee Name

Type for hints...

Status

-- Select -- ▾

Reset

Search

+ Add

(5) Records Found