

# Frontend Interview Task



## Dynamic Data Table Manager (Next.js + Redux + MUI)

---



### Objective

Build a **Dynamic Data Table Manager** using **Next.js**, **Redux Toolkit**, and **Material UI (MUI)**. This project tests your ability to work with dynamic UIs, manage complex state, and implement real-world features like import/export, searching, sorting, and inline editing.

---



### Project Requirements



#### Core Features

##### 1. Table View

- Display a table with these default columns:  
`Name`, `Email`, `Age`, `Role`
- Add **sorting** on column headers (ASC/DESC toggle)
- Add **global search** (searches all fields)
- Add **client-side pagination** (10 rows per page)

##### 2. Dynamic Columns






- A "Manage Columns" modal:
  - Add new fields like `Department`, `Location`
  - Show/hide existing columns using checkboxes
  - Reflect changes dynamically in the table

- Persist column visibility in **localStorage** or **Redux Persist**

### 3. Import & Export

- **Import CSV:**
    - Upload CSV, parse it using a library (e.g. PapaParse)
    - Show errors for invalid format
  - **Export CSV:**
    - Export current table view to a **.csv** file
    - Only include visible columns
- 

### **Bonus Features (Optional but appreciated)**

-  **Inline row editing**
    - Double-click to edit fields inline
    - Validate inputs (e.g., age must be a number)
    - “Save All” and “Cancel All” buttons
  -  Row actions: **Edit**, **Delete** (with confirmation)
  -  Theme toggle (Light/Dark mode using MUI theming)
  -  Column reordering via drag-and-drop
  -  Fully responsive design
-

## Tech Requirements

- React 18 / Next.js 14 (App Router preferred)
- Redux Toolkit for state management
- Material UI (v5+)
- TypeScript
- React Hook Form for forms
- PapaParse for CSV parsing
- FileSaver.js / Blob for export
- localStorage / Redux Persist for preferences