Gitlab Assesment

To Automate the Gitlab(https://admin-demo.nopcommerce.com/login) using Selenium Webdriver.

Step 1: Set up Environments

- I have created Java Project with name called **Gitlab.**
- In src folder i have created com.Gitlab.PageObject, com.Gitlab.Testcases,and com.Gitlab.Utilities.
- In main project directory i have created Driver, Configuration and Screenshots folders.
- I have added selenium common jars, log4j jars into my buildpath.
- I have Configured TestNG Library into my project.
- Convert it to TestNG

Step2 : Create pageobject class for each page

- My first Testcase is to automate the login with functionality, so i have used page object model for each screen, first of all i identify the elements and action items in the page, and i will create loginpage.java file in com.gitlab.Pageobject Package, at below i have mentioned pageobject code.

Com.login.pageobject

```
package com.JavaFramework.PageObjects;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.CacheLookup;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class LoginPage {


    public WebDriver ldriver;

    public LoginPage(WebDriver rdriver)
    {
        ldriver=rdriver;
        PageFactory.initElements(rdriver, this);
    }

    @FindBy(id="Email")
    @CacheLookup
    WebElement txtEmail;

    @FindBy(id="Password")
    @CacheLookup
    WebElement txtPassword;

    @FindBy(xpath="//input[@value='Log in']")
    @CacheLookup
    WebElement btnLogin;

    @FindBy(linkText="Logout")
    WebElement lnkLogout;
    public void setUserName(String uname)
    {
        txtEmail.clear();
        txtEmail.sendKeys(uname);
    }
    public void setPassword(String pwd)
    {
        txtPassword.clear();
        txtPassword.sendKeys(pwd);
    }
    public void clickLogin()
    {
        btnLogin.click();
    }
    public void clickLogout()
    {
        lnkLogout.click();
    }
}
```

Step 3: CreateProjectPageObject

In the same project directory in src folder i will create on more page object class for createproject, so now i will identify the elements and action item for creating project

```
package com.Gitlab.PageObject;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.CacheLookup;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class CreateProjectPage {
public WebDriver ldriver;

    public WebDriver ldriver1;
        public CreateProjectPage(WebDriver rdriver)
        {
            ldriver1=rdriver;
            PageFactory.initElements(rdriver, this);
        }

        @FindBy(xpath="//a[@class='btn btn-success']")
        @CacheLookup
        WebElement NewProjectButton

        @FindBy(id="project_name")
        @CacheLookup
        WebElement projectName;

        @FindBy(id="project_description")
        @CacheLookup
        WebElement Description;

        @FindBy(id="project_visibility_level_20")
        @CacheLookup
        WebElement publicvisibility;
        @FindBy(xpath="//div[@id='blank-project-pane']//input[@value='Create project']")
        @CacheLookup
        WebElement CreateProject;

        @FindBy(xpath="//img[@alt='akash tippa']")
        @CacheLookup
        WebElement profiledrop;

        @FindBy(xpath="//a[@class='sign-out-link']")
        @CacheLookup
        WebElement Logout;
        public void EnterProjectName(String proname)
        {
            projectName.sendKeys(proname);;
        }
        public void EnterDescription(String Desc)
        {
            Description.sendKeys(Desc);
        }
        public void SetVisibility()
        {
            publicvisibility.click();
        }
        public void ClickonCreateProjectbutton()
        {
            CreateProject.click();
        }
        public void profiledropdown()
        {
            profiledrop.click();
        }
        public void Signout()
        {
            Logout.click();
        }
}
```

Step 4: Create BaseClass for testcases

In **Selenium, Base class** is the main **class** which takes care of Browser setup, loading configuration file and other reusable methods like screenshot, handling sync issues and many more. With **base class** you can avoid code duplication and can reuse the code as much you want.

Base class code.

```
package com.Gitlab.TestCases;

import java.io.File;
import java.io.IOException;

import org.apache.commons.io.FileUtils;
import org.apache.log4j.Logger;
import org.apache.log4j.PropertyConfigurator;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeClass;

import com.Gitlab.Utilities.ReadConfig;
public class BaseClass {

        ReadConfig config = new ReadConfig();

        public static WebDriver driver;
        public String baseURL= config.getApplicationURL();
        public String email= config.getEmail();
        public String invalidemail=config.invalidemail();
        public String invalidpassword=config.invalidepassword();
        public String username= config.getUsername();
        public String password= config.getPassword();
        public static Logger logger;

        @BeforeClass
        void SetUp() throws InterruptedException
        {
                logger = Logger.getLogger("Gitlab");
                PropertyConfigurator.configure("log4j.properties");
                        System.setProperty("webdriver.chrome.driver", config.getChromePath());
                        driver=new ChromeDriver();
                        driver.manage().window().maximize();
                        driver.get(baseURL);
```

```
    Thread.sleep(7000);
    }

    @AfterTest
    public void tearDown() {
        if(driver!=null) {
            driver.quit();
        }
    }
    public void captureScreen(WebDriver driver, String tname) throws IOException {
        TakesScreenshot ts = (TakesScreenshot) driver;
        File source = ts.getScreenshotAs(OutputType.FILE);
        File target = new File(System.getProperty("user.dir") + "/Screenshots/" + tname +
".png");
        FileUtils.copyFile(source, target);
        System.out.println("Screenshot taken");
    }
}
```

Step 5: Create Configuration file for Secreat data

properties **files** are mainly used in Java programs to maintain project **configuration** data, database **config** or project settings, etc. Each parameter in properties **file** is stored as a pair of strings, in key-value pair format, where each key is on one line.

Config.properties file

```
baseURL=https://gitlab.com/users/sign_in#register-pane
email=akashtippa@gmail.com
password=Akash@123
username=myname99
invalidemail=bvnop@yopmail.com
invalidpassword=9398448
projectname=TestProject
apikey=a0151b6c77092f8da8c9ec75ec2472e5
sitekey=6LfAERQTAAAAAL4GYSiAMGLbcLyUIBSfPrDNJgeC
chromepath=./Driver\\chromedriver.exe
```

Readconfig.java file

```
package com.Gitlab.Utilities;

import java.io.File;
import java.io.FileInputStream;
import java.util.Properties;

public class ReadConfig {

        Properties pro;

        public ReadConfig()
        {
                File src= new File("./Configurations/config.properties");

                try {
                        FileInputStream fis= new FileInputStream(src);
                        pro = new Properties();
                        pro.load(fis);
                }catch(Exception e){
                        System.out.println("Exceptioin is " +e.getMessage());
                }
        }
        public String getApplicationURL()
        {
                String URL=pro.getProperty("baseURL");
                return URL;
        }
```

```
    public String getEmail()
    {
        String email=pro.getProperty("email");
        return email;

    }
    public String invalidemail()
    {
        String invalidemail=pro.getProperty("invalidemail");
        return invalidemail;
    }
    public String invalidepassword()
    {
        String invalidpassword=pro.getProperty("invalidpassword");
        return invalidpassword;
    }
    public String getUsername()
    {
        String username= pro.getProperty("username");
                return username;
    }
    public String getPassword()
    {
        String Password= pro.getProperty("password");
        return Password;
    }
    public String getChromePath()
    {
        String chromePath = pro.getProperty("chromepath");
        return chromePath;
    }
}
```

Step 6 : Create Testcase for Login

the next process is create Testcase script for login, i have created login test in the com.gitlab.login, then i will create class TC_loginwithemailandusername.java here i will write loginpage class and object for inheriting the loginpage methods.

```
package com.Gitlab.TestCases;

import org.testng.annotations.Test;
import com.Gitlab.PageObject.LoginPage;

public class TC_LoginWithEmailandUsername_002 extends BaseClass{
    @Test(priority=1)
    public void loginwithemailid() throws InterruptedException
    {

        LoginPage login = new LoginPage(driver);
        login.SigninLink();
        Thread.sleep(5000);
        login.EnterUserName(email);
        logger.info("email id is entered");
        login.EnterPassword(password);
        logger.info("Password is Entered");
        login.ClickonSigninbutton();
        logger.info("Clicked on Sign in button");

        Thread.sleep(4000);
        login.profiledropdown();
        login.Signout();
        logger.info("Account signed out");
    }
    @Test(priority=2)
    public void LoginwithUsername() throws InterruptedException{
        LoginPage login = new LoginPage(driver);

        Thread.sleep(5000);

        login.EnterUserName(username);
        logger.info("username is entered");
        login.EnterPassword(password);
        logger.info("Password is Entered");
        login.ClickonSigninbutton();
        logger.info("Clicked on Sign in button");

        Thread.sleep(4000);
        login.profiledropdown();
        login.Signout();
        logger.info("Account signed out");

        Thread.sleep(3000);
    }


}
```

Step 7: Create Testcase for Create Project

The next process is create test case for creating project i have created CreatProject test case in the com.gitlab.Testcase, then i will create class TC_.java here i will write CreateProject class and object for inheriting the createproject methods.

```
package com.Gitlab.TestCases;

import java.io.IOException;

import org.junit.Assert;
import org.openqa.selenium.By;
import org.testng.annotations.Test;

import com.Gitlab.PageObject.CreateProjectPage;
import com.Gitlab.PageObject.LoginPage;

public class TC_CreateProject_003 extends BaseClass {

    private static final boolean ele = false;
    @Test(priority=1)
    public void login() throws InterruptedException
    {

        LoginPage login = new LoginPage(driver);
        login.SigninLink();
        Thread.sleep(5000);
        login.EnterUserName(username);
        login.EnterPassword(password);
        login.ClickonSigninbutton();

        Thread.sleep(5000);
    }
    @Test(priority=2)
    public void CreateNewProject() throws InterruptedException, IOException
    {
        CreateProjectPage create = new CreateProjectPage(driver);

        create.ClickonNewProjectbutton();
        Thread.sleep(5000);

        //create.CreateBlankProject();

        create.EnterProjectName("Genaration Project4");

        logger.info("Project name is Entered");
        create.EnterDescription("This new Project");
        logger.info("Project Description is Entered");
        create.SetVisibility();
        logger.info("Visibility mode public");
        create.ClickonCreateProjectbutton();
        logger.info("Clicked on Create Project button");
        logger.info("Project is Created");


        String Project = driver.findElement(By.xpath("//h1[@class='home-panel-title gl-mt-3 gl-mb-2 gl-font-size-h1 gl-line-height-24 gl-font-weight-bold']")).getText();
        if(Project.contains("Genaration Project4"))
        {
            Assert.assertTrue(true);
            logger.info("Project name is Validated");
            logger.info("test case Passed.....");
        }else
        {
            captureScreen(driver,"CreateNewProject");
            Assert.assertTrue(false);
        }
        Thread.sleep(4000);
        create.profiledropdown();
        create.Signout();
        logger.info("Account signed out");

    }


}
```

Registration 8:
Note:
For Registration Test case, we have capatcha, we need to use external API’s to solve capatcha, to solve this Recapatcha we need to use twowaycapacha service that is paid version, so we cannot execute i skipped the capacha test case Execution

Step 9:
Configuration
IDE: Eclipse
Operating Sytem: Windows 10
Browser: Chrome v86