**Fix getAutoCompleteParams Function (in LabourTask.cshtml)**

javascript

Copy

function getAutoCompleteParams() {

// Find the focused AutoComplete input element

var input = $("input[data-role='autocomplete']:focus");

if (!input.length) return { columnName: "", searchText: "" }; // Fallback

// Get column name from input's name attribute

var columnName = input.attr("name");

// Get current typed value

var searchText = input.val();

return {

columnName: columnName,

searchText: searchText

};

}

**2. Update AutoComplete Editor Template (AutocompleteEditor.cshtml)**

html

Copy

@model object

@{

var fieldName = ViewData.TemplateInfo.GetFullHtmlFieldName("");

}

@(Html.Kendo().AutoComplete()

.Name(fieldName)

.DataTextField("this")

.Filter("contains")

.Placeholder("Type to search...")

.HtmlAttributes(new {

name = fieldName, // Ensure name matches model property

"data-role" = "autocomplete" // Required for Kendo recognition

})

.DataSource(source =>

{

source.Read(read =>

{

read.Url(Url.Action("GetAutoCompleteData", "AutoComplete"))

.Type(HttpVerbs.Get)

.Data("getAutoCompleteParams");

})

.ServerFiltering(true);

})

)

Run HTML

**3. Verify Controller Action (AutoCompleteController.cs)**

csharp

Copy

[HttpGet]

public IActionResult GetAutoCompleteData(string columnName, string searchText)

{

// 1. Validate parameters

if (string.IsNullOrEmpty(columnName))

return BadRequest("Column name is required.");

// 2. Column whitelist

var columnMappings = new Dictionary<string, (string SqlColumn, string Table)>

{

{ "TaskDescription", ("TaskDescription", "[SBCES].[LabourTaskDescriptions]") }

};

if (!columnMappings.TryGetValue(columnName, out var mapping))

return BadRequest("Invalid column name.");

// 3. Parameterized query

using (var connection = \_dbConnection)

{

var sql = $@"

SELECT DISTINCT {mapping.SqlColumn}

FROM {mapping.Table}

WHERE {mapping.SqlColumn} LIKE @SearchText + '%'";

var results = connection.Query<string>(sql, new { SearchText = searchText ?? "" });

return Json(results.ToList()); // Return as JSON array

}

}

**4. Testing Steps**

**A. Network Request Verification (Chrome DevTools)**

1. Open Chrome DevTools (**F12** → **Network** tab).
2. Edit a grid row and type "ad" in the TaskDescription field.
3. Verify the request URL matches:

Copy

https://localhost:44332/AutoComplete/GetAutoCompleteData?columnName=TaskDescription&searchText=ad

* + ✅ Correct parameters: columnName and searchText
  + ❌ If you see filter[filters][0][value]=ad, the parameters are incorrect

**B. Console Error Check**

* Ensure no errors like:
  + Uncaught TypeError: Cannot read properties of undefined
  + 400 (Bad Request)

**C. Widget Initialization Check**

* After opening the edit form, check the console for:

Copy

"AutoComplete initialized for: TaskDescription"

**Key Fixes Explained**

1. **Widget Identification**:
   * data-role="autocomplete" ensures Kendo recognizes the input as an AutoComplete
   * name="TaskDescription" links the input to your model property
2. **Parameter Binding**:
   * getAutoCompleteParams now reliably gets the focused AutoComplete's name/value
   * Eliminated dependency on undefined e.sender
3. **Security**:
   * Column whitelisting (columnMappings) prevents SQL injection
   * Parameterized queries via Dapper