Great question! The scenario you're describing involves using the ClientDetailTemplateId in a Telerik Kendo Grid to load dynamic content into the grid's detail template. To achieve this, you can utilize AJAX to dynamically load the contents of your partial views (PartialTest1.cshtml and PartialTest2.cshtml) into the template.

Here’s a step-by-step guide:

**1. Define the Grid**

Your Kendo Grid should specify the ClientDetailTemplateId for the detail template. We'll use a <script> block with a placeholder to dynamically load the partial view content:

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

.Name("RebuiltPartsGrid")

.Columns(columns =>

{

columns.Bound(c => c.PartName);

columns.Bound(c => c.PartNumber);

})

.DetailTemplateId("template")

.DataSource(dataSource => dataSource

.Ajax()

.Read(read => read.Action("GetRebuiltParts", "Home"))

)

)

<script id="template" type="text/x-kendo-template">

<div id="detail-template-#=PartId#"></div>

</script>

**2. Load Partial Views with JavaScript**

Using the dataBound event of the Kendo Grid, you can hook into when the detail row is expanded and then use AJAX to load the content from your partial views into the placeholder (<div>).

Here’s the JavaScript code:

<script>

$(document).ready(function () {

var grid = $("#RebuiltPartsGrid").data("kendoGrid");

grid.bind("detailInit", function (e) {

// Get the container for the detail template

var container = e.detailCell.find("#detail-template-" + e.data.PartId);

// Choose which partial to load based on some condition

var url = e.data.SomeCondition ? '/Home/PartialTest1' : '/Home/PartialTest2';

// Load the partial view via AJAX

container.load(url);

});

});

</script>

**3. Create Your Controller Actions**

Make sure you have actions in your controller to serve the partial views:

public IActionResult PartialTest1()

{

return PartialView("\_PartialTest1");

}

public IActionResult PartialTest2()

{

return PartialView("\_PartialTest2");

}

**4. Create the Partial Views**

Define the content of your partial views (\_PartialTest1.cshtml and \_PartialTest2.cshtml) as required:

**\_PartialTest1.cshtml**

<div>

<h3>Partial Test 1</h3>

<p>Content from PartialTest1.</p>

</div>

**\_PartialTest2.cshtml**

<div>

<h3>Partial Test 2</h3>

<p>Content from PartialTest2.</p>

</div>

**How It Works:**

1. The ClientDetailTemplateId in the grid points to the <script> block for the detail template.
2. When a detail row is expanded, the detailInit event is triggered.
3. Inside detailInit, you dynamically load either PartialTest1.cshtml or PartialTest2.cshtml based on some condition from the row data.
4. The AJAX load method fetches and injects the content of the partial views into the detail template.

**Tips:**

* Use the model property (e.g., SomeCondition) to decide which partial to load dynamically.
* Ensure your partial views are lightweight for faster loading.
* Consider caching partial view content for better performance.

This approach keeps your code modular and ensures your grid stays dynamic and interactive. Let me know if you need more help!