To implement a full grid model in an ASP.NET Core application using Telerik's Grid component, follow these steps: ### 1. Define Models You will typically have a Product model and a Category model.

csharp

public class Category

{

public int Id { get; set; }

public string Name { get; set; }

}

public class Product

{

public int Id { get; set; }

public string Name { get; set; }

public List CategoryIds { get; set; } // Foreign key to categories

}

### 2. Create a ViewModel for the Grid To simplify binding between your models and the grid, you can create a ViewModel.

csharp

public class ProductViewModel

{

public int Id { get; set; }

public string Name { get; set; }

public List CategoryIds { get; set; }

public List Categories { get; set; } // For populating MultiSelect

}

### 3. Implement Your Controller In your controller, implement actions to provide data to the grid.

csharp

using Microsoft.AspNetCore.Mvc;

using Dapper;

using System.Data.SqlClient;

using System.Collections.Generic;

using System.Linq;

public class ProductsController : Controller

{

private readonly string \_connectionString = "YourConnectionString"; // Update with your connection string

// Action to fetch products along with categories

public IActionResult Index()

{

var products = GetProducts();

var categories = GetCategories();

var productViewModels = products.Select(p => new ProductViewModel

{

Id = p.Id,

Name = p.Name,

CategoryIds = p.CategoryIds,

Categories = categories // Pass categories to ViewModel

}).ToList();

return View(productViewModels);

}

private IEnumerable GetProducts()

{

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

{

return connection.Query("SELECT Id, Name, CategoryIds FROM Products").ToList();

}

}

private IEnumerable GetCategories()

{

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

{

return connection.Query("SELECT Id, Name FROM Categories").ToList();

}

}

}

### 4. Implement the View (Index.cshtml) Bind the data to the Telerik Grid in your view.

@model List<ProductViewModel>

@{

ViewData["Title"] = "Products";

}

<h2>Products</h2>

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

.Name("grid")

.Columns(columns =>

{

columns.Bound(p => p.Id).Title("ID");

columns.Bound(p => p.Name).Title("Name");

columns.Template(p => {

@\* MultiSelect implementation \*@

@Html.Kendo().MultiSelect()

.Name("CategoryIds\_" + p.Id)

.DataTextField("Name")

.DataValueField("Id")

.BindTo(p.Categories)

.Value(string.Join(",", p.CategoryIds)) // Set the value

.HtmlAttributes(new { style = "width: 100%" })

).Title("Categories");

});

})

.Pageable()

.Sortable()

.Filterable()

.DataSource(dataSource => dataSource

.Ajax()

.Read(read => read.Action("Index", "Products"))

)

)