

<%@ Control Language="C#" AutoEventWireup="true" CodeBehind="ucCVPassRate\_Approve.ascx.cs"

Inherits="DataMaps.ucCVPassRate\_Approve" %>

<%@ Register Assembly="System.Web" Namespace="System.Web.UI" TagPrefix="asp" %>

<%@ Register Assembly="DataMapsClassLibrary" Namespace="DataMaps" TagPrefix="cc1" %>

<%@ Register Src="~/usercontrols/ucMetricInsert.ascx" TagName="ucMetricInsert" TagPrefix="uc1" %>

<script src="../scripts/jquery-1.6.2.min.js" type="text/javascript"></script>

<script src="../scripts/jquery-ui-1.8.16.custom.min.js" type="text/javascript"></script>

<script src="../scripts/scripts.js" type="text/javascript"></script>

<link href="../css/jquery-ui-1.8.17.custom.css" rel="stylesheet" type="text/css" />

<script type="text/javascript">

// $(document).ready(function () {

// if ($.cookie('colWidth') != null) {

// var columns = $.cookie('colWidth').split(',');

// var i = 0;

// $('.GridView th').each(function () {

// $(this).width(columns[i++]);

// });

// }

// $(".GridView").colResizable({

// liveDrag: true,

// gripInnerHtml: "<div class='grip'></div>",

// draggingClass: "dragging",

// onResize: onSampleResized

// });

// });

// var onSampleResized = function (e) {

// var columns = $(e.currentTarget).find("th");

// var msg = "";

// columns.each(function () { msg += $(this).width() + ","; })

// $.cookie("colWidth", msg);

// };

function pageLoad() {

$("#Message1").val('');

$("#Message2").val('');

}

function IsDateGreater(DateValue1, DateValue2) {

var DaysDiff;

Date1 = new Date(DateValue1);

Date2 = new Date(DateValue2);

DaysDiff = Math.floor((Date1.getTime() - Date2.getTime()) / (1000 \* 60 \* 60 \* 24));

if (DaysDiff > 0)

return false;

else

return true;

}

</script>

<asp:ScriptManager ID="ScriptManager1"

runat="server"

onasyncpostbackerror="ScriptManager1\_AsyncPostBackError">

</asp:ScriptManager>

<script type="text/javascript" language="javascript">

var divElem = 'AlertDiv';

var messageElem = 'AlertMessage';

var bodyTag = 'bodytag';

Sys.WebForms.PageRequestManager.getInstance().add\_endRequest(EndRequestHandler);

Sys.WebForms.PageRequestManager.getInstance().add\_beginRequest(BeginRequestHandler);

function ToggleAlertDiv(visString) {

if (visString == 'hidden') {

$get(bodyTag).style.backgroundColor = 'white';

}

else {

$get(bodyTag).style.backgroundColor = 'gray';

}

var adiv = $get(divElem);

adiv.style.visibility = visString;

}

function ClearErrorState() {

$get(messageElem).innerHTML = '';

ToggleAlertDiv('hidden');

}

function $$(id, context) {

var el = $("#" + id, context);

if (el.length < 1)

el = $("[id$=\_" + id + "]", context);

return el;

}

function BeginRequestHandler(sender, args) {

$("#Message1").val('');

$("#Message2").val('');

}

function EndRequestHandler(sender, args) {

$("#Message1").val('');

$("#Message2").val('');

if (args.get\_error() != undefined) {

var errorMessage;

if (args.get\_response().get\_statusCode() == '200') {

//errorMessage = args.get\_error().message;

errorMessage = $("#Message1").val();

}

else {

// Error occurred somewhere other than the server page.

errorMessage = 'An unspecified error occurred. ';

}

args.set\_errorHandled(true);

//ToggleAlertDiv('visible');

//$get(messageElem).innerHTML = errorMessage;

//$get(messageElem).innerHTML = $("#Message1").val();

}

else {

//ToggleAlertDiv('visible');

//$get(messageElem).innerHTML = 'No errors';

//alert($("[id$=\_Message1]").attr("id"));

//$get(messageElem).innerHTML = ($("[id$=\_Message1]").attr("id")).value;

//$get(messageElem).innerHTML = ($("[id$=\_Message1]").attr("id")).value;

//$get(messageElem).innerHTML = $("#Message1").val();

}

}

</script>

<asp:Panel ID="Panel0" runat="server">

<table style="width: 100%">

<tr id="Region\_TR" runat="server">

<td id="largo" runat="server" style="height: 35px; text-align: right; width: 11%">

<asp:Label ID="Label1" runat="server" CssClass="label" Text="Region" Font-Size="Small" />

</td>

<td style="width: 20%">

<asp:DropDownList ID="ddlRegion" runat="server" OnSelectedIndexChanged="ddlRegion\_SelectedIndexChanged"

AutoPostBack="True" Font-Size="Small" />

</td>

<td style="text-align: right; width: 10%;">

<asp:Label ID="Label2" runat="server" CssClass="label" Text="Site" Font-Size="Small" />

</td>

<td style="width: 18%">

<asp:DropDownList ID="ddlSite" runat="server" AutoPostBack="True" OnSelectedIndexChanged="ddlSite\_SelectedIndexChanged"

Font-Size="Small" />

</td>

<td style="text-align: right; width: 10%;">

&nbsp;</td>

</tr>

<tr id="Client\_TR" runat="server">

<td align="right" style="text-align: right; vertical-align: middle">

<asp:Label ID="Label4" runat="server" CssClass="label" Text="Client" Font-Size="Small" />

</td>

<td>

<asp:DropDownList ID="ddlClient" runat="server" OnSelectedIndexChanged="ddlClient\_SelectedIndexChanged"

AutoPostBack="True" Font-Size="Small" />

</td>

<td align="right">

<asp:Label ID="Label5" runat="server" CssClass="label" Text="Program" Font-Size="Small" />

</td>

<td>

<asp:DropDownList ID="ddlProgram" runat="server" OnSelectedIndexChanged="ddlProgram\_SelectedIndexChanged"

AutoPostBack="True" Font-Size="Small" />

</td>

<td style="width: 5%" align="right">

<asp:Label ID="Label7" runat="server" CssClass="label" Font-Size="Small"

Text="Metric Category" Visible="False" />

</td>

<td>

<asp:DropDownList ID="ddlCategory" runat="server" AutoPostBack="True"

Font-Size="Small" OnSelectedIndexChanged="ddlCategory\_SelectedIndexChanged"

Visible="False" />

</td>

</tr>

<tr id="DeptoLOB" runat="server">

<td align="right" class="style1" style="text-align: right; vertical-align: middle">

<asp:Label ID="Label8" runat="server" CssClass="label" Font-Size="Small"

Text="Metric Name" />

</td>

<td class="style1">

<asp:DropDownList ID="ddlMetric" runat="server" AutoPostBack="True"

Font-Size="Small" OnSelectedIndexChanged="ddlMetric\_SelectedIndexChanged"

Style="vertical-align: middle">

</asp:DropDownList>

</td>

<td align="right" class="style1">

<asp:Label ID="Label6" runat="server" CssClass="label" Font-Size="Small"

Text=" LOB" />

</td>

<td class="style1">

<asp:DropDownList ID="ddlLOB" runat="server" AutoPostBack="True"

Font-Size="Small" OnSelectedIndexChanged="ddlLOB\_SelectedIndexChanged" />

</td>

<td align="right" class="style1">

<asp:Label ID="Label11" runat="server" CssClass="label" Font-Size="Small"

Text="Symbol" Visible="False" />

</td>

<td class="style1">

<asp:DropDownList ID="ddlSymbol" runat="server" AutoPostBack="True" Style="vertical-align: middle"

Font-Size="Small" Visible="False">

<asp:ListItem Value="cur"></asp:ListItem>

<asp:ListItem Value="%"></asp:ListItem>

<asp:ListItem Value="min"></asp:ListItem>

<asp:ListItem Value="sec"></asp:ListItem>

<asp:ListItem Value="#"></asp:ListItem>

</asp:DropDownList>

</td>

<td class="style1">

<asp:ImageButton ID="ImageButton1" runat="server" Height="25px" ImageUrl="~/images/excel\_icon.png"

OnClick="ImageButton1\_Click" Visible="False" Width="29px" OnPreRender="ImageButton1\_PreRender"

ToolTip="Export Data into Excel" />

</td>

</tr>

</table>

<hr />

</asp:Panel>

<asp:Panel ID="Panel1" runat="server">

<asp:Table ID="Table1" runat="server">

<asp:TableRow ID="TableRow2" runat="server">

<asp:TableCell ID="TableCell7" runat="server">

</asp:TableCell>

<asp:TableCell ID="TableCell5" runat="server">

</asp:TableCell>

<asp:TableCell ID="TableCell9" runat="server">

</asp:TableCell>

<asp:TableCell ID="TableCell6" runat="server">

</asp:TableCell>

<asp:TableCell ID="TableCell8" Style="text-align: right;" Width="600px" runat="server">

</asp:TableCell>

</asp:TableRow>

<asp:TableRow ID="TableRow1" runat="server">

<asp:TableCell ID="TableCell10" runat="server">

</asp:TableCell>

<asp:TableCell ID="TableCell1" runat="server">

<asp:DropDownList ID="ddlMonth" runat="server" Font-Size="Small" Visible="false">

<asp:ListItem Value="Jan"></asp:ListItem>

<asp:ListItem Value="Feb"></asp:ListItem>

<asp:ListItem Value="Mar"></asp:ListItem>

<asp:ListItem Value="Apr"></asp:ListItem>

<asp:ListItem Value="May"></asp:ListItem>

<asp:ListItem Value="Jun"></asp:ListItem>

<asp:ListItem Value="Jul"></asp:ListItem>

<asp:ListItem Value="Aug"></asp:ListItem>

<asp:ListItem Value="Sep"></asp:ListItem>

<asp:ListItem Value="Oct"></asp:ListItem>

<asp:ListItem Value="Nov"></asp:ListItem>

<asp:ListItem Value="Dec"></asp:ListItem>

</asp:DropDownList>

</asp:TableCell>

<asp:TableCell ID="TableCell3" HorizontalAlign="right" runat="server">

</asp:TableCell>

</asp:TableRow>

<%--<asp:TableRow ><asp:TableCell ID="TableCell2" runat="server" ColumnSpan="7">

<asp:Label ID="Message1" runat="server" CssClass="label" ForeColor="Green"></asp:Label>

</asp:TableCell></asp:TableRow>--%>

</asp:Table>

</asp:Panel>

<asp:UpdatePanel ID="UpdatePanel2" runat="server">

<ContentTemplate>

<asp:UpdateProgress ID="UpdateProgress1" runat="server" AssociatedUpdatePanelID="UpdatePanel2">

<ProgressTemplate>

<div style="margin-top: 20px; color: Green">

Processing, please wait...

</div>

</ProgressTemplate>

</asp:UpdateProgress>

<asp:Label ID="Message1" runat="server" Font-Size="Small" ForeColor="Red" Visible="true"></asp:Label>

<asp:Label ID="Message2" runat="server" Font-Size="Small" ForeColor="Red" Visible="true"></asp:Label>

<asp:GridView ID="GVMetrics" runat="server" AllowPaging="True" AutoGenerateColumns="False"

EmptyDataText="No records found." AllowSorting="True" Width="100%"

CssClass="GridView" OnPageIndexChanging="GVMetrics\_PageIndexChanging" DataKeyNames="ChangeRequestID"

OnSorting="GVMetrics\_Sorting" OnRowCommand="GVMetrics\_RowCommand" OnRowDataBound="GVMetrics\_RowDataBound"

OnRowCreated="GVMetrics\_RowCreated"

onselectedindexchanged="GVMetrics\_SelectedIndexChanged">

<Columns>

<asp:BoundField DataField="Region\_Name" HeaderText="Region"

SortExpression="Region\_Name" ShowHeader="False">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="SubRegion\_Name" HeaderText="Sub Region" Visible = "false" SortExpression="SubRegion\_Name">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="Site" HeaderText="Site" SortExpression="Site">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="Client" HeaderText="Client" SortExpression="Client">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small" HorizontalAlign="Left"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="Program" HeaderText="Program" SortExpression="Program">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="LOB" HeaderText="LOB" SortExpression="LOB">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="Month"

HeaderText="Month">

<HeaderStyle Font-Size="X-Small" Wrap="False"></HeaderStyle>

<ItemStyle Font-Size="X-Small" HorizontalAlign="Center"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="Category" HeaderText="Category"

SortExpression="Category" Visible="False">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="MetricName" HeaderText="Metric" SortExpression="MetricName">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="CurrentActual" HeaderText="Current Actual" DataFormatString="{0:n2}">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="NewActual" HeaderText="New Actual" DataFormatString="{0:n2}">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="CurrentGoal" HeaderText="Current Goal" DataFormatString="{0:n2}">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="NewGoal" HeaderText="New Goal" DataFormatString="{0:n2}">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="Symbol\_Code" HeaderText="Symbol">

<HeaderStyle Font-Size="X-Small" ></HeaderStyle>

<ItemStyle Font-Size="X-Small" HorizontalAlign="Center" ></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="Expected" HeaderText="Expected">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="SubmittedBy" HeaderText="Submitted By">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="Justification" HeaderText="Justification">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="ChangeType" HeaderText="Change Type" ReadOnly="True"

SortExpression="ChangeType">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="CStatus" HeaderText="Status"

SortExpression="CStatus">

<ControlStyle Font-Size="X-Small" />

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:TemplateField HeaderStyle-Font-Size="X-Small"

ItemStyle-Font-Size="X-Small" HeaderText="Comments">

<ItemTemplate>

<asp:TextBox ID="txtComment" runat="server" Font-Size="X-Small"

TextMode = "Multiline" MaxLength="200"

Enabled = '<%# IsApplicable((int)Eval("ChangeRequestID"), (string)Eval("ChangeType")) %>'

Width="100px" Height="35px" />

<%--<asp:Label ID="lblComment" runat="server" Font-Size="X-Small" CssClass="label" width="150px" Visible="false" />--%>

</ItemTemplate>

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:TemplateField>

<asp:TemplateField>

<ItemTemplate>

<asp:Button ID="BtnApprove" runat="server" Font-Size="X-Small" Text="Approve" CommandName="Approved"

CommandArgument="<%# Container.DataItemIndex %>"

OnClientClick="return confirm('Approve?');"

Enabled = '<%# IsApplicable((int)Eval("ChangeRequestID"), (string)Eval("ChangeType")) %>'

onclick="BtnApprove\_Click" />

<asp:Button ID="BtnReject" runat="server" Font-Size="X-Small" Text=" Reject " CommandName="Rejected"

CommandArgument="<%# Container.DataItemIndex %>"

OnClientClick="return confirm('Reject?');"

Enabled = '<%# IsApplicable((int)Eval("ChangeRequestID"), (string)Eval("ChangeType")) %>'

onclick="BtnReject\_Click" />

</ItemTemplate>

<ItemStyle Width="50px" />

</asp:TemplateField>

<asp:BoundField DataField="MetricUsageName" HeaderText="Metric Usage"

Visible="False">

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="EffectiveEnd\_DT" DataFormatString="{0:M-dd-yyyy}"

HeaderText="End Date" Visible="False">

<HeaderStyle Font-Size="X-Small" Wrap="False"></HeaderStyle>

<ItemStyle Font-Size="X-Small" HorizontalAlign="Center" Width="60px"></ItemStyle>

</asp:BoundField>

<asp:ButtonField CommandName="Views" HeaderText="Metric Usages" Text="View" Visible="False">

<HeaderStyle Font-Size="X-Small" HorizontalAlign="Center" />

<ItemStyle Font-Size="X-Small" HorizontalAlign="Center" />

</asp:ButtonField>

<asp:BoundField DataField="ChangeRequestID" Visible="false" HeaderText="ID">

<HeaderStyle Font-Size="X-Small"></HeaderStyle>

<ItemStyle Font-Size="X-Small" HorizontalAlign="Center" Width="60px"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="ChangeRequestID" HeaderText="ChangeRequestID"

Visible="False">

<ControlStyle Font-Size="X-Small" />

<HeaderStyle Font-Size="X-Small" />

<ItemStyle Font-Size="X-Small" />

</asp:BoundField>

<asp:BoundField DataField="SubmitterID" HeaderText="SubmitterID"

Visible="False" />

</Columns>

<HeaderStyle CssClass="HeaderStyle" BackColor="#CCCCCC" />

<RowStyle CssClass="RowStyle" />

<SelectedRowStyle CssClass="SelectedRowStyle" />

<AlternatingRowStyle CssClass="AlternatingRowStyle" />

</asp:GridView>

</ContentTemplate>

</asp:UpdatePanel>

<asp:UpdatePanel ID="UpdatePanel1" runat="server">

<ContentTemplate>

<asp:Panel ID="Panel4" runat="server">

<br>

<asp:GridView ID="GridView3" runat="server" AutoGenerateColumns="False" Width="412px">

<Columns>

<asp:BoundField DataField="MetricUsageName" HeaderText="Metric Usage">

<HeaderStyle Font-Size="Small" Width="0px" Wrap="False" />

<ItemStyle Font-Size="Small" Width="0px" Wrap="False" HorizontalAlign="Center" />

</asp:BoundField>

<asp:BoundField DataField="Start" DataFormatString="{0:MMMM d, yyyy}" HeaderText="Start Date">

<HeaderStyle Font-Size="Small" Width="0px" HorizontalAlign="Center" Wrap="False" />

<ItemStyle Font-Size="Small" Width="0px" Wrap="False" HorizontalAlign="Center" />

</asp:BoundField>

<asp:BoundField DataField="End" DataFormatString="{0:MMMM d, yyyy}" HeaderText="End Date">

<HeaderStyle Font-Size="Small" Width="0px" Wrap="False" />

<ItemStyle Font-Size="Small" Width="0px" Wrap="False" HorizontalAlign="Center" />

</asp:BoundField>

</Columns>

<HeaderStyle BackColor="#CCCCCC" CssClass="HeaderStyle" />

<RowStyle CssClass="RowStyle" />

<SelectedRowStyle CssClass="SelectedRowStyle" />

<AlternatingRowStyle CssClass="AlternatingRowStyle" />

</asp:GridView>

</asp:Panel>

</ContentTemplate>

</asp:UpdatePanel>

<div id="AlertDiv">

<div id="AlertMessage">

</div>

<br />

<div id="AlertButtons">

<input id="OKButton" type="button" value="OK" runat="server"

onclick="ClearErrorState()" visible="False" />

</div>

</div>

<asp:HiddenField ID="Outset" runat="server" Value="True" />

<asp:HiddenField ID="AllowUpdate" runat="server" Value="False" />

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.IO;

using System.Linq;

using System.Web;

using System.Web.Script.Services;

using System.Web.Security;

using System.Web.Services;

using System.Web.UI;

using System.Web.UI.WebControls;

using Microsoft.ApplicationBlocks.Data;

using Microsoft.SqlServer.Server;

using System.Text;

using System.Net.Mail;

namespace DataMaps

{

public partial class ucCVPassRate\_Approve : System.Web.UI.UserControl

{

bool InsertMapping = false;

static string \_connStr = AppSettings.sqlConnDWReference;

static int CacheTime = 30;

static int Usage\_Column = 21;

static int EndDate\_Column = 22;

bcCVMetricConfig bcMC = null;

public CUser MCUser

{

get

{

if (Session["User"] == null)

Session["User"] = new CUser();

return (CUser)Session["User"];

}

set

{

Session["User"] = value;

}

}

public DataSet ApproveViewData

{

get

{

if (Session["ApproveViewData"] == null)

Session["ApproveViewData"] = new DataSet();

return (DataSet)Session["ApproveViewData"];

}

set

{

Session["ApproveViewData"] = value;

}

}

#region Blink

public void Blinks(string controlID)

{

string script = "<script type=\"text/javascript\">";

//if (controlID == "Message1") script = script + "Blink('" + Message1.ClientID + "');";

if (controlID == "Message1")

script += "Blink(\"" + Message1.ClientID + "\");";

//Blink("<%=Message1.ClientID%>");

//if (controlID == "Message2") script = script + "Blink('" + Message2.ClientID + "');";

script += " </script>";

Page page = HttpContext.Current.CurrentHandler as Page;

if (page != null && !page.ClientScript.IsClientScriptBlockRegistered("alert3"))

{

page.ClientScript.RegisterStartupScript(page.GetType(), "alert3", script, true);

}

}

#endregion

#region Load

protected string GetUser()

{

//SDM = 130085

//DG = 226946

//VP = 130098

//BD = 2493

string userName = "2493";

try

{

#if (!DEBUG)

{

MembershipUser u;

//userName = System.Web.Security.Membership.GetUser(HttpContext.Current.User.Identity.Name).UserName.ToString();

u = System.Web.Security.Membership.GetUser();

if (u != null)

{

//userId = u.ProviderUserKey.ToString();

userName = u.UserName;

}

else

{

userName = Page.User.Identity.Name;

if (userName.Contains("\\"))

userName = userName.Substring(userName.LastIndexOf("\\") + 1);

}

}

#else

// DG = 232523

// VP = 211975

// BD = 130144

userName = "2493";

#endif

}

catch

{

//userName = "E04922";

userName = Page.User.Identity.Name;

}

return userName;

}

protected void Page\_Load(object sender, EventArgs e)

{

if (IsPostBack) return;

bcMC = new bcCVMetricConfig();

//modify check\_authorization--insertmapping variable;

Check\_Authorization();

//ram

MCUser = bcMC.GetUser("Metric Name Usage", this.GetUser());

LoadRegion();

LoadSite();

LoadClient();

LoadCategories("CVPR");

Load\_Data();

}

protected void Check\_Authorization()

{

bool inicio = System.Convert.ToBoolean(Outset.Value);

if (!inicio)

{

InsertMapping = System.Convert.ToBoolean(AllowUpdate.Value);

return;

}

InsertMapping = true;

Outset.Value = "False";

if (Roles.Enabled)

{

string[] roles = Roles.GetRolesForUser(HttpContext.Current.User.Identity.Name);

InsertMapping = roles.Contains("RefMap.MetricConf.Change");

}

AllowUpdate.Value = System.Convert.ToString(InsertMapping);

}

protected DataSet RetrieveData()

{

SqlParameter[] parameters = new SqlParameter[9];

parameters[0] = new SqlParameter("@RegionID", SqlDbType.Int);

parameters[0].Value = ddlRegion.SelectedValue;

parameters[1] = new SqlParameter("@SubRegionID", SqlDbType.Int);

parameters[1].Value = 0; // ddlSubRegion.SelectedValue;

parameters[2] = new SqlParameter("@SiteID", SqlDbType.Int);

parameters[2].Value = ddlSite.SelectedValue;

parameters[3] = new SqlParameter("@ClientID", SqlDbType.Int);

parameters[3].Value = ddlClient.SelectedValue;

parameters[4] = new SqlParameter("@ProgramID", SqlDbType.Int);

parameters[4].Value = ddlProgram.SelectedValue;

parameters[5] = new SqlParameter("@LOB", SqlDbType.Int);

parameters[5].Value = ddlLOB.SelectedValue;

parameters[6] = new SqlParameter("@CAT", SqlDbType.Int);

parameters[6].Value = ddlCategory.SelectedValue;

parameters[7] = new SqlParameter("@MetricName\_ID", SqlDbType.Int);

parameters[7].Value = ddlMetric.SelectedValue;

parameters[8] = new SqlParameter("@Usage", SqlDbType.Int);

parameters[8].Value = 0;

DataSet ds;

//ds = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_MetricConfigUsage\_Approve\_View", parameters);

ds = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_MetricConfigUsage\_Approve\_View2", parameters);

return ds;

}

protected void RetrieveDataUsage()

{

SqlParameter[] parameters = new SqlParameter[8];

parameters[0] = new SqlParameter("@RegionID", SqlDbType.Int);

parameters[0].Value = ddlRegion.SelectedValue;

parameters[1] = new SqlParameter("@SubRegionID", SqlDbType.Int);

parameters[1].Value = 0; // ddlSubRegion.SelectedValue;

parameters[2] = new SqlParameter("@SiteID", SqlDbType.Int);

parameters[2].Value = ddlSite.SelectedValue;

parameters[3] = new SqlParameter("@ClientID", SqlDbType.Int);

parameters[3].Value = ddlClient.SelectedValue;

parameters[4] = new SqlParameter("@ProgramID", SqlDbType.Int);

parameters[4].Value = ddlProgram.SelectedValue;

parameters[5] = new SqlParameter("@LOB", SqlDbType.Int);

parameters[5].Value = ddlLOB.SelectedValue;

parameters[6] = new SqlParameter("@CAT", SqlDbType.Int);

parameters[6].Value = ddlCategory.SelectedValue;

parameters[7] = new SqlParameter("@MetricName\_ID", SqlDbType.Int);

parameters[7].Value = ddlMetric.SelectedValue;

DataSet ds;

ds = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_MetricConfiguration\_View\_Usage", parameters);

Session["Usage"] = ds;

}

protected void Load\_Data()

{

Session["Usage"] = null;

GridView3.DataSource = null;

GridView3.DataBind();

DataSet ds = RetrieveData();

if (ds.Tables[0].Rows.Count > 0)

RetrieveDataUsage();

GVMetrics.DataSource = ds;

GVMetrics.DataBind();

ApproveViewData = ds;

//BtnApprove.Enabled = true;

//BtnReject.Enabled = false;

if (GVMetrics.Rows.Count > 0)

{

ImageButton1.Visible = true;

}

else

{

ImageButton1.Visible = false;

}

Panel1.Visible = InsertMapping;

//GVMetrics.Columns[0].Visible = InsertMapping;

}

#endregion

#region Select Combo

protected void ddlRegion\_SelectedIndexChanged(object sender, EventArgs e)

{

//LoadSubRegion();

LoadSite();

Load\_Data();

}

protected void ddlSubRegion\_SelectedIndexChanged1(object sender, EventArgs e)

{

LoadSite();

Load\_Data();

}

protected void ddlSite\_SelectedIndexChanged(object sender, EventArgs e)

{

Load\_Data();

}

protected void ddlClient\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadProgram();

Load\_Data();

}

protected void ddlProgram\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadLOB();

Load\_Data();

}

protected void ddlLOB\_SelectedIndexChanged(object sender, EventArgs e)

{

Load\_Data();

}

protected void ddlCategory\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadMetric();

Load\_Data();

}

protected void ddlMetric\_SelectedIndexChanged(object sender, EventArgs e)

{

Load\_Data();

}

#endregion

#region Export Excel

protected void ImageButton1\_PreRender(object sender, EventArgs e)

{

if (sender is ImageButton)

{

ImageButton imgBtn = (ImageButton)sender;

ScriptManager ScriptMgr = (ScriptManager)this.FindControl("ScriptManager1");

ScriptMgr.RegisterPostBackControl(imgBtn);

}

}

protected DataSet LoadAllData()

{

SqlParameter[] parameters = new SqlParameter[8];

parameters[0] = new SqlParameter("@RegionID", SqlDbType.Int);

parameters[0].Value = ddlRegion.SelectedValue;

parameters[1] = new SqlParameter("@SubRegionID", SqlDbType.Int);

parameters[1].Value = 0; // ddlSubRegion.SelectedValue;

parameters[2] = new SqlParameter("@SiteID", SqlDbType.Int);

parameters[2].Value = ddlSite.SelectedValue;

parameters[3] = new SqlParameter("@ClientID", SqlDbType.Int);

parameters[3].Value = ddlClient.SelectedValue;

parameters[4] = new SqlParameter("@ProgramID", SqlDbType.Int);

parameters[4].Value = ddlProgram.SelectedValue;

parameters[5] = new SqlParameter("@LOB", SqlDbType.Int);

parameters[5].Value = ddlLOB.SelectedValue;

parameters[6] = new SqlParameter("@CAT", SqlDbType.Int);

parameters[6].Value = ddlCategory.SelectedValue;

parameters[7] = new SqlParameter("@MetricName\_ID", SqlDbType.Int);

parameters[7].Value = ddlMetric.SelectedValue;

return SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_MetricUsage\_View", parameters);

}

protected void ImageButton1\_Click(object sender, ImageClickEventArgs e)

{

Response.Clear();

Response.Charset = "";

Response.ContentType = "application/vnd.ms-excel";

Response.AddHeader("Content-Disposition", "attachment;filename=Data.xls");

StringWriter sw = new System.IO.StringWriter();

HtmlTextWriter HtmlText = new HtmlTextWriter(sw);

System.Web.UI.WebControls.DataGrid dg = new System.Web.UI.WebControls.DataGrid();

//dg.DataSource = LoadAllData(); ram

dg.DataSource = ApproveViewData;

dg.DataBind();

dg.RenderControl(HtmlText);

try

{

Response.Write(sw.ToString());

Response.End();

}

catch (Exception)

{

}

}

#endregion

#region Load Filters

private void LoadRegion()

{

if (Cache["Regions"] == null)

{

DataSet ds = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_RefMap\_GetRegion");

ddlRegion.DataSource = ds;

Cache.Insert("Regions", ds, null, DateTime.Now.AddMinutes(CacheTime), System.Web.Caching.Cache.NoSlidingExpiration);

}

else

{

ddlRegion.DataSource = (DataSet)Cache["Regions"];

}

ddlRegion.DataTextField = "Region\_Name";

ddlRegion.DataValueField = "Region\_ID";

ddlRegion.DataBind();

ListItem li = new ListItem("--All--", "0");

ddlRegion.Items.Insert(0, li);

}

private void LoadAllSites()

{

SqlParameter[] parameters = new SqlParameter[2];

parameters[0] = new SqlParameter("@RegionId", SqlDbType.Int);

parameters[0].Value = 0;

parameters[1] = new SqlParameter("@SubRegionId", SqlDbType.Int);

parameters[1].Value = 0;

DataSet ds = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_RefMap\_GetSites", parameters);

Cache.Insert("SitesAll", ds, null, DateTime.Now.AddMinutes(CacheTime), System.Web.Caching.Cache.NoSlidingExpiration);

}

private void LoadSite()

{

ddlSite.Items.Clear();

int RegionID = System.Convert.ToInt32(ddlRegion.SelectedValue);

int SubRegionID = 0; // System.Convert.ToInt32(ddlSubRegion.SelectedValue);

if (Cache["SitesAll"] == null) LoadAllSites();

DataSet ds2 = (DataSet)Cache["SitesAll"];

DataTable dt = ds2.Tables[0];

var query = from o in dt.AsEnumerable()

where (o.Field<int>("Region\_ID") == RegionID || RegionID == 0)

&& (o.Field<int>("SubRegion\_ID") == SubRegionID || SubRegionID == 0)

select new

{

Site\_Name = o.Field<string>("Site\_Name"),

Site\_ID = o.Field<int>("Site\_ID")

};

ddlSite.DataSource = query;

ddlSite.DataTextField = "Site\_Name";

ddlSite.DataValueField = "Site\_ID";

ddlSite.DataBind();

ListItem li = new ListItem("--All--", "0");

ddlSite.Items.Insert(0, li);

}

private void LoadClient()

{

ListItem li = new ListItem("--All--", "0");

ddlProgram.Items.Clear();

ddlProgram.Items.Insert(0, li);

ddlLOB.Items.Clear();

ddlLOB.Items.Insert(0, li);

ddlMetric.Items.Clear();

ddlMetric.Items.Insert(0, li);

if (Cache["Clients"] == null)

{

DataSet ds = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_RefMap\_GetClients");

ddlClient.DataSource = ds;

Cache.Insert("Clients", ds, null, DateTime.Now.AddMinutes(CacheTime), System.Web.Caching.Cache.NoSlidingExpiration);

}

else

{

ddlClient.DataSource = (DataSet)Cache["Clients"];

}

ddlClient.DataTextField = "Client\_Name";

ddlClient.DataValueField = "ClientId";

ddlClient.DataBind();

ddlClient.Items.Insert(0, li);

}

private void LoadProgram()

{

ddlProgram.Items.Clear();

int SiteID = 0;

if (Region\_TR.Visible) SiteID = System.Convert.ToInt32(ddlSite.SelectedValue);

if (ddlClient.SelectedValue != "0")

{

SqlParameter[] parameters = new SqlParameter[1];

parameters[0] = new SqlParameter("@Client\_ID", SqlDbType.Int);

parameters[0].Value = ddlClient.SelectedValue;

ddlProgram.DataSource = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_RefMap\_GetProgram", parameters);

ddlProgram.DataTextField = "Program\_Name";

ddlProgram.DataValueField = "ProgramId";

int i = ddlProgram.Items.Count;

i++;

ddlProgram.DataBind();

}

ListItem li = new ListItem("--All--", "0");

ddlProgram.Items.Insert(0, li);

ddlLOB.Items.Clear();

ddlLOB.Items.Insert(0, li);

}

private void LoadLOB()

{

ddlLOB.Items.Clear();

if (ddlClient.SelectedValue != "0")

{

SqlParameter[] parameters = new SqlParameter[2];

parameters[0] = new SqlParameter("@Client\_ID", SqlDbType.Int);

parameters[0].Value = ddlClient.SelectedValue;

parameters[1] = new SqlParameter("@Program\_ID", SqlDbType.Int);

parameters[1].Value = ddlProgram.SelectedValue;

ddlLOB.DataSource = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_RefMap\_GetLOB", parameters);

ddlLOB.DataTextField = "LineofBusiness\_Name";

ddlLOB.DataValueField = "LineofBusiness\_ID";

ddlLOB.DataBind();

int i = ddlLOB.Items.Count;

i++;

}

ListItem li = new ListItem("--All--", "0");

ddlLOB.Items.Insert(0, li);

}

private void LoadCategories(string uType)

{

DataSet ds = null;

if (uType == "CVPR")

ds = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_RefMap\_GetMetricCategory2");

else

ds = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_RefMap\_GetMetricCategory");

ddlCategory.DataSource = ds;

ddlCategory.DataTextField = "MetricCategory\_Name";

ddlCategory.DataValueField = "MetricCategory\_ID";

ddlCategory.DataBind();

ListItem li = new ListItem("--All--", "0");

ddlCategory.Items.Insert(0, li);

}

private void LoadMetric()

{

SqlParameter[] parameters = new SqlParameter[2];

parameters[0] = new SqlParameter("@Category", ddlCategory.SelectedValue);

parameters[1] = new SqlParameter("@Metric", "");

ddlMetric.DataSource = SqlHelper.ExecuteDataset(\_connStr, CommandType.StoredProcedure, "sp\_RefMap\_GetMetrics", parameters);

ddlMetric.DataTextField = "Metric\_Name";

ddlMetric.DataValueField = "MetricName\_ID";

ddlMetric.DataBind();

ListItem li = new ListItem("--All--", "0");

ddlMetric.Items.Insert(0, li);

}

#endregion

#region Sort

private string ConvertSortDirectionToSql(SortDirection sortDirection)

{

string newSortDirection = String.Empty;

switch (sortDirection)

{

case SortDirection.Ascending:

newSortDirection = "ASC";

break;

case SortDirection.Descending:

newSortDirection = "DESC";

break;

}

return newSortDirection;

}

protected void GVMetrics\_Sorting(object sender, GridViewSortEventArgs e)

{

DataSet ds = RetrieveData();

ApproveViewData = ds;

if (ds.Tables[0].Rows.Count > 0)

{

DataView dataView = new DataView(ds.Tables[0]);

dataView.Sort = e.SortExpression + " " + ConvertSortDirectionToSql(e.SortDirection);

//Session["Data"] = dataView; ram

GVMetrics.DataSource = dataView;

GVMetrics.DataBind();

}

}

#endregion

#region Grid GVMetrics

protected void GVMetrics\_PageIndexChanging(object sender, GridViewPageEventArgs e)

{

GVMetrics.PageIndex = e.NewPageIndex;

//GVMetrics.DataSource = Session["Data"]; ram

GVMetrics.DataSource = ApproveViewData;

GVMetrics.DataBind();

//BtnReject.Enabled = false;

}

protected void GVMetrics\_RowCreated(object sender, GridViewRowEventArgs e)

{

string usages = "";

if (e.Row.RowType == DataControlRowType.DataRow)

{

// e.Row.Attributes.Add("onmouseover", "this.previous\_color=this.style.backgroundColor;this.style.backgroundColor='#C1DAD7'");

// e.Row.Attributes.Add("onmouseout", "this.style.backgroundColor=this.previous\_color;");

object objTemp = GVMetrics.DataKeys[e.Row.RowIndex].Value;

int keys2 = System.Convert.ToInt32(objTemp);

if (Session["Usage"] == null) return;

DataSet ds2 = (DataSet)Session["Usage"];

DataTable dt = ds2.Tables[0];

var query = from o in dt.AsEnumerable()

where o.Field<int>("MetricConfigurationID") == keys2

select o;

usages = "Usages:\n\n";

int conteo = 0;

if (query.Count() > 0)

{

foreach (var emp in query)

{

usages = usages + emp[1] + "\n";

conteo++;

}

if (conteo == 1)

e.Row.Cells[Usage\_Column].ToolTip = conteo.ToString() + " Usage";

else

e.Row.Cells[Usage\_Column].ToolTip = conteo.ToString() + " Usages";

e.Row.Cells[Usage\_Column].Font.Underline = true;

e.Row.Cells[Usage\_Column].ForeColor = System.Drawing.Color.Blue;

}

else

e.Row.Cells[Usage\_Column].Text = "";

}

}

protected void GVMetrics\_RowCommand(object sender, GridViewCommandEventArgs e)

{

ClearMessageLabels();

string opcion = e.CommandName;

int index = Convert.ToInt32(e.CommandArgument) % GVMetrics.PageSize;

if (opcion == "Views")

{

//int index = Convert.ToInt32(e.CommandArgument);

string MetricID2 = GVMetrics.DataKeys[index].Values["MetricConfigurationID"].ToString();

int MetricID = System.Convert.ToInt32(MetricID2);

if (Session["Usage"] == null) return;

DataSet ds2 = (DataSet)Session["Usage"];

DataTable dt = ds2.Tables[0];

var query = from o in dt.AsEnumerable()

where o.Field<int>("MetricConfigurationID") == MetricID

select new

{

MetricUsageName = o.Field<string>("MetricUsageName"),

Start = o.Field<System.DateTime>("effectivestart\_DT"),

End = o.Field<System.DateTime>("effectiveend\_DT")

};

GridView3.DataSource = query;

GridView3.DataBind();

}

int changeTypeCol = 17;

int siteCol = 2;

int clientCol = 3;

int programCol = 4;

int lobCol = 5;

int monthCol = 6;

int metricCol = 8;

int oldActualCol = 9;

int newActualCol = 10;

int oldGoalCol = 11;

int newGoalCol = 12;

int requesterNameCol = 15;

int requesterIdCol = 26;

string comment = "";

string changeType = "";

int changerequestid = 0;

string site = "";

string client = "";

string program = "";

string lob = "";

string month = "";

string metric = "";

string oldActual = "";

string newActual = "";

string oldGoal = "";

string newGoal = "";

string requesterName = "";

string requesterID = "";

string strRet = "";

bcMC = new bcCVMetricConfig();

if (opcion == "Approved" || opcion == "Rejected")

{

//Get index

//int index = Convert.ToInt32(e.CommandArgument);

comment = ((System.Web.UI.WebControls.TextBox)GVMetrics.Rows[index].FindControl("txtComment")).Text;

changeType = GVMetrics.Rows[index].Cells[changeTypeCol].Text;

changerequestid = Convert.ToInt32(GVMetrics.DataKeys[index].Values["ChangeRequestID"]);

site = GVMetrics.Rows[index].Cells[siteCol].Text;

client = GVMetrics.Rows[index].Cells[clientCol].Text;

program = GVMetrics.Rows[index].Cells[programCol].Text;

lob = GVMetrics.Rows[index].Cells[lobCol].Text;

month = GVMetrics.Rows[index].Cells[monthCol].Text;

metric = GVMetrics.Rows[index].Cells[metricCol].Text;

oldActual = GVMetrics.Rows[index].Cells[oldActualCol].Text;

newActual = GVMetrics.Rows[index].Cells[newActualCol].Text;

oldGoal = GVMetrics.Rows[index].Cells[oldGoalCol].Text;

newGoal = GVMetrics.Rows[index].Cells[newGoalCol].Text;

requesterName = GVMetrics.Rows[index].Cells[requesterNameCol].Text;

requesterID = GVMetrics.Rows[index].Cells[requesterIdCol].Text;

strRet = "Error";

//check for comment

if (comment.Trim() == "")

Message1.Text = "Comment or justification is required.";

else

{

//bool bTmp;

//if (opcion == "Approved")

// bTmp = true;

//else

// bTmp = false;

strRet = bcMC.ApproveMetricUsageRequest(GetUser(), changerequestid, (opcion == "Approved"), comment, changeType);

if (strRet == "Success")

{

if (opcion == "Approved") //bTmp == true

Message1.Text = "Approval saved successfully.";

else

Message1.Text = "Change request rejected.";

Load\_Data();

if (changeType == "Add Usage" || changeType == "Remove Usage")

ProcessEmail1(GetUser(), changerequestid, opcion, comment, changeType,

site, client, program, lob, month, metric, newGoal, requesterName, requesterID);

else

ProcessEmail2(GetUser(), changerequestid, opcion, comment, changeType,

site, client, program, lob, month, metric, oldActual, newActual, oldGoal, newGoal, requesterName, requesterID);

}

else

Message1.Text = strRet;

}

}

}

protected void GVMetrics\_RowDataBound(object sender, GridViewRowEventArgs e)

{

if (e.Row.RowType == DataControlRowType.DataRow)

{

int cual = e.Row.RowIndex;

if (cual > 0)

{

string fecha = e.Row.Cells[EndDate\_Column].Text;

if (fecha != "")

{

if (DateTime.Parse(fecha) < DateTime.Today)

{

//e.Row.Cells[0].Enabled = false;

e.Row.Cells[EndDate\_Column].ForeColor = System.Drawing.Color.Red;

}

else

{

//e.Row.Cells[0].Enabled = true;

e.Row.Cells[EndDate\_Column].ForeColor = System.Drawing.Color.Black;

}

}

}

}

}

protected bool IsApplicable(int changeRequestId, string changeType)

{

string strCurrentStatus = "";

string strNextStep = "";

if (bcMC == null)

bcMC = new bcCVMetricConfig();

//Get current status of change request

//Then Get next step or approval code

if (changeType == "Add Usage" || changeType == "Remove Usage")

{

strCurrentStatus = bcMC.GetCurrentConfigUsageApprovalStatus(changeRequestId);

strNextStep = bcMC.GetNextApprovalStep();

}

else

{

strCurrentStatus = bcMC.GetCurrentConfigUsageApprovalStatusForGoalActual(changeRequestId);

strNextStep = bcMC.GetNextApprovalStepForGoalActual();

}

//if there's no error in retrieving user details in CVMetricApprovers

if (MCUser.ErrorMsg == "")

if (strNextStep.Contains(MCUser.ApprovalStep))

return true;

else

return false;

else //user does not exist in CVMetricApprovers table

return false;

}

#endregion

protected void BtnCancel\_Click(object sender, EventArgs e)

{

Visibles(false);

}

protected void Visibles(bool show)

{

ClearMessageLabels();

Message1.Visible = show;

Message2.Visible = show;

//BtnApprove.Visible = !show;

//BtnReject.Visible = show;

}

private void ClearMessageLabels()

{

Message1.Text = "";

Message2.Text = "";

}

protected void BtnApprove\_Click(object sender, EventArgs e)

{

ClearMessageLabels();

}

protected void BtnReject\_Click(object sender, EventArgs e)

{

ClearMessageLabels();

}

protected void GVMetrics\_SelectedIndexChanged(object sender, EventArgs e)

{

}

protected void ScriptManager1\_AsyncPostBackError(object sender, AsyncPostBackErrorEventArgs e)

{

if (e.Exception.Data["ExtraInfo"] != null)

{

ScriptManager1.AsyncPostBackErrorMessage =

e.Exception.Message +

e.Exception.Data["ExtraInfo"].ToString();

}

else

{

ScriptManager1.AsyncPostBackErrorMessage =

"An unspecified error occurred.";

}

}

#region EMAIL

//For Pending Actual or Pending Goal

private void ProcessEmail2(string approverID, int changeRequestID, string opcion, string comment, string changeType,

string site, string client, string program, string lob,

string month, string metric,

string oldActual, string newActual, string oldGoal, string newGoal,

string requesterName, string requesterID)

{

string strBody = "";

if (changeType == "Pending Actual")

strBody = "Actual value for " + "\n\n";

else

strBody = "Goal value for " + "\n\n";

strBody += "Site: " + site + "\n";

strBody += "Client: " + client + "\n";

strBody += "Program: " + program + "\n";

strBody += "LOB: " + lob + "\n";

strBody += "Metric: " + metric + "\n";

strBody += "Period: " + month + "\n";

if (changeType == "Pending Actual")

{

if (oldActual != null && oldActual != "&nbsp;")

strBody += "Old Actual: " + oldActual + "\n";

strBody += "New Actual: " + newActual + "\n";

}

else

{

if (oldGoal != null && oldGoal != "&nbsp;")

strBody += "Old Goal: " + oldGoal + "\n";

strBody += "New Goal: " + newGoal + "\n";

}

strBody += "\n" + "has been " + opcion.ToUpper() + "\n";

strBody += "Comment: " + comment + "\n\n";

SendEmail(opcion, strBody, requesterID, requesterName, "BD", "", true);

}

//For Add Usage or Remove Usage

private void ProcessEmail1(string approverID, int changeRequestID, string opcion, string comment, string changeType,

string site, string client, string program, string lob,

string month, string metric, string goal,

string requesterName, string requesterID)

{

string strBody = "";

string strC = bcMC.GetCurrentConfigUsageApprovalStatus(changeRequestID);

//check if pending

if (strC.Contains("P2A") || strC.Contains("P3A"))

strBody = "There is a request to " + changeType.ToUpper() + " for " + "\n\n";

else

strBody = "Request to " + changeType.ToUpper() + " has been " + opcion.ToUpper() + " for " + "\n\n";

strBody += "Comment: " + comment + "\n\n";

strBody += "Details of the request are as follows: " + "\n";

strBody += "Site: " + site + "\n";

strBody += "Client: " + client + "\n";

strBody += "Program: " + program + "\n";

strBody += "LOB: " + lob + "\n";

strBody += "Metric: " + metric + "\n";

strBody += "Period: " + month + "\n";

if (changeType == "Add Usage")

strBody += "Goal: " + goal + "\n\n";

//strBody += "Change Request ID: " + changeRequestID.ToString() + "\n";

if (opcion == "Approved")

{

if (strC.Contains("P2A") && !strC.Contains("P3A"))

SendEmail(opcion, strBody, requesterID, requesterName, "VP", "", false);

if (strC.Contains("P3A") && !strC.Contains("P2A"))

SendEmail(opcion, strBody, requesterID, requesterName, "BD", "", false);

if (strC.Contains("P3A") && strC.Contains("P2A"))

SendEmail(opcion, strBody, requesterID, requesterName, "BD", "VP", false);

if (strC.Contains("AAA"))

SendEmail(opcion, strBody, requesterID, requesterName, "BD", "", true);

}

else

{

if (strC.Contains("R1A"))

SendEmail(opcion, strBody, requesterID, requesterName, "Data Governance", "", true);

else if (strC.Contains("R2A"))

SendEmail(opcion, strBody, requesterID, requesterName, "VP", "", true);

else if (strC.Contains("R3A"))

SendEmail(opcion, strBody, requesterID, requesterName, "BD", "", true);

}

}

protected void SendEmail(string opcion, string strBody, string requesterID, string requesterName, string sendToGroup, string sendToGroup2, bool isFinal)

{

string machine = System.Environment.MachineName.ToString();

//if (machine == "WWS10-001-STAGE" || machine == "LAPTOP\_RMARTIN") return;

int steps = 0;

string SendTo = "";

try

{

MailMessage mail = new MailMessage();

SmtpClient SmtpServer = new SmtpClient("mailedc01");

steps = 1;

string userName = GetUser();

string From = "CVMetric\_ChangeRequest\_No\_Reply@stream.com";

string BCC = "approver@stream.com";

string BCC2 = "requester@stream.com";

try

{

//From = System.Web.Security.Membership.GetUser(userName).Email.ToString();

BCC = System.Web.Security.Membership.GetUser(userName).Email.ToString();

BCC2 = System.Web.Security.Membership.GetUser(requesterID).Email.ToString();

steps = 2;

}

catch

{

//From = "CVMetric\_ChangeRequest\_No\_Reply@stream.com";

BCC = "robert.martin@stream.com";

BCC2 = "emil.kirov@stream.com";

}

mail.From = new MailAddress(From);

mail.Bcc.Add(new MailAddress(BCC));

mail.Bcc.Add(new MailAddress(BCC2));

steps = 3;

SendTo = GetUsers(sendToGroup);

if (sendToGroup2 != "")

SendTo += GetUsers(sendToGroup2);

if (SendTo == "")

{

Message2.Text = "Error sending the email. The list for approvers is empty";

Blinks("Message2");

return;

}

steps = 4;

// take the last ','

int largo = SendTo.Length;

SendTo = SendTo.Substring(0, largo - 1);

steps = 5;

mail.To.Add(SendTo);

steps = 6;

//sp\_RefMap\_GetEmployeeName

//string Nombre = GetUserName();

string Nombre = requesterName;

steps = 7;

string place = "";

try

{

int pos1 = strBody.IndexOf("Site");

int pos2 = strBody.IndexOf("Client");

string sitio = strBody.Substring(pos1, (pos2 - pos1 - 1));

place = " for " + sitio;

}

catch

{

}

mail.Subject = GetEmailSubject((machine == "WWS10-001-STAGE" || machine == "LAPTOP\_RMARTIN"), isFinal, opcion, SendTo, place);

mail.Body = strBody + "\nRequestor: " + Nombre + "\n\nClick this link to access the tool: http://referencesystem.stream.com/ ";

SmtpServer.Port = 25;

//SmtpServer.Credentials = new System.Net.NetworkCredential("90980", "SAL@Si0920");

//SmtpServer.EnableSsl = true;

steps = 10;

SmtpServer.Send(mail);

}

catch

{

Message2.Text = "Error sending the email in step " + steps.ToString() + " , report it to Robert.Martin@Stream.com " + SendTo;

Blinks("Message2");

}

}

private string GetEmailSubject(bool isTestEmail, bool isFinal, string opcion, string SendTo, string place)

{

string strRet = "";

if (opcion == "Approved")

{

if (isFinal)

{

if (SendTo.Length > 10)

strRet = "xStream Data Maps, Request Approved " + place;

else

strRet = "xStream Data Maps, Request Approved.";

}

else

{

if (SendTo.Length > 10)

strRet = "xStream Data Maps, Request to Approve " + place;

else

strRet = "xStream Data Maps, Request to Approve.";

}

}

else

{

if (SendTo.Length > 10)

strRet = "xStream Data Maps, Request Rejected " + place;

else

strRet = "xStream Data Maps, Request Rejected.";

}

if (isTestEmail)

strRet = "Test Email: " + strRet;

return strRet;

}

protected string GetUserName()

{

if (Session["Employee"] != null)

{

return Session["Employee"].ToString();

}

string From = GetUser();

String strSQL = "sp\_RefMap\_GetEmployeeName2";

SqlParameter[] parameters = new SqlParameter[1];

parameters[0] = new SqlParameter("@LoginID", SqlDbType.NVarChar, 100);

//parameters[0].Value = System.Convert.ToInt32(From);

parameters[0].Value = From;

object names = SqlHelper.ExecuteScalar(\_connStr, CommandType.StoredProcedure, strSQL, parameters);

Session["Employee"] = names.ToString();

return names.ToString();

}

protected string GetUsers(string strRole)

{

bcCVMetricConfig bcMC = new bcCVMetricConfig();

if (Cache["UserList"] != null)

{

return Cache["UserList"].ToString();

}

string listado = "";

try

{

string correo1, user1;

string[] users = bcMC.GetUsersInRole(strRole);

for (int i = 0; i < users.Length; i++)

{

user1 = users[i];

correo1 = System.Web.Security.Membership.GetUser(user1).Email.ToString();

listado = listado + correo1 + ",";

}

}

catch

{

listado = "Robert.Martin@stream.com,";

}

if (listado.Length > 40)

{

Cache.Insert("UserList", listado, null, System.Web.Caching.Cache.NoAbsoluteExpiration, new TimeSpan(0, 15, 0));

}

return listado;

}

//protected string GetUserName(string From)

//{

// String strSQL = "sp\_RefMap\_GetEmployeeName";

// SqlParameter[] parameters = new SqlParameter[1];

// parameters[0] = new SqlParameter("@LoginID", SqlDbType.Int);

// parameters[0].Value = System.Convert.ToInt32(From);

// object names = SqlHelper.ExecuteScalar(\_connStr, CommandType.StoredProcedure, strSQL, parameters);

// return names.ToString();

//}

#endregion

}

}