**Violating the clean code principle**

1. **HomePage**
2. General base comments in HomePage.

Solution : Remove those general base comments which is declared under packages.

1. Avoid declaring drivers as final.

Solution : private Webdriver driver;

1. Declared GITHUB\_BASE\_URL in HomePage.

Solution : Remove GITHUB\_BASE\_URL because GITHUB\_BASE\_URL is already opened.

1. Declared open() method in HomePage it duplicates the code.

Solution : Remove open() method from HomePage.

1. Invalid return type for hasUserLoggedIn() method.

Solution : boolean is the return type for hasUserLoggedIn() method and also used terinary operator to simplify the code.

public boolean hasUserLoggedIn(){

return linkLoggedInUser.getText()!=””?true:false;

}

1. HomePage is a page so we place it in a pages package not in model package.
2. **User**
3. General base comments in User.

Solution : Remove those general base comments which is declared under packages.

1. The third if condition which is declared in equals() method is tightly coupled.

Solution : Extract those condition into separate method called isInvalidUsername(), it makes code into loosely coupled.

1. The return statement to verify password is extracted into isValidPassword(), it makes code into loosely coupled.

Solution : if(isValidPassword()){

return false;

}

1. Passing undeclared literals or objects as parameters into method call like SimpeleDateFormat violates the cleancode principle.

Solution : DATE\_PATTERN is a constant so we assign it to a DATE\_PATTERN constant and it would be pass to a SimpleDateFormat.

DateFormat dateformat = new SimpleDateFormat(DATE\_PATTERN);.

1. **SignInPage**
2. General base comments in SignInPage.

Solution : Remove those general base comments in SignInPage which are declared under packages.

1. Declaring variable suffixes with numbers like input1, input2 and also avoid digits between variable name like butt0n.

Solution : Avoid declaring variable suffixes with numbers.

private WebElement usernameFeild;

private WebElement passwordFeild;

private WebElement signInButton;

1. Don’t use direct click(), sendKeys() methods.

Solution : Create method by passing driver and using visibilityOf() as ExpectedCondition.

1. Declare enterDetails() method as public to call sendUsername(), sendPassword(), clickOnSignIn() methods.
2. Simplify enterDetails() method by calling clickOnFields() if the signInButton.isDisplayed() condition is true otherwise throw an Exception.
3. Simplify isReadyToSignIn() method to check if signInButton is displayed or not using terinary operator.
4. **StartPage**
5. General base comments in StartPage.

Solution : Remove those general base comments in StartPage which are declared under packages.

1. When declaring fields avoid using numbers in it.

Example : private WebElement butt0n;

Solution : private WebElement button;

1. Declare “WebDriver” using protected access specifier

to avoid different package non-subclass .

Solution : protected WebDriver driver;

1. Call clickOnFields() method instead of calling click() method in invokeSignIn() method.
2. **Test1**
3. General base comments inTest1 class.

Solution : Remove those general base comments in Test1 class which are declare under packages.

1. When declaring class names numbers can’t be added to the suffix of the class name.

Solution : public class Test{

}

1. Avoiding all statements declared in test() method.

Solution : Create setup() method under @Beforetest annotation by setting driver by applying waits.

1. Declare the “WebElements” like driver and startPage outside the tests by using respective access specifiers.

Solution : protected WebDriver driver;

private StartPage startPage;

1. We initially set up the driver in the method under which we declare @BeforeTest annotation.

Solution : @BeforeTest

public void setup(){

driver =new FirefoxDriver();

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

startPage = new StartPage(driver);

startPage.open();

}

1. Declare constants by using static and final keywords and also declare them as public because we use those in another method.

Solution :

public static final String USERNAME = ”testautomationuser”;

public static final String PASSWORD = ”Time4Death!”;

1. Close the driver close() method before perform quit().

Solution : @AfterTest

public void tearUp(){

driver.close();

}

1. Closure statements should be written in a method under which we declare @After\* annotation.

Solution : @AfterSuite

public void closure(){

driver.quit();

}

1. Create class called ConfigurationTest and place the methods which we declare under @BeforeTest, @AfterTest and @AfterSuite into it.
2. The above ConfigurationTest class should be extended into Test class.

Solution : public class ConfigurationTest extends Test{ }