**­­­­-Create Index Grid**

1. Create the model class to be converted to table in the database as following stricter

Model -> Domian -> ModelName -> class name

Example:

public class SendFormPeople

{

[Key, Column(Order = 1)]

public int SendFormId { get; set; }

public SendForm SendForm { get; set; }

[Key, Column(Order = 2)]

public int EmpId { get; set; }

[ForeignKey("EmpId")]

public Person Person { get; set; }

}

1. Add the Controller and the view
   1. The controller inherits form BaseController.
   2. Define private IHrUnitOfWork \_hrUnitOfWork;
   3. Create a constructor for the controller as follwoing.

public SendFormController(IHrUnitOfWork unitOfWork) : base(unitOfWork)

{

\_hrUnitOfWork = unitOfWork;

}

* 1. Create Index Method

public ActionResult Index()

{

return View();

}

* 1. Create Method to get data to be showed in the grid

Example:

public ActionResult GetForms(int MenuId, int pageSize, int skip)

{

return ApplyFilter<SendFormGridVM>(\_hrUnitOfWork.SendFormRepository.GetForms(CompanyId, Language, MsgUtils.Instance.Trls("TargetEmp"), MsgUtils.Instance.Trls("TargetDept"), MsgUtils.Instance.Trls("TargetJob")), null, MenuId, pageSize, skip);

}

1. Add the View to the Index Method.
2. Add the page to the menu from

الاعدادت>التخصيص>مصمم المنيو

1. Add the page to user roles.
2. Add the page from مصمم الجداول
3. Define the fields you want to show.
4. Go to index view and add the following
5. @using System.Web.Script.Serialization;

@{

Layout = null;

}

@Html.Partial("\_RequestFilters", new ViewDataDictionary { { "GridName", "SendFormDiv" }, { "hasDepts", false }, { "hasRange", false }, { "AddUrl", Url.Action("Detail", "SendForm") } })

<div id="SendFormDiv"></div>

<script>

Grids.indexGrid({

read: "@Url.Action("GetForms", "SendForm")?MenuId=@Request.QueryString["MenuId"]",

show: "@Url.Action("Details", "SendForm")",

edit: "@Url.Action("Details", "SendForm")",

objectName: "SendForm",

gridName: "SendFormDiv",

admin: "@User.Identity.CanCustomize()",

lang: "@User.Identity.GetCulture()",

columns: [{ field: "TargetText", sortable: false }, { field: "FormId", values: @Html.Raw(ViewBag.FormNameList)}],

serverPaging: true,

serverFiltering: true,

serverSorting: true,

pageable: { pageSize: 25, pageSizes: [25, 50, 75, 100, 125, 150, 175] }

});

</script>

1. Replace your own methods.

**­­­­-Delete Record from index grid**

1. Method name “Delete+PageName”
2. Takes parameter Id
3. Define var message=”OK”.
4. Defin DataSource<PageViewModel> source.
5. Get the model by Id and use Automapper.
6. To save changes source.Errors = SaveChanges(Language).
7. Check if source.Errors.count>0 then return Json(source) else return json(message).

Example

public ActionResult DeleteJob(int id)

{

var message = "OK";

DataSource<PositionViewModel> Source = new DataSource<PositionViewModel>();

Job Job = \_hrUnitOfWork.JobRepository.Get(id);

if (Job != null)

{

AutoMapper(new Models.AutoMapperParm

{

Source = Job,

ObjectName = "Job",

Transtype = TransType.Delete

});

\_hrUnitOfWork.JobRepository.Remove(Job);

\_hrUnitOfWork.JobRepository.RemoveLName(Language,Job.Name);

}

Source.Errors = SaveChanges(Language);

if (Source.Errors.Count() > 0)

return Json(Source);

else

return Json(message);

}