**Update VehicleSeriesEditor to Match Buses Field**

Update the VehicleSeriesEditor.cshtml file to ensure that the MultiSelect properly binds to the Buses field in the grid.

html

CopyEdit

@model List<int> <!-- Expect a list of selected ListId values -->

@(Html.Kendo().MultiSelect()

.Name("Buses") // This must match the field name in the grid

.DataTextField("Description") // Display Description in the MultiSelect dropdown

.DataValueField("ListId") // Bind to ListId for the values

.BindTo((IEnumerable)ViewData["BusesList"]) // Provide the full list of available buses

)

**2. Pass a Correct BusesList to ViewData**

Ensure the controller action (Index) is passing a correctly formatted ViewData["BusesList"] to the editor template.

csharp

CopyEdit

public async Task<IActionResult> Index()

{

// Query to fetch all available buses for the MultiSelect

var query = "SELECT ListId, Description FROM SBCES.LISTOFBUSES";

// Fetch the list of buses

var busesList = await \_dbConnection.QueryAsync<BusesModel>(query);

// Pass the buses list to ViewData for the MultiSelect

ViewData["BusesList"] = busesList;

// Return the view

return View();

}

**3. Fix GetRebuiltPartsView for MultiSelect Value Binding**

The grid's Buses field binds to a list of selected ListId values. The controller should return the Buses property as a list of IDs (ListId) for proper value binding in the MultiSelect editor.

csharp

CopyEdit

public async Task<JsonResult> GetRebuiltPartsView([DataSourceRequest] DataSourceRequest request)

{

try

{

var query = @"

SELECT

rbm.RebuiltStockNum,

rbm.Keyword,

rbl.ListId,

lb.Description

FROM SBCES.RbMasterlist rbm

LEFT JOIN SBCES.RBLISTOFBUSES rbl ON rbm.RebuiltStockNum = rbl.RebuiltStockNum

LEFT JOIN SBCES.LISTOFBUSES lb ON rbl.ListId = lb.ListId";

var rebuiltPartsDict = new Dictionary<string, RebuiltPartsViewModel>();

var data = await \_dbConnection.QueryAsync<RebuiltPartsViewModel, BusesModel, RebuiltPartsViewModel>(

query,

(part, bus) =>

{

if (!rebuiltPartsDict.TryGetValue(part.RebuiltStockNum, out var rebuiltPart))

{

rebuiltPart = part;

rebuiltPartsDict.Add(rebuiltPart.RebuiltStockNum, rebuiltPart);

}

if (bus != null)

{

rebuiltPart.Buses.Add(bus);

}

return rebuiltPart;

},

splitOn: "ListId"

);

// Convert Buses to a list of ListId values for binding

foreach (var part in rebuiltPartsDict.Values)

{

part.Buses = part.Buses.Select(b => new BusesModel { ListId = b.ListId, Description = b.Description }).ToList();

}

return Json(rebuiltPartsDict.Values.ToDataSourceResult(request));

}

catch (Exception ex)

{

return Json(new DataSourceResult { Errors = "Error occurred while fetching rebuilt parts." });

}

}

**4. Fix Index.cshtml to Use the Correct MultiSelect Editor**

The grid column for Buses must properly use the editor template (VehicleSeriesEditor) and bind to the Buses field.

html

CopyEdit

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

.Name("RebuiltPartsGrid")

.Columns(columns =>

{

columns.Bound(p => p.RebuiltStockNum).Title("Rebuilt Stock #").Width(150).Editable(false); // PK, not editable

columns.Bound(p => p.Keyword).Title("Keyword").Width(200);

// MultiSelect column for Buses

columns.Bound(p => p.Buses)

.ClientTemplate("#= Buses.map(function(bus) { return bus.Description; }).join(', ') #") // Display descriptions

.EditorTemplateName("VehicleSeriesEditor") // Use custom MultiSelect editor

.Title("Buses").Width(300);

columns.Command(command =>

{

command.Edit(); // Inline Edit Command

command.Destroy(); // Delete button

}).Title("Actions").Width(150);

})

.ToolBar(toolbar => toolbar.Create()) // Create button

.Editable(editable => editable.Mode(GridEditMode.InLine)) // Inline editing

.Pageable()

.Sortable()

.Filterable()

.DataSource(dataSource => dataSource

.Ajax()

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

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

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

.Destroy(destroy => destroy.Url(Url.Action("Delete", "RebuiltParts")).Type(HttpVerbs.Post)) // Delete action

.Model(model =>

{

model.Id(p => p.RebuiltStockNum); // Set the primary key

model.Field(p => p.Buses).DefaultValue(new List<int>()); // Default empty list for MultiSelect

})

)

)

**Summary of Fixes**

1. **MultiSelect Name**: Ensure .Name("Buses") matches the grid's bound field.
2. **Editor Binding**: The VehicleSeriesEditor must handle a list of selected ListId values (List<int>).
3. **Buses List**: Ensure ViewData["BusesList"] contains the full list of available buses (ListId and Description).
4. **Controller Adjustments**: Ensure the Buses property is returned as a list of ListId values for proper value binding in the MultiSelect editor.
5. **Default Values**: Set a default value (new List<int>()) for the Buses field in the grid model.