* **Add to Enumeration:** DashboardEnum.cs

Venue = 2

* **Add Dashboard class that inherits from BaseDashboard**

Note: Add class to \mvc\Dashboards

*namespace MedCore.Dashboards*

*{*

*public class VenueDashBoard : BaseDashboard*

*{*

*public DashboardEnum DashboardType = DashboardEnum.UserAnmin;*

*}*

*}*

* **Implement Load – function that is responsible for initial load**

public override void Load(ViewDataDictionary ViewData, User currentUser)

{

var context = new HttpContextWrapper(HttpContext.Current);

var currentPerson = (MedCoreLib.BusinessObjects.Person)context.Session[SessionItems.Person];

ViewData["List"] = Venue. LoadLatest(currentPerson);

ViewData["GroupedList"] = VenueState.LoadLatests(currentPerson.Id);

ViewData["Dashboard"] = DashboardType;

ViewData["DashboardControl"] = "\_VenueList";

}

* **(optional)Implement RefreshWidget – function that is responsible for ajax refresh**

public override Dictionary<string, object> RefreshWidget(ViewDataDictionary ViewData, ControllerContext ControllerContext, TempDataDictionary TempData, User currentUser, long lastId, int count, string widgetitems)

* **Implement LoadLatest for a list**

public static List<Venue> LoadLatest(Person pOwner, long lastestId = 0)

{

var conn = DataBase.ConnectionFromContext();

var venues = new List<Venue>();

var cmd = new SqlCommand("Venue\_Populate\_Latests", conn) { CommandType = CommandType.StoredProcedure };

cmd.Parameters.Add(new SqlParameter("@OwnerId", SqlDbType.BigInt) { Value = pOwner.Id });

using (var rdr = cmd.ExecuteReader(CommandBehavior.SingleResult))

{

while (rdr.Read())

{

var od = (Venue)(new Venue()).FromDataRow(rdr);

venues.Add(od);

}

rdr.Close();

}

return venues;

}

CREATE PROCEDURE [dbo].[Venue\_Populate\_Latests]

@OwnerId bigint

AS

BEGIN

SELECT v.VenueId,v.Name,v.DateStart

,v.VenueStateId,v.Name as VenueStateName

,pc.PredefinedColorId, pc.Code as PredefinedColorCode

FROM Venue v

INNER JOIN VenueState vs ON vs.VenueStateId=v.VenueStateId

INNER JOIN PredefinedColor pc ON pc.PredefinedColorId=vs.PredefinedColorId

WHERE v.OwnerId = @OwnerId AND v.DeletedBy IS NULL

END

* **(optoinal) Define a Group class.**

It should inherit from AggregateBase

public class VenueState : AggregateBase

* **Implement LoadLatests for diagram**

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LIB.AdvancedProperties;

using LIB.BusinessObjects;

using LIB.Tools.AdminArea;

using LIB.Tools.BO;

using LIB.Tools.Utils;

namespace MedCoreLib.BusinessObjects

{

[Bo(Group = AdminAreaGroupenum.Venue

, ModulesAccess = (long)(Modulesenum.ControlPanel)

, DisplayName = "Venue Statuses"

, SingleName = "Venue Status")]

public class VenueState : AggregateBase

{

#region Constructors

/// <summary>

/// Initializes a new instance of the <see cref="DisplayMode"/> class.

/// </summary>

public VenueState()

: base(0)

{

}

/// <summary>

/// Initializes a new instance of the <see cref="DisplayMode"/> class.

/// </summary>

/// <param name="id">

/// The id.

/// </param>

public VenueState(long id)

: base(id)

{

}

#endregion

public override int GetCount()

{

return VenuesCount;

}

public static Dictionary<long, AggregateBase> LoadLatests(long pOwner)

{

var conn = DataBase.ConnectionFromContext();

var venueStates = new Dictionary<long, AggregateBase>();

var cmd = new SqlCommand("VenueState\_Populate\_Latests", conn) { CommandType = CommandType.StoredProcedure };

if (pOwner > 0)

cmd.Parameters.Add(new SqlParameter("@OwnerId", SqlDbType.BigInt) { Value = pOwner });

using (var rdr = cmd.ExecuteReader(CommandBehavior.SingleResult))

{

while (rdr.Read())

{

var state = (VenueState)(new VenueState()).FromDataRow(rdr);

state.Color = rdr["Color"].ToString();

venueStates.Add(state.Id, state);

}

rdr.Close();

}

return venueStates;

}

#region Properties

[Template(Mode = Template.Name)]

public string Name { get; set; }

[Common(EditTemplate = EditTemplates.Hidden), Db(\_Editable = false, \_Populate = false)]

public int VenuesCount { get; set; }

[Template(Mode = Template.DropDown)]

public PredefinedColor PredefinedColor { get; set; }

#endregion

}

}

CREATE PROCEDURE [dbo].[VenueState\_Populate\_Latests]

@OwnerId bigint

AS

BEGIN

SELECT vs.VenueStateId,vs.Name,COUNT(v.VenueId) as VenuesCount

,pc.Color

FROM [Venue] v

INNER JOIN [VenueState] vs ON vs.VenueStateId=v.VenueStateId

INNER JOIN PredefinedColor pc ON pc.PredefinedColorId=vs.PredefinedColorId

WHERE v.DeletedBy iS NULL AND vs.DeletedBy IS NULL

GROUP BY vs.VenueStateId,vs.Name,pc.Color

END

* **Add DashBoardController Logic: DashBoardController.cs**

else if (currentUser.HasAtLeastOnePermission((long)BasePermissionenum.VenueAdmin))

{

return new VenueDashBoard();

}

* **Add List View ex(\_VenueList.cshtml)**

@using LIB.BusinessObjects

@using MedCore.Models.Objects

@if (((List<Venue>)ViewData["List"]).Count > 0)

{

<script type="text/javascript">

var gLatestDashboardId = @(((List<Venue>)ViewData["List"]).First(i => i.Id>0).Id);

var gDashboardItemsCount = @(((List<Venue>)ViewData["List"]).Count);

</script>

<form class="table-widget">

<div class="dashboard-ico">

<div class="outer-ico">

<div class="widget-caption">

Venues

</div>

<div class="data-grid-container">

<div class="data-grid">

<div class="data-grid-title-row">

<div class="data-grid-title">N</div>

<div class="data-grid-title">Name</div>

<div class="data-grid-title">Date Statrt</div>

<div class="data-grid-title">Curent State</div>

</div>

@Html.Partial("Controls/\_VenuesRows")

</div>

</div>

</div>

</div>

<div class="clear" style="height:1px"></div>

</form>

}

* **Add Row View (\_VenuesRows.cshtml)**

@using MedCore.Models.Objects

@using MedCoreLib.BusinessObjects

@{

Layout = "~/Views/Master/\_Ajax.cshtml";

}

@{ var index = ((List<Venue>)ViewData["List"]).Count+ (int)ViewData["DashboardListCount"];}

@foreach (var venue in ((List<Venue>)ViewData["List"]))

{

<a href="@LIB.Tools.Utils.URLHelper.GetUrl("ControlPanel/EditItem/Venue/MedCore.Models.Objects/" + venue.Id )" target="\_blank" class="data-grid-data-row data-grid-data-row-@(venue.Id) data-grid-data-row-type-@(venue.VenueState.PredefinedColor.Code)">

<div class="data-grid-data"><b>@(venue.Id)</b></div>

<div class="data-grid-data">@(venue.Name)</div>

<div class="data-grid-data">@(venue.DateStart.ToString("dd.MM.yyyy HH:mm"))</div>

<div class="data-grid-data">@(venue.VenueState.Name)</div>

<input type="hidden" name="widgetitems" value="@(venue.Id):@(venue.VenueState.Id)" />

</a>

}

* Create Folder for view AdminAreaGroup.cs

public static AdminAreaGroup Venue = new AdminAreaGroup() { Group = AdminAreaGroupenum.Venue, Name = LIB.Tools.Utils.Translate.GetTranslatedValue("Venue", "BO") };