To implement an **AutoComplete feature for all grid columns** during **Add** and **Edit** actions in a **Telerik ASP.NET Core Grid**, and to make it reusable across the application, you can follow these steps:

**1. Create a Common Controller for AutoComplete**

Create a common controller that will handle the AutoComplete functionality for all columns. This controller will take a SQL query or a Dapper call as input and return the data for the AutoComplete.

**Example: CommonController.cs**

csharp

Copy

using Microsoft.AspNetCore.Mvc;

using System.Collections.Generic;

using System.Data.SqlClient;

using Dapper;

namespace YourNamespace.Controllers

{

public class CommonController : Controller

{

private readonly string \_connectionString;

public CommonController(IConfiguration configuration)

{

\_connectionString = configuration.GetConnectionString("DefaultConnection");

}

[HttpGet]

public IActionResult GetAutoCompleteData(string query, string columnName)

{

// Example: Use Dapper to fetch data from the database

using (var connection = new SqlConnection(\_connectionString))

{

connection.Open();

var sql = $"SELECT DISTINCT {columnName} FROM YourTableName WHERE {columnName} LIKE @Query + '%'";

var results = connection.Query<string>(sql, new { Query = query });

return Json(results);

}

}

}

}

**2. Create a Common Editor Template for AutoComplete**

Create a common editor template that will be used for all columns that require AutoComplete. This template will call the common controller action.

**Example: AutoCompleteEditor.cshtml**

Place this file in the Views/Shared/EditorTemplates folder.

html

Copy

@model string

@(Html.Kendo().AutoComplete()

.Name(ViewData.TemplateInfo.HtmlFieldPrefix) // Dynamic name based on the field

.DataTextField("Text")

.Filter("contains")

.Placeholder("Type to search...")

.DataSource(source =>

{

source.Read(read =>

{

read.Action("GetAutoCompleteData", "Common") // Common controller action

.Data("onAdditionalData"); // Pass additional data (e.g., column name)

})

.ServerFiltering(true);

})

)

<script>

function onAdditionalData() {

// Get the column name dynamically

var columnName = "@ViewData.ModelMetadata.PropertyName";

return {

query: $("#" + columnName).data("kendoAutoComplete").value(),

columnName: columnName

};

}

</script>

Run HTML

**3. Update Your Grids to Use the Common Editor Template**

Update your grid definitions to use the common editor template for the columns that require AutoComplete.

**Example: LabourTask.cshtml**

html

Copy

@(Html.Kendo().Grid<BCES.Models.Admin.TaskModel>()

.Name("TaskGrid")

.Columns(columns =>

{

columns.Bound(p => p.TaskId).Title("Task Id").Hidden(true);

columns.Bound(p => p.TaskDescription).Title("Task Description")

.EditorTemplateName("AutoCompleteEditor"); // Use the common editor template

columns.Command(command =>

{

if (@ViewBag.RoleId == 1)

{

command.Edit();

}

if (@ViewBag.RoleId == 1)

{

command.Destroy();

}

}).Width(200);

})

.ToolBar(toolbar =>

{

if (@ViewBag.RoleId == 1)

{

toolbar.Create();

}

})

.Editable(editable => editable.Mode(GridEditMode.InLine))

.Pageable()

.Sortable()

.Filterable(ftb => ftb.Mode(GridFilterMode.Row))

.DataSource(dataSource => dataSource

.Ajax()

.Model(model =>

{

model.Id(u => u.TaskId);

model.Field(c => c.TaskDescription);

})

.Read(read => read.Url(Url.Action("ReadTasks", "Task")).Type(HttpVerbs.Get))

.Create(create => create.Url(Url.Action("CreateTask", "Task")).Type(HttpVerbs.Post))

.Update(update => create.Url(Url.Action("UpdateTask", "Task")).Type(HttpVerbs.Post))

.Destroy(delete => create.Url(Url.Action("DeleteTask", "Task")).Type(HttpVerbs.Post))

)

)

Run HTML

**Example: NonStockCodedParts.cshtml**

html

Copy

@(Html.Kendo().Grid<BCES.Models.Parts.NscPartsUsedViewModel>()

.Name("NscPartsUsedGrid")

.Columns(columns =>

{

columns.Bound(p => p.OrigSuppNum).Title("Supplier Number")

.EditorTemplateName("AutoCompleteEditor"); // Use the common editor template

columns.Bound(p => p.OrigSupplierName).Title("Supplier Name")

.EditorTemplateName("AutoCompleteEditor"); // Use the common editor template

columns.Bound(p => p.KeyWord).Title("Keyword")

.EditorTemplateName("AutoCompleteEditor"); // Use the common editor template

columns.Bound(p => p.Description).Title("Description")

.EditorTemplateName("AutoCompleteEditor"); // Use the common editor template

columns.Bound(p => p.CoreCost).Title("Per Unit Cost")

.Format("{0:C2}").EditorTemplateName("Currency");

columns.Command(command =>

{

if (@ViewBag.RoleId == 1)

{

command.Edit();

}

if (@ViewBag.RoleId == 1)

{

command.Destroy();

}

}).Width(200);

})

.ToolBar(toolbar =>

{

if (@ViewBag.RoleId == 1)

{

toolbar.Create();

}

})

.Editable(editable => editable.Mode(GridEditMode.InLine))

.Pageable()

.Sortable()

.Filterable(ftb => ftb.Mode(GridFilterMode.Row))

.DataSource(dataSource => dataSource

.Ajax()

.Model(model =>

{

model.Id(u => u.NscPartsUsedId);

})

.Read(read => read.Url(Url.Action("GetNscPartsUsedView", "NscPartsUsed")).Type(HttpVerbs.Get))

.Create(create => create.Url(Url.Action("CreateNscPartsUsed", "NscPartsUsed")).Type(HttpVerbs.Post))

.Update(update => create.Url(Url.Action("UpdateNscPartsUsed", "NscPartsUsed")).Type(HttpVerbs.Post))

.Destroy(delete => create.Url(Url.Action("DeleteNscPartsUsed", "NscPartsUsed")).Type(HttpVerbs.Post))

)

)

Run HTML

**4. Summary**

* **Common Controller**: Handles the AutoComplete data fetching logic.
* **Common Editor Template**: Provides a reusable AutoComplete editor for all columns.
* **Grid Definitions**: Use the common editor template for columns that require AutoComplete.

This approach ensures that you can reuse the AutoComplete functionality across your application without duplicating code. Let me know if you need further clarification!