**Coding Standards**

Version: 1.0

**1. Naming Conventions**

**Class Names: Pascal Casing**

Ex. Employe Operations Class Name: **Operations**

**Method Name: Pascal Casing**

Ex. Insert Employee Method Name: **InsertEmployee**

**Property Name: Pascal Casing**

Ex. Department Name Property: **DepartmentName**

**Variable Names:** Camel casing prefixed with first three letters of its type in small.

**Class Level Variable:** prefix with **\_ (underscore)** before type letters in small

Ex. User Name: \_**strUserName – Type: string**

Ex. User Id: \_**intUserId – Type: int**

Ex. Is User Active: \_**blnIsUserActive – Type: Boolean**

Ex. User Class Object: \_**objUser – Type: User**

**Method Level Variable:**

Ex. User Name: **strUserName – Type: string**

Ex. User Id: **intUserId – Type: int**

Ex. Is User Active: **blnIsUserActive – Type: Boolean**

Ex. User Class Object: **objUser – Type: User**

**Folder Name inside a project:** camel Casing.

Ex. User Admin: **UserAdmin**

**UI Page Name:** camel Casing.

Ex. User Admin Page: **userAdmin.aspx**

**Class Name:** camel Casing.

Ex. User Admin Page: **userAdmin.cs**

Database Table Name: camel Casing prefixed with first “tbl” and three letters of it type

Ex. User Table: **tblMstuser**

Database Field Name: all Small letters

Ex. User Id: **userid**

Database Stored Procedure Name: camel Casing prefix with “usp\_”

Ex. User Id: **userid**

UI Control Names: Camel casing prefixed by first three letters of the control type in small.

Ex. User Name Text box: **txtUserName – Type: Text**

Ex. Location Drop down List: **ddlLocation – Type: Drop Down List**

Ex. Is Active Check Box: **chkIsUserActive – Type: Check Box**

Ex. Authority Radio Button: **rdoAuthority – Type: Radio button**

**Using Name Spaces:**

All the classes need to be grouped under name spaces. As described below.

Layer Name: Business

Module Name: User

Class Main Function: Manager

**Namespace for the above: SIMS.Business.Services.User.Manager**

**2. Code commenting**

**Class Commenting:**

/// <Summary>

/// <Description>Class description</Description>

/// <ChangeHistory>

/// <ChangeDate> Developer name</ChangeDate>

/// <ChangedBy> Developer name</ChangedBy>

/// <ChangeDescription> Chnages done</ChangeDescription>

/// </ChangeHistory>

/// </Summary>

**Method Commenting:**

/// <Summary>

/// <Description>What is the main function of the method</Description>

/// <MethodName>Its Name</MethodName>

/// <RetunType>what it returns</RetunType>

/// <InputParams>

/// <param1></param1>

/// <param2></param2>

/// </InputParams>

/// </Summary>

**3. Indentation and Tabs**

1. Use TAB for indentation. Do not use SPACES. Define the Tab size as 4.
2. Comments should be in the same level as the code (use the same level of indentation).
3. Curly braces ( {} ) should be in the same level as the code outside the braces.
4. Use one blank line to separate logical groups of code.
5. There should be one and only one single blank line between each method inside the class.
6. The curly braces should be on a separate line and not in the same line as if, for etc.
7. Use #region to group related pieces of code together. If you use proper grouping using #region, the page should like this when all definitions are collapsed.

**4. Programming Best Practices.**

For Best practices please refer Capgemini .Net coding practices.