**Correct Approach**

1. **Ensure ListOfBusIds is Part of RebuiltPartsViewModel**:
   * The RebuiltPartsViewModel should have a ListOfBusIds property, which is already the case in your code:

csharp

Copy

public List<int?> ListOfBusIds { get; set; }

1. **Ensure the Form Data is Structured Correctly**:
   * When the form is submitted, the ListOfBusIds should be sent as part of the RebuiltPartsViewModel object.
   * For example, if you're using Telerik's inline editing, the grid should send the data in the following format:

json

Copy

{

"RebuiltStockNum": "123",

"MmsStockCode": "ABC",

"ListOfBusIds": [1, 2, 3],

// Other properties...

}

1. **Update the AddRebuiltPartsView Method**:
   * Remove the redundant listOfBusIds parameter and rely on the RebuiltPartsViewModel to contain the ListOfBusIds property.

csharp

Copy

[HttpPost]

[AllowAnonymous]

public async Task<IActionResult> AddRebuiltPartsView(

[DataSourceRequest] DataSourceRequest request,

[FromForm] RebuiltPartsViewModel rebuiltPartsViewModel) // Only this parameter is needed

{

try

{

if (rebuiltPartsViewModel == null)

{

return BadRequest("Invalid data.");

}

// Ensure ListOfBus is initialized (if needed)

rebuiltPartsViewModel.ListOfBus ??= new List<BusesModel>();

// Insert the rebuilt part into the database

var insertPartSql = @"

INSERT INTO SBCES.RbMasterlist

(

RebuiltStockNum, MmsStockCode, Keyword, JobNumber, CorePartNum,

DetailedDesc, CoreCharge, EstimatedCost, BuyNewCost, RemanCost,

ExternalCost, LastModifiedBy, DateEntered, IsActive

)

VALUES

(

@RebuiltStockNum, @MmsStockCode, @Keyword, @JobNumber, @CorePartNum,

@DetailedDesc, @CoreCharge, @EstimatedCost, @BuyNewCost, @RemanCost,

@ExternalCost, @LastModifiedBy, GETDATE(), @IsActive

)";

await \_dbConnection.ExecuteAsync(insertPartSql, new

{

rebuiltPartsViewModel.RebuiltStockNum,

rebuiltPartsViewModel.MmsStockCode,

rebuiltPartsViewModel.Keyword,

rebuiltPartsViewModel.JobNumber,

rebuiltPartsViewModel.CorePartNum,

rebuiltPartsViewModel.DetailedDesc,

rebuiltPartsViewModel.CoreCharge,

rebuiltPartsViewModel.EstimatedCost,

rebuiltPartsViewModel.BuyNewCost,

rebuiltPartsViewModel.RemanCost,

ExternalCost = "", // Adjust as needed

LastModifiedBy = User.Identity.Name, // Or your logic

IsActive = rebuiltPartsViewModel.IsActive ? 1 : 0

});

// Insert associated buses

if (rebuiltPartsViewModel.ListOfBusIds?.Any() == true)

{

var insertBusesSql = @"

INSERT INTO SBCES.RbListOfBuses (RebuiltStockNum, ListId)

VALUES (@RebuiltStockNum, @ListId)";

foreach (var listId in rebuiltPartsViewModel.ListOfBusIds.Where(id => id.HasValue))

{

await \_dbConnection.ExecuteAsync(insertBusesSql, new

{

rebuiltPartsViewModel.RebuiltStockNum,

ListId = listId.Value

});

}

}

return Json(new { success = true });

}

catch (Exception ex)

{

return StatusCode(500, $"Internal server error: {ex.Message}");

}

}

**Verify Client-Side Form Data**

Ensure the client-side form data is structured correctly. For example, if you're using Telerik's Kendo UI Grid with inline editing, the ListOfBusIds should be included in the data sent to the server.

**Example of Client-Side Data Submission**

When the user clicks "Save," the grid should send data in the following format:

json

Copy

{

"RebuiltStockNum": "123",

"MmsStockCode": "ABC",

"ListOfBusIds": [1, 2, 3],

// Other properties...

}

**Ensure the Grid Configuration is Correct**

In your Index.cshtml, ensure the grid's dataSource is configured to include ListOfBusIds:

csharp

Copy

.DataSource(dataSource => dataSource

.Ajax()

.Model(model =>

{

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

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

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

model.Field(c => c.ListOfBusIds).DefaultValue(new List<int?>()); // Ensure this is included

})

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

)

**Debugging Tips**

1. **Check the Request Payload**:
   * Use browser developer tools (e.g., Chrome DevTools) to inspect the network request when the form is submitted. Verify that ListOfBusIds is included in the payload.
2. **Log the Model State**:
   * Add logging to check if the rebuiltPartsViewModel is being populated correctly:

csharp

Copy

if (rebuiltPartsViewModel == null)

{

\_logger.LogError("RebuiltPartsViewModel is null.");

return BadRequest("Invalid data.");

}

\_logger.LogInformation($"ListOfBusIds: {string.Join(", ", rebuiltPartsViewModel.ListOfBusIds)}");

1. **Test with a Simple Example**:
   * Create a simple form to test the binding of ListOfBusIds:

html

Copy

<form method="post" action="/RebuiltParts/AddRebuiltPartsView">

<input type="text" name="RebuiltStockNum" value="123" />

<input type="text" name="MmsStockCode" value="ABC" />

<input type="text" name="ListOfBusIds" value="1,2,3" />

<button type="submit">Submit</button>

</form>

Run HTML

**Conclusion**

You do not need the [FromForm(Name = "ListOfBusIds")] string listOfBusIds parameter if ListOfBusIds is already part of the RebuiltPartsViewModel. Ensure the form data is structured correctly and the ListOfBusIds property is included in the payload. If the issue persists, inspect the request payload and verify the client-side configuration.