**1. Update the RebuiltPartsViewModel.cs**

Ensure that the VehicleSeries property is properly defined to hold the selected values from the MultiSelect.

csharp

Copy

public class RebuiltPartsViewModel

{

public int RbMasterlistId { get; set; }

public string RebuiltStockNum { get; set; }

public string MmsStockCode { get; set; }

public string DetailedDesc { get; set; }

public string Keyword { get; set; }

public string JobNumber { get; set; }

public string CorePartNum { get; set; }

public decimal CoreCharge { get; set; }

public decimal BuyNewCost { get; set; }

public decimal RemanCost { get; set; }

public List<BusesModel> VehicleSeries { get; set; } = new List<BusesModel>();

}

**2. Create a Controller Method to Fetch Data for MultiSelect**

Add a method in your controller to fetch the data for the MultiSelect using Dapper.

csharp

Copy

using Dapper;

using Microsoft.AspNetCore.Mvc;

using System.Collections.Generic;

using System.Data;

using System.Threading.Tasks;

public class RebuiltPartsController : Controller

{

private readonly IDbConnection \_dbConnection;

public RebuiltPartsController(IDbConnection dbConnection)

{

\_dbConnection = dbConnection;

}

[HttpGet]

public async Task<IActionResult> GetVehicleSeries()

{

try

{

var sql = @"

SELECT

RBLISTOFBUSES.listid AS ListId,

LISTOFBUSES.description AS Description

FROM

SBCES.RBLISTOFBUSES

INNER JOIN

SBCES.LISTOFBUSES ON RBLISTOFBUSES.listid = LISTOFBUSES.listid";

var vehicleSeries = await \_dbConnection.QueryAsync<BusesModel>(sql);

return Json(vehicleSeries);

}

catch (Exception ex)

{

return StatusCode(500, "Internal server error: " + ex.Message);

}

}

}

**3. Update the Index.cshtml**

Add the MultiSelect column to the grid and configure it to use the controller method for data fetching.

html

Copy

@(Html.Kendo().Grid<RebuiltPartsViewModel>()

.Name("RebuiltPartsGrid")

.Columns(columns =>

{

columns.Bound(c => c.RebuiltStockNum).Title("Rebuilt Stock Code").Filterable(ftb => ftb.Cell(cell => cell.Operator("contains").SuggestionOperator(FilterType.Contains)));

columns.Bound(c => c.MmsStockCode).Title("MMS Stock Code").Filterable(ftb => ftb.Cell(cell => cell.Operator("contains").SuggestionOperator(FilterType.Contains)));

columns.Bound(c => c.DetailedDesc).Title("Description").Filterable(ftb => ftb.Cell(cell => cell.Operator("contains").SuggestionOperator(FilterType.Contains)));

columns.Bound(c => c.Keyword).Title("Keyword").Filterable(ftb => ftb.Cell(cell => cell.Operator("contains").SuggestionOperator(FilterType.Contains)));

columns.Bound(c => c.JobNumber).Title("Job Number").Filterable(ftb => ftb.Cell(cell => cell.Operator("contains").SuggestionOperator(FilterType.Contains)));

columns.Bound(c => c.CorePartNum).Title("Core Code").Filterable(ftb => ftb.Cell(cell => cell.Operator("contains").SuggestionOperator(FilterType.Contains)));

columns.Bound(c => c.CoreCharge).Title("Core Cost").Filterable(ftb => ftb.Cell(cell => cell.Operator("gte"))).Format("{0:C2}").EditorTemplateName("Currency");

columns.Bound(c => c.BuyNewCost).Title("Buy Cost").Filterable(ftb => ftb.Cell(cell => cell.Operator("gte"))).Format("{0:C2}").EditorTemplateName("Currency");

columns.Bound(c => c.RemanCost).Title("Remanufactured Cost").Filterable(ftb => ftb.Cell(cell => cell.Operator("gte"))).Format("{0:C2}").EditorTemplateName("Currency");

columns.Bound(c => c.VehicleSeries).Title("Vehicle Series")

.ClientTemplate("#=VehicleSeries.map(function(item) { return item.Description; }).join(', ') #")

.EditorTemplateName("VehicleSeriesEditor");

columns.Command(command =>

{

command.Edit().Text(" ").IconClass("k-icon k-i-edit").HtmlAttributes(new { title = "Edit" });

command.Custom("Archive").Text(" ").IconClass("k-icon k-i-folder").Click("onArchiveClick").HtmlAttributes(new { title = "Archive" });

command.Custom("Save to Excel").Text(" ").IconClass("k-icon k-i-file-excel").Click("onSaveToExcelClick").HtmlAttributes(new { title = "Save to Excel" });

command.Custom("Email Estimate").Text(" ").IconClass("k-icon k-i-email").Click("onEmailEstimateClick").HtmlAttributes(new { title = "Email Estimate" });

}).Title("Actions").Width(200).HtmlAttributes(new { @class = "action-buttons" });

})

.ToolBar(toolbar => { toolbar.Create(); })

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

.Pageable()

.Sortable()

.ClientDetailTemplateId("template")

.Resizable(r => r.Columns(true))

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

.DataSource(dataSource => dataSource

.Ajax()

.Model(model =>

{

model.Id(c => c.RbMasterlistId);

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

model.Field(c => c.RebuiltStockNum).Editable(true);

model.Field(c => c.MmsStockCode).Editable(true);

model.Field(c => c.VehicleSeries).DefaultValue(new List<BusesModel>()).Editable(true);

})

.Read(read => read.Url(Url.Action("GetRebuiltPartsView", "RebuiltParts")).Type(HttpVerbs.Get))

.Create(create => create.Url(Url.Action("AddRebuiltPartsView", "RebuiltParts")).Type(HttpVerbs.Post))

.Update(update => update.Url(Url.Action("UpdateRebuiltPartsView", "RebuiltParts")).Type(HttpVerbs.Post))

)

)

Run HTML

**4. Create the Editor Template for MultiSelect**

Create a new editor template named VehicleSeriesEditor.cshtml in the ~/Views/Shared/EditorTemplates folder.

html

Copy

@model List<BCES.Models.Parts.BusesModel>

@(Html.Kendo().MultiSelectFor(m => m)

.Name("VehicleSeries")

.DataTextField("Description")

.DataValueField("ListId")

.Placeholder("Select vehicle series...")

.AutoBind(false)

.DataSource(source =>

{

source.Read(read =>

{

read.Action("GetVehicleSeries", "RebuiltParts");

})

.ServerFiltering(true);

})

)

Run HTML

**5. Update the RebuiltPartsController to Handle MultiSelect Data**

Ensure that the UpdateRebuiltPartsView and AddRebuiltPartsView methods in the controller can handle the VehicleSeries data.

csharp

Copy

[HttpPost]

public IActionResult UpdateRebuiltPartsView([FromBody] RebuiltPartsViewModel model)

{

try

{

// Update logic here, including handling VehicleSeries

return Ok();

}

catch (Exception ex)

{

return StatusCode(500, "Internal server error: " + ex.Message);

}

}

[HttpPost]

public IActionResult AddRebuiltPartsView([FromBody] RebuiltPartsViewModel model)

{

try

{

// Add logic here, including handling VehicleSeries

return Ok();

}

catch (Exception ex)

{

return StatusCode(500, "Internal server error: " + ex.Message);

}

}

**6. Exception Handling and Logging**

Ensure that you have proper exception handling and logging in place to catch and log any errors that occur during the data fetching or updating processes.

**7. File Locations**

* **Controller**: ~/Controllers/RebuiltPartsController.cs
* **View**: ~/Views/RebuiltParts/Index.cshtml
* **Editor Template**: ~/Views/Shared/EditorTemplates/VehicleSeriesEditor.cshtml
* **Model**: ~/Models/Parts/RebuiltPartsViewModel.cs and ~/Models/Parts/BusesModel.cs

By following these steps, you should be able to integrate a Telerik MultiSelect in a Kendo Grid column with inline editing, fetching data using Dapper, and handling server-side logic