```
<!DOCTYPE html>
<html lang="it" data-theme="system">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width,initial-
scale=1,maximum-scale=1,user-scalable=no"/>
  <title>Agenda Alloggi & Pulizie — Pro</title>

  <!-- vis-timeline -->
  <link rel="stylesheet" href="https://unpkg.com/vis-
timeline@7.7.0/styles/vis-timeline-graph2d.min.css">
  <script src="https://unpkg.com/vis-timeline@7.7.0/
standalone/umd/vis-timeline-graph2d.min.js"></script>

  <style>
    /* ============= THEME TOKENS ============= */
    :root{
      --radius:12px;
      --shadow: 0 6px 24px rgba(0,0,0,.18);
      --ring: 0 0 0 2px rgba(43,116,255,.35);
      --aside-w: 360px;               /* larghezza pannello
*/
      --splitter-w: 10px;             /* larghezza maniglia
*/
    }
    /* Dark */
    :root[data-theme="dark"], :root[data-
theme="system"].prefers-dark{
      --bg:#0b1220; --panel:#0e1626; --fg:#e8eefc; --
muted:#9fb1cf; --border:#1f2a44;
      --stay:#22c1dc; --stay-border:#1aa0b7;
      --clean:#f8ae3b; --clean-border:#d8901d;
      --row-bg:#0f1628; --today:#182236;
      --pill:#0b1220; --pill-bg:#ffe08a;
      --brand:#2b74ff; --bad:#ef4444;
      --input-bg: var(--panel);
      --kbd:#2a3551;
      --heat:#2b74ff33; --heat-peak:#2b74ff;
    }
    /* Light */
    :root[data-theme="light"]{
      --bg:#f5f7fb; --panel:#ffffff; --fg:#0c172b; --
muted:#5a6a86; --border:#d7deea;
      --stay:#22c1dc; --stay-border:#1aa0b7;
      --clean:#f8ae3b; --clean-border:#d8901d;
      --row-bg:#f2f6ff; --today:#e9f1ff;
      --pill:#3a2d00; --pill-bg:#fff0b3;
      --brand:#2b74ff; --bad:#d12c2c;
```

```
      --input-bg:#ffffff;
      --kbd:#ebeff7;
      --heat:#2b74ff22; --heat-peak:#2b74ff;
    }

    /* rimuove contorni box-day */
    .vis-time-axis .vis-grid.vis-vertical { border-left: none
!important; }
    .vis-time-axis .vis-text { opacity:.8 }

    /* gutter giorno: una linea morbida, non un bordo del box
*/
    .day-gutter {
      position:absolute; top:0; bottom:0; width:2px;
      background: linear-gradient(to bottom, transparent 0,
rgba(255,255,255,.25) 30%, rgba(0,0,0,.2) 70%, transparent
100%);
      pointer-events:none;
    }

    /* pill smussata agli estremi dell'evento per dare
l'effetto "incastro" */
    .vis-item.vis-range .vis-item-content { border-radius:
12px; }
    .vis-item.cleaning .vis-item-content { border: 1px dashed
currentColor; }

    *{box-sizing:border-box}
    html,body{height:100%}
    body{margin:0; background:var(--bg); color:var(--fg);
font-family: Inter, system-ui, -apple-system, Segoe UI,
Roboto, Arial, sans-serif;}

    /* ======= LAYOUT: main + aside ======= */
    .page{
      display:grid;
      grid-template-columns: 1fr var(--splitter-w) var(--
aside-w);
      gap:0;
      transition:grid-template-columns .2s ease;
      max-width:1400px; margin:0 auto;
    }
    .page[data-aside="closed"]{ grid-template-columns: 1fr
var(--splitter-w) 0; }
    .splitter{
      cursor:col-resize; width:var(--splitter-w);
      position:relative;
```

```
      background:linear-gradient(90deg, transparent, var(--
border), transparent);
    }
    .page[data-aside="closed"] .splitter::after{
      content:'◀';
      position:absolute; top:50%; right:-4px;
transform:translateY(-50%);
      background:var(--panel); color:var(--fg);
      border:1px solid var(--border); border-radius:10px;
      padding:2px 6px; font-size:12px; cursor:pointer; box-
shadow:var(--shadow);
    }

    .aside{
      position:sticky; top:0; height:100vh;
      background:var(--panel); border-left:1px solid var(--
border);
      padding:12px; overflow:auto; box-shadow:var(--shadow);
    }
    .aside header{display:flex; align-items:center; gap:8px;
margin-bottom:8px}
    .aside .title{font-weight:800}
    .aside .ghost{opacity:.6; font-weight:600; font-
size:12px}

    .shell{padding:14px;}
    .top{position:sticky; top:0; z-index:20; display:flex;
align-items:center; gap:10px; flex-wrap:wrap; margin-
bottom:10px; background:linear-gradient(180deg,var(--
bg),rgba(11,18,32,.0)); padding-bottom:6px;}
    .title{font-weight:800; font-size:18px;}
    .spacer{flex:1}
    .legend{display:flex; gap:8px; align-items:center}
    .chip{display:inline-flex; align-items:center; gap:8px;
padding:6px 10px; border:1px solid var(--border);
background:var(--panel); border-radius:999px; font-size:12px;
color:var(--muted)}
    .dot{width:10px; height:10px; border-radius:999px;
display:inline-block}

    .toolbar{display:flex; align-items:center; gap:8px; flex-
wrap:wrap; margin-bottom:10px;}
    input[type="date"], button, select, input[type="search"],
textarea{
      padding:10px 12px; border:1px solid var(--border);
border-radius:10px; background:var(--input-bg); color:var(--
fg); font-size:14px; box-shadow:none; outline:none;
    }
```

```
    textarea{width:100%; min-height:90px; resize:vertical}
    input::placeholder{color:#6e82a8}
    button{cursor:pointer; transition:filter .15s ease,
transform .02s ease}
    button:active{transform:translateY(1px)}
    button.primary{background:var(--brand); color:#fff;
border-color:var(--brand)}
    button:hover{filter:brightness(1.05)}
    button:focus-visible, input:focus-visible, select:focus-
visible, textarea:focus-visible{ box-shadow: var(--ring); }

    .segmented{display:inline-flex; border:1px solid var(--
border); border-radius:10px; overflow:hidden}
    .segmented button{border:0; background:var(--panel);
padding:8px 12px}
    .segmented button.active{background:var(--kbd);
color:#fff}

    #range{font-weight:700}
    #timeline{height:72vh; border:1px solid var(--border);
border-radius:var(--radius); overflow:auto; background:var(--
panel); box-shadow:var(--shadow);}
    #list{display:none}

    /* vis theme */
    .vis-timeline{border:0; background:var(--panel)}
    .vis-panel.vis-left{background:var(--panel); border-
right:1px solid var(--border)}
    .vis-labelset .vis-label{border-bottom:1px solid var(--
border); background:var(--row-bg); color:var(--fg)}
    .vis-labelset .vis-label .vis-inner{padding:var(--row-
pad,8px) 10px; font-weight:700; font-size:13px; white-
space:nowrap; overflow:hidden; text-overflow:ellipsis}
    .vis-time-axis .vis-text{color:var(--muted)}
    .vis-time-axis .vis-grid.vis-minor{border-color:#1b2743}
    .vis-time-axis .vis-grid.vis-major{border-color:#253154}
    .vis-current-time{background:var(--bad)}

    .vis-item{border-radius:10px; border-width:1px;
color:#0b1220; font-weight:800; line-height:20px}
    .vis-item.stay .vis-item-content{padding:6px 10px; font-
size:12px; white-space:nowrap}
    .vis-item.cleaning{background:var(--clean); border-
color:var(--clean-border); height:16px;}
    .vis-item.cleaning .vis-item-content{padding:0 8px; font-
size:11px; line-height:16px;}
```

```
    .pill{display:inline-flex; align-items:center; gap:6px;
padding:0 8px; height:18px; border-radius:999px;
background:var(--pill-bg); color:var(--pill); font-
weight:900; font-size:11px;}
    .muted{color:var(--muted); font-weight:600}

    /* KBD */
    kbd{font:600 11px/1 system-ui; padding:4px 6px; border-
radius:6px; background:var(--kbd); border:1px solid var(--
border);}

    /* Mobile */
    @media (max-width: 900px){
      .page{grid-template-columns: 1fr 0 0;}
      .shell{padding:10px}
      .legend{display:none}
      .title{font-size:16px}
      .toolbar .desktop-only{display:none}
      #timeline{height:60vh}
      .segmented{width:100%}
      .segmented button{flex:1}
    }

    /* Zen focus */
    html[data-zen="1"] .vis-item{ filter:saturate(.15)
opacity:.55 }
    html[data-zen="1"] .vis-item.cleaning{ filter:none;
opacity:1 }

    /* aside widgets */
    .card{border:1px solid var(--border); border-radius:12px;
padding:10px; background:var(--panel); margin-bottom:10px}
    .chips{display:flex; flex-wrap:wrap; gap:6px}
    .chip-btn{border:1px solid var(--border);
background:var(--input-bg); border-radius:999px; padding:6px
10px; font-size:12px; cursor:pointer}
    .chip-btn:hover{filter:brightness(1.04)}
    .row{display:flex; gap:8px; align-items:center}
    .row > *{flex:1}
    .ghost-small{font-size:12px; color:var(--muted)}
    .handle{background:var(--kbd); border:1px solid var(--
border); border-radius:8px; padding:4px 8px; font-size:12px;
cursor:pointer}
    .conf{display:flex; align-items:center; justify-
content:space-between; gap:6px; padding:6px 8px; border-
radius:8px; border:1px solid var(--border); margin-
bottom:6px}
```

```
    /* Barra di scorrimento orizzontale */
    #scrollX { height:16px; overflow-x:auto; overflow-
y:hidden; background:var(--panel); border-top:1px solid
var(--border); border-radius:10px; }
    #scrollXInner { height:1px; }
  </style>
</head>
<body>
  <div class="page" id="page" data-aside="open">
    <!-- MAIN -->
    <div>
      <div class="shell">
        <div class="top">
          <div class="title">Agenda Alloggi &amp; Pulizie</
div>
          <div class="spacer"></div>

          <!-- Theme selector -->
          <select id="themeSel" title="Tema" class="desktop-
only">
            <option value="system">Sistema</option>
            <option value="light">Chiaro</option>
            <option value="dark">Scuro</option>
          </select>

          <label class="desktop-only" title="Aumenta spazi e
target tattili"><input id="comfort" type="checkbox">
comfort</label>

          <div class="legend desktop-only">
            <span class="chip"><span class="dot"
style="background:var(--stay)"></span> Soggiorno</span>
            <span class="chip"><span class="dot"
style="background:var(--clean)"></span> Pulizie</span>
          </div>
        </div>

        <div class="toolbar" role="toolbar" aria-
label="Comandi">
          <div class="segmented" role="tablist" aria-
label="Vista">
            <button id="tabCal" class="active"
onclick="setMode('cal')" aria-selected="true">Calendario</
button>
            <button id="tabList" onclick="setMode('list')"
aria-selected="false">Lista</button>
          </div>
```

```
          <span class="muted" style="margin-
left:6px">Periodo</span>
          <input id="from" type="date" aria-label="Da">
          <input id="to"   type="date" aria-label="A">
          <button id="btn" class="desktop-only"
onclick="refresh()">Aggiorna</button>

          <select id="propFilter" title="Filtra per unità"
aria-label="Filtra per unità">
            <option value="">Tutte le unità</option>
          </select>
          <label class="desktop-only" title="Mostra solo
pulizie"><input id="onlyClean" type="checkbox"> solo
pulizie</label>
          <label class="desktop-only" title="Mostra solo
rientri"><input id="onlyRientro" type="checkbox"> solo
rientro</label>
          <input id="q" type="search" placeholder="Cerca
unità/ospite…" aria-label="Cerca" style="min-width:160px">

          <div class="spacer"></div>
          <button class="desktop-only" onclick="zoom('week')"
title="Settimana (W)"><kbd>W</kbd> Settimana</button>
          <button class="desktop-only"
onclick="zoom('month')" title="Mese (M)"><kbd>M</kbd> Mese</
button>
          <button onclick="moveDays(-1)" title="Giorno
precedente (←)"><kbd>←</kbd> Giorno</button>
          <button class="primary" onclick="goToday()"
title="Oggi (T)"><kbd>T</kbd> Oggi</button>
          <button onclick="moveDays(1)" title="Giorno
successivo (→)">Giorno <kbd>→</kbd></button>

          <div class="spacer"></div>
          <span id="range" style="font-weight:700"></span>
          <span id="count" class="muted" style="margin-
left:8px"></span>

          <span class="muted desktop-only" style="margin-
left:8px">Omni-search: <kbd>Ctrl</kbd>/<kbd>⌘</kbd>+<kbd>K</
kbd></span>
        </div>

        <!-- Calendario -->
        <div id="timeline" aria-hidden="false"></div>
        <!-- Barra di scorrimento orizzontale -->
        <div id="scrollX"><div id="scrollXInner"></div></div>
```

```
        <!-- Lista (smartphone) -->
        <div id="list" aria-hidden="true"></div>
      </div>
    </div>

    <!-- SPLITTER -->
    <div class="splitter" id="splitter" title="Trascina per
ridimensionare o clicca per aprire/chiudere"></div>

    <!-- ASIDE / COPILOT -->
    <aside class="aside" id="aside">
      <header>
        <div class="title">Copilot</div>
        <span class="ghost">strumenti rapidi</span>
        <div class="spacer"></div>
        <button id="toggleAside" class="handle"
title="Chiudi/Apri">⇢</button>
      </header>

      <!-- Omni search -->
      <div class="card">
        <div class="ghost-small" style="margin-
bottom:6px">Scrivi comandi o filtri (es. “oggi”, “settimana”,
“solo pulizie”, “Milano 211”, “dal 2025-10-01 al
2025-10-10”)</div>
        <input id="omni" type="search" placeholder="Cosa vuoi
fare?" />
        <div class="chips" style="margin-top:8px">
          <button class="chip-btn" data-cmd="oggi">Oggi</
button>
          <button class="chip-btn" data-
cmd="settimana">Settimana</button>
          <button class="chip-btn" data-cmd="mese">Mese</
button>
          <button class="chip-btn" data-cmd="solo
pulizie">Solo pulizie</button>
          <button class="chip-btn" data-cmd="solo
rientro">Solo rientro</button>
          <button class="chip-btn" data-cmd="zen">Zen</
button>
        </div>
      </div>

      <!-- Heatmap occupazione -->
      <div class="card">
        <div class="row" style="justify-content:space-
between">
```

```
          <div class="title" style="font-
size:14px">Occupazione</div>
          <span id="occPct" class="ghost-small">—</span>
        </div>
        <canvas id="occCanvas" width="320" height="72"
style="width:100%; height:72px; display:block"></canvas>
        <div class="ghost-small">Prossimi 30 giorni · più
pieno = più colore</div>
      </div>

      <!-- Conflict Radar -->
      <div class="card" id="confCard" style="display:none">
        <div class="title" style="font-size:14px">Anomalie</
div>
        <div id="confList"></div>
        <div class="ghost-small" id="confMore"></div>
      </div>

      <!-- Note rapide -->
      <div class="card">
        <div class="row">
          <input id="noteDate" type="date" />
          <button id="noteToday" class="chip-btn"
style="flex:0">Oggi</button>
        </div>
        <textarea id="noteTxt" placeholder="Appunti/
istruzioni per il giorno..."></textarea>
        <div class="row">
          <button id="saveNote" class="primary">Salva nota</
button>
          <button id="clearNote">Cancella</button>
        </div>
        <div id="noteInfo" class="ghost-small" style="margin-
top:6px"></div>
      </div>

      <!-- Totali pulizie (richiede getCleaningTotals lato
server) -->
      <div class="card">
        <div class="title" style="font-size:14px">Totali
Pulizie</div>
        <div id="totalsBody">Caricamento…</div>
        <button class="chip-btn" id="exportCsv">Esporta CSV</
button>
      </div>

      <!-- Utilità -->
      <div class="card">
```

```
        <div class="row">
          <button id="jumpNow">Centrati sull’ora attuale</
button>
          <button id="resetFilters">Reset filtri</button>
        </div>
      </div>
    </aside>
  </div>

  <script>
    /*
==========================================================
     * 1) DATI DAL SERVER
     *
==========================================================
*/
    const INIT_FROM = '<?!= fromISO ?>';
    const INIT_TO   = '<?!= toISO   ?>';
    const INIT_ROWS = <?!= JSON.stringify(rows) ?>; // slim:
{g,t,s,e,l,r}

    /*
==========================================================
     * 2) NORMALIZZAZIONE NOMI (DEDUP)
     *
==========================================================
*/
    const ALIAS_MAP = {
      "Domus Art | Vivi l'Arte nel Centro di Ferrara": "Domus
Art",
      "[Centro Storico] Elegante open space deposito bag":
"Ferrara Sotto",
      "Charme Ferrara | Vista su Piazza Trento – Trieste":
"Ferrara Sopra",
      "Loft Love Milano – Design Max 4 people": "Milano 209",
      "Loft con Balcone | 20' Duomo – Olympic Village":
"Milano 211",
    };
    const ALWAYS_GROUPS = [ "Milano 211" ];

    const $ = id => document.getElementById(id);
    function canonUnit(u){ const x = String(u||'').trim();
return ALIAS_MAP[x] || x; }
    function mapRow(r){ const gCanon = canonUnit(r.g); return
Object.assign({}, r, { gOrig: r.g, g: gCanon }); }
    function itemIdFromRow(r){ return r._id || (r.g + '|' +
r.t + '|' + r.s + '|' + r.e); }
```

```
    /*
================================================================
     * 3) STATO
     *
================================================================
*/
    let mode = (window.innerWidth <= 900) ? 'list' : 'cal';
    let allRows = INIT_ROWS.map(mapRow);
    let rows = allRows.slice();
    let timeline, groupsDS, itemsDS;

    const fromEl = () => $('from');
    const toEl   = () => $('to');
    const qEl    = () => $('q');
    const propEl = () => $('propFilter');
    const onlyCleanEl = () => $('onlyClean');
    const onlyRientroEl = () => $('onlyRientro');

    /*
================================================================
     * 3.1 Preferenze
     *
================================================================
*/
    const PREFS_KEY='hn_agenda_prefs_v4';
    function savePrefs(){
      const prefs={
        onlyClean: onlyCleanEl().checked,
        onlyRientro: onlyRientroEl().checked,
        prop: propEl().value,
        mode, comfort:
document.documentElement.dataset.comfort==='1',
        theme: document.documentElement.dataset.theme ||
'system',
        aside: $('page').dataset.aside || 'open',
        asideW:
getComputedStyle(document.documentElement).getPropertyValue('
--aside-w').trim()
      };
      localStorage.setItem(PREFS_KEY, JSON.stringify(prefs));
    }
    function loadPrefs(){
      try{
        const
p=JSON.parse(localStorage.getItem(PREFS_KEY)||'{}');
        if (typeof p.onlyClean==='boolean')
onlyCleanEl().checked=p.onlyClean;
```

```
        if (typeof p.onlyRientro==='boolean')
onlyRientroEl().checked=p.onlyRientro;
        if(p.prop) propEl().value=p.prop;
        if(p.mode) setMode(p.mode);
        document.documentElement.dataset.comfort =
p.comfort ? '1':'0';
        const theme = p.theme || 'system';
        document.documentElement.dataset.theme = theme;
        $('themeSel').value = theme;
        applySystemThemeClass();
        if (p.aside) $('page').dataset.aside = p.aside;
        if (p.asideW)
document.documentElement.style.setProperty('--aside-w',
p.asideW);
      }catch{}
    }

    /*
=============================================================
     * 4) UI MODALITÀ
     *
=============================================================
*/
    function setMode(m){
      mode = m;
      $('tabCal').classList.toggle('active', m==='cal');
      $('tabList').classList.toggle('active', m==='list');
      $('timeline').style.display = (m==='cal') ? 'block' :
'none';
      $('timeline').setAttribute('aria-hidden', m!=='cal');
      $('list').style.display = (m==='list') ? 'block' :
'none';
      $('list').setAttribute('aria-hidden', m!=='list');
      if (m === 'cal') { ensureTimeline();
applyDataBatched(rows); renderDayGutters(); }
      if (m === 'list') { renderList(rows); }
      savePrefs();
    }

    /*
=============================================================
     * 5) RANGE & NAV
     *
=============================================================
*/
    function fmtRange(fromISO, toISO){
      const f = new Date(fromISO + 'T00:00:00'); const t =
new Date(toISO + 'T23:59:59');
```

```
      const opt = { day:'2-digit', month:'short',
year:'numeric' };
      return f.toLocaleDateString(undefined, opt) + ' → ' +
t.toLocaleDateString(undefined, opt);
    }
    function setRangeLabel(){ $('range').textContent =
fmtRange(fromEl().value, toEl().value); }

    function moveDays(d){
      const f = new Date(fromEl().value + 'T00:00:00');
      const t = new Date(toEl().value   + 'T00:00:00');
      f.setDate(f.getDate()+d); t.setDate(t.getDate()+d);
      fromEl().value = f.toISOString().slice(0,10);
      toEl().value   = t.toISOString().slice(0,10);
      setRangeLabel(); refreshDebounced();
    }
    function goToday(){
      const today = new Date();
      const f = new Date(today);
      const t = new Date(today); t.setDate(today.getDate() +
(window.innerWidth<=900 ? 1 : 30));
      fromEl().value = f.toISOString().slice(0,10);
      toEl().value   = t.toISOString().slice(0,10);
      setRangeLabel(); refreshDebounced();
    }
    function zoom(modeZoom){
      const from = new Date(fromEl().value + 'T00:00:00');
      let to = new Date(from);
      if (modeZoom === 'week')  to.setDate(from.getDate()+7);
      if (modeZoom === 'month') to.setMonth(from.getMonth()
+1);
      to.setDate(to.getDate()-1);
      toEl().value = to.toISOString().slice(0,10);
      setRangeLabel(); refreshDebounced();
    }

    /*
============================================================
     * 6) FILTRI
     *
============================================================
*/
    function buildFiltersFromRows(rowsSlim){
      const set = new Set(ALWAYS_GROUPS.map(String));
      rowsSlim.forEach(r => set.add(r.g));
      const cur = decodeURIComponent(propEl().value || '');
      propEl().innerHTML = '<option value="">Tutte le unità</
option>' +
```

```
Array.from(set).sort((a,b)=>a.localeCompare(b,'it')).map(v =>
`<option value="${encodeURIComponent(v)}">${v}</
option>`).join('');
      if (set.has(cur)) propEl().value =
encodeURIComponent(cur);
    }
    function applyFilters(){
      const q = (qEl().value || '').toLowerCase().trim();
      const prop = decodeURIComponent(propEl().value || '');
      const onlyClean = onlyCleanEl().checked;
      const onlyRientro = onlyRientroEl().checked;

      rows = allRows.filter(r => {
        if (onlyClean && r.t !== 'c') return false;
        if (onlyRientro && !(r.t === 'c' && /rientro/
i.test(r.l || ''))) return false;
        if (prop && r.g !== prop) return false;
        if (q) {
          const hay = ((r.g||'') + ' ' +
(r.l||'')).toLowerCase();
          if (!hay.includes(q)) return false;
        }
        return true;
      });
      $('count').textContent = rows.length + ' elementi';
      drawHeatmapDebounced();
      updateConflictsUI();
    }

    /*
================================================================
     * 7) TIMELINE
     *
================================================================
*/
    function ensureTimeline(){
      if (timeline) return;
      groupsDS = new vis.DataSet([]);
      itemsDS  = new vis.DataSet([]);
      const container = document.getElementById('timeline');
      const options = {
        stack: true,
        orientation: 'top',
        zoomKey: 'ctrlKey',
        moveable: true,
        zoomable: true,
        margin: { item: 6, axis: 8 },
```

```
        start: new Date(fromEl().value + 'T00:00:00'),
        end:   new Date(toEl().value   + 'T23:59:59'),
        showCurrentTime: true,
        groupHeightMode: 'fixed',
        groupOrder: (a,b) =>
a.content.localeCompare(b.content,'it'),
        tooltip: { followMouse: true },
        locale: 'it'
      };
      timeline = new vis.Timeline(container, itemsDS,
groupsDS, options);

      // Rotellina: verticale = scroll gruppi; Shift o scroll
orizzontale = PAN orizzontale
      container.addEventListener('wheel', (ev)=>{
        if (ev.ctrlKey) return; // lascia lo zoom a vis
        const wantHoriz = ev.shiftKey || Math.abs(ev.deltaX)
> Math.abs(ev.deltaY);
        if (wantHoriz){
          const range = timeline.getWindow();
          const span = range.end - range.start;
          const px = container.clientWidth || 1;
          const deltaPx = (Math.abs(ev.deltaX) >
Math.abs(ev.deltaY)) ? ev.deltaX : ev.deltaY; // con Shift
uso deltaY
          const msPerPx = span / px;
          const shiftMs = deltaPx * msPerPx;
          const newStart = new Date(range.start.valueOf() +
shiftMs);
          const newEnd = new Date(range.end.valueOf() +
shiftMs);
          timeline.setWindow(newStart, newEnd, { animation:
false });
        } else {
          container.scrollTop += ev.deltaY;
        }
        ev.preventDefault();
      }, {passive:false});

      // Click item -> Calendar
      timeline.on('itemclick', function (props) {
        const it = props.item && itemsDS.get(props.item);
        if (!it) return;
        const q = encodeURIComponent(it._rid || it.group ||
'');
        if (q) window.open('https://calendar.google.com/
calendar/u/0/r/search?q=' + q, '_blank');
      });
```

```
      // gutter iniziali
      renderDayGutters();
    }

    // Colore stabile per unità
    function colorFromName(name){
      let h=0; for (let i=0;i<name.length;i++) h = (h*31 +
name.charCodeAt(i))>>>0;
      const hue = h % 360, sat = 75, light = 55;
      return `hsl(${hue} ${sat}% ${light}%)`;
    }

    function toItem(r){
      const isClean = (r.t === 'c');
      const base = isClean ? 'var(--clean)' :
colorFromName(r.g);
      const content = isClean
        ? (r.l && /rientro/i.test(r.l) ? `<span class="pill">
🧹 ${r.l}</span>` : '🧹 Pulizie')
        : (r.l || '');
      return {
        id: itemIdFromRow(r),
        group: r.g,
        start: r.s,
        end: r.e,
        className: isClean ? 'cleaning' : 'stay',
        style: `background:${base}; border-color:$
{isClean?'var(--clean-border)':base};`,
        content,
        title: [ r.g, (isClean ? 'Pulizie' : 'Soggiorno') +
(r.l ? ' · ' + r.l : ''), new Date(r.s).toLocaleString() + '
→ ' + new Date(r.e).toLocaleString() ].join('\n'),
        _rid: r.r
      };
    }

    function computeConflictsSet(arr){
      const overlaps=(a,b,c,d)=> (new Date(a)<new Date(d) &&
new Date(c)<new Date(b));
      const byG={};
      arr.forEach(r=>{ (byG[r.g] ||= []).push(r); });
      const ids=new Set();
      Object.values(byG).forEach(list=>{
        const stays=list.filter(r=>r.t==='s');
        const cleans=list.filter(r=>r.t==='c');
        cleans.forEach(c=>{
```

```
          stays.forEach(s=>{ if(overlaps(c.s,c.e,s.s,s.e))
{ ids.add(itemIdFromRow(c)); ids.add(itemIdFromRow(s)); }});
        });
        for(let i=0;i<cleans.length;i++){
          for(let j=i+1;j<cleans.length;j++){

if(overlaps(cleans[i].s,cleans[i].e,cleans[j].s,cleans[j].e))
{
              ids.add(itemIdFromRow(cleans[i]));
ids.add(itemIdFromRow(cleans[j]));
            }
          }
        }
      });
      return ids;
    }

    function applyDataBatched(data){
      if (!timeline) return;

      timeline.setWindow(
        new Date(fromEl().value + 'T00:00:00'),
        new Date(toEl().value   + 'T23:59:59'),
        { animation: false }
      );

      const uniq = new Set(ALWAYS_GROUPS.map(String));
      data.forEach(r => uniq.add(r.g));
      const curIds = new Set(groupsDS.getIds());
      const wantIds = new Set(uniq);
      uniq.forEach(name => { if (!curIds.has(name))
groupsDS.add({ id: name, content: name }); });
      curIds.forEach(id => { if (!wantIds.has(id))
groupsDS.remove(id); });

      const confSet = computeConflictsSet(allRows);

      const newItems = data.map(r => {
        const it = toItem(r);
        if (confSet.has(it.id)){
          it.style += ';box-shadow:0 0 0 2px var(--bad)
inset;';
        }
        return it;
      });

      const newIds = new Set(newItems.map(x => x.id));
      const oldIds = new Set(itemsDS.getIds());
```

```
      const toAdd = [];
      const toUpdMap = new Map();
      newItems.forEach(it => { oldIds.has(it.id) ?
toUpdMap.set(it.id, it) : toAdd.push(it); });
      const toUpd = Array.from(toUpdMap.values());
      const toDel = Array.from(oldIds).filter(id => !
newIds.has(id));

      const BATCH = 400;
      function runBatches(list, op){
        let i = 0;
        return new Promise(resolve=>{
          (function step(){
            const chunk = list.slice(i, i+BATCH);
            if (!chunk.length) return resolve();
            op(chunk); i += BATCH;
            (window.requestIdleCallback ||
window.requestAnimationFrame)(step);
          })();
        });
      }
      Promise.resolve()
        .then(()=> runBatches(toAdd, chunk =>
itemsDS.add(chunk)))
        .then(()=> runBatches(toUpd, chunk =>
itemsDS.update(chunk)))
        .then(()=> runBatches(toDel, chunk =>
itemsDS.remove(chunk)));
    }

    /*
==============================================================
     * 7.1 Giunzioni tra giorni (gutter verticali)
     *
==============================================================
*/
    function renderDayGutters(){
      if (!timeline) return;
      const root = timeline.dom.center;
      root.querySelectorAll('.day-
gutter').forEach(n=>n.remove());
      const range = timeline.getWindow();
      const start = new Date(range.start);
start.setHours(0,0,0,0);
      const end   = new Date(range.end);
end.setHours(0,0,0,0);
```

```
      for (let d = new Date(start); d <= end;
d.setDate(d.getDate()+1)){
        const x = timeline.timeToScreen(d);
        const g = document.createElement('div');
        g.className = 'day-gutter';
        g.style.left = `${x}px`;
        root.appendChild(g);
      }
    }
    // ridisegna i gutter quando cambia il range
    const _reGutters = ()=> renderDayGutters();
    document.addEventListener('visibilitychange',
_reGutters);
    window.addEventListener('resize', _reGutters);

    /*
================================================================
     * 7.2 Barra di scorrimento in basso sincronizzata
     *
================================================================
*/
    const sc = document.getElementById('scrollX');
    const inner = document.getElementById('scrollXInner');
    function syncScrollBar(){
      if (!timeline) return;
      inner.style.width = (timeline.dom.center.scrollWidth ||
(timeline.dom.center.clientWidth*3))+'px';
      sc.scrollLeft = timeline.dom.center.scrollLeft;
    }
    sc.addEventListener('scroll', ()=>{ if(timeline)
timeline.dom.center.scrollLeft = sc.scrollLeft; });
    const _center = ()=> timeline && timeline.dom &&
timeline.dom.center;
    const centerElObserver = new
MutationObserver(syncScrollBar);
    const ensureObserver = ()=>{
      if (_center()){
        _center().addEventListener('scroll',
()=>{ sc.scrollLeft = timeline.dom.center.scrollLeft; });
        centerElObserver.observe(_center(),
{ attributes:true, attributeFilter:['style','class'] });
        syncScrollBar();
      }
    };

    /*
================================================================
     * 8) LIST VIEW
```

```
     *
================================================================
*/
    function renderList(data){
      const host = $('list');
      host.innerHTML = '';
      const byDay = new Map();
      data.forEach(r => {
        const k = (r.s || r.e || '').slice(0,10);
        if (!byDay.has(k)) byDay.set(k, []);
        byDay.get(k).push(r);
      });
      const days = Array.from(byDay.keys()).sort();
      const todayISO = new Date().toISOString().slice(0,10);

      days.forEach(dayISO => {
        const wrap = document.createElement('div');
        const h = document.createElement('h3');
        const d = new Date(dayISO + 'T00:00:00');
        h.textContent = d.toLocaleDateString(undefined,
{ weekday:'short', day:'2-digit', month:'short' }) +
(dayISO===todayISO ? ' · Oggi' : '');
        wrap.appendChild(h);

        const arr = byDay.get(dayISO).slice().sort((a,b)=>{
          if (a.t!==b.t) return (a.t==='c'?-1:1);
          return new Date(a.s)-new Date(b.s);
        });

        arr.forEach(r => {
          const card = document.createElement('div');
          card.className = 'card';
          card.style.cssText = 'display:flex; align-
items:center; justify-content:space-between; gap:8px;';
          const left = document.createElement('div');
          const right = document.createElement('div');

          const badge = document.createElement('span');
          badge.style.cssText = 'font-size:11px; padding:2px
8px; border-radius:999px; color:#0b1220; background:'+
(r.t==='c' ? 'var(--clean)' : 'var(--stay)') + '; font-
weight:900';
          badge.textContent = (r.t==='c' ? 'Pulizie' :
'Soggiorno');

          const unit = document.createElement('div');
          unit.style.cssText = 'font-weight:800; font-
size:13px';
```

```
          unit.textContent = r.g;

          const meta = document.createElement('div');
          meta.style.cssText = 'font-size:12px; color:var(--
muted)';
          const s = new Date(r.s), e = new Date(r.e);
          meta.textContent = s.toLocaleTimeString([],
{hour:'2-digit', minute:'2-digit'}) + ' – ' +
e.toLocaleTimeString([], {hour:'2-digit', minute:'2-digit'});

          const rientro = document.createElement('div');
          rientro.style.cssText = 'font-size:12px';
          if (r.t==='c' && r.l) rientro.textContent = r.l;

          left.appendChild(badge);
          left.appendChild(unit);
          left.appendChild(meta);
          if (rientro.textContent) left.appendChild(rientro);

          const btn = document.createElement('button');
          btn.textContent = 'Apri in Calendar';
          btn.onclick = ()=>{
            const q = encodeURIComponent(r.r || r.g || '');
            window.open('https://calendar.google.com/
calendar/u/0/r/search?q=' + q, '_blank');
          };

          right.appendChild(btn);
          card.appendChild(left);
          card.appendChild(right);
          wrap.appendChild(card);
        });

        host.appendChild(wrap);
      });

      $('count').textContent = data.length + ' elementi';
      if (!days.length) host.innerHTML = '<div
class="muted">Nessun elemento nel periodo.</div>';
    }

    /*
================================================================
     * 9) SERVER CALLS
     *
================================================================
*/
    let _debTimer;
```

```
    function refreshDebounced(){ clearTimeout(_debTimer);
_debTimer = setTimeout(refresh, 200); }
    function refresh(){
      const from = fromEl().value, to = toEl().value;
      if (!from || !to) return;
      const btn = $('btn'); if (btn) { btn.disabled = true;
btn.textContent = 'Aggiorno…'; }
      let watchdog = setTimeout(() => { if (btn)
{ btn.disabled = false; btn.textContent = 'Aggiorna'; } },
10000);

      google.script.run
        .withSuccessHandler(res => {
          clearTimeout(watchdog);
          allRows = (Array.isArray(res) ? res :
[]).map(mapRow);
          buildFiltersFromRows(allRows);
          applyFilters();
          if (mode==='cal'){ ensureTimeline();
applyDataBatched(rows); renderDayGutters();
syncScrollBar(); }
          else { renderList(rows); }
          if (btn){ btn.disabled = false; btn.textContent =
'Aggiorna'; }
          toast('Aggiornato','success');
        })
        .withFailureHandler(err => {
          clearTimeout(watchdog);
          if (btn){ btn.disabled = false; btn.textContent =
'Aggiorna'; }
          alert('Errore lato server:\n' + ((err &&
err.message) ? err.message : String(err)));
          toast('Errore aggiornamento','error');
        })
        .getRowsSlim(from, to);
    }

    /*
================================================================
     * 10) THEMING, COMFORT, SHORTCUTS
     *
================================================================
*/
    function applySystemThemeClass(){
      const prefersDark = window.matchMedia &&
window.matchMedia('(prefers-color-scheme: dark)').matches;
```

```
      document.documentElement.classList.toggle('prefers-
dark', (document.documentElement.dataset.theme==='system' &&
prefersDark));
    }
    $('themeSel').addEventListener('change', ()=>{
      const val = $('themeSel').value;
      document.documentElement.dataset.theme = val;
      applySystemThemeClass();
      if(mode==='cal' && timeline) timeline.redraw();
      savePrefs(); drawHeatmapDebounced();
    });
    if (window.matchMedia){
      window.matchMedia('(prefers-color-scheme:
dark)').addEventListener('change',
()=>{ applySystemThemeClass(); drawHeatmapDebounced(); });
    }

    $('comfort').addEventListener('change',()=>{
      document.documentElement.dataset.comfort = $
('comfort').checked?'1':'0';
      if(mode==='cal' && timeline) timeline.redraw();
      savePrefs();
    });

    window.addEventListener('keydown', (e)=>{
      if ((e.ctrlKey||e.metaKey) &&
e.key.toLowerCase()==='k'){ e.preventDefault(); $
('omni').focus(); return; }
      if (e.key==='ArrowLeft'){ moveDays(-1); }
      if (e.key==='ArrowRight'){ moveDays(1); }
      if (e.key.toLowerCase()==='w'){ zoom('week'); }
      if (e.key.toLowerCase()==='m'){ zoom('month'); }
      if (e.key.toLowerCase()==='t'){ goToday(); }
    });

    /*
================================================================
     * 11) ASIDE: toggle, splitter, omni, heatmap, note,
conflict radar
     *
================================================================
*/
    $('toggleAside').onclick = ()=>{
      const page = $('page');
      page.dataset.aside = (page.dataset.aside === 'open') ?
'closed' : 'open';
```

```
      if (mode==='cal' && timeline)
setTimeout(()=>{ timeline.redraw(); renderDayGutters();
syncScrollBar(); }, 150);
      savePrefs();
    };

    // Splitter drag + click per riaprire
    (function(){
      const split = $('splitter');
      let down=false, startX=0, startW=0;
      split.addEventListener('mousedown', e=>{
        const page = $('page');
        if (page.dataset.aside==='closed'){ // click singolo
per aprire
          page.dataset.aside='open';
          if (mode==='cal' && timeline)
setTimeout(()=>{ timeline.redraw(); renderDayGutters();
syncScrollBar(); }, 150);
          savePrefs();
          return;
        }
        down=true; startX=e.clientX;

startW=parseInt(getComputedStyle(document.documentElement).ge
tPropertyValue('--aside-w'));
        document.body.style.userSelect='none';
      });
      window.addEventListener('mousemove', e=>{
        if(!down) return;
        const dx = startX - e.clientX;
        const newW = Math.max(260, Math.min(560, startW +
dx));
        document.documentElement.style.setProperty('--aside-
w', newW+'px');
      });
      window.addEventListener('mouseup', ()=>{
        if(!down) return;
        down=false; document.body.style.userSelect='';
        if (mode==='cal' && timeline) { timeline.redraw();
renderDayGutters(); syncScrollBar(); }
        savePrefs();
      });
      split.addEventListener('dblclick', ()=>{
        const page=$('page');
        page.dataset.aside =
(page.dataset.aside==='open')?'closed':'open';
```

```
        if (mode==='cal' && timeline)
setTimeout(()=>{ timeline.redraw(); renderDayGutters();
syncScrollBar(); }, 150);
        savePrefs();
      });
    })();

    // Omni chips
    [...document.querySelectorAll('.chip-btn[data-
cmd]')].forEach(b=>{
      b.addEventListener('click', ()=>
runOmni(b.dataset.cmd));
    });
    $('omni').addEventListener('keydown', e=>{
      if (e.key === 'Enter'){ e.preventDefault(); runOmni($
('omni').value.trim()); }
    });

    function runOmni(text){
      const q = (text||'').toLowerCase();
      if (!q) return;

      if (/^oggi$/.test(q)){ goToday(); return; }
      if (/^settimana$/.test(q)){ zoom('week'); return; }
      if (/^mese$/.test(q)){ zoom('month'); return; }
      if (/^solo pulizie$/.test(q)){ $
('onlyClean').checked=true; applyFilters(); (mode==='cal')?
applyDataBatched(rows):renderList(rows); savePrefs(); return;
}
      if (/^solo rientro$/.test(q)){ $
('onlyRientro').checked=true; applyFilters(); (mode==='cal')?
applyDataBatched(rows):renderList(rows); savePrefs(); return;
}
      if (/^zen$/.test(q))
{ document.documentElement.dataset.zen =
(document.documentElement.dataset.zen==='1'?'0':'1'); return;
}
      if (/^chiaro$/.test(q))
{ document.documentElement.dataset.theme='light'; $
('themeSel').value='light'; applySystemThemeClass();
savePrefs(); drawHeatmapDebounced(); return; }
      if (/^scuro$/.test(q))
{ document.documentElement.dataset.theme='dark'; $
('themeSel').value='dark'; applySystemThemeClass();
savePrefs(); drawHeatmapDebounced(); return; }
      if (/^sistema$/.test(q))
{ document.documentElement.dataset.theme='system'; $
```

```
('themeSel').value='system'; applySystemThemeClass();
savePrefs(); drawHeatmapDebounced(); return; }

      const m = q.match(/dal\s+(\d{4}-\d{2}-\d{2})\s+al\s+
(\d{4}-\d{2}-\d{2})/);
      if (m){
        fromEl().value=m[1]; toEl().value=m[2];
setRangeLabel(); refreshDebounced(); return;
      }

      const opt = [...propEl().options].find(o =>
decodeURIComponent(o.value).toLowerCase().includes(q));
      if (opt){ propEl().value=opt.value; applyFilters();
(mode==='cal')?applyDataBatched(rows):renderList(rows);
savePrefs(); return; }

      qEl().value = text;
      applyFilters();
      (mode==='cal')?applyDataBatched(rows):renderList(rows);
    }

    /* Heatmap occupazione */
    const drawHeatmapDebounced = debounce(drawHeatmap, 120);
    function drawHeatmap(){
      const canvas = $('occCanvas'); if (!canvas) return;
      const ctx = canvas.getContext('2d');
      const rect = canvas.getBoundingClientRect();
      canvas.width = Math.floor(rect.width *
(window.devicePixelRatio||1));
      canvas.height = Math.floor(72 *
(window.devicePixelRatio||1));
      ctx.scale((window.devicePixelRatio||1),
(window.devicePixelRatio||1));
      ctx.clearRect(0,0,rect.width,72);

      const start = new Date(fromEl().value + 'T00:00:00');
      const days = 30;
      const units = new Set(allRows.map(r=>r.g));
      const totalU = Math.max(units.size, 1);

      const pct = [];
      for(let i=0;i<days;i++){
        const d0 = new Date(start); d0.setDate(d0.getDate()
+i);
        const d1 = new Date(d0); d1.setDate(d0.getDate()+1);
        const dayUnits = new Set();
        allRows.forEach(r=>{
          if (r.t!=='s') return;
```

```
          const s=new Date(r.s), e=new Date(r.e);
          if (e>d0 && s<d1) dayUnits.add(r.g);
        });
        pct.push(dayUnits.size/totalU);
      }

      const w = rect.width/days, h=56, baseY=10;
      let sum=0;
      for(let i=0;i<days;i++){
        sum += pct[i];
        const x=i*w;
        const barH = Math.max(2, h*pct[i]);
        ctx.fillStyle = pct[i]>0.7 ?
getComputedStyle(document.documentElement).getPropertyValue('
--heat-peak') :
getComputedStyle(document.documentElement).getPropertyValue('
--heat');
        ctx.fillRect(x, baseY + (h-barH), Math.max(1,w-2),
barH);
      }
      const avg = sum/days;
      $('occPct').textContent = 'media ' +
Math.round(avg*100) + '%';
    }

    /* Conflict radar (UI) */
    function findConflicts(arr){
      const overlaps=(a,b,c,d)=> (new Date(a)<new Date(d) &&
new Date(c)<new Date(b));
      const byG={};
      arr.forEach(r=>{ (byG[r.g] ||= []).push(r); });
      const list=[];
      Object.entries(byG).forEach(([g,rows])=>{
        const stays=rows.filter(r=>r.t==='s');
        const cleans=rows.filter(r=>r.t==='c');
        cleans.forEach(c=>{
          stays.forEach(s=>{ if(overlaps(c.s,c.e,s.s,s.e))
list.push({type:'Pulizie sovrapposte al soggiorno', g,
when:c.s, c, s}); });
        });
        for(let i=0;i<cleans.length;i++){
          for(let j=i+1;j<cleans.length;j++){

if(overlaps(cleans[i].s,cleans[i].e,cleans[j].s,cleans[j].e))
              list.push({type:'Pulizie sovrapposte tra loro',
g, when:cleans[i].s, c:cleans[i], s:cleans[j]});
          }
        }
```

```
      });
      return list.sort((a,b)=> new Date(a.when)-new
Date(b.when));
    }

    function updateConflictsUI(){
      const confs = findConflicts(allRows);
      const card = $('confCard'), listEl=$('confList'),
moreEl=$('confMore');
      if (!confs.length){ card.style.display='none';
return; }
      card.style.display='block';
      listEl.innerHTML = confs.slice(0,8).map(c=>{
        const ds=new Date(c.when).toLocaleString();
        return `<div class="conf">
          <div class="t"><strong>${c.g}</strong><br><span
class="ghost-small">${c.type}</span><br><span class="ghost-
small">${ds}</span></div>
          <button class="chip-btn" data-jump="$
{c.when}">Vai</button>
        </div>`;
      }).join('');
      moreEl.textContent = confs.length>8 ? `+$
{confs.length-8} altre` : '';
      [...listEl.querySelectorAll('[data-
jump]')].forEach(b=>{
        b.addEventListener('click',()=> jumpTo(new
Date(b.dataset.jump)));
      });
    }

    function jumpTo(date){
      if(!timeline) return;
      const d = new Date(date);
      const range= timeline.getWindow();
      const span= range.end - range.start;
      const start = new Date(d - span/2);
      const end = new Date(d + span/2);
      timeline.setWindow(start, end, {animation:true});
    }

    /* Note rapide */
    function noteKey(dateISO){ return 'hn_notes_v1:'+dateISO;
}
    function loadNote(){
      const d = $('noteDate').value;
      const t = localStorage.getItem(noteKey(d)) || '';
      $('noteTxt').value = t;
```

```
      $('noteInfo').textContent = t ? 'Salvata' : '—';
    }
    $('noteToday').onclick = ()=>{ $('noteDate').value = new
Date().toISOString().slice(0,10); loadNote(); };
    $('saveNote').onclick =
()=>{ localStorage.setItem(noteKey($('noteDate').value), $
('noteTxt').value); $('noteInfo').textContent='Salvata';
toast('Nota salvata','success'); };
    $('clearNote').onclick =
()=>{ localStorage.removeItem(noteKey($('noteDate').value));
$('noteTxt').value=''; $('noteInfo').textContent='—';
toast('Nota rimossa'); };
    $('noteDate').addEventListener('change', loadNote);

    // Utilità
    $('jumpNow').onclick = ()=>{
      if(!timeline) return;
      const now = new Date();
      const range = timeline.getWindow();
      const span = range.end - range.start;
      const start = new Date(now - span/2);
      const end = new Date(now + span/2);
      timeline.setWindow(start, end, {animation:true});
    };
    $('resetFilters').onclick = ()=>{
      qEl().value=''; onlyCleanEl().checked=false;
onlyRientroEl().checked=false; propEl().value='';
      applyFilters(); (mode==='cal')?
applyDataBatched(rows):renderList(rows); savePrefs();
    };

    /*
================================================================
     * 12) TOAST & UTILS
     *
================================================================
*/
    function toast(msg,type='info'){
      const t=document.createElement('div');
      t.textContent=msg;
      t.style.cssText='position:fixed; right:14px;
bottom:14px; background:var(--panel); color:var(--fg);
border:1px solid var(--border); border-radius:10px;
padding:10px 12px; box-shadow:var(--shadow); z-index:9999;';
      if(type==='success') t.style.borderColor='#2ecc71';
      if(type==='error') t.style.borderColor='var(--bad)';
      document.body.appendChild(t);
```

```
      setTimeout(()=>{ t.style.transition='opacity .3s ease,
transform .3s ease'; t.style.opacity='0';
t.style.transform='translateY(6px)';
setTimeout(()=>t.remove(), 300); }, 1600);
    }
    function debounce(fn, ms){ let t; return
(...a)=>{ clearTimeout(t); t=setTimeout(()=>fn(...a), ms); };
}

    /*
================================================================
     * 13) INIT
     *
================================================================
*/
    function setRangeDefaults(){
      const f = INIT_FROM || new
Date().toISOString().slice(0,10);
      const t = (window.innerWidth<=900 ? f : (INIT_TO ||
f));
      fromEl().value = f; toEl().value = t;
    }

    (function(){
      applySystemThemeClass();

      setRangeDefaults();
      setMode(mode);
      setRangeLabel();
      buildFiltersFromRows(allRows);
      applyFilters();
      if (mode==='cal'){ ensureTimeline();
applyDataBatched(rows); renderDayGutters(); ensureObserver();
syncScrollBar(); } else { renderList(rows); }

      // listeners
      window.onresize = () => { if (mode==='cal' && timeline)
{ timeline.redraw(); renderDayGutters(); syncScrollBar(); }
drawHeatmapDebounced(); };
      $('from').addEventListener('change',
()=>{ setRangeLabel(); refreshDebounced();
drawHeatmapDebounced(); });
      $('to').addEventListener('change',
()=>{ setRangeLabel(); refreshDebounced();
drawHeatmapDebounced(); });
      $('q').addEventListener('input', ()=>{ applyFilters();
(mode==='cal') ? applyDataBatched(rows) : renderList(rows);
savePrefs(); });
```

```
      $('propFilter').addEventListener('change',
()=>{ applyFilters(); (mode==='cal') ? applyDataBatched(rows)
: renderList(rows); savePrefs(); });
      $('onlyClean').addEventListener('change',
()=>{ applyFilters(); (mode==='cal') ? applyDataBatched(rows)
: renderList(rows); savePrefs(); });
      $('onlyRientro').addEventListener('change',
()=>{ applyFilters(); (mode==='cal') ? applyDataBatched(rows)
: renderList(rows); savePrefs(); });

      // load prefs
      loadPrefs();

      // init note & heatmap & conflicts
      $('noteDate').value = new
Date().toISOString().slice(0,10);
      loadNote();
      drawHeatmapDebounced();
      updateConflictsUI();
    })();

    // Totali pulizie (se il backend espone
getCleaningTotals)
    function loadTotals(fromISO, toISO){
      if (!(google && google.script && google.script.run))
return;
      google.script.run
        .withSuccessHandler(monthly=>{
          const el=$('totalsBody'); el.innerHTML='';
          const months = Object.keys(monthly).sort();
          if (!months.length){ el.textContent='Nessuna
pulizia nel periodo.'; return; }
          months.forEach(m=>{
            const {total, perUnit} = monthly[m];
            const row=document.createElement('div');
            row.style.marginBottom='6px';
            row.innerHTML = `<b>${m}</b>: € $
{Number(total).toFixed(2)}<br>`+
              Object.entries(perUnit).map(([u,v])=>`${u}: € $
{Number(v).toFixed(2)}`).join('<br>');
            el.appendChild(row);
          });
          $('exportCsv').onclick=()=>{
            const lines=['mese,unità,totale'];
            Object.entries(monthly).forEach(([m,v])=>{
              lines.push(`${m},TOTALE,${v.total}`);
```

```
Object.entries(v.perUnit).forEach(([u,tot])=>lines.push(`${m},${u},${tot}`));
            });
            const blob=new Blob([lines.join('\n')],{type:'text/csv'});
            const a=document.createElement('a');
            a.href=URL.createObjectURL(blob); a.download='totali-pulizie.csv'; a.click();
          };
        })
        .getCleaningTotals(fromISO,toISO);
    }
    loadTotals(INIT_FROM, INIT_TO);
  </script>
</body>
</html>
```