Esportazione report in Excel da Material React Table
==========

1. Panoramica di Material React Table e funzione di export

- MRT non include un exporter nativo; si integra con librerie esterne (export-to-csv, xlsx, pdfmake).
- I dati filtrati/ordinati visibili in tabella sono disponibili via:
`table.getFilteredRowModel().rows.map(r => r.original)`
- Gli esempi ufficiali mostrano come aggiungere pulsanti di export nella toolbar.

2. Posizionamento del bottone "Scarica report"

Opzioni:
Toolbar superiore	Menu kebab	Floating Action Button
Coerente con UX	UI pulita	Sempre visibile

Consiglio: utilizzare `renderTopToolbarCustomActions`.

3. Implementazione rapida con SheetJS (xlsx)

```tsx
import * as XLSX from 'xlsx';
// ...
const exportExcel = () => {
const rows = table.getFilteredRowModel().rows.map(r => r.original);
const ws = XLSX.utils.json_to_sheet(rows);
const wb = XLSX.utils.book_new();
XLSX.utils.book_append_sheet(wb, ws, 'Report');
XLSX.writeFile(wb, 'report.xlsx');
};
```

4. Tipi di report possibili

- Dati filtrati (default)
- Completo (tutte le colonne/righe)
- Analitico (pivot + KPI)

5. Snippet completo con modale

```tsx
<Button onClick={() => setOpen(true)}>Scarica report</Button>
<Dialog open={open}>
<ListItemButton onClick={handleDownloadFiltrato}>
Dati filtrati (Excel)
</ListItemButton>
</Dialog>
```

6. Alternative a SheetJS

Libreria	Peso browser	Licenza	Pro	Contro
SheetJS	7.5■MB	Apache	Leggera, API semplice	Styling avanzato Pro
ExcelJS	22■MB	MIT	Controllo completo su stili e formule	Peso maggiore
xlsx-populate	~	MIT	Ottima per template esistenti	Meno funzionalità
excel4node	Server only	MIT	Utile lato back-end	Non gira in browser

7. Raccomandazioni

- **Solo export rapido**: SheetJS.
- **Report formattato**: ExcelJS.
- **Template preesistente**: xlsx-populate.
- **Dataset enorme o ACL**: generare su server con ExcelJS/excel4node.

8. Esempio minimal con ExcelJS

```tsx
import ExcelJS from 'exceljs';
import { saveAs } from 'file-saver';
async function downloadExcel(rows) {
const wb = new ExcelJS.Workbook();
const ws = wb.addWorksheet('Report');
ws.addRow(['ID', 'Nome']);
rows.forEach(r => ws.addRow([r.id, r.name]));
const buf = await wb.xlsx.writeBuffer();
saveAs(new Blob([buf]), 'report.xlsx');
}
```

9. Prossimi passi

- Gestione loading con `Backdrop`.
- Aggiunta di stili e colonne larghe auto.
- Filename dinamico (`report_YYYY-MM-DD.xlsx`).
- Backend export per dataset > 50k righe.

Creato da ChatGPT, 26 giugno 2025