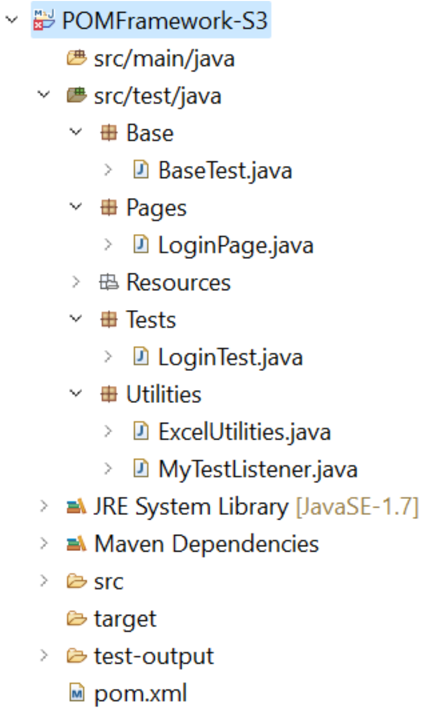
Page Object Model

Using The Page Object Model (POM) Approach To Organize Test Cases Efficiently.

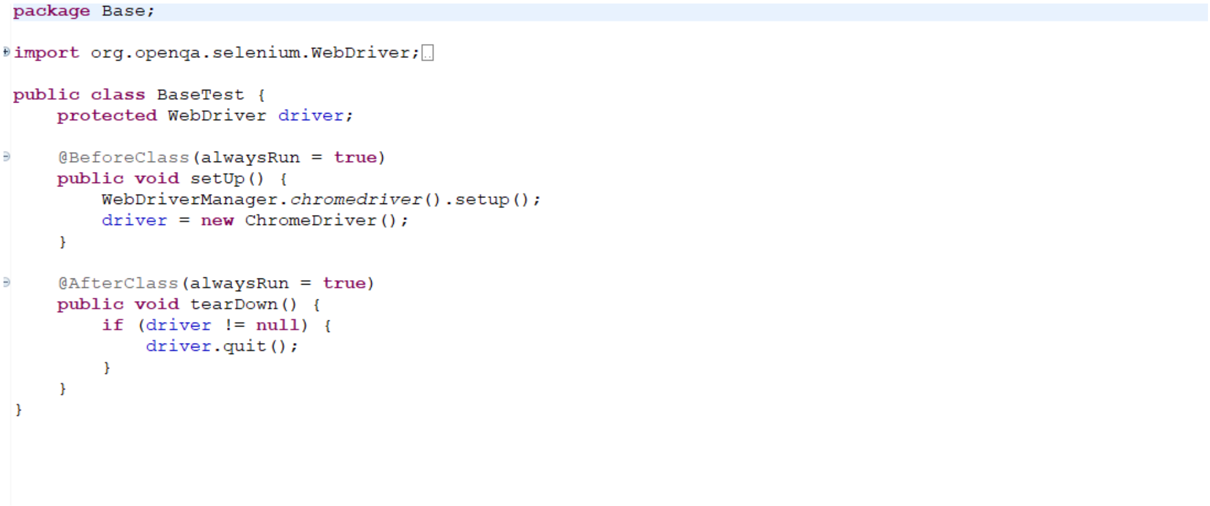
This Framework Will Handle Test Data And Allow Tests To Be Executed In Different Environments.

The Testng Framework Test Data Management Managing Test Data For Various Test Scenarios.

Page Object Model (POM) Organizing Test Cases Using POM For Efficient Maintenance And Readability. Maven Integration Managing Dependencies And Build Processes Using Maven



```
POMFramework-S3
    src/main/java
    src/test/java
        Base
            BaseTest.java
        Pages
            LoginPage.java
        Resources
        Tests
            LoginTest.java
        Utilities
            ExcelUtilities.java
            MyTestListener.java
    JRE System Library [JavaSE-1.7]
    Maven Dependencies
    src
    target
    test-output
    pom.xml
```



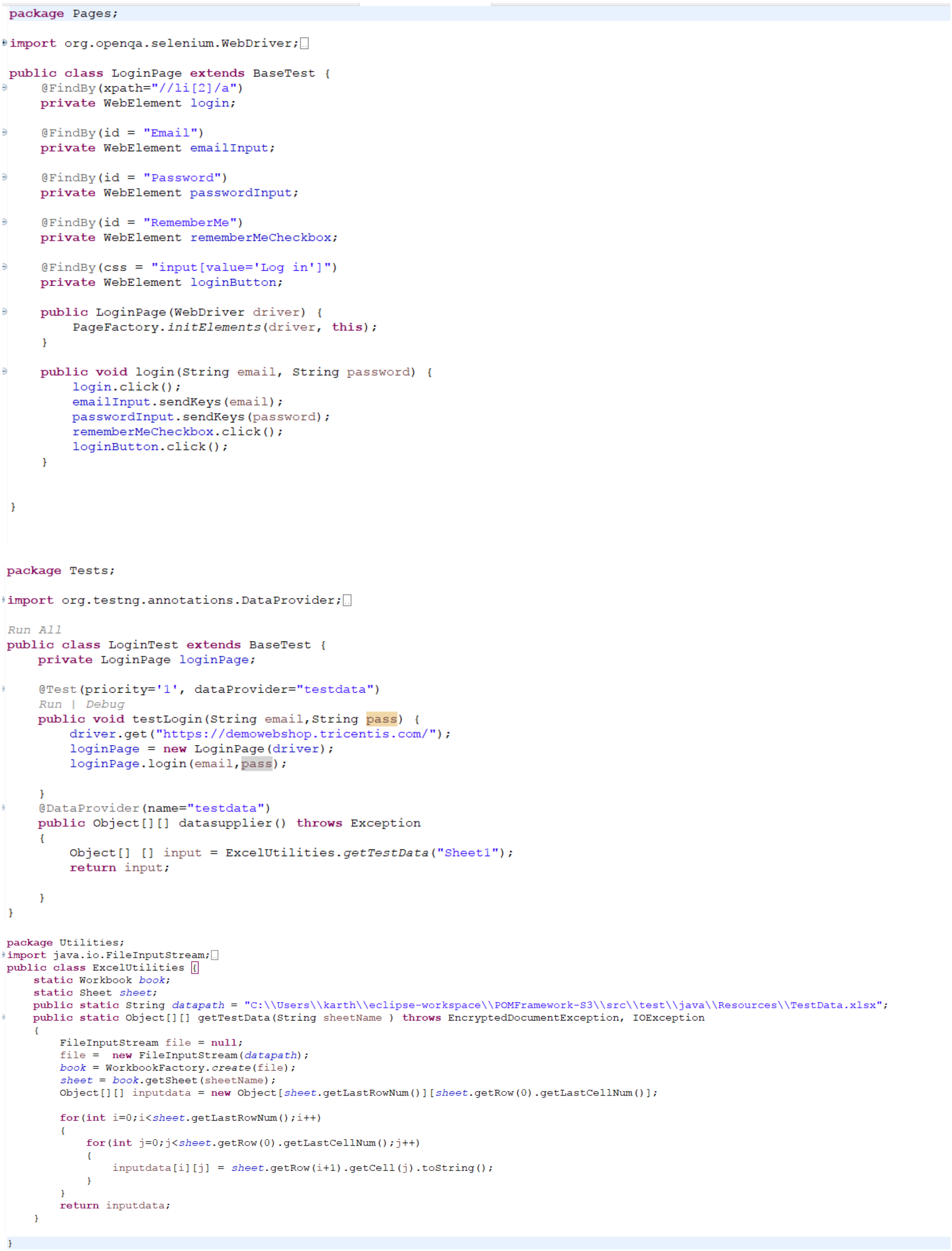
```
package Base;

import org.openqa.selenium.WebDriver;

public class BaseTest {
    protected WebDriver driver;

    @BeforeClass(alwaysRun = true)
    public void setUp() {
        WebDriverManager.chromedriver().setup();
        driver = new ChromeDriver();
    }

    @AfterClass(alwaysRun = true)
    public void tearDown() {
        if (driver != null) {
            driver.quit();
        }
    }
}
```



```
package Pages;

import org.openqa.selenium.WebDriver;□

public class LoginPage extends BaseTest {
    @FindBy(xpath="//li[2]/a")
    private WebElement login;

    @FindBy(id = "Email")
    private WebElement emailInput;

    @FindBy(id = "Password")
    private WebElement passwordInput;

    @FindBy(id = "RememberMe")
    private WebElement rememberMeCheckbox;

    @FindBy(css = "input[value='Log in']")
    private WebElement loginButton;

    public LoginPage(WebDriver driver) {
        PageFactory.initElements(driver, this);
    }

    public void login(String email, String password) {
        login.click();
        emailInput.sendKeys(email);
        passwordInput.sendKeys(password);
        rememberMeCheckbox.click();
        loginButton.click();
    }


}
```

```
package Tests;

import org.testng.annotations.DataProvider;□

Run All
public class LoginTest extends BaseTest {
    private LoginPage loginPage;

    @Test(priority='1', dataProvider="testdata")
    Run | Debug
    public void testLogin(String email,String pass) {
        driver.get("https://demowebshop.tricentis.com/");
        loginPage = new LoginPage(driver);
        loginPage.login(email,pass);

    }
    @DataProvider(name="testdata")
    public Object[][] datasupplier() throws Exception
    {
        Object[] [] input = ExcelUtilities.getTestData("Sheet1");
        return input;

    }
}
```

```
package Utilities;
import java.io.FileInputStream;□
public class ExcelUtilities {
    static Workbook book;
    static Sheet sheet;
    public static String datapath = "C:\\Users\\karth\\eclipse-workspace\\POMFramework-S3\\src\\test\\java\\Resources\\TestData.xlsx";
    public static Object[][] getTestData(String sheetName ) throws EncryptedDocumentException, IOException
    {
        FileInputStream file = null;
        file =  new FileInputStream(datapath);
        book = WorkbookFactory.create(file);
        sheet = book.getSheet(sheetName);
        Object[][] inputdata = new Object[sheet.getLastRowNum()][sheet.getRow(0).getLastCellNum()];

        for(int i=0;i<sheet.getLastRowNum();i++)
        {
            for(int j=0;j<sheet.getRow(0).getLastCellNum();j++)
            {
                inputdata[i][j] = sheet.getRow(i+1).getCell(j).toString();
            }
        }
        return inputdata;
    }

}
```

GitHub repository : https://github.com/Karthick-Office/Project01.git

My “POMFramework-S3” Project is present under the “POMFramework-S3” folder in the GitHub