Code Review Checklist

- Assertion is not to be done in page object file.
- Assertion to be done in Test Case level.

Example:

```
@Test(dataProvider = "dataexcel")
public void TestCase_Search(String searchword, String expectedTitle) throws Exception {

    boolean clickLink = iAutomateDemoPage.clickTestNGLinkOnGoogle();
    IsTrue(clickLink, passMessage: "SUCCESSFULLY clicked on " + searchword + " link/url",
            failMessage: "Could NOT click on " + searchword + " link/url");

    String title = iAutomateDemoPage.TitleOfPage();
    Equals(title, expectedTitle, passMessage: "SUCCESSFULLY asserted title on the page as " + title,
            failMessage: "Could NOT assert title on the page");
}
```

Note: IsTrue, Equals etc. assertion wrappers are inside testbase class which you extend in your testcase.

- Web-Element initializing done with variables in Title Casing not Camel casing. since Camel casing is used to represent variables

Example:

```
@FindBy(xpath = AddToCartConstants.CARTITEM)
public WebElement CartItem;
```

- Following is the naming convention you can utilize as per element type

Category	UI/Control type	Prefix	Example
Basic	Button	btn	btnExit
Basic	Check box	chk	chkReadOnly
Basic	Combo box	cbo	cboEnglish
Basic	Common dialog	dlg	dlgFileOpen
Basic	Date picker	dtp	dtpPublished
Basic	Dropdown List / Select tag	ddl	ddlCountry
Basic	Form	frm	frmEntry
Basic	Frame	fra	fraLanguage
Basic	Image	img	imgIcon
Basic	Label	lbl	lblHelpMessage
Basic	Links/Anchor Tags	lnk	lnkForgotPwd
Basic	List box	lst	lstPolicyCodes
Basic	Menu	mnu	mnuFileOpen

Basic	Radio button / group	rdo	rdoGender
Basic	RichTextBox	rtf	rtfReport
Basic	Table	tbl	tblCustomer
Basic	TabStrip	tab	tabOptions
Basic	Text Area	txa	txaDescription
Basic	Text box	txt	txtLastName
Complex	Chevron	chv	chvProtocol
Complex	Data grid	dgd	dgdTitles
Complex	Data list	dbl	dblPublisher
Complex	Directory list box	dir	dirSource
Complex	Drive list box	drv	drvTarget
Complex	File list box	fil	filSource
Complex	Panel/Fieldset	pnl	pnlGroup
Complex	ProgressBar	prg	prgLoadFile
Complex	Slider	sld	sldScale
Complex	Spinner	spn	spnPages
Complex	StatusBar	sta	staDateTime
Complex	Timer	tmr	tmrAlarm
Complex	Toolbar	tlb	tlbActions
Complex	TreeView	tre	treOrganization

- Appropriate method naming should be done to have easy readability of code. Use camel casing. Avoid underscore in method name.
- The methods written should be clear enough to understand what each statement is expected to execute.
- Put code in try catch blocks and avoid multiple catch block.
- Method should have return type.
- Log messages which are clearly understandable - **logInfo** for informational and **logError** for error message

PFB Screenshot for above points.

Example:

```
public boolean sendKeys(WebElement element, String txt) {
    try {
        if(!isElementDisplayedByWebElement(element))
            return false;

        element.sendKeys(txt);
        logInfo("Entered text " + txt + " successfully");
        return true;
    } catch (Exception ex) {
        logError("Could not enter text-" + txt + " " + ex.getMessage());
        return false;
    }
}
```

- Every Function should have Java Doc.

Example:

```
/**
 * This method is used to set with provided input text
 * @author
 * @param element [WebElement] to send data to
 * @param txt [String] : Data to be send
 * @return false if the element is not displayed
 * @return true if data entered successfully else false
 */
public boolean sendKeys(WebElement element, String txt) {
```

- Page objects should be unique as well as robust.
- For data search functionalities, minimum texts are used.
- Avoid hard coding values.
- No code duplication to be done to avoid redundancy.
- Avoid unnecessary imports statement and unused variables.
- Avoid static method and variable.
- Remove console print Statements (SOPs)