AGrid

Powerful Data Grids for Salesforce

Dev Component

Configuration Manual

PRODUCT	FEATURE
AGrid	**Dev Component**
VERSION	**DOCUMENT TYPE**
1.0	Feature Configuration Manual

Table of Contents

Auto-updating Table of Contents

Right-click below and select "Update Field" in Microsoft Word to regenerate this list automatically when sections change.

1. Introduction

1.1 Overview

AGrid Dev Component is a Lightning Web Component intended for Lightning Web Component developers to display data in an AGrid Table. To leverage this table, the developer prepares JSON data and provides it as input to the component.

This manual provides a complete reference for configuring and integrating the Dev Component into custom Lightning Web Components, including configuration properties, column definitions, data structures, related-record handling, status management, and the full event model.

1.2 Component Reference

Component Name	`devTable`
Namespace	`agrid`
Component Type	Lightning Web Component (LWC)

ScreenShot:

```
<agrid-dev-table
    data={data}
    columns={columns}
    configuration-properties={configurationProperties}
    related-records={relatedRecords}
    status={status}
    onloadmore={handleLoadMore}
    onsearch={handleSearch}
    onfilter={handleFilter}
    onclearfilter={handleClearFilter}
    onsort={handleSort}
    ondelete={handleDelete}
    oncreate={handleCreate}
    onshowrelatedrecords={handleShowRelatedRecords}
    onsave={handleSave}
    onrefresh={handleRefresh}>
</agrid-dev-table>
```

1.3 Supported Record Types

The Dev Component supports rendering both SObject and non-SObject records. SObject records have a unique Id field. Likewise, when rendering non-SObject records, each record must also contain a unique field named Id.

> **NOTE**
> Both SObject and non-SObject records require a unique `Id` field to be present on every record. This is used internally for record tracking, selection, and update operations.

1.4 Component Attributes

AGrid Dev Component exposes the following attributes for configuration:

- **Configuration Properties** — global grid behavior, styling, sorting, and list-level actions
- **Columns** — column definitions including field metadata, formatting, and row actions
- **Data** — the array of records to display in the table
- **Related Records** — list of records shown in lookup dropdowns
- **Status** — result of user actions (save, delete, create, etc.) sent back to the grid

Each attribute is documented in detail in the sections that follow.

2. Configuration Properties

Attribute name: `configuration-properties`

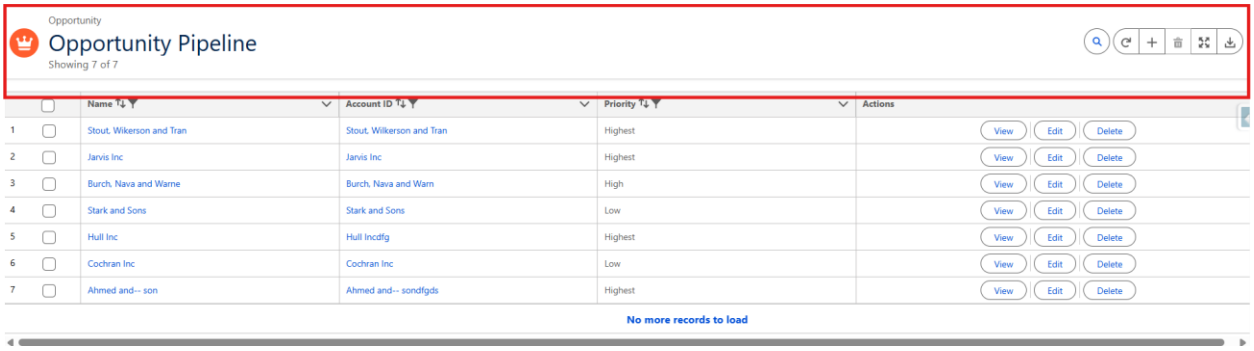
Configuration properties define the global behavior, appearance, and functional capabilities of the AGrid table. They are passed as a JSON object to the component.

2.1 Sample Configuration

```
{
  "objectName": "Opportunity",
  "objectIconName": "standard:opportunity",
  "title": "Opportunity Pipeline",
  "isSObject": true,
  "defaultLoadsizeDesktop": 25,
  "columnFreezeCount": 2,
  "sortFieldLevel": 3,
  "hideSelectionCheckbox": false,
  "hideTableHeader": false,
  "totalRecordsCount": 10000,
  "saveRecordsLocally": false,
  "gridStyle": {
    "rowBorderThickness": 1,
    "columnBorderThickness": 1,
    "gridFontSize": "small",
    "tableHeight": 360
  },
  "sortFields": [
    {
      "fieldName": "CreatedDate",
      "sortDirection": "ASC",
      "order": "1"
    },
    {
      "fieldName": "Stage",
      "sortDirection": "ASC",
      "order": "2"
    }
  ],
  "listActions": [
    {
      "showAction": true,
      "action": "standard_action",
      "actionType": "Icon",
      "iconName": "utility:add",
      "isBulkCreate": true,
      "name": "Create"
    },
    {
      "showAction": true,
      "action": "standard_action",
```

```
      "actionType": "Icon",
      "iconName": "utility:delete",
      "name": "Delete"
    },
    {
      "showAction": true,
      "action": "standard_action",
      "actionType": "Icon",
      "iconName": "utility:download",
      "name": "Export"
    },
    {
      "showAction": true,
      "action": "standard_action",
      "actionType": "Icon",
      "iconName": "utility:expand",
      "name": "Focus Mode"
    }
  ]
}
```

Configuration Properties Screenshot:



2.2 Top-Level Properties

Attribute	Type	Default	Description
`objectName`	*String*	–	Specifies the object name for the list view.
`objectIconName`	*String*	–	Defines which icon should be displayed before the list view title.
`title`	*String*	–	Specifies the name of the list view.
`isSObject`	*Boolean*	–	Set to true when records are Salesforce records; false for non-Salesforce records.

defaultLoadsizeDesktop	*Number*	25	Number of records to load when the user scrolls to the end of the table.
columnFreezeCount	*Number*	0	Number of columns to freeze. Maximum of 3 columns can be frozen.
sortFieldLevel	*Number*	1	Number of columns that can be sorted simultaneously. Maximum of 3.
hideSelectionCheckbox	*Boolean*	false	Hides the record selection checkbox when set to true.
hideTableHeader	*Boolean*	false	Hide the table header when set to true.
totalRecordsCount	*Number*	0	Total number of records available for the object. Used for record count display.
saveRecordsLocally	*Boolean*	false	When enabled, edited records are saved within the list view; no save events are fired.
gridStyle	*JSON*	–	Object controlling visual styling of the grid. See section 2.3.
sortFields	*Array*	[ ]	Array of fields used to sort records. Maximum 3 fields. See section 2.4.
listActions	*Array*	[ ]	Array of standard list-level actions. See section 2.5.

2.3 gridStyle Object

The gridStyle property controls the visual appearance of the grid.

Attribute	Type	Default	Description
rowBorderThickness	*Number*	1	Border thickness for rows. Allowed values: 0–5.
columnBorderThickness	*Number*	1	Border thickness for columns. Allowed values: 0–5.
gridFontSize	*String*	small	Font size for the list view. Allowed values: Small, Medium, Large.
tableHeight	*Number*	360	Height of the table in pixels.

2.4 sortFields Array

Each entry in the sortFields array is a JSON object describing a sort rule. A maximum of 3 sort rules is supported.

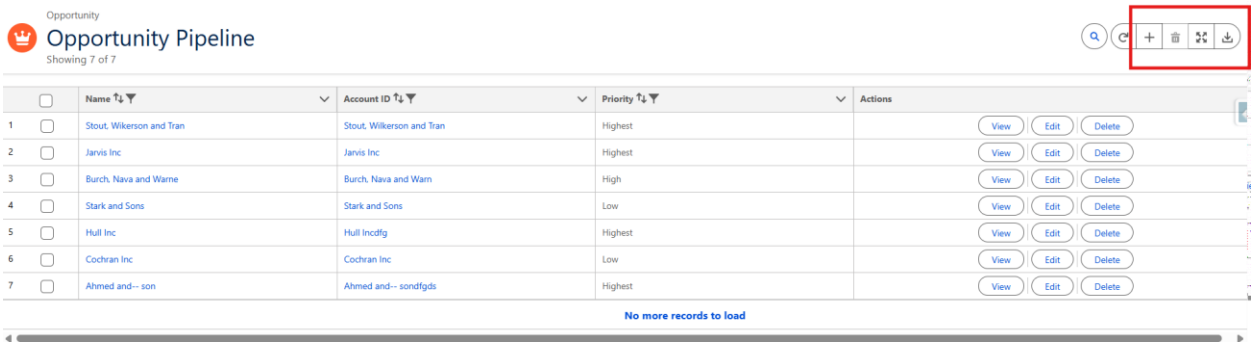
Attribute	Type	Default	Description
`fieldName`	*String*	–	Name of the field to be sorted.
`sortDirection`	*String*	ASC	Sort direction. Allowed values: ASC, DESC.
`order`	*Number*	–	Order in which this sort is applied (1, 2, or 3).

2.5 listActions Array

The Dev Component supports the following four standard list-level actions. Custom actions are not supported.

- Create
- Delete
- Export
- Focus Mode

List Action ScreenShot:



Each action is provided as a JSON object with the following attributes:

Attribute	Type	Default	Description
`showAction`	*Boolean*	`true`	Controls whether the action is shown or hidden in the list view.
`name`	*String*	–	Name of the action. Must be one of: Create, Delete, Export, Focus Mode.

`action`	*String*	–	For standard list actions, the value must be standard_action.
`actionType`	*String*	–	How the action is rendered. Allowed values: Button, Icon.
`iconName`	*String*	–	Required when actionType is Icon. Not required when actionType is Button.
`isBulkCreate`	*Boolean*	`false`	Applies only to the Create action. When true, supports bulk record creation; when false, opens the standard create modal for a single record. This attribute is ignored for other actions.

BEHAVIOR NOTE

For the Create action on SObject records, when **`isBulkCreate`** is false, the standard Salesforce create modal is opened. When the user clicks Save, the record is created and **no event is fired**.

3. Columns

Attribute name: `columns`

The columns attribute is an array of column definitions that controls what is displayed in each column of the grid, as well as column behavior such as sorting, filtering, and row actions.

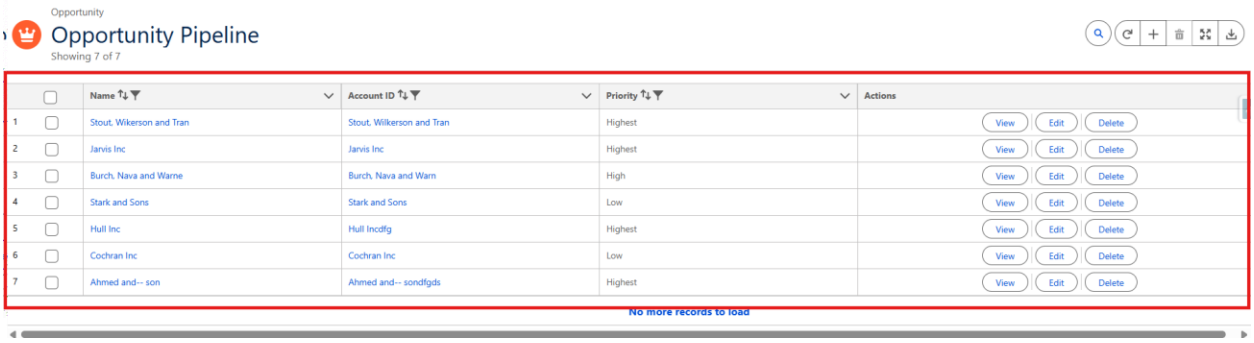
3.1 Sample Columns Configuration


```
[
  {
    "label": "Name",
    "columnType": "FIELD",
    "fieldName": "Name",
    "fieldType": "STRING",
    "order": 1,
    "required": false,
    "editable": false,
    "createable": false,
    "sortable": false,
    "searchable": false,
    "filterable": false,
    "referenceFieldNavigation": "Modal Window",
    "isNameField": true,
    "isRichTextArea": false,
    "cellStyles": {
      "width": 100,
      "textWrap": "Clip_Text"
    }
  },
  {
    "label": "Amount",
    "columnType": "FIELD",
    "fieldName": "Amount",
    "fieldType": "Currency",
    "order": 2,
    "required": false,
    "editable": false,
    "createable": false,
    "sortable": false,
    "searchable": false,
    "filterable": false,
    "referenceFieldNavigation": "Modal Window",
    "isNameField": false,
    "isRichTextArea": false,
    "cellStyles": {
      "width": 100,
      "textWrap": "Clip_Text"
    },
    "summary": {
      "showSumValue": true,
```

```
      "showAverageValue": true,
      "showMinValue": true,
      "showMaxValue": true,
      "sum": 1000,
      "average": 1000,
      "min": 1,
      "max": 400
    }
  },
  {
    "label": "Priority",
    "columnType": "FIELD",
    "fieldName": "Priority__c",
    "fieldType": "PICKLIST",
    "order": 3,
    "required": false,
    "editable": false,
    "createable": false,
    "sortable": false,
    "searchable": false,
    "filterable": false,
    "referenceFieldNavigation": "Modal Window",
    "isNameField": false,
    "isRichTextArea": false,
    "picklistValues": [
      {
        "label": "Highest",
        "value": "Highest"
      },
      {
        "label": "High",
        "value": "High"
      },
      {
        "label": "Low",
        "value": "Low"
      }
    ],
    "cellStyles": {
      "width": 100,
      "textWrap": "Clip_Text"
    }
  },
  {
    "label": "Account Id",
    "columnType": "FIELD",
    "fieldName": "AccountId",
    "fieldType": "REFERENCE",
    "order": 4,
    "required": false,
    "editable": false,
    "createable": false,
    "sortable": false,
```

```
    "searchable": false,
    "filterable": false,
    "referenceField": "Account.Name",
    "referenceFieldNavigation": "Modal Window",
    "isNameField": false,
    "isRichTextArea": false,
    "cellStyles": {
      "width": 100,
      "textWrap": "Clip_Text"
    }
  },
  {
    "label": "Actions",
    "columnType": "ACTION",
    "order": 5,
    "actions": [
      {
        "showAction": true,
        "action": "standard_action",
        "actionType": "Button",
        "name": "View"
      },
      {
        "showAction": true,
        "action": "standard_action",
        "actionType": "Button",
        "name": "Edit"
      },
      {
        "showAction": true,
        "action": "standard_action",
        "actionType": "Button",
        "name": "Delete"
      }
    ]
  }
]
```

Column ScreenShot:



3.2 Column Attributes

Attribute	Type	Default	Description
`label`	*String*	–	Column label displayed in the table header.
`columnType`	*String*	`FIELD`	Type of column. Allowed values: FIELD, ACTION.
`fieldName`	*String*	–	API name of the field this column is bound to.
`fieldType`	*String*	–	Salesforce field data type (e.g., STRING, PICKLIST, REFERENCE).
`order`	*Number*	–	Order in which the column is rendered.
`required`	*Boolean*	`false`	When true, the field cannot be left empty during edit; an error is shown on save.
`editable`	*Boolean*	`false`	When true, the column supports inline editing.
`createable`	*Boolean*	`false`	When true, the column appears in the create screen.
`sortable`	*Boolean*	`false`	When true, users can sort records by this column.
`searchable`	*Boolean*	`false`	When true, the column is included in global search.
`filterable`	*Boolean*	`false`	When true, the column supports column-level filtering.
`referenceField`	*String*	–	Reference field path in the format relationshipName.fieldName.

referenceFieldNavigation	*String*	`Modal Window`	How to open the referenced record. Allowed values: Same Tab, New Tab, Modal Window.
isNameField	*Boolean*	`false`	When true, the field is rendered as a hyperlink to the record detail page.
isRichTextArea	*Boolean*	`false`	When true, the field is rendered with rich-text preview and edit support.
scale	*Number*	–	Number of decimal places to display for numeric fields.
picklistValues	*Array*	`[ ]`	Picklist options for inline edit and filtering.
actions	*Array*	`[ ]`	Row-level actions. Required when columnType is ACTION. See section 3.4.
cellStyles	*JSON*	–	Cell-level style configuration. See section 3.5.
summary	*JSON*	–	Header-level summary configuration. See section 3.6.

3.3 columnType Values

The Dev Component supports two column types:

- **FIELD** — bound to a specific field on the record. Use this for displaying record data.
- **ACTION** — renders a column of row-level action buttons (View, Edit, Delete).

3.4 Row Actions

The Dev Component supports the following three standard row-level actions. Custom row actions are not supported.

- View
- Edit
- Delete

Each row action is defined as a JSON object with these attributes:

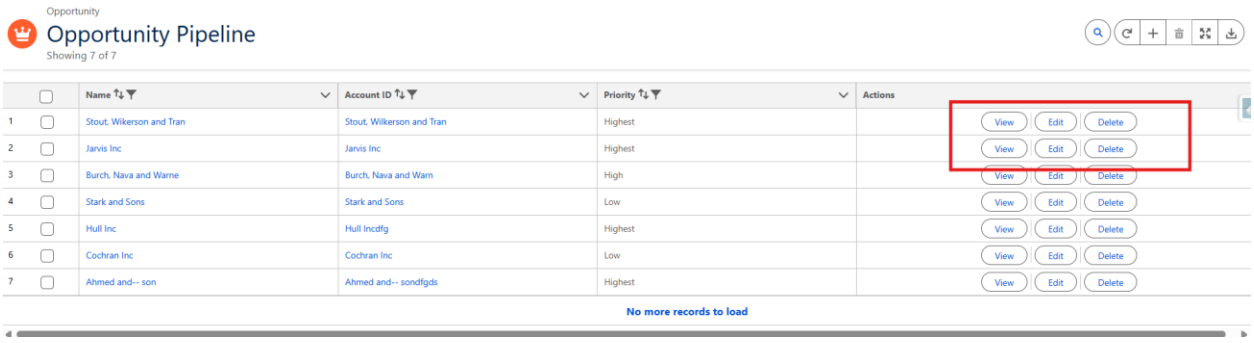
Attribute	Type	Default	Description
showAction	*Boolean*	`true`	Show or hide the action.

`name`	*String*	–	Action name. Must be one of: View, Edit, Delete.
`action`	*String*	–	For standard row actions, the value must be standard_action.
`actionType`	*String*	–	How the action is rendered. Allowed values: Link, Button, Icon.
`iconName`	*String*	–	Required when actionType is Icon.

BEHAVIOR NOTE

For the Edit action on SObject records, the standard Salesforce edit modal is shown. When the user clicks Save, the record is saved and **no event is fired**.

Row Action ScreenShot:



3.5 cellStyles Object

The cellStyles property controls the appearance of cells in a column.

Attribute	Type	Default	Description
`width`	*Number*	–	Width of the column in pixels.
`textWrap`	*String*	`Clip_Text`	Text wrapping behavior. Allowed values: Wrap_Text, Clip_Text.

3.6 summary Object

If the user wants to display summary values in the table header, they should be provided in JSON format with the following attributes.

Attribute	Type	Default	Description

showSumValue	*Boolean*	`false`	Display the sum value in the column header.
showAverageValue	*Boolean*	`false`	Display the average value in the column header.
showMinValue	*Boolean*	`false`	Display the minimum value in the column header.
showMaxValue	*Boolean*	`false`	Display the maximum value in the column header.
sum	*Number*	–	Sum value of the column.
average	*Number*	–	Average value of the column.
min	*Number*	–	Minimum value of the column.
max	*Number*	–	Maximum value of the column.

Note on referenceField: If the relationship name is `Account` and the related Name field is `Name`, then `referenceField` must be set to `Account.Name`.

4. Data

Attribute name: `data`

The data attribute contains the array of records to display in the table.

If a column references a relationship field, the referenceField path defined in the column wrapper should be queried and included in the record.

To render non-Salesforce records in the table, each record must contain an Id attribute with a unique value.

4.1 Sample Data

```
[
    {
        "attributes": {
            "type": "Opportunity",
            "url": "/services/data/v66.0/sobjects/Opportunity/006Ec00000WAqMHIA1"
        },
        "Id": "006Ec00000WAqMHIA1",
        "Name": "Mavis Lights",
        "AccountId": "001Ec00001cpIbTIAU",
        "Account": {
            "attributes": {
                "type": "Account",
                "url": "/services/data/v66.0/sobjects/Account/001Ec00001cpIbTIAU"
            },
            "Name": "Frye, Compton and Chung",
            "Id": "001Ec00001cpIbTIAU"
        }
    },
    {
        "attributes": {
            "type": "Opportunity",
            "url": "/services/data/v66.0/sobjects/Opportunity/006Ec00000WAqMHIB2"
        },
        "Id": "006Ec00000WAqMHIB2",
        "Name": "Cortes Islands",
        "AccountId": "001Ec00001cpIbTIAU",
        "Account": {
            "attributes": {
                "type": "Account",
                "url": "/services/data/v66.0/sobjects/Account/001Ec00001cpIbTIAU"
            },
            "Name": "Frye, Compton and Chung",
            "Id": "001Ec00001cpIbTIAU"
        }
    }
]
```

5. Related Records

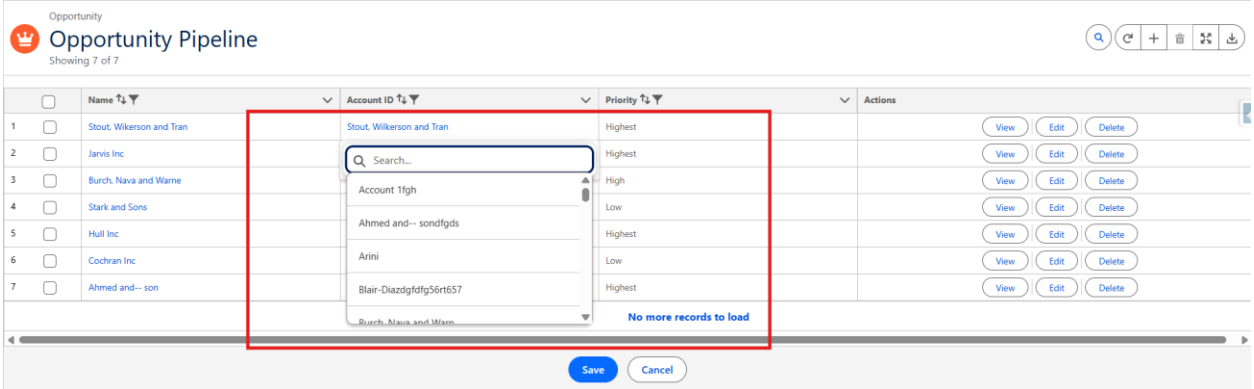
Attribute name: `related-records`

The related-records attribute provides the list of records to display in lookup dropdowns when the user clicks a reference field input.

5.1 Attributes

Attribute	Type	Default	Description
`label`	*String*	–	Display label, typically the record name.
`value`	*String*	–	Record Id used as the lookup value.

Related Records ScreenShot:



5.2 Sample Data

```
[
  {
    "label": "Sample Account for Entitlements",
    "value": "001Ru000017MnhfIAC"
  },
  {
    "label": "Teklist",
    "value": "001Ru000017YWqXIAW"
  },
  {
    "label": "Yakidoo",
    "value": "001Ru000017YWq0IAG"
  },
  {
    "label": "Twitterbridge",
```

```
    "value": "001Ru000017YWqtIAG"
  },
  {
    "label": "Fiveclub",
    "value": "001Ru000017YWr5IAG"
  },
  {
    "label": "Buzzster",
    "value": "001Ru000017YWrBIAW"
  }
]
```

6. Status

Attribute name: `status`

The status attribute is used by the consumer of the Dev Component to communicate the result of user actions (such as save, delete, create, refresh, search, sort, filter) back to the grid. The grid uses this object to update its internal state accordingly.

6.1 Sample Status Object

```
{
    "isSuccess": true,
    "successRecords": [],
    "records": [],
    "deletedRecordIds": [],
    "errors": {
        "001Ru000017YWqtIAG": "Record Update failed due to invalid value"
    },
    "action": "save",
    "toastMessage": "Records saved successfully."
}
```

6.2 Status Attributes

Attribute	Type	Default	Description
isSuccess	*Boolean*	–	True if the user action succeeded; false otherwise.
action	*String*	–	Action that was performed. Must match the corresponding event name (e.g., save, delete, create, refresh, search, sort, filter, clearfilter, loadmore).
successRecords	*Array*	[ ]	Records that were updated successfully (used after a Save action).
deletedRecordIds	*Array*	[ ]	Record Ids that were deleted successfully (used after a Delete action).
errors	*JSON*	{ }	Map of record Id to error message for failed records.
records	*Array*	[ ]	Updated record list. Used for Refresh, Search, Filter, ClearFilter, Sort, Create, and Load More actions.
toastMessage	*String*	–	Message displayed in the success or error toast.

7. Events

The Dev Component emits a set of events that the consumer must handle in order to drive the grid behavior. After handling each event, the consumer should update the status attribute and, where applicable, totalRecordsCount in configuration-properties.

Events ScreenShot:

7.1 Event Summary

Event	Dispatch Name	Description
Load More	`onloadmore`	Fired when the table scroll reaches the end.
Search	`onsearch`	Fired when the user enters a value in the global search box.
Filter	`onfilter`	Fired when the user applies a column-level filter.
Clear Filter	`onclearfilter`	Fired when the user clears a column-level filter.
Sort	`onsort`	Fired when the user sorts the records.
Delete	`ondelete`	Fired when the user deletes records (row or list action).
Create	`oncreate`	Fired when the user saves new records (bulk create only).

Show Related Records	`onshowrelatedrecords`	Fired when the user clicks a lookup input.
Save	`onsave`	Fired when the user clicks Save after inline editing.
Refresh	`onrefresh`	Fired when the user clicks the refresh icon.

7.2 Load More Event

Dispatch name: `onloadmore`

Fired when the table scroll reaches the end. The event has no detail payload.

After receiving the event, the consumer should fetch and provide the next set of records via the records attribute on the status object. Then update totalRecordsCount in configuration-properties.

Sample Status Update

```
{
    "isSuccess": true,
    "records": [
        {
            "attributes": { "type": "Opportunity", "url": "..." },
            "Id": "006Ec00000WAqMHIA1",
            "Name": "Mavis Lights",
            "AccountId": "001Ec00001cpIbTIAU"
        },
        {
            "attributes": { "type": "Opportunity", "url": "..." },
            "Id": "006Ec00000WAqMHIB2",
            "Name": "Cortes Islands",
            "AccountId": "001Ec00001cpIbTIAU"
        }
    ],
    "action": "loadmore"
}
```

7.3 Search Event

Dispatch name: `onsearch`

Fired when the user enters a value in the global search box.

Event Structure

```
detail: {
    searchValue: "Search Text"
}
```

Event Attributes

Attribute	Type	Default	Description
`searchValue`	*String*	–	Search string entered by the user in the global search box.

After performing the search, update the status object with the matching records and update totalRecordsCount.

```
{
    "isSuccess": true,
    "records": [ /* matching records */ ],
    "action": "search"
}
```

7.4 Filter Event

Dispatch name: `onfilter`

Fired when the user applies a column-level filter.

Event Structure

```
detail: {
    fieldName: "Name",
    operator:  "=",
    value:     "AGrid",
    fromValue: null,
    toValue:   null
}
```

Event Attributes

Attribute	Type	Default	Description
`fieldName`	*String*	–	API name of the field on which the filter is applied.
`operator`	*String*	–	Selected filter operator.
`value`	*Any*	–	Filter value entered by the user.
`fromValue`	*Any*	`null`	From value, set when the operator is Between or Not Between.
`toValue`	*Any*	`null`	To value, set when the operator is Between or Not Between.

Use these attributes to filter the records, then update the status object with the filtered result and update totalRecordsCount.

```
{
    "isSuccess": true,
    "records": [ /* filtered records */ ],
    "action": "filter"
}
```

7.5 Clear Filter Event

Dispatch name: `onclearfilter`

Fired when the user clears a column-level filter.

Event Structure

```
detail: {
    fieldName: "Name"
}
```

Event Attributes

Attribute	Type	Default	Description
`fieldName`	*String*	–	API name of the field whose filter was cleared.

Use fieldName to remove the corresponding filter, re-fetch the records, and update the status object.

```
{
    "isSuccess": true,
    "records": [ /* unfiltered records */ ],
    "action": "clearfilter"
}
```

7.6 Sort Event

Dispatch name: `onsort`

Fired when the user sorts the records. The event detail contains an array of sort objects.

Event Structure

```
detail: [
    { fieldName: 'Name',        sortDirection: 'ASC', order: 1 },
    { fieldName: 'CreatedDate', sortDirection: 'ASC', order: 2 }
```

```
]
```

Event Attributes

Attribute	Type	Default	Description
`fieldName`	*String*	–	API name of the field being sorted.
`sortDirection`	*String*	ASC	Sort direction. Allowed values: ASC, DESC.
`order`	*Number*	–	Order in which this sort rule is applied.

After sorting the records, update the records attribute on the status object and update both totalRecordsCount and sortFields in configuration-properties.

```
{
    "isSuccess": true,
    "records": [ /* sorted records */ ],
    "action": "sort"
}
```

7.7 Delete Event

Dispatch name: `ondelete`

Fired when the user deletes records. This event is dispatched for both row-level and list-level Delete actions.

Event Structure

```
detail: {
    recordIds: [ '001Ec00001cpIbTIAU', '001Ec00001cpIbTIAV' ]
}
```

Event Attributes

Attribute	Type	Default	Description
`recordIds`	*Array*	[ ]	List of record Ids to delete.

After deleting the records, update the status object with the deleted record Ids:

```
{
    "isSuccess": true,
    "deletedRecordIds": [ '001Ec00001cpIbTIAU', '001Ec00001cpIbTIAV' ],
```

```
    "action": "delete",
    "toastMessage": "Records deleted successfully."
}
```

If deletion fails, return error details and a failure toast:

```
{
    "isSuccess": false,
    "deletedRecordIds": [],
    "action": "delete",
    "errors": {
        "001Ru000017YWqtIAG": "The user does not have delete permission."
    },
    "toastMessage": "Record deletion failed."
}
```

7.8 Create Event

Dispatch name: `oncreate`

> **IMPORTANT**
>
> The **`oncreate`** event is dispatched only for **bulk create**. For single-record creation on SObject records, the standard Salesforce create modal handles persistence and no event is fired.

Fired when the user saves new records during bulk create.

Event Structure

```
detail: {
    records: [
        {
            "attributes": { "type": "Opportunity", "url": "..." },
            "Id": "006Ec00000WAqMHIA1",
            "Name": "Mavis Lights",
            "AccountId": "001Ec00001cpIbTIAU"
        },
        {
            "attributes": { "type": "Opportunity", "url": "..." },
            "Id": "006Ec00000WAqMHIB2",
            "Name": "Cortes Islands",
            "AccountId": "001Ec00001cpIbTIAU"
        }
    ]
}
```

Event Attributes

Attribute	Type	Default	Description
records	*Array*	[ ]	List of new records to create.

After creating the records successfully, the consumer should re-fetch the data and provide the updated records.

```
{
    "isSuccess": true,
    "records": [ /* updated list of records */ ],
    "action": "create",
    "toastMessage": "Records created successfully."
}
```

If creation fails, return a failure toast:

```
{
    "isSuccess": false,
    "action": "create",
    "toastMessage": "Records creation failed."
}
```

7.9 Show Related Records Event

Dispatch name: `onshowrelatedrecords`

Fired when the user clicks a lookup input to select a related record.

Event Structure

```
detail: {
    objectName: 'Contact',
    fieldName:  'AccountId',
    searchValue: 'Search Text',
}
```

Event Attributes

Attribute	Type	Default	Description
objectName	*String*	–	Name of the object to query for related records.
fieldName	*String*	–	API name of the reference field.
searchValue	*String*	–	Search string entered by the user in the lookup search box.

Use objectName and fieldName to query the related records and use searchValue to filter those records, then pass the resulting list to the `related-records` attribute in the format described in section 5.

7.10 Save Event

Dispatch name: `onsave`

Fired when the user clicks Save after inline editing.

Event Structure

```
detail: {
    changedRecords: [
        {
            "attributes": { "type": "Opportunity", "url": "..." },
            "Id": "006Ec00000WAqMHIA1",
            "Name": "Mavis Lights",
            "AccountId": "001Ec00001cpIbTIAU"
        },
        {
            "attributes": { "type": "Opportunity", "url": "..." },
            "Id": "006Ec00000WAqMHIB2",
            "Name": "Cortes Islands",
            "AccountId": "001Ec00001cpIbTIAU"
        }
    ]
}
```

Event Attributes

Attribute	Type	Default	Description
changedRecords	*Array*	[ ]	List of records with updated field values.

After saving the records, return the updated records via successRecords:

```
{
    "isSuccess": true,
    "successRecords": [ /* successfully saved records */ ],
    "action": "save",
    "toastMessage": "Records saved successfully."
}
```

If the update fails, return per-record error messages and a failure toast:

```
{
    "isSuccess": false,
    "action": "save",
    "errors": {
```

```
        "001Ru000017YWqtIAG": "The user does not have update permission."
    },
    "toastMessage": "Records update failed."
}
```

7.11 Refresh Event

Dispatch name: `onrefresh`

Fired when the user clicks the refresh icon. The event has no detail payload.

After receiving the event, the consumer should re-fetch the data and provide the updated records via the records attribute on the status object. Then update totalRecordsCount.

```
{
    "isSuccess": true,
    "records": [ /* refreshed records */ ],
    "action": "refresh"
}
```

Appendix A: Quick Reference

A.1 Event-to-Status Mapping

After handling each event, set the action attribute on the status object to the corresponding event string.

Event	action value	Required Status Fields
`onloadmore`	`loadmore`	isSuccess, records
`onsearch`	`search`	isSuccess, records
`onfilter`	`filter`	isSuccess, records
`onclearfilter`	`clearfilter`	isSuccess, records
`onsort`	`sort`	isSuccess, records
`ondelete`	`delete`	isSuccess, deletedRecordIds, errors, toastMessage
`oncreate`	`create`	isSuccess, records, toastMessage
`onsave`	`save`	isSuccess, successRecords, errors, toastMessage
`onrefresh`	`refresh`	isSuccess, records

A.2 Configuration Limits

Attribute	Type	Default	Description
`columnFreezeCount`	*Number*	3	Maximum number of columns that can be frozen.
`sortFieldLevel`	*Number*	3	Maximum number of columns that can be sorted simultaneously.
`rowBorderThickness`	*Number*	`0–5`	Allowed range for row border thickness.
`columnBorderThickness`	*Number*	`0–5`	Allowed range for column border thickness.
`listActions`	*Count*	4	Standard list actions: Create, Delete, Export, Focus Mode.
`rowActions`	*Count*	3	Standard row actions: View, Edit, Delete.

A.3 Glossary

- **LWC** — Lightning Web Component, the Salesforce UI framework on which the Dev Component is built.
- **SObject** — A Salesforce standard or custom object record.
- **Inline Edit** — Editing a cell value directly in the table without opening a modal.
- **Bulk Create** — Creating multiple records simultaneously through a dedicated bulk creation interface.
- **Lazy Loading** — Loading additional records on demand as the user scrolls, rather than all at once.