

FRAMEWORK:

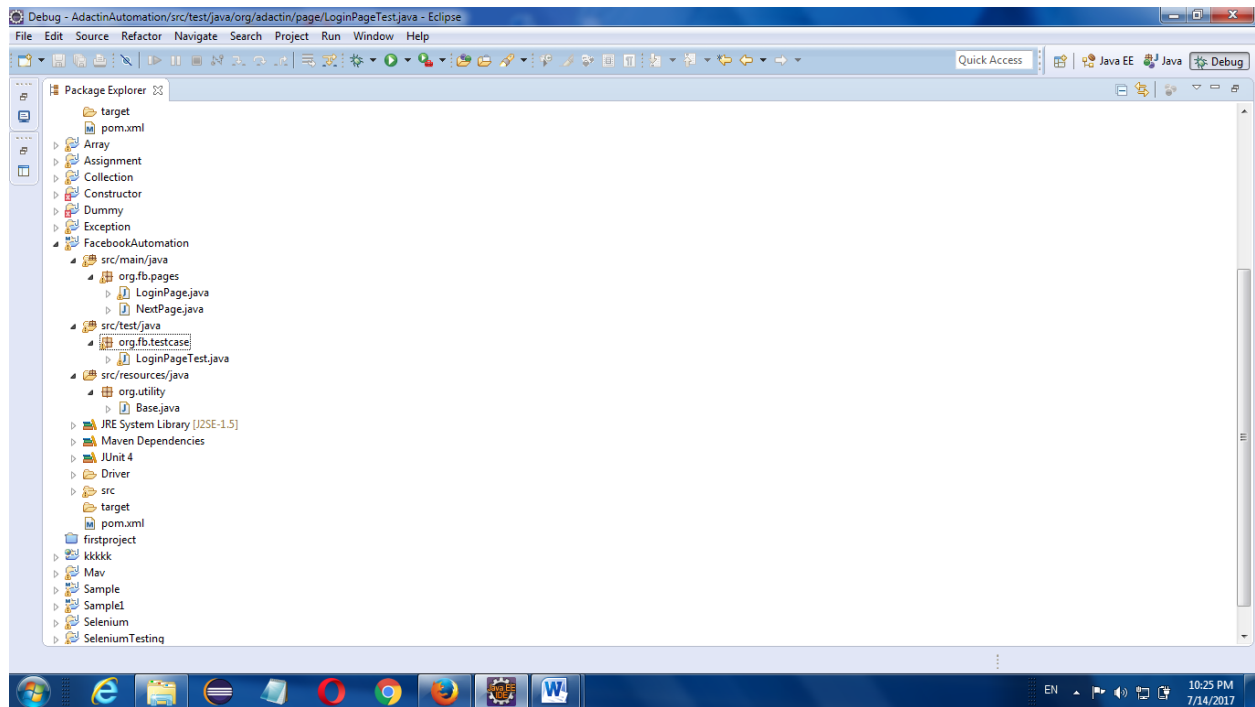
- A framework is a set of concept and practices, that provide support for automation testing.
- Different frameworks are available like,
 - POM(Page Object Model)
 - Junit
 - Cucumber
 - Data driven

POM:

- Page Object Model
- POM is an object repository design pattern in selenium webdriver
- POM creates our testing code maintainable and reusable
- Page factory is an optimized way to create object repository

POM Steps & Rules:

- Create a maven project/java project
- We have to create 3 source folders(packages)
 1. Src/main/java
 - It contains POM information
 2. Src/test/java (Junit,TestNG)
 - It contains login, asserts & registration
 3. src/resources/java
 - It contains reusable methods/codes. Ex, browser coding, radio button, scroll down and etc.



- In the main package /POM package, we have to create different class for different web pages,
- For example, in Facebook login page, we have to input user name, password and click login. This is one class and if we click login then will go to new home page that is another class. For that different pages we have to create different class
- Then all the annotations should be in private and use getters & setters
- Create a default constructor and contains one page factory
- This is a gateway to initialize the value
- In the test package(Junit), all the methods/annotations should be in public(access specifier)
- This package get the variables from main pom package and initialize(input) the value
- In the base package contains browser configuration, scroll down codes, check box code and etc.
- we can use this browser configuration/whatever in our Junit package

Example:

Facebook login using POM framework:

1.Base package: (resource)

```
public class Base {
    static WebDriver driver;
```

```

public static WebDriver getDriver(String browser) {
    if(browser.equals("chrome")){
        System.setProperty("webdriver.chrome.driver",
            "C:/Users/siva/workspace/Pom/Driver/chromedriver.exe");
        driver = new ChromeDriver();

    } else if(browser.equals("firefox")){
        System.setProperty("webdriver.gecko.driver",
            "C:/Users/siva/workspace/Pom/Driver/geckodriver.exe");
        driver = new FirefoxDriver();

    } else if(browser.equals("ie")){
        System.setProperty("webdriver.ie.driver",
            "C:/Users/siva/workspace/Pom/Driver/IEDriverServer.exe");
        driver = new InternetExplorerDriver();

    } else if(browser.equals("opera")){
        System.setProperty("webdriver.opera.driver",
            "C:/Users/siva/workspace/Pom/Driver/operadriver.exe");
        driver = new OperaDriver();

    }

    driver.manage().window().maximize();
    driver.get("https://www.facebook.com/");
    return driver;
}
}

```

- Here we can configure all browser setup's
- In the Junit package we can use which browser we want

2. POM Package: (Main)

Class 1:

```

public class LoginPage {
    static WebDriver driver;
    @FindBy(name="email")
    private WebElement txtUserName;
    @FindBy(id="pass")
    private WebElement txtUserPassword;
    @FindBy(id="loginbutton")
    private WebElement btnLoginButton;
    @FindBy(xpath="//*[@class='fb_logo img sp_sh983SE11WH sx_84be92']")

```

```

private WebElement imgFbLogo;

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

public WebElement getImgFbLogo() {
    return imgFbLogo;
}

public void setImgFbLogo(WebElement imgFbLogo) {
    this.imgFbLogo = imgFbLogo;
}

public String getTxtUserName() {
    return txtUserName.getAttribute("value");
}

public void setTxtUserName(String txtUserName) {
    this.txtUserName.sendKeys(txtUserName);
}

public String getTxtUserPassword() {
    return txtUserPassword.getAttribute("value");
}

public void setTxtUserPassword(String txtUserPassword) {
    this.txtUserPassword.sendKeys(txtUserPassword);
}

public WebElement getBtnLoginButton() {
    return btnLoginButton;
}

public void setBtnLoginButton(WebElement btnLoginButton) {
    this.btnLoginButton = btnLoginButton;
}
}

```

Here,

- We first declare all the annotations like username, password , login and etc.
- Then create one default constructor with page factory which is used to initialize the value from the Junit.

```

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

```

- Then click generate getters and setters
- If we click login button, it will move to next page
- If we check or insert something in next page, we have to create one more base class
(i.e.) NextPage (another class –see in below)

Class 2:

```
public class NextPage {
    WebDriver driver;
    @FindBy(xpath="//*[text()='Recover Your Account']")
    private WebElement txtRecover;

    public WebElement getTxtRecover() {
        return txtRecover;
    }

    public void setTxtRecover(WebElement txtRecover) {
        this.txtRecover = txtRecover;
    }

    public NextPage(WebDriver ldriver) {
        this.driver=ldriver;
        PageFactory.initElements(driver, this);
    }
}
```

3.Test package (JUnit):

```
public class LoginPageTest {
    static WebDriver driver;
    LoginPage login;
    static Base base;
    NextPage next;
    @Before
    public void launchBrowser() {
        base=new Base();
        driver=base.getDriver("firefox");
    }
    @Test
    public void verifyLogin() {
        login=new LoginPage(driver);
        next=new NextPage(driver);
        // verify the fb logo is available
        Assert.assertTrue(login.getImgFbLogo().isDisplayed());
        // set the user name
    }
}
```

```

login.setTxtUserName("venkat91675@gmail.com");
//check the user name you entered
Assert.assertEquals("venkat91675@gmail.com", login.getTxtUserName());
// set the password
login.setTxtUserPassword("venkat12345");
//check the user name you entered
Assert.assertEquals("venkat12345", login.getTxtUserPassword());
//click login
login.getBtnLoginButton().click();
//verify next page recover option is available
Assert.assertTrue(next.getTxtRecover().isDisplayed());

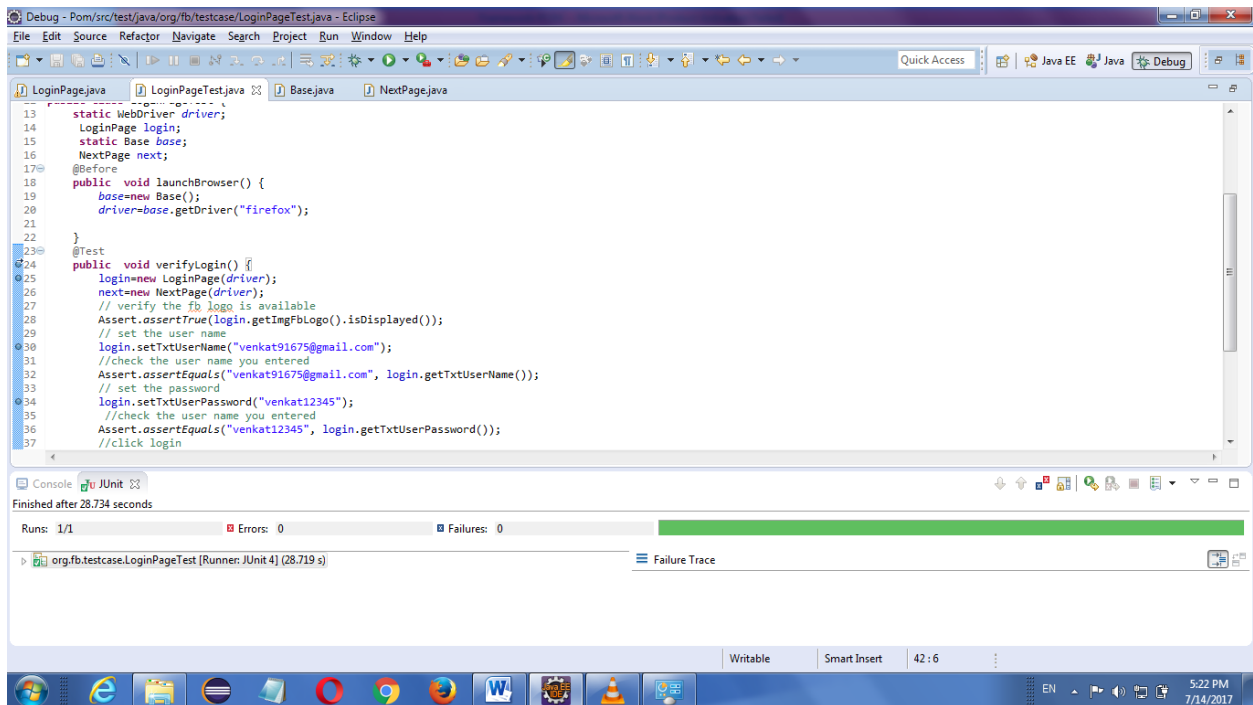
}
@After
public void closeBrowser() {
    driver.quit();
}

}

```

- We can perform all test action here
- First launch the browser which one we want
- Then Verify the login details
- Assert → class
- assertTrue() → is a method used to check the particular value is available or not.
- If its not available, it will fail the test case
- As well as assertEquals() is also one of the method to check
- Then close the browser

Output:



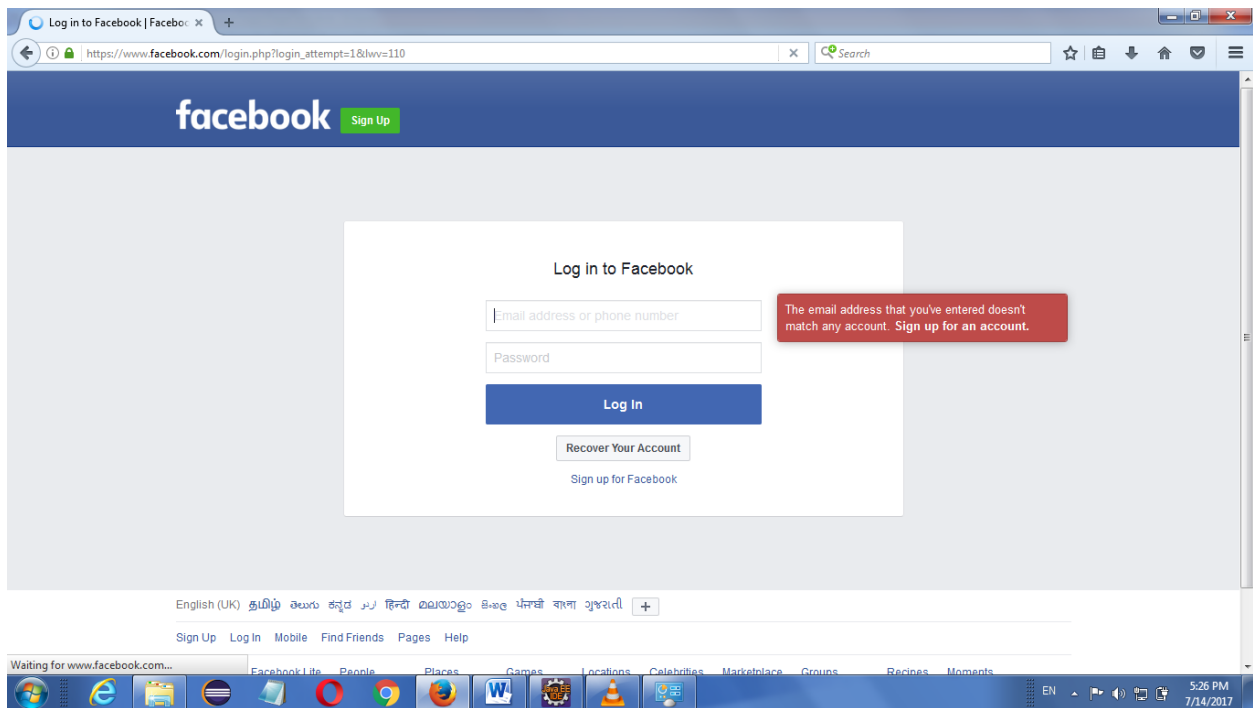
The screenshot shows the Eclipse IDE interface. The top editor displays the `LoginPageTest.java` file with the following code:

```
13 static WebDriver driver;
14 LoginPage login;
15 static Base base;
16 NextPage next;
17 @Before
18 public void launchBrowser() {
19     base=new Base();
20     driver=base.getDriver("firefox");
21 }
22 @Test
23 public void verifyLogin() {
24     login=new LoginPage(driver);
25     next=new NextPage(driver);
26     // verify the fb logo is available
27     Assert.assertTrue(login.getImgFbLogo().isDisplayed());
28     // set the user name
29     login.setTxtUserName("venkat91675@gmail.com");
30     //check the user name you entered
31     Assert.assertEquals("venkat91675@gmail.com", login.getTxtUserName());
32     // set the password
33     login.setTxtUserPassword("venkat12345");
34     //check the user name you entered
35     Assert.assertEquals("venkat12345", login.getTxtUserPassword());
36     //click login
37 }
```

The bottom console shows the test results:

```
Finished after 28.734 seconds
Runs: 1/1 Errors: 0 Failures: 0
org.fb.testcase.LoginPageTest [Runner: JUnit 4] (28.719 s)
```

The status bar at the bottom indicates the file is writable, smart insert is enabled, and the cursor is at line 42, column 6.



EXERCISE : POM

1. Adactin.com automation using POM framework

Base class:

```
public class Base {
    static WebDriver driver;
    public static WebDriver getDriver(String browser) {
        if(browser.equals("chrome")){
            System.setProperty("webdriver.chrome.driver",
                               "C:/Users/siva/workspace/Pom/Driver/chromedriver.exe");
            driver = new ChromeDriver();

        }else if(browser.equals("firefox")){
            System.setProperty("webdriver.gecko.driver",
                               "C:/Users/siva/workspace/Pom/Driver/geckodriver.exe");
            driver = new FirefoxDriver();

        }else if(browser.equals("ie")){
            System.setProperty("webdriver.ie.driver",
                               "C:/Users/siva/workspace/Pom/Driver/IEDriverServer.exe");
            driver = new InternetExplorerDriver();

        }else if(browser.equals("opera")){
            System.setProperty("webdriver.opera.driver",
                               "C:/Users/siva/workspace/Pom/Driver/opera.driver.exe");
            driver = new OperaDriver();

        }
        driver.manage().window().maximize();
        driver.get("http://www.adactin.com/HotelApp/");
        return driver;
    }
}
```

Main class: 1

```
public class LoginPage1 {
    static WebDriver driver;
    @FindBy(xpath="//*[@id='username']")
```



```

private WebElement txtUserName;
@FindBy(id="password")
private WebElement txtUserPassword;
@FindBy(id="login")
private WebElement btnLoginButton;
@FindBy(xpath="//*[@alt='AdactIn Group']")
private WebElement imgAdactinLogo;
@FindBy(xpath="//*[@class='header_title']")
private WebElement imgHeader;
@FindBy(xpath="//*[text()='Welcome to AdactIn Group of Hotels']")
private WebElement imgWelcome;
@FindBy(xpath="//*[@alt='Hotel Image 1']")
private WebElement imgHotel1;
@FindBy(xpath="//*[@alt='Hotel Image 2']")
private WebElement imgHotel2;
@FindBy(xpath="//*[@alt='Hotel Image 3']")
private WebElement imgHotel3;
@FindBy(xpath="//*[@alt='Hotel Image 4']")
private WebElement imgHotel4;
@FindBy(xpath="//*[text()='Existing User Login - Build 1']")
private WebElement txtExUser;
@FindBy(xpath="//*[text()='Forgot Password?']")
private WebElement txtForPass;
@FindBy(xpath="//*[text()='New User Register Here']")
private WebElement txtNewUser;
@FindBy(xpath="//*[text()='Important Note:']")
private WebElement txtImpNote;

public WebElement getImgAdactinLogo() {
    return imgAdactinLogo;
}
public void setImgAdactinLogo(WebElement imgAdactinLogo) {
    this.imgAdactinLogo = imgAdactinLogo;
}
public WebElement getImgHeader() {
    return imgHeader;
}
public void setImgHeader(WebElement imgHeader) {
    this.imgHeader = imgHeader;
}
public WebElement getImgWelcome() {
    return imgWelcome;
}
public void setImgWelcome(WebElement imgWelcome) {
    this.imgWelcome = imgWelcome;
}

```

```
public WebElement getImgHotel1() {  
    return imgHotel1;  
}  
public void setImgHotel1(WebElement imgHotel1) {  
    this.imgHotel1 = imgHotel1;  
}  
public WebElement getImgHotel2() {  
    return imgHotel2;  
}  
public void setImgHotel2(WebElement imgHotel2) {  
    this.imgHotel2 = imgHotel2;  
}  
public WebElement getImgHotel3() {  
    return imgHotel3;  
}  
public void setImgHotel3(WebElement imgHotel3) {  
    this.imgHotel3 = imgHotel3;  
}  
public WebElement getImgHotel4() {  
    return imgHotel4;  
}  
public void setImgHotel4(WebElement imgHotel4) {  
    this.imgHotel4 = imgHotel4;  
}  
public WebElement getTxtExUser() {  
    return txtExUser;  
}  
public void setTxtExUser(WebElement txtExUser) {  
    this.txtExUser = txtExUser;  
}  
public WebElement getTxtForPass() {  
    return txtForPass;  
}  
public void setTxtForPass(WebElement txtForPass) {  
    this.txtForPass = txtForPass;  
}  
public WebElement getTxtNewUser() {  
    return txtNewUser;  
}  
public void setTxtNewUser(WebElement txtNewUser) {  
    this.txtNewUser = txtNewUser;  
}  
public WebElement getTxtImpNote() {  
    return txtImpNote;  
}  
public void setTxtImpNote(WebElement txtImpNote) {
```

```

        this.txtImpNote = txtImpNote;
    }
    public LoginPage1(WebDriver ldriver) {
        this.driver=ldriver;
        PageFactory.initElements(driver, this);
    }
    public WebElement getimgAdactinLogo() {
        return imgAdactinLogo;
    }
    public void setimgAdactinLogo(WebElement imgAdactinLogo) {
        this.imgAdactinLogo = imgAdactinLogo;
    }
    public String getTxtUserName() {
        return txtUserName.getAttribute("value");
    }
    public void setTxtUserName(String txtUserName) {
        this.txtUserName.sendKeys(txtUserName);
    }
    public String getTxtUserPassword() {
        return txtUserPassword.getAttribute("value");
    }
    public void setTxtUserPassword(String txtUserPassword) {
        this.txtUserPassword.sendKeys(txtUserPassword);
    }
    public WebElement getBtnLoginButton() {
        return btnLoginButton;
    }
    public void setBtnLoginButton(WebElement btnLoginButton) {
        this.btnLoginButton = btnLoginButton;
    }
}

```

Main class: 2

```

public class NextPage1 {
    WebDriver driver;
    @FindBy(xpath="//*[text()='Logout']")
    private WebElement txtLogout;

    public WebElement getTxtLogout() {

```

```

        return txtLogout;
    }

    public void setTxtLogout(WebElement txtLogout) {
        this.txtLogout = txtLogout;
    }

    public NextPage1(WebDriver ldriver) {
        this.driver=ldriver;
        PageFactory.initElements(driver, this);
    }
}

```

Junit class:

```

public class LoginPageTest1 {
    static WebDriver driver;
    LoginPage1 login1;
    static Base b;
    NextPage1 next1;
    @Before
    public void launchBrowser() {
        driver=b.getDriver("firefox");
    }
    @Test
    public void verifyLogin() {
        login1=new LoginPage1(driver);
        next1=new NextPage1(driver);
        Assert.assertTrue(login1.getimgAdactinLogo().isDisplayed());
        Assert.assertTrue(login1.getImgHeader().isDisplayed());
        Assert.assertTrue(login1.getImgWelcome().isDisplayed());
        Assert.assertTrue(login1.getImgHotel1().isDisplayed());
        Assert.assertTrue(login1.getImgHotel2().isDisplayed());
        Assert.assertTrue(login1.getImgHotel3().isDisplayed());
        Assert.assertTrue(login1.getImgHotel4().isDisplayed());
        Assert.assertTrue(login1.getTxtExUser().isDisplayed());
        Assert.assertTrue(login1.getTxtForPass().isDisplayed());
    }
}

```

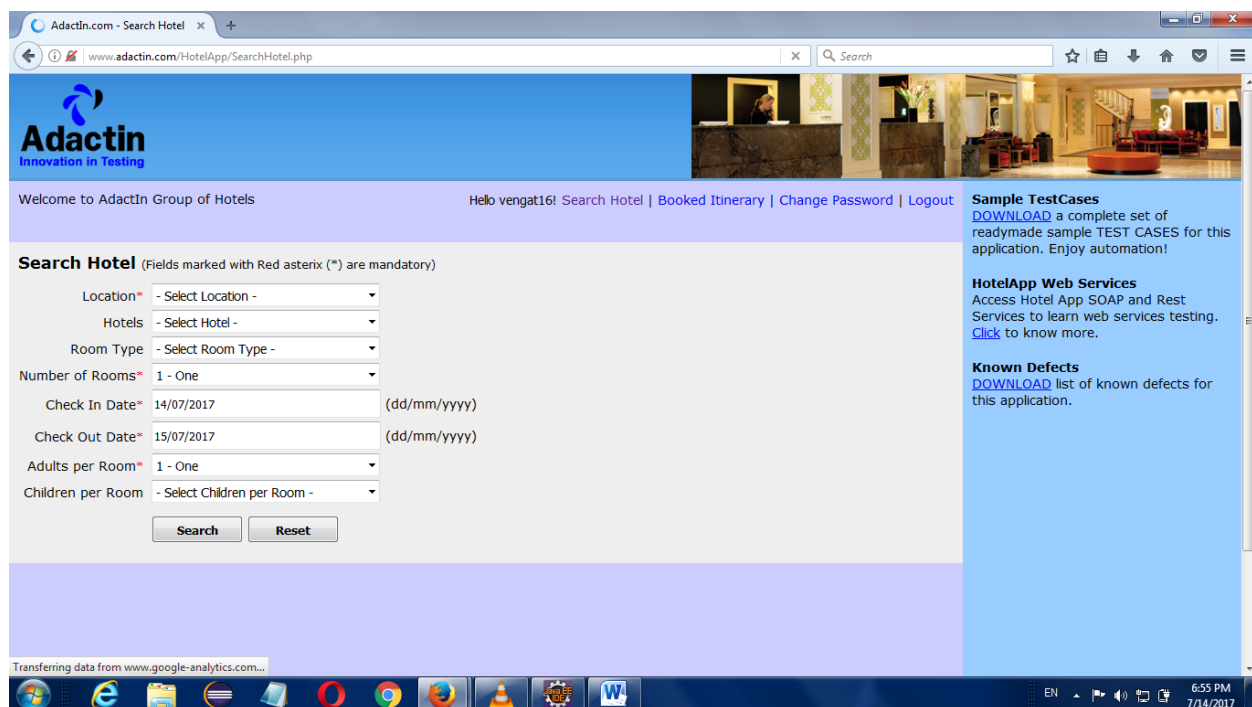
```

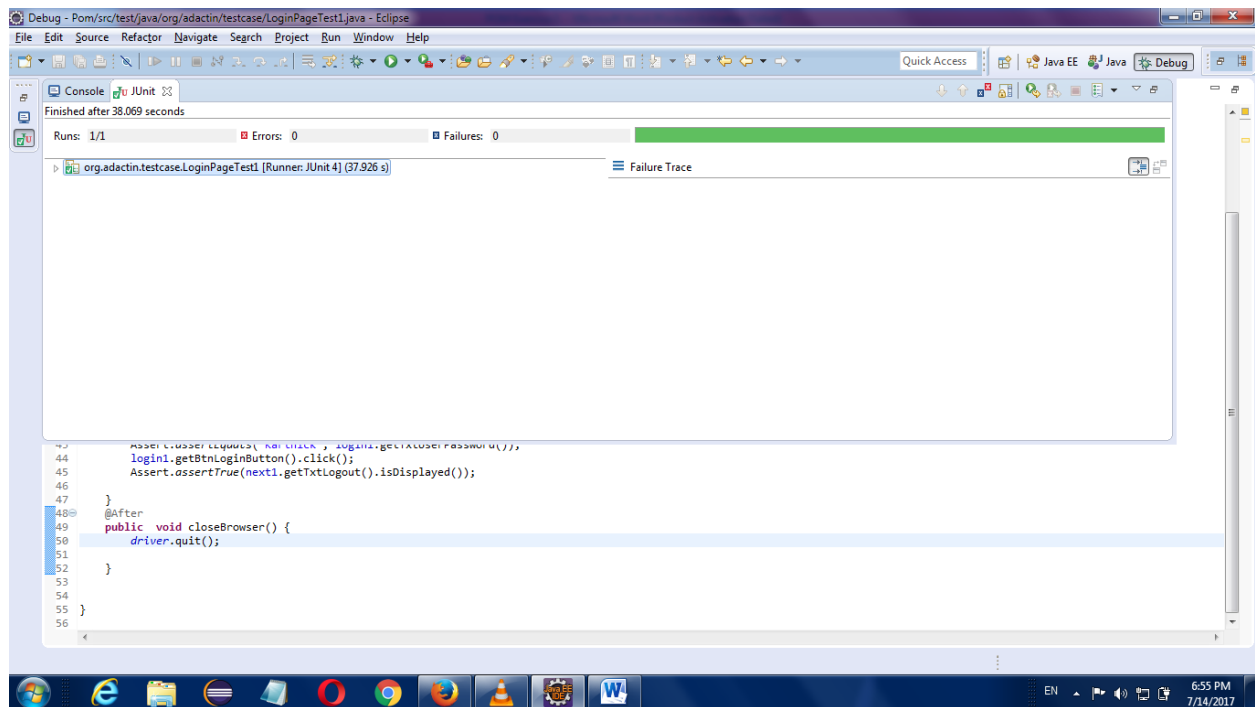
        Assert.assertTrue(login1.getTxtNewUser().isDisplayed());
        Assert.assertTrue(login1.getTxtImpNote().isDisplayed());
        login1.setTxtUserName("vengat16");
        Assert.assertEquals("vengat16", login1.getTxtUserName());
        login1.setTxtUserPassword("Karthick");
        Assert.assertEquals("Karthick", login1.getTxtUserPassword());
        login1.getBtnLoginButton().click();
        Assert.assertTrue(next1.getTxtLogout().isDisplayed());

    }
    @After
    public void closeBrowser() {
        driver.quit();
    }
}

```

Output:





REUSABLE METHODS:

- We can create some reusable methods in base class like sendkeys, getvalue, isDisplayed and etc

1.isDisplayed:

Base class:

```
public boolean elementFound(WebDriver driver, int time, WebElement
element) {
    boolean res = false;
    driver.manage().timeouts().implicitlyWait(time,
TimeUnit.SECONDS);
    try{
        res = element.isDisplayed();
    }catch(Exception e){
        e.printStackTrace();
    }
    return res;
}
```

```
}
```

Junit class:

```
public void verifyLogin() {  
    login=new LoginPage(driver);  
    next=new NextPage(driver);  
    Assert.assertTrue(base.elementFound(driver,10,login.getImgFbLogo()));  
}
```

2.sendkeys:

Base class:

```
public void setText(WebElement element, String name) {  
    if (name != null) {  
        element.clear();  
        element.sendKeys(name);  
    }  
}
```

Junit class:

```
public void verifyLogin() {  
    login=new LoginPage(driver);  
    next=new NextPage(driver);  
    base.setText(login.getTxtUserName(),"venkat91675@gmail.com");  
    base.setText(login.getTxtUserPassword(),"venkat12345");  
}}
```

3.getvalue:

Base value:

```
public String getText(WebElement element) {  
  
    String name = element.getAttribute("value");  
    return name;  
}
```

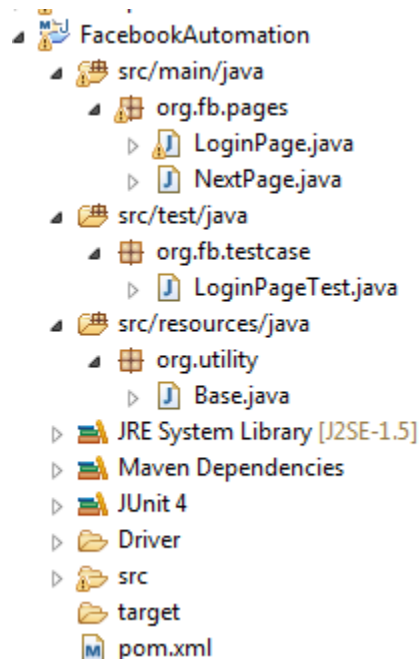
Junit class:

```
public void verifyLogin() {  
    login=new LoginPage(driver);  
    next=new NextPage(driver);  
    Assert.assertTrue(base.elementFound(driver,10,login.getImgFbLogo()));  
    base.setText(login.getTxtUserName(),"venkat91675@gmail.com");  
    Assert.assertEquals("venkat91675@gmail.com",  
base.getText(login.getTxtUserName()));  
    base.setText(login.getTxtUserPassword(),"venkat12345");  
    Assert.assertEquals("venkat12345", base.getText(login.getTxtUserPassword()));  
    login.getBtnLoginButton().click();  
    Assert.assertTrue(base.elementFound(driver, 10, next.getTxtRecover()));  
}
```

Example program:

1.Facebook automation using POM & JUnit framework

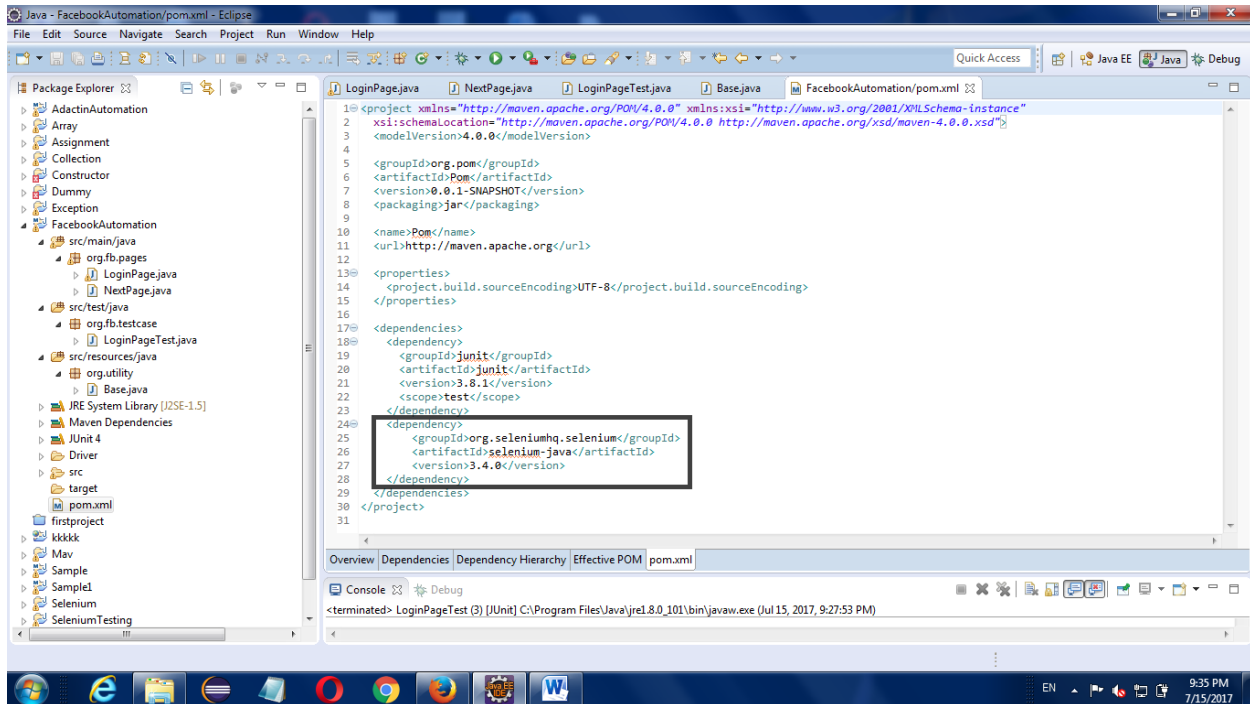
1. Project Structure:



2. Maven dependencies:

- If we want to add some jar file in maven dependency, click pom.xml
- Go to pom.xml and add particular dependency code and save
- Then it will added to maven dependencies

- For ex, if we want to add selenium jar, just copy code from internet and add in pom.xml



3. Base class:

```

public class Base {
    static WebDriver driver;
    WebDriverWait wait;

    public WebDriver getDriver(String browser) {
        if (browser.equals("chrome")) {
            System.setProperty("webdriver.chrome.driver",
                "C:/Users/siva/workspace/Pom/Driver/chromedriver.exe");
            driver = new ChromeDriver();
        } else if (browser.equals("firefox")) {
            System.setProperty("webdriver.gecko.driver",
                "C:/Users/siva/workspace/Pom/Driver/geckodriver.exe");
            driver = new FirefoxDriver();
        } else if (browser.equals("ie")) {
            System.setProperty("webdriver.ie.driver",
                "C:/Users/siva/workspace/Pom/Driver/IEDriverServer.exe");

```

```

        driver = new InternetExplorerDriver();

    } else if (browser.equals("opera")) {
        System.setProperty("webdriver.opera.driver",
"C:/Users/siva/workspace/Pom/Driver/operadriver.exe");
        driver = new OperaDriver();

    }
    driver.manage().window().maximize();
    driver.get("https://www.facebook.com/");
    return driver;
}

public boolean elementToBeVisible(WebDriver driver, int time,
    WebElement element) {
    boolean flag = false;
    try {
        wait = new WebDriverWait(driver, time);
        wait.until(ExpectedConditions.visibilityOf(element));
        flag = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return flag;
}

public boolean elementFound(WebDriver driver, int time, WebElement
element) {
    boolean res = false;
    driver.manage().timeouts().implicitlyWait(time,
TimeUnit.SECONDS);
    try {
        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return res;
}

public boolean elementFound(WebElement element) {
    boolean b = false;
    try {
        b = element.isDisplayed();
    } catch (Exception e) {

```

```

        e.printStackTrace();
    }
    return b;
}

public void setText(WebElement element, String name) {
    if (name != null && elementFound(element)) {
        element.clear();
        element.sendKeys(name);
    }
}

public String getText(WebElement element) {
    String name = null;
    if (elementFound(element)) {
        name = element.getAttribute("value");
    }
    return name;
}

public void clickBtn(WebElement element) {
    if (elementFound(element)) {
        element.click();
    }
}
}

```

4. Junit/Test class:

```

public class LoginPageTest {
    static WebDriver driver;
    LoginPage login;
    static Base base;
    NextPage next;
    @Before
    public void launchBrowser() {
        base = new Base();
        driver=base.getDriver("firefox");
    }
    @Test

```

```

    public void verifyLogin() {
        login=new LoginPage(driver);
        next=new NextPage(driver);

        Assert.assertTrue(base.elementFound(driver,10,login.getImgFbLogo()));
        base.setText(login.getTxtUserName(),"venkat91675@gmail.com");
        Assert.assertEquals("venkat91675@gmail.com",
            base.getText(login.getTxtUserName()));
        base.setText(login.getTxtUserPassword(),"venkat12345");
        Assert.assertEquals("venkat12345",
            base.getText(login.getTxtUserPassword()));
        login.getBtnLoginButton().click();
        Assert.assertTrue(base.elementFound(driver, 10,
            next.getTxtRecover()));

    }
    @After
    public void closeBrowser() {
        driver.quit();
    }
}

```

5. Main class:

Class 1:

```

public class LoginPage {
    static WebDriver driver;
    @FindBy(id="email")
    private WebElement txtUserName;
    @FindBy(id="pass")
    private WebElement txtUserPassword;
    @FindBy(id="u_0_r")
    private WebElement btnLoginButton;
    @FindBy(xpath="//*[@class='fb_logo img sp_sh983SE11WH sx_84be92']")
    private WebElement imgFbLogo;

    public LoginPage(WebDriver ldriver) {
        this.driver=ldriver;
        PageFactory.initElements(driver, this);
    }
    public WebElement getImgFbLogo() {
        return imgFbLogo;
    }
}

```

```

    public void setImgFbLogo(WebElement imgFbLogo) {
        this.imgFbLogo = imgFbLogo;
    }
    public WebElement getTxtUserName() {
        return txtUserName;
    }
    public void setTxtUserName(WebElement txtUserName) {
        this.txtUserName=txtUserName;
    }
    public WebElement getTxtUserPassword() {
        return txtUserPassword;
    }
    public void setTxtUserPassword(WebElement txtUserPassword) {
        this.txtUserPassword=txtUserPassword;
    }
    public WebElement getBtnLoginButton() {
        return btnLoginButton;
    }
    public void setBtnLoginButton(WebElement btnLoginButton) {
        this.btnLoginButton = btnLoginButton;
    }
}

```

Class 2:

```

public class NextPage {
    WebDriver driver;
    @FindBy(xpath="//*[text()='Recover Your Account']")
    private WebElement txtRecover;

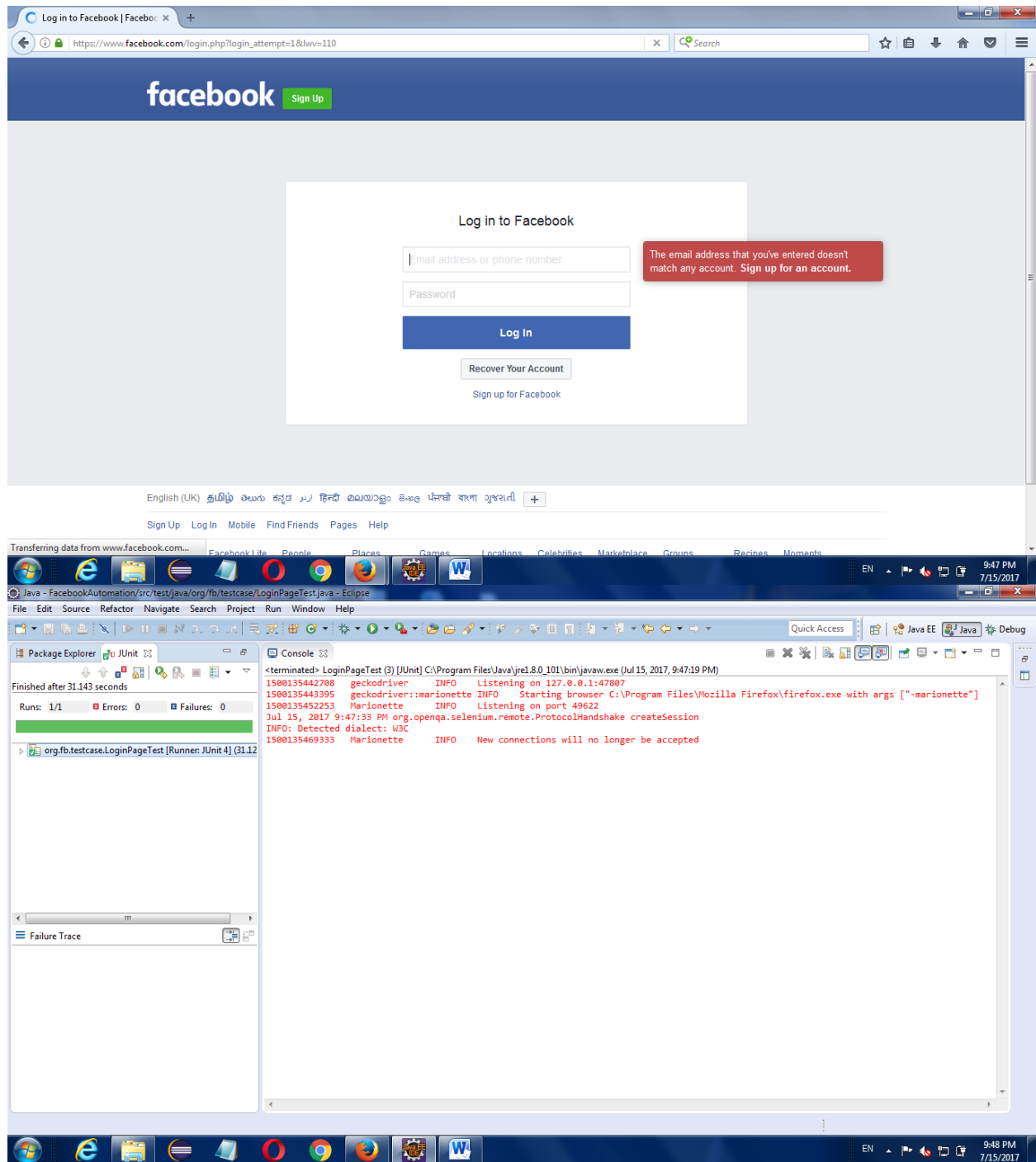
    public WebElement getTxtRecover() {
        return txtRecover;
    }

    public void setTxtRecover(WebElement txtRecover) {
        this.txtRecover = txtRecover;
    }

    public NextPage(WebDriver ldriver) {
        this.driver=ldriver;
        PageFactory.initElements(driver, this);
    }
}

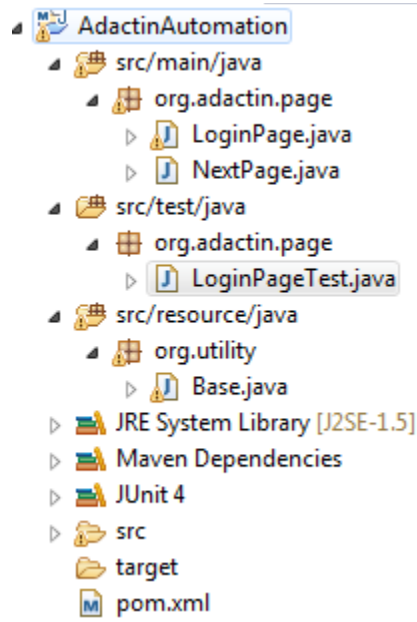
```

Output:



2. Adactin Hotel automation using POM & JUnit Framework(Updated reusable methods)- verify login functionality

1. Project structure:



2. Base class:

```
public class Base {
    static WebDriver driver;
    WebDriverWait wait;

    public static WebDriver getDriver(String browser) {
        if (browser.equals("chrome")) {
            System.setProperty("webdriver.chrome.driver",
                "C:/Users/siva/workspace/Pom/Driver/chromedriver.exe");
            driver = new ChromeDriver();

        } else if (browser.equals("firefox")) {
            System.setProperty("webdriver.gecko.driver",
                "C:/Users/siva/workspace/Pom/Driver/geckodriver.exe");
            driver = new FirefoxDriver();

        } else if (browser.equals("ie")) {
            System.setProperty("webdriver.ie.driver",
                "C:/Users/siva/workspace/Pom/Driver/IEDriverServer.exe");
            driver = new InternetExplorerDriver();
        }
    }
}
```

```

        } else if (browser.equals("opera")) {
            System.setProperty("webdriver.opera.driver",
"C:/Users/siva/workspace/Pom/Driver/operadriver.exe");
            driver = new OperaDriver();

        }
        driver.manage().window().maximize();
        driver.get("http://www.adactin.com/HotelApp/");
        return driver;
    }

    public boolean elementToBeVisible(WebDriver driver, int time,
        WebElement element) {
        boolean flag = false;
        try {
            wait = new WebDriverWait(driver, time);
            wait.until(ExpectedConditions.visibilityOf(element));
            flag = true;
        } catch (Exception e) {
            e.printStackTrace();
        }
        return flag;
    }

    public boolean elementFound(WebDriver driver, int time, WebElement element)
    {
        boolean res = false;
        driver.manage().timeouts().implicitlyWait(time, TimeUnit.SECONDS);
        try {
            res = element.isDisplayed();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return res;
    }

    public boolean elementFound(WebElement element) {
        boolean b = false;
        try {
            b = element.isDisplayed();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

```



```

        return b;
    }

    public void setText(WebElement element, String name) {
        if (name != null && elementFound(element)) {
            element.clear();
            element.sendKeys(name);
        }
    }

    public String getText(WebElement element) {
        String name = null;
        if (elementFound(element)) {
            name = element.getAttribute("value");
        }
        return name;
    }

    public void clickBtn(WebElement element) {
        if (elementFound(element)) {
            element.click();
        }
    }
}

```

3. JUnit class:

```

public class LoginPageTest {
    static WebDriver driver;
    LoginPage login;
    static Base base;
    NextPage next;

    @Before
    public void launchBrowser() {
        base = new Base();
        driver = Base.getDriver("firefox");
    }

    @Test
    public void verifyLogin() {

```

```

        login = new LoginPage(driver);
        next = new NextPage(driver);
        Assert.assertTrue(base.elementFound(driver, 1,
            login.getimgAdactinLogo()));
        Assert.assertTrue(base.elementFound(login.getImgHeader()));
        Assert.assertTrue(base.elementFound(login.getImgWelcome()));
        Assert.assertTrue(base.elementFound(login.getImgHotel1()));
        Assert.assertTrue(base.elementFound(login.getImgHotel2()));
        Assert.assertTrue(base.elementFound(login.getImgHotel3()));
        Assert.assertTrue(base.elementFound(login.getImgHotel4()));
        Assert.assertTrue(base.elementFound(login.getTxtExUser()));
        Assert.assertTrue(base.elementFound(login.getTxtForPass()));
        Assert.assertTrue(base.elementFound(login.getTxtNewUser()));
        Assert.assertTrue(base.elementFound(login.getTxtImpNote()));
        base.setText(login.getTxtUserName(), "vengat16");
        Assert.assertEquals("vengat16",
            base.getText(login.getTxtUserName()));
        base.setText(login.getTxtUserPassword(), "Karthick");
        Assert.assertEquals("Karthick",
            base.getText(login.getTxtUserPassword()));
        login.getTxtUserPassword().sendKeys(Keys.ENTER);
        base.elementToBeVisible(driver, 20, login.getBtnLoginButton());
        base.clickBtn(login.getBtnLoginButton());
        base.elementToBeVisible(driver, 20, next.getTxtLogout());
        Assert.assertTrue(base.elementFound(driver, 20,
            next.getTxtLogout()));
    }

    @After
    public void closeBrowser() {
        driver.quit();
    }
}

```

4. Main class:

Class 1:

```

public class LoginPage {
    static WebDriver driver;
    @FindBy(xpath="//*[@id='username']")
    private WebElement txtUserName;
    @FindBy(id="password")
    private WebElement txtUserPassword;
}

```

```

@FindBy(id="login")
private WebElement btnLoginButton;
@FindBy(xpath="//*[@alt='AdactIn Group']")
private WebElement imgAdactinLogo;
@FindBy(xpath="//*[@class='header_title']")
private WebElement imgHeader;
@FindBy(xpath="//*[text()='Welcome to AdactIn Group of Hotels']")
private WebElement imgWelcome;
@FindBy(xpath="//*[@alt='Hotel Image 1']")
private WebElement imgHotel1;
@FindBy(xpath="//*[@alt='Hotel Image 2']")
private WebElement imgHotel2;
@FindBy(xpath="//*[@alt='Hotel Image 3']")
private WebElement imgHotel3;
@FindBy(xpath="//*[@alt='Hotel Image 4']")
private WebElement imgHotel4;
@FindBy(xpath="//*[text()='Existing User Login - Build 1']")
private WebElement txtExUser;
@FindBy(xpath="//*[text()='Forgot Password?']")
private WebElement txtForPass;
@FindBy(xpath="//*[text()='New User Register Here']")
private WebElement txtNewUser;
@FindBy(xpath="//*[text()='Important Note:']")
private WebElement txtImpNote;

public WebElement getImgAdactinLogo() {
    return imgAdactinLogo;
}
public void setImgAdactinLogo(WebElement imgAdactinLogo) {
    this.imgAdactinLogo = imgAdactinLogo;
}
public WebElement getImgHeader() {
    return imgHeader;
}
public void setImgHeader(WebElement imgHeader) {
    this.imgHeader = imgHeader;
}
public WebElement getImgWelcome() {
    return imgWelcome;
}
public void setImgWelcome(WebElement imgWelcome) {
    this.imgWelcome = imgWelcome;
}
public WebElement getImgHotel1() {
    return imgHotel1;
}

```

```
public void setImgHotel1(WebElement imgHotel1) {
    this.imgHotel1 = imgHotel1;
}
public WebElement getImgHotel2() {
    return imgHotel2;
}
public void setImgHotel2(WebElement imgHotel2) {
    this.imgHotel2 = imgHotel2;
}
public WebElement getImgHotel3() {
    return imgHotel3;
}
public void setImgHotel3(WebElement imgHotel3) {
    this.imgHotel3 = imgHotel3;
}
public WebElement getImgHotel4() {
    return imgHotel4;
}
public void setImgHotel4(WebElement imgHotel4) {
    this.imgHotel4 = imgHotel4;
}
public WebElement getTxtExUser() {
    return txtExUser;
}
public void setTxtExUser(WebElement txtExUser) {
    this.txtExUser = txtExUser;
}
public WebElement getTxtForPass() {
    return txtForPass;
}
public void setTxtForPass(WebElement txtForPass) {
    this.txtForPass = txtForPass;
}
public WebElement getTxtNewUser() {
    return txtNewUser;
}
public void setTxtNewUser(WebElement txtNewUser) {
    this.txtNewUser = txtNewUser;
}
public WebElement getTxtImpNote() {
    return txtImpNote;
}
public void setTxtImpNote(WebElement txtImpNote) {
    this.txtImpNote = txtImpNote;
}
public LoginPage(WebDriver ldriver) {
```

```

        this.driver=ldriver;
        PageFactory.initElements(driver, this);

    }
    public WebElement getimgAdactinLogo() {
        return imgAdactinLogo;
    }
    public void setimgAdactinLogo(WebElement imgAdactinLogo) {
        this.imgAdactinLogo = imgAdactinLogo;
    }
    public WebElement getTxtUserName() {
        return txtUserName;
    }
    public void setTxtUserName(WebElement txtUserName) {
        this.txtUserName=txtUserName;
    }
    public WebElement getTxtUserPassword() {
        return txtUserPassword;
    }
    public void setTxtUserPassword(WebElement txtUserPassword) {
        this.txtUserPassword=txtUserPassword;
    }
    public WebElement getBtnLoginButton() {
        return btnLoginButton;
    }
    public void setBtnLoginButton(WebElement btnLoginButton) {
        this.btnLoginButton = btnLoginButton;
    }

}

```

Class 2:

```

public class NextPage {
    WebDriver driver;
    @FindBy(xpath="//*[text()='Logout']")
    private WebElement txtLogout;

    public WebElement getTxtLogout() {
        return txtLogout;
    }

    public void setTxtLogout(WebElement txtLogout) {

```

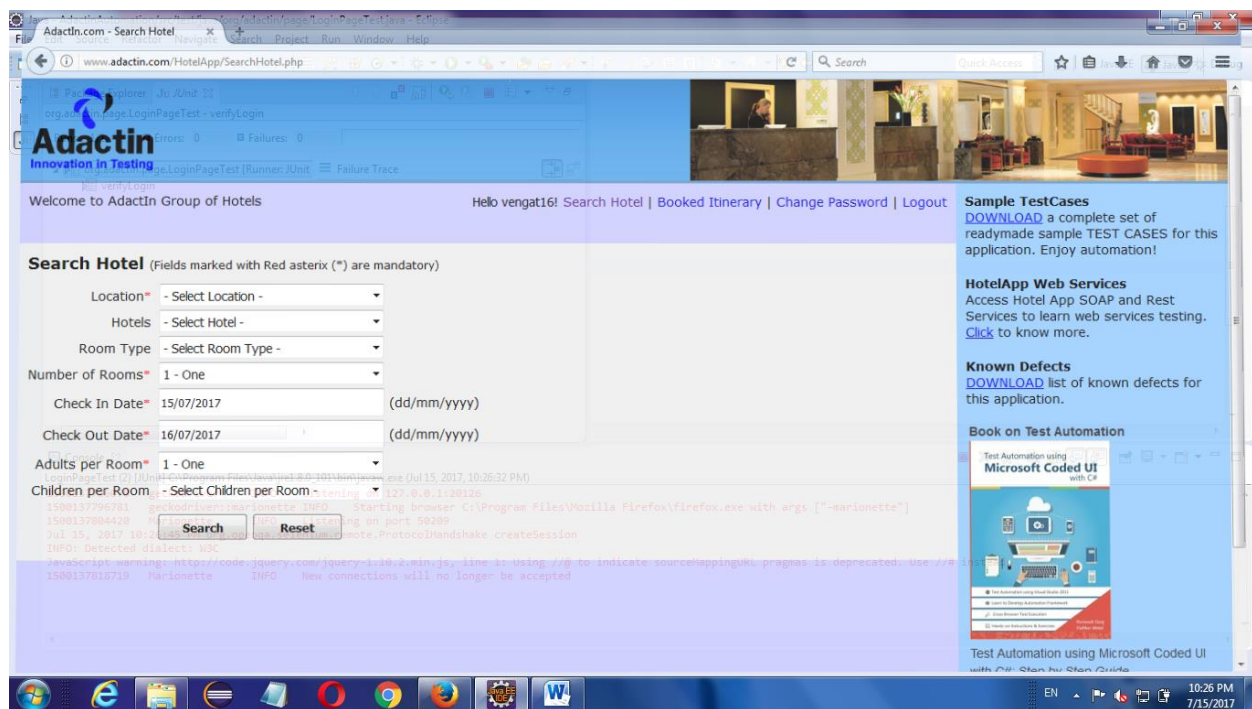
```

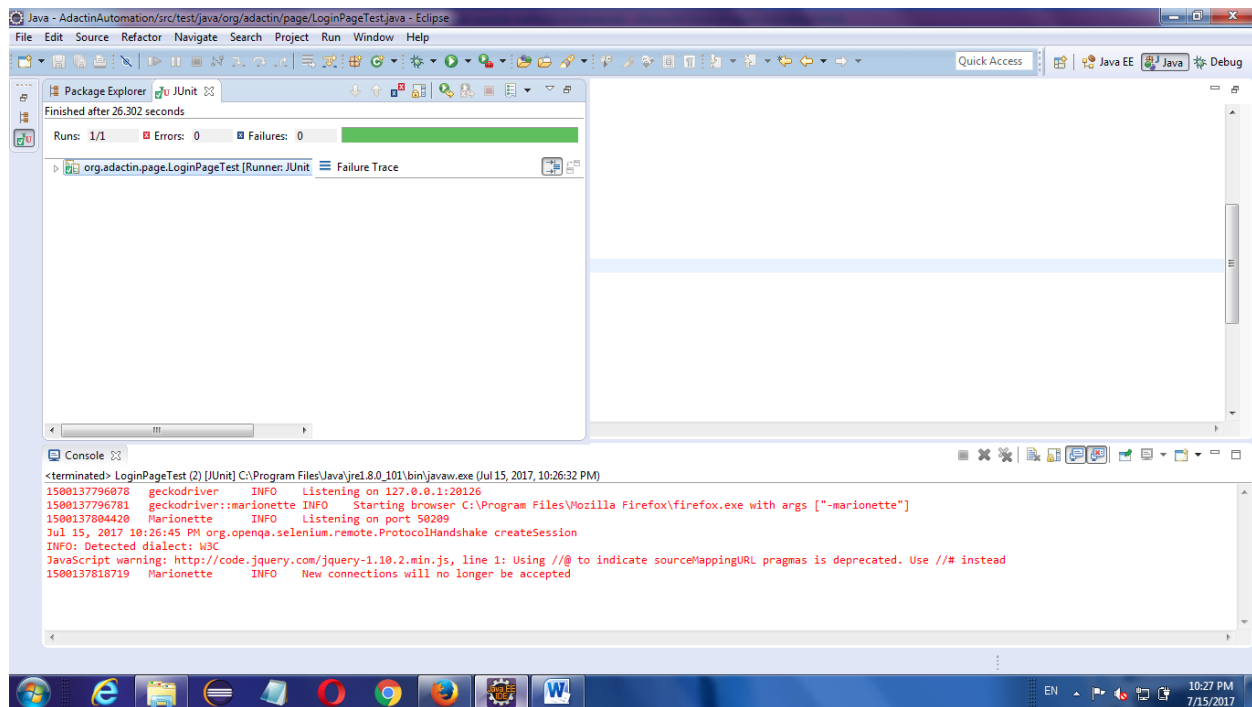
        this.txtLogout = txtLogout;
    }

    public NextPage(WebDriver ldriver) {
        this.driver=ldriver;
        PageFactory.initElements(driver, this);
    }
}

```

Output:





SCREENSHOT REUSABLE METHOD

- We can add screenshot in reusable methods

```
public void getScreenShot(String screenShotFileName) {
    File screenShotLocation = new File("./screenshot/" +
    screenShotFileName
    + ".png");
    TakesScreenshot screenshot = (TakesScreenshot) driver;
    File file = screenshot.getScreenshotAs(OutputType.FILE);
    try {
        FileUtils.copyFile(file, screenShotLocation);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

- We can use this screenshot method wherever we want in our test class

JSON METHOD

- JSON is **Java Script Object Notation**, an open standard format that uses human readable text to transmit data objects consisting of attribute-value pairs.
Ex, { "someKey": "someValue", "anotherKey": "value" } or

STEPS:

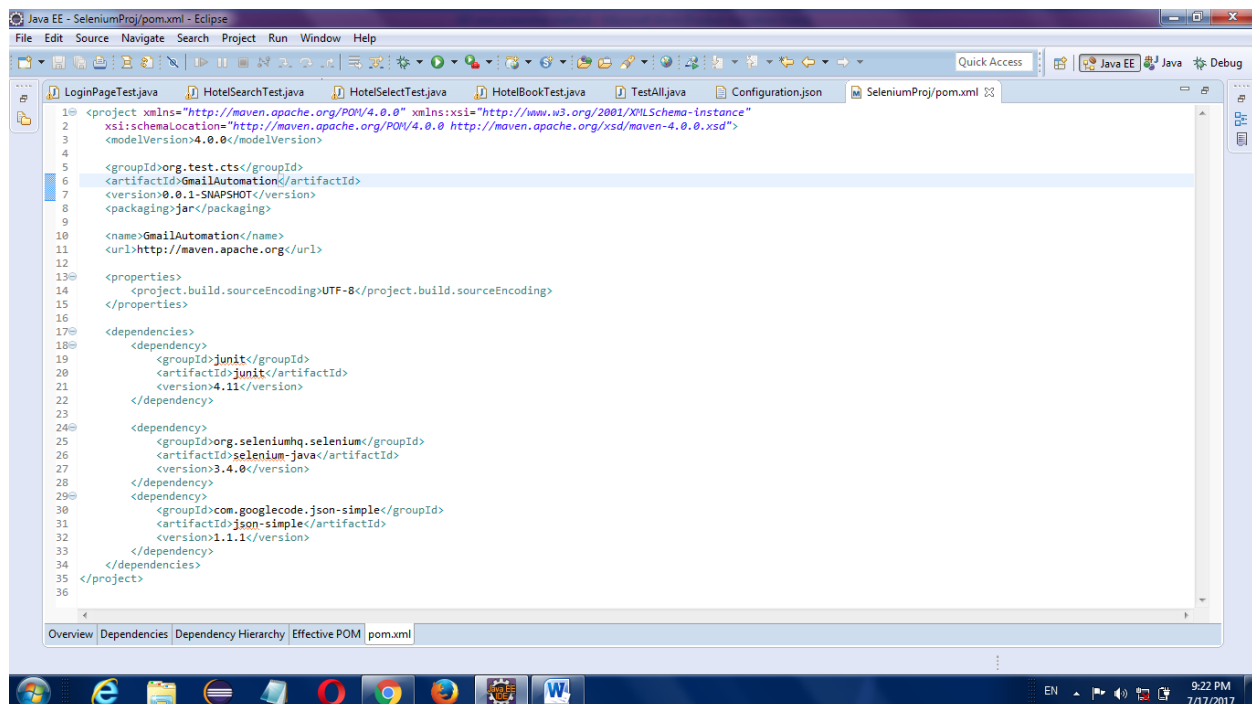
1. Right click your project → create one folder in the name of JSON.

2. In that folder, right click → create one file
3. In that file , we can create some attribute-value pairs whatever we want like below format

```
{ "url": "https://www.facebook.com/", "browser": "chrome" }
```

4. Add JSON jar in maven dependencies

- Here we just add JSON dependency code in the pom.xml
- Then click ctrl+S
- Then right click → Run as → maven install
- Then right click → maven → update project → select project → click ok



BROWSER COFIGURATION USING JSON CODE :

```
public class Base {
    static WebDriver driver;
    WebDriverWait wait;
    static File fl = new File("./JSON/Configuration.json");

    public static WebDriver getDriver() {
        JSONObject jsonObject = JSONReadFromFile();
        String browser = (String) jsonObject.get("browser");

        File f = new File("./driver");
        if (browser.equals("chrome")) {
            System.setProperty("webdriver.chrome.driver",
f.getAbsolutePath())
```



```

        + "/chromedriver.exe");
        driver = new ChromeDriver();

    } else if (browser.equals("firefox")) {
        System.setProperty("webdriver.gecko.driver",
f.getAbsolutePath()
        + "/geckodriver.exe");
        driver = new FirefoxDriver();

    } else if (browser.equals("ie")) {
        System.setProperty("webdriver.ie.driver",
f.getAbsolutePath()
        + "/IEDriverServer.exe");
        driver = new InternetExplorerDriver();

    }

    driver.manage().window().maximize();
    driver.get((String) jsonObject.get("url"));
    return driver;
}

public static JSONObject JSONReadFromFile() {
    JSONParser parser = new JSONParser();
    JSONObject jsonObject = null;
    try {

        Object obj = parser.parse(new
FileReader(fl.getAbsolutePath()));

        jsonObject = (JSONObject) obj;

    } catch (Exception e) {
        e.printStackTrace();
    }
    return jsonObject;
}

}

```

Configuration file:

```

{"url":"https://www.facebook.com/","browser":"chrome"}

```

Example program:

1. Facebook automation using JSON concept and screenshot method:

1. Base class:

```
public class Base {
    static WebDriver driver;
    WebDriverWait wait;
    static File fl = new File("./JSON/Configuration.json");

    public static WebDriver getDriver() {
        JSONObject jsonObject = JSONReadFromFile();
        String browser = (String) jsonObject.get("browser");

        File f = new File("./driver");
        if (browser.equals("chrome")) {
            System.setProperty("webdriver.chrome.driver", f.getAbsolutePath()
                + "/chromedriver.exe");
            driver = new ChromeDriver();

        } else if (browser.equals("firefox")) {
            System.setProperty("webdriver.gecko.driver", f.getAbsolutePath()
                + "/geckodriver.exe");
            driver = new FirefoxDriver();

        } else if (browser.equals("ie")) {
            System.setProperty("webdriver.ie.driver", f.getAbsolutePath()
                + "/IEDriverServer.exe");
            driver = new InternetExplorerDriver();

        }

        driver.manage().window().maximize();
        driver.get((String) jsonObject.get("url"));
        return driver;
    }

    public boolean elementToBeVisible(WebDriver driver, int time,
        WebElement element) {
        boolean flag = false;
        try {
            wait = new WebDriverWait(driver, time);
            wait.until(ExpectedConditions.visibilityOf(element));
            flag = true;
        } catch (Exception e) {
```

```
e.printStackTrace();
}
return flag;
}
```

```
public boolean alertIsPresent(WebDriver driver, int time) {
    boolean flag = false;
    try {
        wait = new WebDriverWait(driver, time);
        wait.until(ExpectedConditions.alertIsPresent());
        flag = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return flag;
}
```

```
public boolean elementToBeClickable(WebDriver driver, int time,
WebElement element) {
    boolean flag = false;
    try {
        wait = new WebDriverWait(driver, time);
        wait.until(ExpectedConditions.elementToBeClickable(element));
        flag = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return flag;
}
```

```
public boolean elementFound(WebDriver driver, int time, WebElement element)
{
    boolean res = false;
    driver.manage().timeouts().implicitlyWait(time, TimeUnit.SECONDS);
    try {

        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();

    }
    return res;

}
```

```
public boolean elementFound(WebElement element) {
```

```

    boolean res = false;
    try {
        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return res;
}

public void setText(WebElement element, String name) {
    if (name != null && elementFound(element)) {
        element.clear();
        element.sendKeys(name);
    }

}

public String getText(WebElement element) {
    String name = null;
    if (elementFound(element)) {
        name = element.getAttribute("value");
    }
    return name;
}

public void clickBtn(WebElement element) {
    if (elementFound(element)) {
        element.click();
    }
}

public static JSONObject JSONReadFromFile() {
    JSONParser parser = new JSONParser();
    JSONObject jsonObject = null;
    try {

        Object obj = parser.parse(new FileReader(fl.getAbsolutePath()));

        jsonObject = (JSONObject) obj;

    } catch (Exception e) {
        e.printStackTrace();
    }
    return jsonObject;
}

```

```

    }

    public void getScreenShot(String screenShotFileName) {
        File screenShotLocation = new File("./screenshot/" + screenShotFileName
        + ".png");
        TakesScreenshot screenshot = (TakesScreenshot) driver;
        File file = screenshot.getScreenshotAs(OutputType.FILE);
        try {
            FileUtils.copyFile(file, screenShotLocation);
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}

```

➤ Here we using JSON concept for getting browser and url

JUnit class:

```

public class LoginPageTest {
    static WebDriver driver;
    LoginPage loginPage;
    HomePage homePage;
    Base base = new Base();

    @BeforeClass
    public static void launchBrowser() {
        driver = Base.getDriver();
    }

    @Test
    public void verifyLoginDetails() {

        loginPage = new LoginPage(driver);
        homePage = new HomePage(driver);
        Assert.assertTrue(base.elementFound(driver, 10,
            loginPage.getImgFbLogo()));
        base.getScreenShot("facebookPage");

        base.setText(loginPage.getTxtUserName(),
            "ramesh@gmail.com");
        base.getScreenShot("username");
        Assert.assertEquals("ramesh@gmail.com",

```

```

        base.getText(loginPage.getTxtUserName());
        base.getScreenShot("password");
        loginPage.setTxtPassword("12345");
        base.setText(loginPage.getTxtPassword(), "12345");
        base.getScreenShot("login");
        Assert.assertEquals("12345",
base.getText(loginPage.getTxtPassword()));
        base.clickBtn(loginPage.getBtnLogin());
        base.getScreenShot("after login");

    }

    @AfterClass
    public static void closeBrowser() {
        driver.quit();
    }
}

```

- In this test class, we have using screenshot method for homepage , username, password and after login
- In this all screenshot will be stored in screenshot folder inside the project

Main class:

Class 1

```

public class LoginPage extends Base {
    WebDriver driver;

    @FindBy(id = "email")
    private WebElement txtUserName;

    @FindBy(id = "pass")
    private WebElement txtPassword;

    @FindBy(xpath = "//*[@text()='Log In']")
    private WebElement btnLogin;

    @FindBy(xpath = "//a[@title='Go to Facebook home']")
    private WebElement imgFbLogo;

    public WebElement getImgFbLogo() {
        return imgFbLogo;
    }
}

```

```

    public void setImgFbLogo(WebElement imgFbLogo) {
        this.imgFbLogo = imgFbLogo;
    }

    public void setTxtUserName(WebElement txtUserName) {
        this.txtUserName = txtUserName;
    }

    public void setTxtPassword(WebElement txtPassword) {
        this.txtPassword = txtPassword;
    }

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

    public WebElement getTxtUserName() {
        return txtUserName;
    }

    public WebElement getTxtPassword() {
        return txtPassword;
    }

    public void setTxtPassword(String txtPassword) {
        this.txtPassword.sendKeys(txtPassword);
    }

    public WebElement getBtnLogin() {
        return btnLogin;
    }

    public void setBtnLogin(WebElement btnLogin) {
        this.btnLogin = btnLogin;
    }
}

```

Class 2

```

public class HomePage {
    WebDriver driver;
    @FindBy(xpath = "//*[@text()='Home']")
    private WebElement imgHomePageLogo;
}

```

```

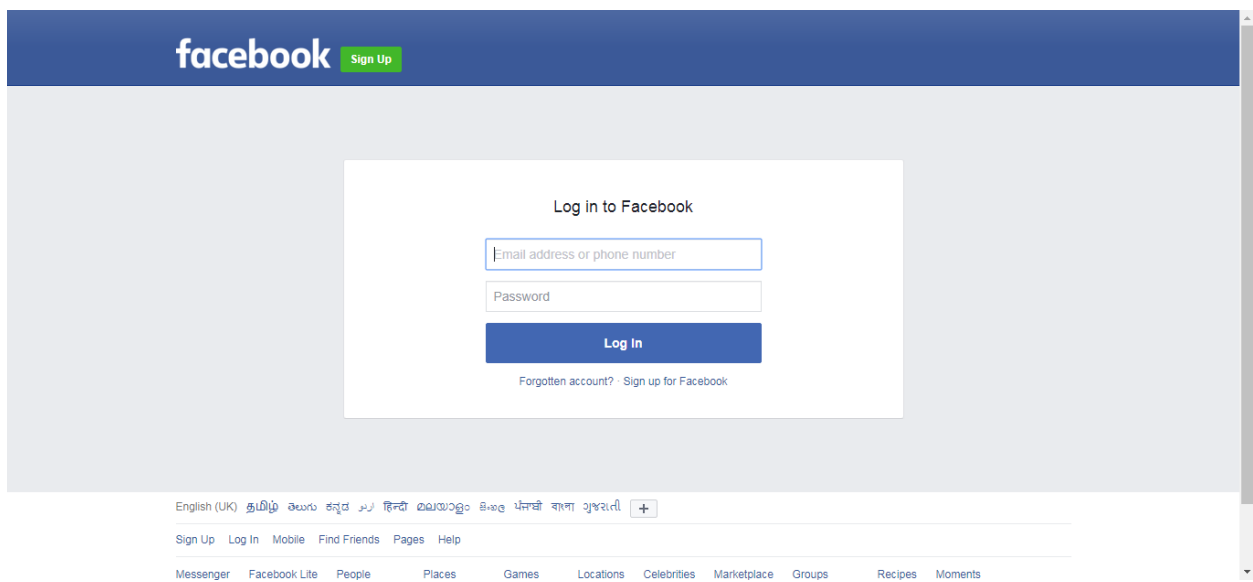
    public HomePage(WebDriver driver) {
        this.driver = driver;
        PageFactory.initElements(driver, this);
    }

    public WebElement getImgHomePageLogo() {
        return imgHomePageLogo;
    }

    public void setImgHomePageLogo(WebElement imgHomePageLogo) {
        this.imgHomePageLogo = imgHomePageLogo;
    }
}

```

Screenshots:



facebook

Email or Phone

Password

Log In

Forgotten account?

Facebook helps you connect and share with the people in your life.



Create an account

It's free and always will be.

First name

Surname

Mobile number or email address

New password

Birthday

16

Jul

1999

Why do I need to provide my date of birth?

☐ Female

☐ Male

By clicking Create an account, you agree to our [Terms](#) and confirm that you have read our [Data Policy](#), including our [Cookie Use Policy](#). You may receive SMS message notifications from Facebook and can opt out at any time.

Create an account

facebook

Email or Phone

ramesh@gmail.com

Password

.....

Log In

Forgotten account?

Facebook helps you connect and share with the people in your life.



Create an account

It's free and always will be.

First name



Surname

Mobile number or email address

New password

Birthday

16

Jul

1999

Why do I need to provide my date of birth?

☐ Female

☐ Male

By clicking Create an account, you agree to our [Terms](#) and confirm that you have read our [Data Policy](#), including our [Cookie Use Policy](#). You may receive SMS message notifications from Facebook and can opt out at any time.

Create an account

facebook

Email or Phone

ramesh@gmail.com

Forgot your account?

Password

Log In

Facebook helps you connect and share with the people in your life.

Create an account

It's free and always will be.

First name

Surname

Mobile number or email address

New password

16

Jul

1999

Why do I need to provide my date of birth?

Female

Male

By clicking Create an account, you agree to our Terms and confirm that you have read our Data Policy, including our Cookie Use Policy. You may receive SMS message notifications from Facebook and can opt out at any time.

Create an account

facebook

Email or Phone

ramesh@gmail.com

Forgot your account?

Password

Log In

Facebook helps you connect and share with the people in your life.

Create an account

It's free and always will be.

First name

Surname

Mobile number or email address

New password

16

Jul

1999

Why do I need to provide my date of birth?

Female

Male

By clicking Create an account, you agree to our Terms and confirm that you have read our Data Policy, including our Cookie Use Policy. You may receive SMS message notifications from Facebook and can opt out at any time.

Create an account

EXERCISE:

1. Demo site registration with file upload:

Base class:

```

public class Base {
    static WebDriver driver;
    WebDriverWait wait;
    static File fl = new File("./JSON/Configuration.json");

    public static WebDriver getDriver() {

```

```

JSONObject jsonObject = JSONReadFromFile();
String browser = (String) jsonObject.get("browser");

File f = new File("./driver");
if (browser.equals("chrome")) {
    System.setProperty("webdriver.chrome.driver", f.getAbsolutePath()
        + "/chromedriver.exe");
    driver = new ChromeDriver();

} else if (browser.equals("firefox")) {
    System.setProperty("webdriver.gecko.driver", f.getAbsolutePath()
        + "/geckodriver.exe");
    driver = new FirefoxDriver();

} else if (browser.equals("ie")) {
    System.setProperty("webdriver.ie.driver", f.getAbsolutePath()
        + "/IEDriverServer.exe");
    driver = new InternetExplorerDriver();
}

driver.manage().window().maximize();
driver.get((String) jsonObject.get("url"));
return driver;
}

public boolean elementToBeVisible(WebDriver driver, int time,
    WebElement element) {
    boolean flag = false;
    try {
        wait = new WebDriverWait(driver, time);
        wait.until(ExpectedConditions.visibilityOf(element));
        flag = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return flag;
}

public boolean elementFound(WebDriver driver, int time, WebElement element) {
    boolean res = false;
    driver.manage().timeouts().implicitlyWait(time, TimeUnit.SECONDS);
    try {
        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return res;
}

```

```

}

public boolean elementFound(WebElement element) {
    boolean b = false;
    try {
        b = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return b;
}

public void setText(WebElement element, String name) {
    if (name != null && elementFound(element)) {
        element.clear();
        element.sendKeys(name);
    }
}

public String getText(WebElement element) {
    String name = null;
    if (elementFound(element)) {
        name = element.getAttribute("value");
    }
    return name;
}

public void clickBtn(WebElement element) {
    if (elementFound(element)) {
        element.click();
    }
}

public void dropDownSelect(WebElement element,String name) {
    Select s=new Select(element) ;
    try{
        s.selectByValue(name);}
    catch(Exception e){
        e.printStackTrace();
    }
}

public void dropDownSelectVText(WebElement element,String name) {
    Select s=new Select(element) ;
    s.selectByVisibleText(name);
}

```

```

public static JSONObject JSONReadFromFile() {
    JSONParser parser = new JSONParser();
    JSONObject jsonObject = null;
    try {

        Object obj = parser.parse(new FileReader(fl.getAbsolutePath()));

        jsonObject = (JSONObject) obj;

    } catch (Exception e) {
        e.printStackTrace();
    }
    return jsonObject;
}

public static void uploadFiles(File path) {
    try {
        Robot robot = new Robot();
        robot.setAutoDelay(3000);
        StringSelection selection = new StringSelection(
            path.getAbsolutePath());
        Toolkit.getDefaultToolkit().getSystemClipboard()
            .setContents(selection, null);

        // press control+v
        robot.keyPress(KeyEvent.VK_CONTROL);
        robot.keyPress(KeyEvent.VK_V);
        robot.setAutoDelay(3000);
        // release control+v
        robot.keyRelease(KeyEvent.VK_CONTROL);
        robot.keyRelease(KeyEvent.VK_V);
        // press enter
        robot.setAutoDelay(3000);
        robot.keyPress(KeyEvent.VK_ENTER);
        robot.keyRelease(KeyEvent.VK_ENTER);

    } catch (AWTException e) {
        e.printStackTrace();
    }
}
}

```

JUnit class:

```

public class RegisterTest extends Base {
    static WebDriver driver;

```

RegisterPage page;

@BeforeClass

public static void launchBrowser() {
 driver = Base.getDriver();

}

@Test

public void registerDetails() {
 page = **new** RegisterPage();
 setText(page.getTxtFirstName(), "venkat");
 setText(page.getTxtLastName(), "ram");
 setText(page.getTxtAddress(), "xyzcolony,xyz street,xyz");
 setText(page.getTxtEmailAdress(), "venkat6@gmail.com");
 setText(page.getTxtPhoneNumber(), "9996599965");
 clickBtn(page.getBtnGender());
 clickBtn(page.getBtnCricket());
 clickBtn(page.getBtnHockey());
 clickBtn(page.getBtnMovies());
 clickBtn(page.getTxlanguage());
 dropDownSelectVText (page.getDrpDwnSkills(), "Analytics");
 dropDownSelectVText (page.getDrpDwnCountries(), "Afghanistan");
 clickBtn(page.getDrpdwnSelectCountry());
 clickBtn(page.getDrpdwnSelectCountryDen());
 dropDownSelectVText (page.getDrpdwnYear(), "1993");
 dropDownSelectVText (page.getDrpdwnMonth(), "11");
 dropDownSelectVText (page.getDrpdwnDay(), "16");
 setText(page.getTxtPassword(), "Karthick123");
 setText(page.getTxtConfirmPassword(), "Karthick123");
 clickBtn(page.getBtnBrowse());
 uploadFiles(**new** File("./Files/Core java material.pdf"));
 Assert.assertTrue(elementFound(page.getImgbrowse()));
 clickBtn(page.getBtnSubmit());

}

@AfterClass

public static void closeBrowser() {
 driver.quit();

}

}

Main class:

```
public class RegisterPage extends Base {  
    @FindBy(xpath="//*[@ ng-model='FirstName']")  
    private WebElement txtFirstName;  
    @FindBy(xpath="//*[@ ng-model='LastName']")  
    private WebElement txtLastName;  
    @FindBy(xpath="//*[@ ng-model='Adress']")  
    private WebElement txtAddress;  
    @FindBy(xpath="//*[@ ng-model='EmailAddress']")  
    private WebElement txtEmailAddress;  
    @FindBy(xpath="//*[@ ng-model='Phone']")  
    private WebElement txtPhoneNumber;  
    @FindBy(xpath="//*[@ value='Male']")  
    private WebElement btnGender;  
    @FindBy(id = "checkbox1")  
    private WebElement btnCricket;  
    @FindBy(id = "checkbox2")  
    private WebElement btnMovies;  
    @FindBy(id = "checkbox3")  
    private WebElement btnHockey;  
    @FindBy(id = "msdd")  
    private WebElement txtlanguage;  
    @FindBy(id="Skills")  
    private WebElement drpdwnskills;  
    @FindBy(id="countries")  
    private WebElement drpdwnCountries;  
    @FindBy(xpath="//*[@role='combobox']")  
    private WebElement drpdwnSelectCountry;  
    @FindBy(xpath="//*[@id='select2-country-results']/li[4]")  
    private WebElement drpdwnSelectCountryDen;  
    @FindBy(id="imagesrc")  
    private WebElement imgbrowse;  
    @FindBy(id="yearbox")  
    private WebElement drpdwnYear;  
    @FindBy(xpath="//*[@placeholder='Month']")  
    private WebElement drpdwnMonth;  
    @FindBy(id="daybox")  
    private WebElement drpdwnDay;  
    @FindBy(id="firstpassword")  
    private WebElement txtPassword;  
    @FindBy(id="secondpassword")  
    private WebElement txtConfirmPassword ;  
    @FindBy(id="submitbtn")  
    private WebElement btnSubmit ;  
}
```

```
@FindBy(xpath="//*[@id='section']/div/h2")
private WebElement imgRegister ;
@FindBy(id = "imagesrc")
private WebElement btnBrowse;

public WebElement getImgbrowse() {
    return imgbrowse;
}

public WebElement getDrpdwnSelectCountryDen() {
    return drpdwnSelectCountryDen;
}

public WebElement getTxtFirstName() {
    return txtFirstName;
}

public WebElement getTxtLastName() {
    return txtLastName;
}

public WebElement getTxtAddress() {
    return txtAddress;
}

public WebElement getTxtEmailAddress() {
    return txtEmailAddress;
}

public WebElement getTxtPhoneNumber() {
    return txtPhoneNumber;
}

public WebElement getBtnGender() {
    return btnGender;
}

public WebElement getBtnCricket() {
    return btnCricket;
}

public WebElement getBtnMovies() {
    return btnMovies;
}

public WebElement getBtnHockey() {
```



```
        return btnHockey;
    }

    public WebElement getTxtlanguage() {
        return txtlanguage;
    }

    public WebElement getDrpDwnSkills() {
        return drpdwnskills;
    }

    public WebElement getDrpDwnCountries() {
        return drpdwnCountries;
    }

    public WebElement getDrpdwnSelectCountry() {
        return drpdwnSelectCountry;
    }

    public WebElement getDrpdwnYear() {
        return drpdwnYear;
    }

    public WebElement getDrpdwnMonth() {
        return drpdwnMonth;
    }

    public WebElement getDrpdwnDay() {
        return drpdwnDay;
    }

    public WebElement getTxtPassword() {
        return txtPassword;
    }

    public WebElement getTxtConfirmPassword() {
        return txtConfirmPassword;
    }

    public WebElement getBtnSubmit() {
        return btnSubmit;
    }

    public WebElement getImgRegister() {
        return imgRegister;
    }
}
```

```
public WebElement getBtnBrowse() {  
    return btnBrowse;  
}  
  
public RegisterPage() {  
    PageFactory.initElements(Base.driver, this);  
}  
}
```

Output:

