-Done by
Akashkiruthic & Karan Mahto

Ext JS Grid Demo - Project Documentation

1. Project Overview

This project demonstrates the implementation of an interactive grid using Ext JS 6.2. The grid displays employee details and allows adding and removing employees dynamically. It also includes pagination and sorting features.

2. Folder Structure

Project folder structure:

```
ExtJS_Grid_Project/
│-- index.html  (Main entry point)
│-- app.js  (Application logic for Ext JS Grid)
│-- assets/
│   ├── css/  (Additional styles, if needed)
│   ├── js/   (Additional JavaScript files, if needed)
│-- extjs/  (Optional local Ext JS framework files, if not using CDN)
```

3. Implementation Steps

Step 1: Setup the Project Structure

1. Create a project folder (`ExtJS_Grid_Project`).

2. Inside the folder, create `index.html` and `app.js` files.

3. Optionally, create an `assets` folder for additional styles or scripts.

Step 2: Create the index.html File

This file serves as the main entry point and includes the necessary scripts for Ext JS.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Ext JS Grid Demo</title>
   <script src="https://cdnjs.cloudflare.com/ajax/libs/extjs/6.2.0/ext-all.js"></script>
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/extjs/6.2.0/classic/theme-
neptune/resources/theme-neptune-all.css">
   <script defer src="app.js"></script>
</head>
<body>
   <h1>Loading Ext JS Grid...</h1>
   <div id="grid-container"></div>
</body>
</html>
```

Step 3: Implement app.js

This file contains the Ext JS logic to create and manage the grid.

Code:

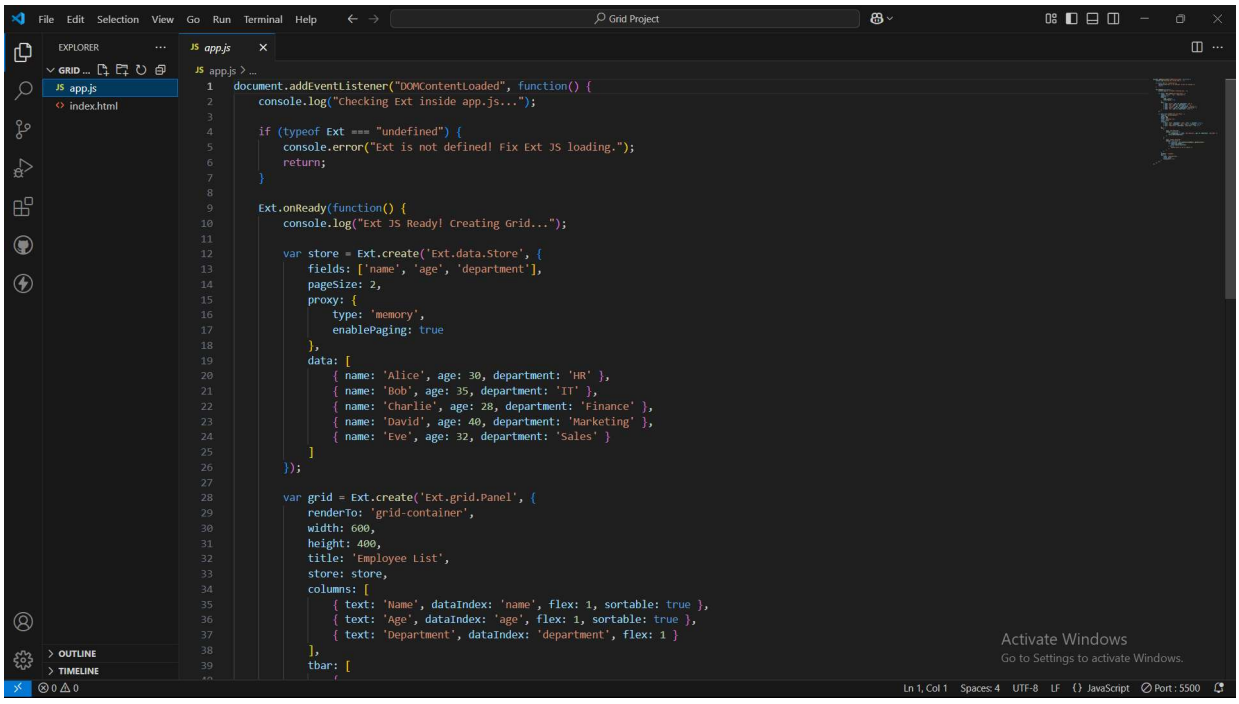
```
document.addEventListener("DOMContentLoaded", function() {
   console.log("Checking Ext inside app.js...");
   if (typeof Ext === "undefined") {
      console.error("Ext is not defined! Fix Ext JS loading.");
      return;
   }
   Ext.onReady(function() {
```

```
console.log("Ext JS Ready! Creating Grid...");
var store = Ext.create('Ext.data.Store', {
    fields: ['name', 'age', 'department'],
    pageSize: 2,
    proxy: {
        type: 'memory',
        enablePaging: true
    },
    data: [
        { name: 'Alice', age: 30, department: 'HR' },
        { name: 'Bob', age: 35, department: 'IT' },
        { name: 'Charlie', age: 28, department: 'Finance' },
        { name: 'David', age: 40, department: 'Marketing' },
        { name: 'Eve', age: 32, department: 'Sales' }
    ]
});
var grid = Ext.create('Ext.grid.Panel', {
    renderTo: 'grid-container',
    width: 600,
    height: 400,
    title: 'Employee List',
    store: store,
    columns: [
        { text: 'Name', dataIndex: 'name', flex: 1, sortable: true },
        { text: 'Age', dataIndex: 'age', flex: 1, sortable: true },
        { text: 'Department', dataIndex: 'department', flex: 1 }
    ],
    tbar: [
        {
            text: 'Add Employee',
            handler: function() {
                var newEmployee = { name: 'New Employee', age: 25, department: 'New Dept' };
                store.add(newEmployee);
            }
```

```
            },
            {
                text: 'Remove Selected',
                handler: function() {
                    var selection = grid.getSelectionModel().getSelection();
                    if (selection.length) {
                        store.remove(selection);
                    } else {
                        alert('Select a row to remove.');
                    }
                }
            }
        ],
        selModel: 'rowmodel',
        bbar: {
            xtype: 'pagingtoolbar',
            store: store,
            displayInfo: true
        }
    });
});
```

-Done by
Akashkiruthic & Karan Mahto

app.js code screenshot



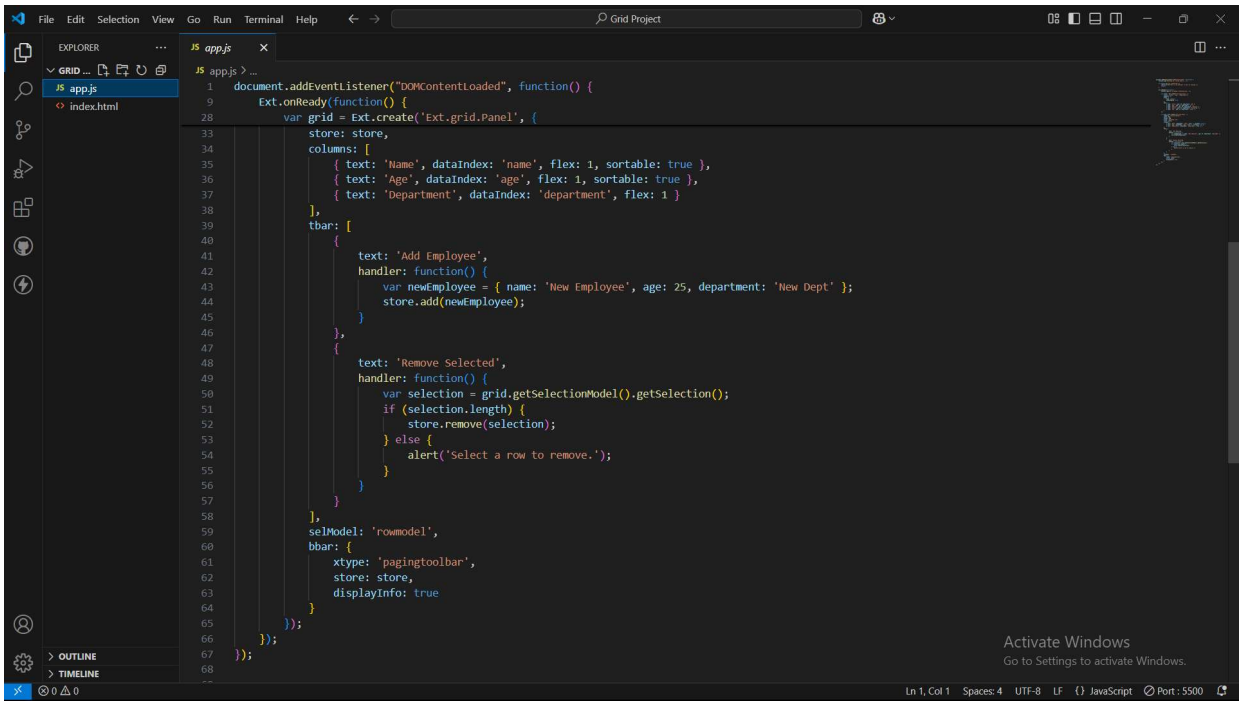
```
document.addEventListener("DOMContentLoaded", function() {
    console.log("Checking Ext inside app.js...");

    if (typeof Ext === "undefined") {
        console.error("Ext is not defined! Fix Ext JS loading.");
        return;
    }

    Ext.onReady(function() {
        console.log("Ext JS Ready! Creating Grid...");

        var store = Ext.create('Ext.data.Store', {
            fields: ['name', 'age', 'department'],
            pageSize: 2,
            proxy: {
                type: 'memory',
                enablePaging: true
            },
            data: [
                { name: 'Alice', age: 30, department: 'HR' },
                { name: 'Bob', age: 35, department: 'IT' },
                { name: 'Charlie', age: 28, department: 'Finance' },
                { name: 'David', age: 40, department: 'Marketing' },
                { name: 'Eve', age: 32, department: 'Sales' }
            ]
        });

        var grid = Ext.create('Ext.grid.Panel', {
            renderTo: 'grid-container',
            width: 600,
            height: 400,
            title: 'Employee List',
            store: store,
            columns: [
                { text: 'Name', dataIndex: 'name', flex: 1, sortable: true },
                { text: 'Age', dataIndex: 'age', flex: 1, sortable: true },
                { text: 'Department', dataIndex: 'department', flex: 1 }
            ],
            tbar: [
```

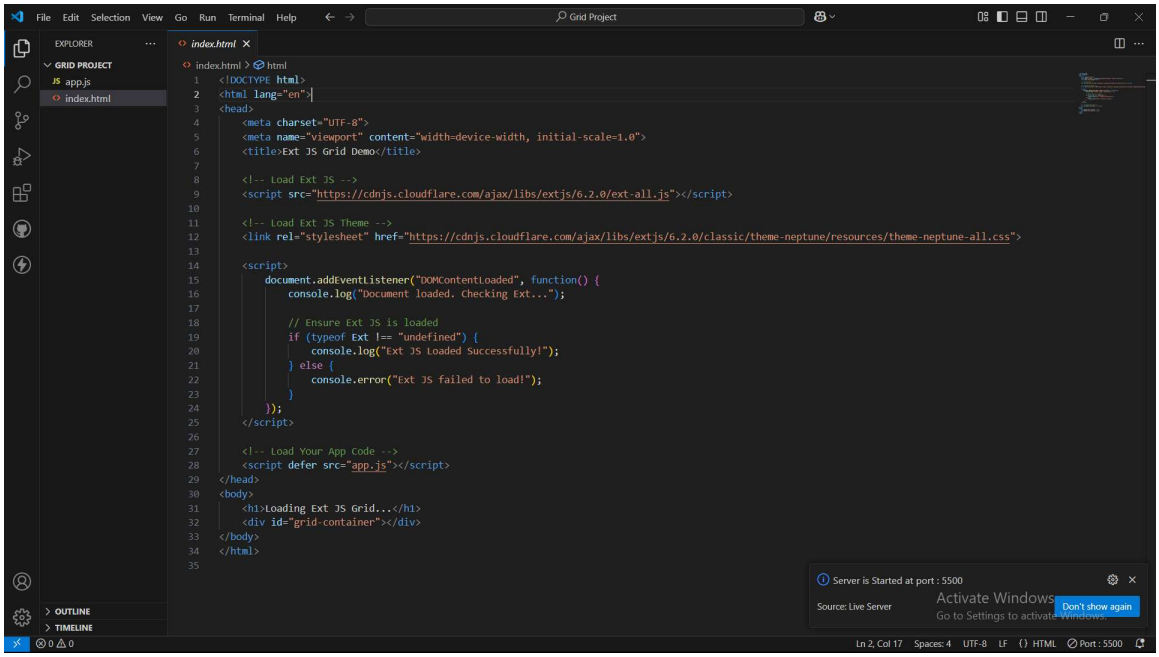


```
document.addEventListener("DOMContentLoaded", function() {
    Ext.onReady(function() {
        var grid = Ext.create('Ext.grid.Panel', {
            store: store,
            columns: [
                { text: 'Name', dataIndex: 'name', flex: 1, sortable: true },
                { text: 'Age', dataIndex: 'age', flex: 1, sortable: true },
                { text: 'Department', dataIndex: 'department', flex: 1 }
            ],
            tbar: [
                {
                    text: 'Add Employee',
                    handler: function() {
                        var newEmployee = { name: 'New Employee', age: 25, department: 'New Dept' };
                        store.add(newEmployee);
                    }
                },
                {
                    text: 'Remove Selected',
                    handler: function() {
                        var selection = grid.getSelectionModel().getSelection();
                        if (selection.length) {
                            store.remove(selection);
                        } else {
                            alert('Select a row to remove.');
                        }
                    }
                }
            ],
            selModel: 'rowmodel',
            bbar: {
                xtype: 'pagingtoolbar',
                store: store,
                displayInfo: true
            }
        });
    });
});
```

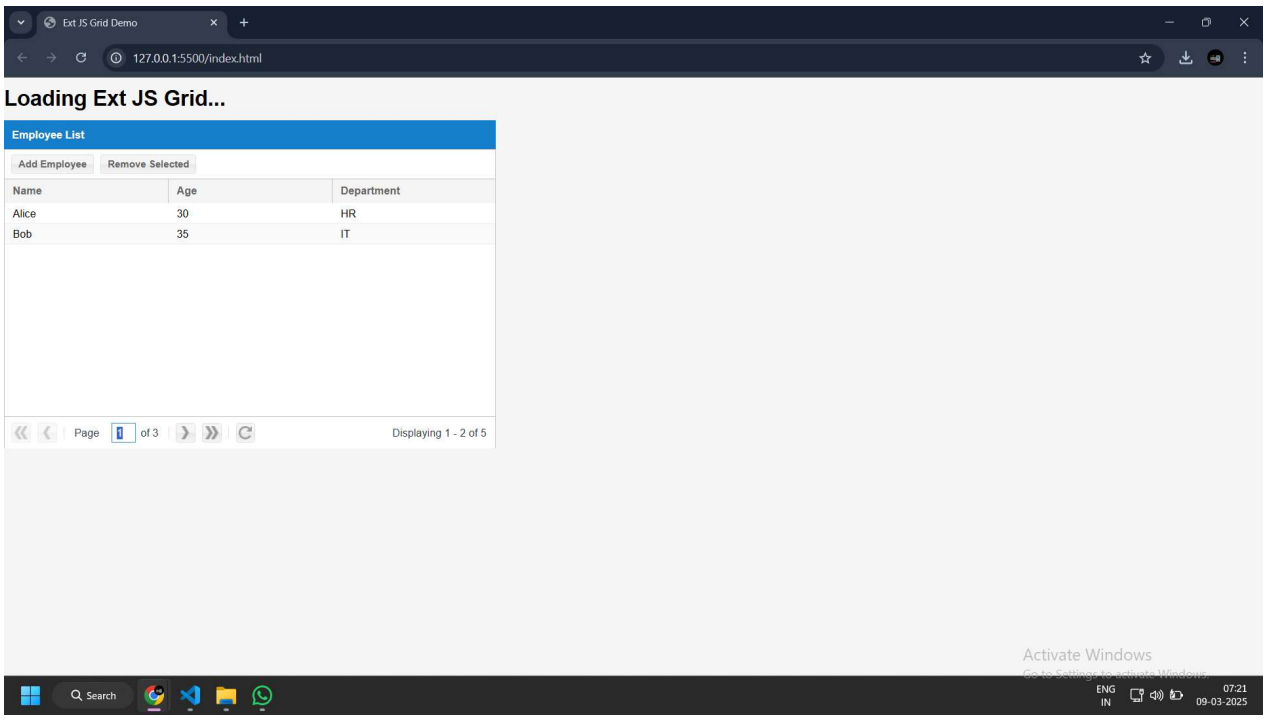
-Guided By SharathKumar

-Done by
Akashkiruthic & Karan Mahto

Index.html ScreenShot



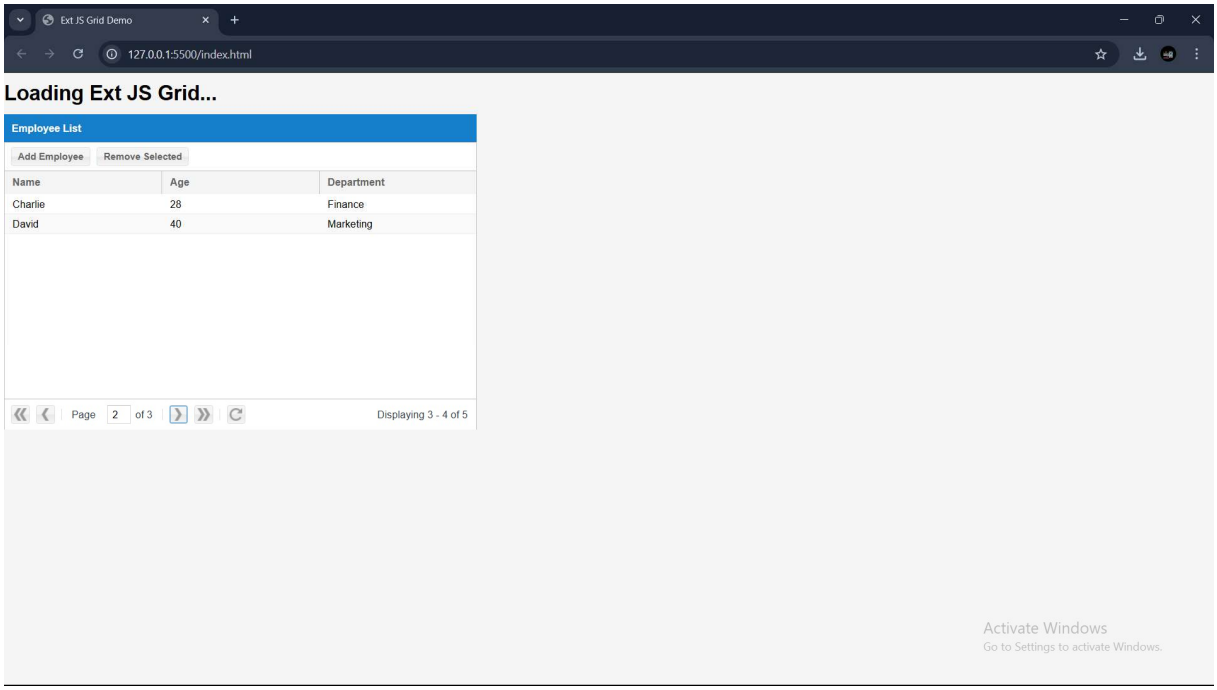
OutPutScreenShot

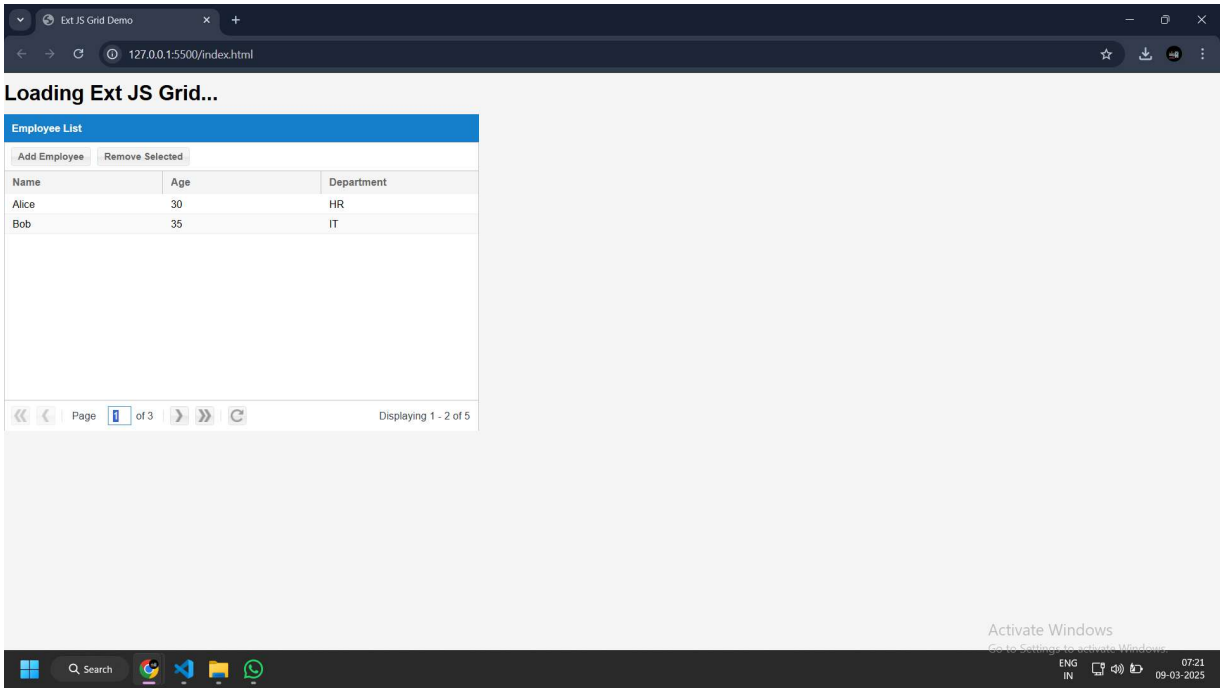


-Guided By SharathKumar

-Done by
Akashkiruthic & Karan Mahto

Output ScreenShot





-Guided By SharathKumar

-Done by
Akashkiruthic & Karan Mahto

4. Explanation of Code

- ***Handling Document Load***

  Ensures that the script executes only after the HTML document has fully loaded.

- ***Creating Data Store***

  Creates an in-memory store with employee data and enables pagination.

- ***Creating the Grid***

  Defines columns and functionalities such as sorting and paging.

- **Toolbar Features**

  Provides buttons to add and remove employees dynamically.

- **Selection Model**

  Allows users to select rows before removing employees.

- **Paging Toolbar**

  Provides navigation controls for paging through the data.

5. Running the Project

Option 1: Open in a Browser

1. Open `index.html` in a browser.
2. Check the console for any errors.

Option 2: Serve via Local Server

1. **Use a local serve**r such as `Live Server` or PythonSimpleHTTPServer.

2. Run the following command:

```
python -m http.server 8000  # Python 3
```

3. Open `http://localhost:8000` in a browser.

6. Conclusion

This project effectively demonstrates the implementation of an interactive Ext JS Grid with features like data display, CRUD operations, sorting, and pagination. By leveraging Ext JS, it ensures a structured and scalable approach to managing tabular data in web applications. The use of an in-memory store allows quick modifications, making it ideal for prototyping. The inclusion of toolbars enhances user interaction by providing seamless row addition and deletion. Pagination ensures efficient handling of large datasets. The project structure follows best practices, making it easy to extend and maintain. Developers can integrate external APIs for dynamic data fetching. By hosting on a local server, the application runs smoothly across different environments. Future improvements may include database integration and advanced filtering. Overall, this project lays a strong foundation for building feature-rich Ext JS applications