1. **For Logging using Log4J:**

```
<dependency>
   <groupId>log4j</groupId>
   <artifactId>log4j</artifactId>
   <version>1.2.17</version>
</dependency>
```

*** Code for Log4J.properties:**

CreateLog4j.Properties file in UtilityPAckage

// Here we have defined root logger - Remove these commented lines after copying in the file Log4j.Properties file

```
log4j.rootLogger=INFO,CONSOLE,R,HTML,TTCC

// Here we define the appender
log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender
log4j.appender.R=org.apache.log4j.RollingFileAppender
log4j.appender.TTCC=org.apache.log4j.RollingFileAppender
log4j.appender.HTML=org.apache.log4j.FileAppender

// Here we define log file location
log4j.appender.R.File=./log/testlog.log
log4j.appender.TTCC.File=./log/testlog1.log
log4j.appender.HTML.File=./log/application.html

// Here we define the layout and pattern
log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
log4j.appender.CONSOLE.layout.ConversionPattern= %5p [%t] (%F:%L)- %m%n
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R.layout.ConversionPattern=%d - %c -%p - %m%n
log4j.appender.TTCC.layout=org.apache.log4j.TTCCLayout
log4j.appender.TTCC.layout.DateFormat=ISO8601
log4j.appender.HTML.layout=org.apache.log4j.HTMLLayout
log4j.appender.HTML.layout.Title=Application log
log4j.appender.HTML.layout.LocationInfo=true
```

Code for Logging:

(To add in our Base Class)

```
import java.util.logging.Logger;

public static Logger loger;     //Code Used for Log4j - to log
```

At Initialization() method of BaseClass.java:

```
//Code for Log4j use
loger.info("Opened the URL");
loger = Logger.getLogger("Automation Framework Log");    //For Logging with
Log4j
PropertyConfigurator.configure("C:\\Users\\Raji\\IdeaProjects\\SelAutomationPro
j2\\src\\main\\resources\\Log4j.Properties");
//Until above code for Log4j use
```

2. To Access an EXCEL File for different Login credentials:

ApachePOI:

```
<dependency>
   <groupId>org.apache.poi</groupId>
   <artifactId>poi</artifactId>
   <version>5.2.2</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.apache.poi/poi-ooxml -->
<dependency>
   <groupId>org.apache.poi</groupId>
   <artifactId>poi-ooxml</artifactId>
   <version>5.2.2</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.commons/commons-lang3 -->
<dependency>
   <groupId>org.apache.commons</groupId>
   <artifactId>commons-lang3</artifactId>
   <version>3.0</version>
</dependency>
<!-- https://mvnrepository.com/artifact/commons-io/commons-io -->
<dependency>
   <groupId>commons-io</groupId>
   <artifactId>commons-io</artifactId>
   <version>2.11.0</version>
</dependency>
```

***// code for Read EXCEL FILE:**

Create ReadXL.java file in UtilityPAckage

```
public class ReadXL {
        public static FileInputStream fi;
        public static FileOutputStream fo;
        public static XSSFWorkbook wb;
        public static XSSFSheet ws;
        public static XSSFRow row;
```

```
	public static XSSFCell cell;



	public static int getRowCount(String xlfile,String xlsheet) throws IOException
	{
		fi=new FileInputStream(xlfile);
		wb=new XSSFWorkbook(fi);
		ws=wb.getSheet(xlsheet);
		int rowcount=ws.getLastRowNum();
		wb.close();
		fi.close();
		return rowcount;
	}


	public static int getCellCount(String xlfile,String xlsheet,int rownum) throws IOException
	{
		fi=new FileInputStream(xlfile);
		wb=new XSSFWorkbook(fi);
		ws=wb.getSheet(xlsheet);
		row=ws.getRow(rownum);
		int cellcount=row.getLastCellNum();
		wb.close();
		fi.close();
		return cellcount;
	}


	public static String getCellData(String xlfile,String xlsheet,int rownum,int colnum) throws
IOException
	{
		fi=new FileInputStream(xlfile);
		wb=new XSSFWorkbook(fi);
		ws=wb.getSheet(xlsheet);
		row=ws.getRow(rownum);
		cell=row.getCell(colnum);
		String data;
		try
		{
			DataFormatter formatter = new DataFormatter();
	String cellData = formatter.formatCellValue(cell);
	return cellData;
		}
```

```
		catch (Exception e)
		{
			data="";
		}
		wb.close();
		fi.close();
		return data;
	}

	public static void setCellData(String xlfile,String xlsheet,int rownum,int colnum,String
data) throws IOException
	{
		fi=new FileInputStream(xlfile);
		wb=new XSSFWorkbook(fi);
		ws=wb.getSheet(xlsheet);
		row=ws.getRow(rownum);
		cell=row.createCell(colnum);
		cell.setCellValue(data);
		fo=new FileOutputStream(xlfile);
		wb.write(fo);
		wb.close();
		fi.close();
		fo.close();
	}


}
```

***// Code for DataDriven:**

Create LoginPageDDT.java file in PageTestPackage

```
public class LoginPageDDT extends BaseClass  {
	HomePage hp;
	LoginPage lp;
	LandingPage landpage;

	LoginPageDDT(){
		super();
	}
	@BeforeMethod
	public void setup() {
		initialization();
		hp=new HomePage(driver);
		lp=new LoginPage(driver);
```

```
		lp=hp.clicklb();

	}
	@AfterMethod
	public void teardown() {
		driver.close();
		logger.info("browser is closed");
	}
	@Test(dataProvider="logindata")
	public void loginddt(String user,String pwd) throws InterruptedException {
		lp.setusername(user);
		logger.info("sending user name from excel");
		lp.setpassword(pwd);
		logger.info("sending password from excel");
		lp.clickSubmit();

		landpage=new LandingPage();

		Thread.sleep(3000);
		try {
	if(landpage.checktable()) {
		System.out.println("successful login");
		logger.info("check to see if login is successful");
	}
		}
	catch(Exception e) {
		System.out.println("login failed");
		logger.info("check to see if login is unsuccessful");
		try {
			hp.captureScreen(driver,"whiteboxtest");
		} catch (IOException e1) {
			// TODO Auto-generated catch block
			e1.printStackTrace();
		}
		Assert.assertFalse(true);
	}
	}

	@DataProvider(name="logindata")
	String [][]getdata() throws IOException{
		String
path="C:\\Users\\vinay\\Downloads\\WhiteBoxFramework-master\\WhiteBoxFramework\\src\\test
\\resources\\testdata.xlsx";
		int rownum=ReadXL.getRowCount(path, "Sheet1");
```

```
		int colcount=ReadXL.getCellCount(path, "Sheet1", 1);
		String logindata[][]=new String[rownum][colcount];
		//0 is for header..column is 0
		for(int i=1;i<=rownum;i++) {
			for(int j=0;j<colcount;j++ ) {
				logindata[i-1][j]=ReadXL.getCellData(path, "Sheet1", i, j);//1  0
			}
		}
		return logindata;
	}

}
```

3. For Reporting Purposes - Extent Reports:

```
<!-- https://mvnrepository.com/artifact/com.aventstack/extentreports -->
<dependency>
   <groupId>com.aventstack</groupId>
   <artifactId>extentreports</artifactId>
   <version>3.0.0</version>
</dependency>
```

//Code for Reporting:

Create Reporting.java file in UtilityPAckage

[Import class / maven dependency add at the red-color marked words]

```
public class Reporting extends TestListenerAdapter
{
	public ExtentHtmlReporter htmlReporter;
	public ExtentReports extent;
	public ExtentTest logger;


	public void onStart(ITestContext testContext)
	{
		System.out.println("extent reports");
		String timeStamp = new
SimpleDateFormat("yyyy.MM.dd.HH.mm.ss").format(new Date());//time stamp
		String repName="Test-Report-"+timeStamp+".html";
```

```
		htmlReporter=new ExtentHtmlReporter(System.getProperty("user.dir")+
"/test-output/"+repName);//specify location of the report
		htmlReporter.loadXMLConfig(System.getProperty("user.dir")+
"/extent-config.xml");

		extent=new ExtentReports();

		extent.attachReporter(htmlReporter);
		extent.setSystemInfo("Host name","localhost");
		extent.setSystemInfo("Environemnt","QA");
		extent.setSystemInfo("user","krishna");

		htmlReporter.config().setDocumentTitle("WhiteBox Test Project"); // Tile of report
		htmlReporter.config().setReportName("Functional Test Automation Report"); //
name of the report
		htmlReporter.config().setTestViewChartLocation(ChartLocation.TOP); //location
of the chart
		htmlReporter.config().setTheme(Theme.DARK);
	}

	public void onTestSuccess(ITestResult tr)
	{
		System.out.println("on success of test");
		logger=extent.createTest(tr.getName()); // create new entry in th report

logger.log(Status.PASS,MarkupHelper.createLabel(tr.getName(),ExtentColor.GREEN)); // send
the passed information to the report with GREEN color highlighted
	}

	public void onTestFailure(ITestResult tr)
	{
		System.out.println("on failure of test");
		logger=extent.createTest(tr.getName()); // create new entry in th report

logger.log(Status.FAIL,MarkupHelper.createLabel(tr.getName(),ExtentColor.RED)); // send the
passed information to the report with GREEN color highlighted

		String
screenshotPath=System.getProperty("user.dir")+"\\Screenshots\\"+tr.getName()+".png";
		System.out.println(screenshotPath);

		File f = new File(screenshotPath);

		if(f.exists())
```

```
                {
                try {
                        logger.fail("Screenshot is below:" +
logger.addScreenCaptureFromPath(screenshotPath));
                        }
                catch (IOException e)
                                {
                                e.printStackTrace();
                                }
                }

        }

        public void onTestSkipped(ITestResult tr)
        {
                logger=extent.createTest(tr.getName()); // create new entry in th report

logger.log(Status.SKIP,MarkupHelper.createLabel(tr.getName(),ExtentColor.ORANGE));
        }

        public void onFinish(ITestContext testContext)
        {
                extent.flush();
        }
}
```