Data driven testing:  
  
**Generic functionalities:**

Developing libraries to and set the data:

Create class in the name of getData below **utils.genric** package  
  
public class getData

{

Public static string fromexcel(String fname,String sname,int rIndex,int cindex)

{

String data=null:

Try{

File f =new file(“file path”+ fname+ “.xlcx”);

fileInputStream fis=new FileInputstream(f);

wb=workbookfatory.create(fis);

sheet s=wb.getSheet(sname);

Row r=s.getRow(rindex);

Cell c=r.getCell(cindes);

Data=c.toString();

}

Catch(Exception e)

{

}

}

}

Create a class in the name of setData below utils.generic package

public class getData

{

Public static string fromexcel(String fname,String sname,int rIndex,int cindex,String cvalue)

{

String data=null:

Try{

File f =new file(“file path”+ fname+ “.xlcx”);

fileInputStream fis=new FileInputstream(f);

wb=workbookfatory.create(fis);

sheet s=wb.getSheet(sname);

Row r=s.getRow(rindex);

Cell c=r.getCell(cindes);

c.setCellValue(cvalue);

fileoutputstream fos=new FileoutputStream(f);

wb.write(fos)

}

Catch(Exception e)

{

}

}

}

**Developing abstractions for selenium functionality:**

**Alert generic Functionalities:**

Create a class in the name of “handleAlert” and develop functionalities to handle it(develop below utils.genric package)

Public class handleAlert

{

Public static void AcceptAlert(webdriver driver)

{

Alert alt=driver.switchTo.alert();

Alt.accept();  
}

Public static void dismissAlert(webdriver driver)

{

Alert alt=driver.switchTo.alert();

Alt.dismiss();  
}

Public static void GetTextAlert(webdriver driver)

{

Alert alt=driver.switchTo.alert();

String s=Alt.getText();

Return s;  
}

Public static void SendTextAlert(webdriver driver, String text)

{

Alert alt=driver.switchTo.alert();

Alt.sendKeys(text);  
}

}

**Developing DropDown generic functionalities:**

public class handleDDL

{

Public static void SelectDDlByIndex(WebElement ddl,int index)

{

Select sel=new Select(ddl);

Sel.selectByIndex(index);

}

Public static void SelectDDlByValue(WebElement ddl,String Value)

{

Select sel=new Select(ddl);

Sel.selectByValue(Value);

}

Public static void SelectDDlByVisibility(WebElement ddl,String VisibleText)

{

Select sel=new Select(ddl);

Sel.selectByVisibleText(VisibleText);

}

Public static Boolean VerifyDDL(WebElement ddl)

{

Select sel=new Select();

Boolean b=Sel.isMultiple();

Return b;

}

Public static void SelectDDlByIndex(WebElement ddl,int index)

{

Select sel=new Select(ddl);

Try{

Sel.selectByIndex(index);

}

Ctach(Execption e)

{

System.out.println(“Invalid Operation);

}

}

Public static void DeSelectDDlByIndex(WebElement ddl)

{

Select sel=new Select(ddl);

Try{

Sel.deSelectByAll();

}

Ctach(Execption e)

{

System.out.println(“Invalid Operation);

}

}

Public static void DeSelectDDlByValue(WebElement ddl, String value)

{

Select sel=new Select(ddl);

Try{

Sel.deSelectByValue(value);

}

Ctach(Execption e)

{

System.out.println(“Invalid Operation);

}

}

Public static void DeSelectDDlByVisibleText(WebElement ddl, String VisbleText)

{

Select sel=new Select(ddl);

Try{

Sel.deSelectByViibleTextl(VisibleText);

}

Ctach(Execption e)

{

System.out.println(“Invalid Operation);

}

}

}

**Developing Project Specific libraries:**

Steps:  
1 Got to test-folder name project  
2.open tdata.xls and store below information In a configuration sheet.

|  |  |  |
| --- | --- | --- |
| Browser NAme | URL |  |
| GC | http://....... |  |
|  |  |  |

Create a class in the name “createdriver “ of below project specific package

Public class createdriver

{

Public static wbdriver{

webDriver driver;

String browserName=getData.fromExcel(“tdata”,”configuration”,1,0);

String url=getData.fromExcel(“tdata”,”configuration”,1,1);

//this will take the data from the exel sheet with name of “tdata”

If(breoserName.equalsIngmoreCase(“ff”)

{

System.setProperty();

Driver=new FirefoxDriver();  
}

Else If(broserName.equalsIngmoreCase(“GC”)

{

System.setProperty();

Driver=new ChromeDriver();  
}

Else

{  
System.out.printLn(“Invalid BrowserName”):

}

Driver.manage().window().max();

driver.get(url)

Return driver;  
  
}

**Developing UI library or Page Object Model(POM):**

1.Define a class in the name of login Page below UI package  
  
public class loginpage

{

WebDriver driver;

Publiv loginpage(Webdriver driver)

{

This.driver=driver;

}

Public void waitForLoginPage()

{

WebdriverWait wait=new WebdriverWait(driver,30);

Wait.until(expectedConditions.visibilityOfElemntLocated(“Xpath OF the Element”);  
}  
public webelement getUserNameTextBox();

{

WebElement element=driver.findelement(By.name(“username”);

Return element;  
}  
public webelement getPasswordTextBox();

{

WebElement element=driver.findelement(By.name(“pwd”);

Return element;  
}

public webelement getLoginButton();

{

WebElement element=driver.findelement(By.xpath(“//input/Login”);

Return element;  
}

public webelement getDivisionHeaderText();

{

WebElement element=driver.findelement(By.cssSelector(“.loginlogo”);

Return element;  
}

public webelement getErrorMessageText();

{

WebElement element=driver.findelement(By.xpath(“xpath of the error message”);

Return element;  
}}  
  
**Developing TestScript or conversion of Manual Testcase to Automation Test Screipt:  
Mapping of test Case in automation:**

TS01 TC01 TC02

TestSuits TS02 TC03  
 TC04

MAnual TS03 TC05

Testclass01 @Test🡪TC01 @Test🡪TC02

TestSuits Testclass02 @Test🡪TC03  
.xml @Test🡪TC04

Automation Testclass03 @Test🡪TC05

**Scenario 1:**

SC01Check Login   
TC05 check invalid Login  
  
Steps to login  
1. Go to page  
2. Enter valid uname  
3. Enter invalid pwd  
4. Click on login button

Rules To define testcases:  
1. Create testNG class below test package(Regression or functional or, smoke)  
a. class name should match with the Scenario description  
b. class name ends with Scenario ID  
c. class name starts with @test  
Example: TestLoginSC\_01

2. Create webdriver reference in class globaly.

3. create page object class references globally.

4.Create before method and initialize webDriver , all page object classes in which the test is using

5. Implement testby using @Test annotation  
a. Rule for a method name  
i. Method name should starts with test  
ii. Method Name should match with test description  
iii. Method name should end with testcase ID  
Ex: testInvalidLoginTC\_01  
b. Implement the test according to test steps  
c. compare Actual and Expected using Assert.

6. Define After Method and close the browser  
  
the code for the above scenario is as below  
  
public class TestLoginSC\_01  
{  
 WebDriver driver;

LoginPage login;

@beforeMethod  
 Public void setup()

{

driver=createDriver.getDriver();

login=new loginPage(driver);

@test

Public void testInvalidLoginTC\_05()

{

Login.waitForLoginPage();

Login.getUserNAmeTextBox().send Keys(“Aadmin”);

Login.getPAsswordTextBox().sendKeys(“mbfkfdbd”);

Login.getLogoinbutton().click();

String actualErrormessage=login.getErrorMessage().getText();  
 String ExpecedErrorMessage=”Usre name jdnjfdc”;  
 Assert.assertEquals(actualErroMessage,ExpectedErrorMessage);

}

@AfterMethods  
public void teardown()

{

Driver.close();  
}

}

**Version Control Tools:**  
  
These are the tool which is used to handle code base

Version control tool will have two types  
1. Internet  
2. Intranet

**How To download and install GIT:**1.Go to Below URL  
<https://git-scm.com/>

2. Click on download for windows

Automatically file will get downloaded in exe

3. After successful download install GIT

**Steps to Create repository:**

Every project will have depositries and the code base is maintained in a centralized repository.

1. Go to below URL  
   https://github.com
2. Sign up
3. Sign in
4. Click on new repository
5. Provide project name as a repository name(must be same as the project name)
6. Provide description
7. Choose public repository
8. Select initialize repository and click on create repository(this is done only once in life cycle of the project)

Once repository is created push the framework code into repository.

1. go to any folder rt. click and choose GIT bash

2. use below command the clone of created repository   
 git clone<repository path>  
 **git clone** <https://github.com/Retheshks/cosmic-cubes.git> please ask to your senior to get this link

3. Add newly developed scripts to depository(into the folder created in the drive)

**Steps to add new files:**

1.cd cosmic-cubes(optional)  
 git add –A  
 2.🡪 -A(all Files)  
 3. Before commiting the scrpitr checkout the brach  
 git checkout -b<branch name>  
 git checkout -b ”dev/frameworks”

**Steps to commit the script:**

1. Git config –global user.email [retheshks78@gmail.com](mailto:retheshks78@gmail.com)
2. Git config –global user.name “Retheshks”
3. Git commit -m<message>  
   git commit -m “FrameWork”
4. Push the commited thing to github
5. Git push origin <Branch>  
   git push origin dev/framework  
     
   life cycle of git hub

Clone

commit

Create a branch and checkout

Added new files

Create PR

push

merge

**review**

Steps to create a Pull Request(PR)

1. Go to GITUHB.com and signing
2. Go to repository
3. Click on compare and create pull request
4. Provide description of changes and add previewers
5. Click on pull request button   
   note: every new pull request will have a unique number

Steps to merge pull request

1. Go to repository ad click on pull request tab
2. Choose a pull request and click on it
3. Click on merge branch
4. Click on confirm merge

Automatically the branch is merged to master

Steps to

1. Take a clone of repository
2. Identify the missing page objects for a test
3. Develop missing page objects
4. Import the project to eclips
5. Add testscript   
   git add  
     
   url to get the project from the git hub is as below  
   <https://github.com/Retheshks/cosmic-cubes>

Assignment :  
1. go to acttime perform login  
2. click on users and add a new users  
online url for the acttime  
demo.acttime.com