Generate a complete bundle /paradigm/intdraction-design, with full UI and psychometrics, analytical psychology interpolating with platform awareness, graph metabolism and extend type in business intellience,building a fully interactive immersive Media interaction design paradigm playground with drag and drop multimodal cards building storyboard following composer code* with abstract, structure, topology, definition, formalism, schema, directives, config and node js scripts, to generate a full a analysis over the structure and all your knowledge interpolated to our interactions and success, creation, integrating the structure of github project in reference, following code, ethos, structure, dynamic, intelligence, schema, business intelligence to create a complete graphical playground UI with metrics data visualization and psychometric tests, running differential analysis on user over differential metrics from interactions, time series over meta data, heuristics and interactive forms, immersive gaming experience assessing their psyche in terms of analytical psychology, to build entire MEDIA INTERACTION DESIGN PARADIGM. Example formalism type stricture: META-PHYSIQUE | paradigms in zeroth observer empirism
The abstract metabolism (interaction media design) as an art systemic which medium overlaps on spaces, cosmos and abstract time, projecting heuristics to hyperspace, humming noises disturbance, fizzy atmosphere, while dimensions collapses reality into paradigms that shape reality out of paradoxes.
```
KEWORDS

Haiku
Inkspot
Ikigai


LAYER
TOPIC: Sensitivity | Sensibility
ABSTRACT: lexical paradigms, not synonym

SINGLE AXIS
(Blossom,emotion,substance,deepness,emancipation,critics,meditation)

HEURISTICS
Out of loving

THESIS
Communication is bidirectional and leveraging calculus proof therefore evoking. Evoking is in regard to something

FORMALISM
σ(n+1) = σ(n)  et σ(n+1) = -σ(n) E ϕ(t0-t)
σ(n+1) = σ(n)+1 E ϕ
```

REF
https://en.wikiversity.org/wiki/Zeroth_order_logic.

INPUT PRESET: ```Use generative art theory and generic algorithms across contemporary and pioneers implementation (previous link, ressources on Verona, Tiles). Extrapolates grids accross all layout as a layer heatmap, scaling from viewport to containers and single unit element: pixel, square, as a formal definition of a grid unit ordinated in template layout definition, accross nested classes. The classes interpolate accross different domains, using Munsell colorimetric 3D space as complexe projection, generating unit (square) pixel from ismorphic projection through complexe to surface planes, which integrate the heatmap as (seeded) noise function, with pixel unit precision, randmized in space, distributed in time and heatmap evolution. Add mouse and touch interactions as timeseries of pas interactions with the heatmap, diffential and offsets, integrating waves function, curl and divergence, through pattern extrapolation in heatmap nested classes definition. Label this as interaction, which is a second layer of the static design grid, the pixel the follows game of life rules with the type definition as randomized paramaters extrapolated to heuristics maps
Use allocation for PNP in the context of Conway Game of life, refactoring implementation toward web standardsinterpolate allocative properties system as an heuristics decisions tree graph
as emergent and evolutive rules towards Turing completeness
(the heatmap is an underlayer background, following generative art tiles deformation)``

c’est le pattern de base, et du coup on cherche un fonction bien stochastique avec les déformations
un raccourcis vers un schéma, interprété comme des fonctions signées distantes, équation différentielle ordinaire, multiple ODE
use kinectic and Meta kinektic, abstract machines for UX dynamics from Jean tinguely
creating: « une narrative artistique pionière sur quelque chose de raw »```.
Generate complete bundle with downloadable link

*composer code:

<!DOCTYPE html>
<html lang="en" data-theme="light">
<head>
<meta charset="utf-8"/>
<meta name="viewport" content="width=device-width, initial-scale=1"/>
<title>Factoryz Notepad — Magnetic Grid</title>

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/
bootstrap.min.css" rel="stylesheet">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/codemirror@5.65.16/
lib/codemirror.css">
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/codemirror@5.65.16/
theme/material-darker.css">
<link rel="stylesheet" href="/public/cms.css" />
<style>
:root{
  --cell-size: 140px;
  --gap: 6px;
  --border: #e5e7eb;
  --panel-w: 380px;
  --navbar-h: 56px;
}
*{box-sizing:border-box}
html,body{height:100%}
body{font-family:"Helvetica Neue",Helvetica,Arial,system-ui,-apple-
system,Segoe UI,Roboto,'Apple Color Emoji','Segoe UI Emoji';
background:#0d1117; color:#e5e7eb}
body::before{
  content:"";
  position:fixed;
  inset:0;
  background: radial-gradient(900px 700px at 10% 10%, rgba(28,42,58,0.6),
transparent 60%),
              radial-gradient(1200px 900px at 90% 20%, rgba(12,32,28,0.4),
transparent 60%);
  z-index:-2;
}
#bg{position:fixed; inset:0; z-index:-3;}
#bg canvas{position:absolute; inset:0; width:100%; height:100%;}
html[data-theme="dark"] body{background:#0b0d10; color:#e5e7eb}
html[data-theme="dark"] .card{background:#0f1319; color:#e5e7eb}
html[data-theme="dark"] .border{border-color:#1f2937 !important}

.navbar-brand{letter-spacing:.2px}

.grid-wrap{overflow:auto}
.grid{
  position:relative;
  display:grid;
  grid-auto-rows: var(--cell-size);
  grid-auto-flow: dense;
  gap: var(--gap);
  justify-content: start;
```

```
  align-content: start;
  padding-bottom: 40vh; /* space for drag */
}
.grid.placeholder-on .placeholder{display:block}
.placeholder{
  display:none;
  border:1px dashed var(--border);
  border-radius: .5rem;
  background: repeating-linear-gradient(45deg, rgba(0,0,0,.03), rgba(0,0,0,.03)
8px, transparent 0, transparent 16px);
}

.cell{
  position: relative;
  border:1px solid var(--border);
  border-radius:.6rem;
  overflow:hidden;
  background: var(--baseColor, #4b5563);
  transition: box-shadow .15s ease, transform .15s ease;
  user-select: none;
  outline: none;
}
.cell.locked{opacity:.9; cursor:not-allowed}
.cell:hover{box-shadow:0 10px 24px rgba(0,0,0,.18)}
.cell:focus-visible{outline:2px solid #60a5fa; outline-offset:2px}
.cell .badge-lock{position:absolute; top:.35rem; right:.35rem; z-index:4}

.cell img.cover{
  position:absolute; inset:0; width:100%; height:100%; object-fit:cover;
opacity:.88; z-index:1;
}

.cell-content{
  position:relative;
  z-index:2;
  height:100%;
  padding:.6rem;
  display:flex;
  flex-direction:column;
  align-items:center;
  justify-content:center;
  text-align:center;
  color:#fff;
}
.cell-content.has-media{ justify-content:space-between; align-items:stretch; }
.cell-content.has-media iframe{ flex:1; min-height:120px; border-radius:.5rem; }
.cell-video{
```

```
  position:absolute;
  inset:0;
  width:100%;
  height:100%;
  object-fit:cover;
  z-index:0;
  opacity:.95;
}
.cell-sine{
  position:absolute;
  left:10px;
  right:10px;
  bottom:10px;
  height:18px;
  z-index:2;
  opacity:.75;
  pointer-events:none;
}
.cell-sine svg{ width:100%; height:100%; display:block; }
.cell-sine path{
  stroke: rgba(255,255,255,0.85);
  stroke-width: 2;
  fill: none;
  filter: drop-shadow(0 2px 6px rgba(0,0,0,0.35));
}
.cell-content h3{font-size:1rem; margin:.1rem 0}
.cell-content .subtitle{font-size:.8rem; opacity:.95}
.cell-content .subtext{font-size:.75rem; opacity:.95}
.cell .links{display:flex; gap:.25rem; flex-wrap:wrap; margin-top:.25rem}
.cell .links a{font-size:.7rem; color:#fff; text-decoration:none;
background:rgba(0,0,0,.35); padding:.15rem .35rem; border-radius:.4rem}
.cell .links a:hover{text-decoration:underline}

.handle-drag, .handle-resize, .handle-resize-r, .handle-resize-b{
  position:absolute; z-index:5;
}
.handle-drag{
  top:.35rem; left:.35rem;
  width:22px; height:22px;
  border-radius:.4rem;
  background:rgba(0,0,0,.25);
  display:flex; align-items:center; justify-content:center;
  color:#fff; font-size:14px; cursor:grab;
}
.handle-drag:active{cursor:grabbing}
.handle-resize{ right:.2rem; bottom:.2rem; width:14px; height:14px; border-
right:3px solid rgba(255,255,255,.9); border-bottom:3px solid
```

```
rgba(255,255,255,.9); cursor:nwse-resize; opacity:.85}
.handle-resize-r{ right:-2px; top:0; width:8px; height:100%; cursor:ew-resize; }
.handle-resize-b{ left:0; bottom:-2px; height:8px; width:100%; cursor:ns-
resize; }

.drop-indicator{
  position:absolute; pointer-events:none; z-index:2000;
  border:2px dashed #0ea5e9; border-radius:.6rem;
  background: rgba(14,165,233,.08);
  box-shadow: inset 0 0 0 1px rgba(14,165,233,.15);
}

.mini-toolbar{
  position:absolute; z-index:6; top:.35rem; right:.35rem;
  display:flex; gap:.25rem;
}
.mini-toolbar .btn{ --bs-btn-padding-y: .1rem; --bs-btn-padding-x: .35rem; --
bs-btn-font-size: .7rem }

/* Slide panel */
.side-panel{
  position: fixed;
  top: var(--navbar-h);
  right: 0;
  width: var(--panel-w);
  max-width: 96vw;
  height: calc(100% - var(--navbar-h));
  border-left: 1px solid var(--border);
  background: var(--panel-bg, #fff);
  transform: translateX(100%);
  transition: transform .25s ease;
  z-index: 1060;
  overflow:auto;
  padding:.5rem;
  box-shadow:-10px 0 24px rgba(0,0,0,.18);
}
html[data-theme="dark"] .side-panel{ --panel-bg:#0f1319 }
.side-panel.open{ transform: translateX(0) }
.panel-overlay{ position:fixed; inset:0; background:rgba(0,0,0,.35);
display:none; z-index:1050 }

/* Lightbox centered preview */
.lightbox{ position:fixed; inset:0; display:none; align-items:center; justify-
content:center; z-index:2000 }
.lightbox.show{ display:flex }
.lightbox .backdrop{ position:absolute; inset:0; background:rgba(0,0,0,.55);
backdrop-filter: blur(2px) }
```

```
.lightbox .content{ position:relative; width:min(96vw,1100px);
height:min(92vh,820px); background:#fff; border-radius:.8rem; overflow:auto;
box-shadow:0 24px 48px rgba(0,0,0,.35) }
html[data-theme="dark"] .lightbox .content{ background:#0f1319;
color:#e5e7eb }
.lightbox .close{ position:absolute; top:.5rem; right:.75rem }

/* Placeholder badge */
.placeholder::before{
  content:"";
}

/* Tooltips */
[data-title]{ position:relative }
[data-title]:hover::after{
  content:attr(data-title);
  position:absolute; white-space:nowrap; z-index:20;
  bottom:100%; right:0; transform:translateY(-4px);
  background:rgba(0,0,0,.85); color:#fff; font-size:.7rem; padding:.2rem .4rem;
border-radius:.35rem;
}

code.inline{ background:rgba(0,0,0,.2); padding:.05rem .25rem; border-
radius:.25rem; }

.magnetic-bridge{margin-top:2rem}
.bridge-card{background:#fff;border:1px solid var(--border);border-
radius:1rem;padding:1.25rem;box-shadow:0 20px 60px rgba(15,23,42,.08)}
html[data-theme="dark"] .bridge-card{background:#0f1319}
.bridge-card h5{margin-bottom:.6rem}
.bridge-console{background:#0b1120;color:#e5e7eb;padding:1rem;border-
radius:.8rem;font-size:.85rem;min-height:120px;overflow:auto}
.secret-toggle{display:flex;align-items:center;gap:.4rem}
.secret-toggle input{width:18px;height:18px}
#gridOverlay{
  position:absolute;
  inset:0;
  pointer-events:none;
  opacity:0;
  background:
    linear-gradient(to right, rgba(255,255,255,0.08) 1px, transparent 1px),
    linear-gradient(to bottom, rgba(255,255,255,0.06) 1px, transparent 1px);
  background-size: var(--cell-size) var(--cell-size);
  mix-blend-mode: overlay;
}
.hidden{display:none}
.bridge-card form .form-control{margin-bottom:.6rem}
```

```
.bridge-card .btn{min-width:180px}
.fibo-menu{
  display:grid;
  grid-template-columns:repeat(3, minmax(0, 1fr));
  grid-auto-rows: 48px;
  gap:8px;
}
.fibo-item{
  border:1px solid rgba(148,163,184,.3);
  border-radius:12px;
  background:rgba(15,23,42,.6);
  color:#e2e8f0;
  display:flex;
  align-items:center;
  justify-content:center;
  font-size:.75rem;
  cursor:grab;
  user-select:none;
}
.fibo-item.dragging{opacity:.5}
.fibo-1{grid-column:span 1;grid-row:span 1;}
.fibo-2{grid-column:span 2;grid-row:span 1;}
.fibo-3{grid-column:span 3;grid-row:span 1;}
.fibo-5{grid-column:span 3;grid-row:span 2;}

</style>
</head>
<body>
<div id="bg">
  <canvas id="gl"></canvas>
  <canvas id="fallback" class="hidden"></canvas>
  <div id="gridOverlay" aria-hidden="true"></div>
</div>
<nav class="navbar navbar-expand-lg border-bottom bg-body-tertiary sticky-
top" id="topNav">
  <div class="container-fluid">
    <a class="navbar-brand fw-semibold" href="/">Singularity Composer</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target="#navmenu">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navmenu">
      <ul class="navbar-nav me-auto">
        <li class="nav-item dropdown">
          <button class="nav-link dropdown-toggle btn btn-link" type="button"
data-bs-toggle="dropdown">Data</button>
          <ul class="dropdown-menu">
```

```
          <li><button class="dropdown-item" type="button" data-
action="import-url">Load from URL...</button></li>
          <li><button class="dropdown-item" type="button" data-
action="import-file">Import JSON...</button></li>
          <li><hr class="dropdown-divider"/></li>
          <li><button class="dropdown-item" type="button" data-action="save-
local">Save to Local Storage</button></li>
          <li><button class="dropdown-item" type="button" data-action="load-
local">Load from Local Storage</button></li>
          <li><button class="dropdown-item" type="button" data-action="clear-
local">Clear Local Storage</button></li>
          <li><hr class="dropdown-divider"/></li>
          <li><button class="dropdown-item" type="button" data-
action="download-json">Download JSON</button></li>
          <li><button class="dropdown-item" type="button" data-
action="export-md">Export Markdown</button></li>
          <li><button class="dropdown-item" type="button" data-
action="export-html">Export HTML Snapshot</button></li>
          <li><button class="dropdown-item" type="button" data-
action="export-csv">Export CSV</button></li>
          <li><button class="dropdown-item" type="button" data-
action="export-txt">Export TXT</button></li>
          <li><button class="dropdown-item" type="button" data-
action="export-svg">Export SVG</button></li>
          <li><button class="dropdown-item" type="button" data-
action="export-pdf">Export PDF (Golden Ratio)</button></li>
          <li><hr class="dropdown-divider"/></li>
          <li><button class="dropdown-item" type="button" data-action="api-
save">Save to API</button></li>
          <li><button class="dropdown-item" type="button" data-action="api-
load">Load from API</button></li>
          <li><hr class="dropdown-divider"/></li>
          <li><button class="dropdown-item" type="button" data-action="open-
playbook-presets">Playbook Presets</button></li>
          <li><button class="dropdown-item" type="button" data-
action="manage-planches">Planches (Cloud)</button></li>
        </ul>
      </li>
      <li class="nav-item dropdown">
        <button class="nav-link dropdown-toggle btn btn-link" type="button"
data-bs-toggle="dropdown">View</button>
        <ul class="dropdown-menu">
          <li><button class="dropdown-item" type="button" data-
action="toggle-theme">Toggle Theme</button></li>
          <li><button class="dropdown-item" type="button" data-action="zoom-
in">Zoom In</button></li>
          <li><button class="dropdown-item" type="button" data-action="zoom-
```

```
out">Zoom Out</button></li>
          <li><button class="dropdown-item" type="button" data-action="reset-
zoom">Reset Zoom</button></li>
          <li><hr class="dropdown-divider"/></li>
          <li class="px-3 py-1 small text-muted">Group n (cols)</li>
          <li class="px-3 py-2">
           <div class="input-group input-group-sm">
            <input id="group-n" class="form-control" type="number" value="2"
min="1"/>
             <button class="btn btn-outline-secondary" data-action="group-
apply">Apply to Selected</button>
           </div>
          </li>
         </ul>
        </li>
        <li class="nav-item dropdown">
         <button class="nav-link dropdown-toggle btn btn-link" type="button"
data-bs-toggle="dropdown">Tools</button>
         <ul class="dropdown-menu">
          <li><button class="dropdown-item" type="button" data-
action="filters">Filters…</button></li>
          <li><button class="dropdown-item" type="button" data-
action="reorder-rows">Reorder Rows…</button></li>
          <li><button class="dropdown-item" type="button" data-action="bulk-
edit">Bulk Edit…</button></li>
         </ul>
        </li>
      </ul>
      <div class="d-flex gap-2 align-items-center">
       <div class="form-check form-switch d-flex align-items-center gap-2 text-
muted small m-0">
        <input class="form-check-input" type="checkbox"
id="autoSaveSwitch">
        <label class="form-check-label text-nowrap"
for="autoSaveSwitch">Auto-save</label>
       </div>
       <div class="form-check form-switch d-flex align-items-center gap-2 text-
muted small m-0 me-2">
        <input class="form-check-input" type="checkbox" id="publicSwitch">
        <label class="form-check-label text-nowrap"
for="publicSwitch">Public</label>
       </div>
       <div class="btn-group btn-group-sm">
        <button class="btn btn-outline-secondary" data-lang="en">EN</button>
        <button class="btn btn-outline-secondary" data-lang="fr">FR</button>
       </div>
       <button class="btn btn-outline-secondary btn-sm" id="togglePanelBtn"
```

```
aria-controls="sidePanel" aria-expanded="false">Panel</button>
      <button class="btn btn-primary btn-sm" data-action="add-cell">Add
Cell</button>
    </div>
   </div>
  </div>
</nav>

<div class="container-fluid py-2">
  <div class="row g-2">
   <div class="col-12">
    <div class="grid-wrap border rounded-3 p-2">
     <div id="grid" class="grid" role="grid" aria-label="Magnetic grid"></div>
    </div>
   </div>
  </div>
</div>

<div id="panelOverlay" class="panel-overlay" tabindex="-1" aria-
hidden="true"></div>

<aside id="sidePanel" class="side-panel" aria-hidden="true">
  <div class="card shadow-sm mb-2">
   <div class="card-header d-flex align-items-center justify-content-
between">
    <span class="fw-semibold">Editor</span>
    <div class="form-check form-switch m-0">
     <input class="form-check-input" type="checkbox" id="livePreviewSwitch"
checked>
     <label class="form-check-label small" for="livePreviewSwitch">Live
preview</label>
    </div>
   </div>
   <div class="card-body small">
    <div class="mb-2">
     <label class="form-label small" for="jsonUrl">JSON URL</label>
     <div class="input-group input-group-sm">
      <input type="url" id="jsonUrl" class="form-control" placeholder="https://
example.com/data.json">
      <button class="btn btn-outline-secondary" data-action="import-
url">Load</button>
     </div>
    </div>
    <hr/>
    <form id="cellForm" class="small">
     <input type="hidden" id="cell-id">
     <div class="row g-2">
```

```
        <div class="col-6">
          <label class="form-label" for="cell-type">Type</label>
          <select id="cell-type" class="form-select form-select-sm">
            <option value="text">Text</option>
            <option value="markdown">Markdown</option>
            <option value="html">HTML</option>
            <option value="js">JS</option>
            <option value="css">CSS</option>
            <option value="code">Code</option>
            <option value="phenotype">Phenotype 3D</option>
            <option value="exosphere">Exosphere</option>
            <option value="drawing">Drawing</option>
            <option value="link">Link</option>
            <option value="image">Image</option>
            <option value="video">Video</option>
            <option value="audio">Audio</option>
          </select>
        </div>
        <div class="col-6"><label class="form-label" for="cell-title">Title</
label><input class="form-control form-control-sm" id="cell-title" type="text"
list="titleSuggestions"></div>
        <div class="col-6"><label class="form-label" for="cell-
subtitle">Subtitle</label><input class="form-control form-control-sm"
id="cell-subtitle" type="text"></div>
        <div class="col-6"><label class="form-label" for="cell-
exponent">Exponent</label><input class="form-control form-control-sm"
id="cell-exponent" type="text"></div>
        <div class="col-12"><label class="form-label" for="cell-
subtext">Subtext</label><input class="form-control form-control-sm"
id="cell-subtext" type="text"></div>
        <div class="col-12"><label class="form-label" for="cell-
description">Description / Markdown / HTML</label><textarea class="form-
control form-control-sm" id="cell-description" rows="4"></textarea></div>
        <div class="col-12"><label class="form-label" for="cell-link">Primary
Link (label|URL or URL)</label><input class="form-control form-control-sm"
id="cell-link" type="text" placeholder="Docs|https://... or https://..."></div>
        <div class="col-12"><label class="form-label" for="cell-image">Media
URL (image/video/audio) (optional)</label><input class="form-control form-
control-sm" id="cell-image" type="url" placeholder="https://..."></div>
        <div class="col-12">
          <label class="form-label" for="cell-upload">Upload (image / audio /
video / html / css / js / md)</label>
          <div class="input-group input-group-sm">
            <input class="form-control form-control-sm" id="cell-upload"
type="file" multiple accept="image/*,audio/*,video/
*,.mp4,.webm,.ogg,.html,.css,.js,.md,.txt,.yaml,.yml,.json,.py,.sh,.bash,.glsl,.vert,
.frag,.wgsl">
```

```
          <button class="btn btn-outline-secondary" type="button" id="cell-upload-apply">Apply</button>
        </div>
      </div>
      <div class="col-6"><label class="form-label" for="cell-color">Base Color</label><input class="form-control form-control-color" id="cell-color" type="color" value="#1e90ff"></div>
      <div class="col-6"><label class="form-label" for="cell-hoverColor">Hover Color</label><input class="form-control form-control-color" id="cell-hoverColor" type="color" value="#6ec6ff"></div>
      <div class="col-6"><label class="form-label" for="cell-group">Group</label><input class="form-control form-control-sm" id="cell-group" type="text" placeholder="e.g. Gallery" list="groupSuggestions"></div>
      <div class="col-6"><label class="form-label" for="cell-tags">Tags (comma)</label><input class="form-control form-control-sm" id="cell-tags" type="text" list="tagSuggestions"></div>
      <div class="col-6"><label class="form-label" for="cell-timestamp">Timeline</label><input class="form-control form-control-sm" id="cell-timestamp" type="datetime-local"></div>
      <div class="col-6"><div class="form-check mt-4"><input class="form-check-input" type="checkbox" id="cell-locked"><label class="form-check-label" for="cell-locked">Locked</label></div></div>
      <div class="col-4"><label class="form-label" for="cell-row">Row</label><input class="form-control form-control-sm" id="cell-row" type="number" min="0" value="0"></div>
      <div class="col-4"><label class="form-label" for="cell-col">Col</label><input class="form-control form-control-sm" id="cell-col" type="number" min="0" value="0"></div>
      <div class="col-4"><label class="form-label" for="cell-rowspan">Row Span</label><input class="form-control form-control-sm" id="cell-rowspan" type="number" min="1" value="1"></div>
      <div class="col-4"><label class="form-label" for="cell-colspan">Col Span</label><input class="form-control form-select-sm form-control-sm" id="cell-colspan" type="number" min="1" value="1"></div>
      <div class="col-4"><label class="form-label" for="cell-priority">Priority</label><select class="form-select form-select-sm" id="cell-priority"><option value="auto">Auto</option><option value="critical">Critical</option><option value="high">High</option><option value="medium">Medium</option><option value="low">Low</option></select></div>
      <div class="col-4"><label class="form-label" for="cell-scale">Scale</label><select class="form-select form-select-sm" id="cell-scale"><option value="auto">Auto</option><option value="hero">Hero</option><option value="lead">Lead</option><option value="detail">Detail</option><option value="micro">Micro</option></select></div>
    </div>
    <div class="d-flex gap-2 mt-3">
```

```
        <button class="btn btn-success btn-sm" type="button" data-
action="add-cell">Add</button>
        <button class="btn btn-primary btn-sm" type="submit" data-
action="update-cell">Update</button>
        <button class="btn btn-outline-info btn-sm" type="button" data-
action="autocomplete-cell">Auto-fill</button>
        <button class="btn btn-outline-light btn-sm" type="button" data-
action="vibe-cell">Vibe</button>
        <button class="btn btn-danger btn-sm" type="button" data-
action="delete-cell">Delete</button>
        <button class="btn btn-outline-secondary btn-sm ms-auto"
type="button" data-action="download-json">Download</button>
       </div>
       <div class="mt-2">
        <button class="btn btn-outline-light btn-sm" type="button"
id="openDrawingStudio">Open Drawing Studio</button>
       </div>
     </form>
     <datalist id="titleSuggestions"></datalist>
     <datalist id="groupSuggestions"></datalist>
     <datalist id="tagSuggestions"></datalist>
   </div>
  </div>

  <div class="card shadow-sm mb-2">
   <div class="card-header">Fibonacci Menu</div>
   <div class="card-body small">
    <div class="fibo-menu" id="fiboMenu">
     <button class="fibo-item fibo-3" draggable="true" data-action="manage-
planches">Planches</button>
     <button class="fibo-item fibo-2" draggable="true" data-action="open-
playbook-presets">Playbooks</button>
     <button class="fibo-item fibo-2" draggable="true" data-action="api-
save">Save API</button>
     <button class="fibo-item fibo-2" draggable="true" data-action="api-
load">Load API</button>
     <button class="fibo-item fibo-1" draggable="true" data-action="save-
local">Save</button>
     <button class="fibo-item fibo-1" draggable="true" data-action="load-
local">Load</button>
     <button class="fibo-item fibo-2" draggable="true" data-action="export-
pdf">Export PDF</button>
     <button class="fibo-item fibo-2" draggable="true" data-action="export-
html">Export HTML</button>
     <button class="fibo-item fibo-1" draggable="true" data-action="zoom-
in">Zoom +</button>
     <button class="fibo-item fibo-1" draggable="true" data-action="zoom-
```

```
out">Zoom -</button>
      <button class="fibo-item fibo-1" draggable="true" data-action="reset-zoom">Zoom 1x</button>
      <button class="fibo-item fibo-1" draggable="true" data-action="toggle-theme">Theme</button>
      <button class="fibo-item fibo-2" draggable="true" id="fiboOpenDrawing">Drawing Studio</button>
      <button class="fibo-item fibo-2" draggable="true" data-action="download-json">Download JSON</button>
      <button class="fibo-item fibo-1" draggable="true" data-action="add-cell">Add Cell</button>
      <button class="fibo-item fibo-1" draggable="true" data-action="vibe-cell">Vibe</button>
      <button class="fibo-item fibo-1" draggable="true" data-action="autocomplete-cell">Auto-fill</button>
    </div>
   </div>
  </div>

  <div class="card shadow-sm mb-2">
   <div class="card-header">Filters</div>
   <div class="card-body small">
    <div class="row g-2">
     <div class="col-6"><label class="form-label">Tag</label><input id="filter-tag" class="form-control form-control-sm" type="text" placeholder="tag"></div>
     <div class="col-6"><label class="form-label">Group</label><input id="filter-group" class="form-control form-control-sm" type="text" placeholder="group"></div>
     <div class="col-6"><label class="form-label">Date from</label><input id="filter-from" class="form-control form-control-sm" type="date"></div>
     <div class="col-6"><label class="form-label">Date to</label><input id="filter-to" class="form-control form-control-sm" type="date"></div>
    </div>
    <div class="d-flex gap-2 mt-2">
     <button class="btn btn-outline-secondary btn-sm" data-action="apply-filters">Apply</button>
     <button class="btn btn-outline-secondary btn-sm" data-action="clear-filters">Clear</button>
    </div>
   </div>
  </div>

  <div class="card shadow-sm">
   <div class="card-header">Notes Stack</div>
   <div class="card-body small" id="notesStack">
    <p class="text-muted m-0">Notes added to cells will appear here.</p>
```

```
  </div>
 </div>

 <div class="card shadow-sm mt-2">
  <div class="card-header">Media Library</div>
  <div class="card-body small">
   <div id="mediaLibrary" class="d-grid gap-2"></div>
  </div>
 </div>

 <div class="card shadow-sm mt-2">
  <div class="card-header">Code Playground</div>
  <div class="card-body small">
   <div class="mb-2">
    <label class="form-label small" for="codeMode">Mode</label>
    <select id="codeMode" class="form-select form-select-sm">
     <option value="js">JS</option>
     <option value="html">HTML</option>
     <option value="code">Code</option>
    </select>
   </div>
   <div class="mb-2">
    <label class="form-label small" for="codePreset">Preset</label>
    <div class="input-group input-group-sm">
     <select id="codePreset" class="form-select form-select-sm"></select>
     <button class="btn btn-outline-secondary" id="codePresetLoad">Load</
button>
    </div>
   </div>
   <div class="mb-2">
    <label class="form-label small" for="codeTitle">Title</label>
    <input id="codeTitle" class="form-control form-control-sm"
placeholder="Playground snippet">
   </div>
   <div class="mb-2">
    <label class="form-label small" for="codeInput">Code</label>
    <textarea id="codeInput" class="form-control form-control-sm" rows="6"
placeholder="Paste code..."></textarea>
   </div>
   <div class="d-flex gap-2">
    <button class="btn btn-outline-light btn-sm" id="codeInsert">Insert into
cell</button>
    <button class="btn btn-outline-secondary btn-sm" id="codeNew">New
cell</button>
    <button class="btn btn-outline-warning btn-sm" id="codeVibe">Vibe</
button>
   </div>
```

```
  </div>
 </div>

 <div class="card shadow-sm mt-2" id="drawingAssistCard">
  <div class="card-header">Drawing Assist</div>
  <div class="card-body small">
   <div class="form-check">
    <input class="form-check-input" type="checkbox" id="assistSmooth"
checked>
    <label class="form-check-label" for="assistSmooth">Smooth stroke</
label>
   </div>
   <div class="form-check">
    <input class="form-check-input" type="checkbox" id="assistStabilize"
checked>
    <label class="form-check-label" for="assistStabilize">Stabilize hand
jitter</label>
   </div>
   <div class="form-check">
    <input class="form-check-input" type="checkbox" id="assistSnap">
    <label class="form-check-label" for="assistSnap">Snap to grid</label>
   </div>
   <div class="row g-2 mt-2">
    <div class="col-6">
     <label class="form-label small" for="assistSnapSize">Snap size</label>
     <input class="form-control form-control-sm" id="assistSnapSize"
type="number" min="4" max="64" value="12">
    </div>
    <div class="col-6">
     <label class="form-label small" for="assistStabilizeLevel">Stabilize</
label>
     <input class="form-control form-control-sm" id="assistStabilizeLevel"
type="number" min="0.1" max="0.9" step="0.05" value="0.35">
    </div>
   </div>
   <div class="d-flex gap-3 mt-2">
    <div class="form-check">
     <input class="form-check-input" type="checkbox" id="assistMirrorX">
     <label class="form-check-label" for="assistMirrorX">Mirror X</label>
    </div>
    <div class="form-check">
     <input class="form-check-input" type="checkbox" id="assistMirrorY">
     <label class="form-check-label" for="assistMirrorY">Mirror Y</label>
    </div>
   </div>
   <small class="text-muted d-block mt-2">Tip: Hold Shift to pan, scroll to
zoom.</small>
```

```
    </div>
  </div>

  <div class="card shadow-sm mt-2">
    <div class="card-header">Realtime Lab</div>
    <div class="card-body small">
      <div class="mb-2">
        <label class="form-label small">Phenotype Widget</label>
        <div class="d-flex gap-2 mb-2">
          <select id="phenotypeModel" class="form-select form-select-sm">
            <option value="neurobio">Neuro-Bio</option>
            <option value="brainfield">Brainfield</option>
          </select>
          <button class="btn btn-outline-light btn-sm"
id="phenotypeSeed">Seed</button>
          <button class="btn btn-outline-secondary btn-sm"
id="phenotypeMutate">Mutate</button>
        </div>
        <div id="phenotypeWidget" class="phenotype-widget" data-
phenotype="panel"></div>
      </div>
      <div class="mb-2">
        <label class="form-label small">Chat</label>
        <div class="chat-panel">
          <div class="chat-log" id="composerChatLog">No messages yet.</div>
          <div class="input-group input-group-sm mt-2">
            <input class="form-control" id="composerChatInput"
placeholder="Message">
            <button class="btn btn-outline-secondary"
id="composerChatSend">Send</button>
          </div>
        </div>
      </div>
    </div>
  </div>
</aside>

<!-- Centered overlay for preview -->
<div id="lightbox" class="lightbox" aria-hidden="true">
  <div class="backdrop" data-close="1"></div>
  <div class="content">
    <button type="button" class="btn-close close" aria-label="Close"></
button>
    <div class="p-3 border-bottom d-flex align-items-center justify-content-
between">
      <div class="d-flex align-items-center gap-2">
        <span class="badge rounded-pill text-bg-secondary" id="lb-id">#</
```

```
span>
       <strong id="lb-title">Title</strong>
       <small class="text-muted" id="lb-subtitle"></small>
      </div>
     <small class="text-muted" id="lb-meta"></small>
    </div>
    <div class="p-3" id="lightboxBody"></div>
  </div>
</div>

<div id="planchesModal" class="modal fade" tabindex="-1">
  <div class="modal-dialog modal-lg">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Manage Planches</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
      </div>
      <div class="modal-body">
        <ul class="nav nav-tabs mb-3" id="plancheTabs" role="tablist">
          <li class="nav-item" role="presentation"><button class="nav-link active"
id="p-list-tab" data-bs-toggle="tab" data-bs-target="#p-list" type="button"
role="tab">My Planches</button></li>
          <li class="nav-item" role="presentation"><button class="nav-link"
id="p-save-tab" data-bs-toggle="tab" data-bs-target="#p-save"
type="button" role="tab">Save Current</button></li>
        </ul>
        <div class="tab-content">
          <div class="tab-pane fade show active" id="p-list" role="tabpanel">
            <div id="planchesList" class="list-group">
              <div class="text-center text-muted py-4">Loading...</div>
            </div>
          </div>
          <div class="tab-pane fade" id="p-save" role="tabpanel">
            <form id="plancheSaveForm">
              <input type="hidden" id="plancheId">
              <div class="mb-3">
                <label class="form-label">Name</label>
                <input type="text" class="form-control" id="plancheName" required
placeholder="e.g. My Masterpiece">
              </div>
              <div class="form-check mb-3">
                <input class="form-check-input" type="checkbox"
id="planchePublic">
                <label class="form-check-label" for="planchePublic">Public (visible
in Playbook)</label>
              </div>
```

```
            <div class="mb-3">
              <label class="form-label">Publish Targets</label>
              <div class="d-flex flex-wrap gap-3">
                <label class="form-check">
                  <input class="form-check-input" type="checkbox"
id="plancheTargetDisplay" checked>
                  <span class="form-check-label">Display page</span>
                </label>
                <label class="form-check">
                  <input class="form-check-input" type="checkbox"
id="plancheTargetHome">
                  <span class="form-check-label">Homepage live feed</span>
                </label>
                <label class="form-check">
                  <input class="form-check-input" type="checkbox"
id="plancheTargetLive">
                  <span class="form-check-label">Live feed dashboard</span>
                </label>
              </div>
            </div>
            <button type="submit" class="btn btn-primary">Save Planche</
button>
          </form>
        </div>
      </div>
    </div>
  </div>
</div>

<div id="playbookPresetsModal" class="modal fade" tabindex="-1">
  <div class="modal-dialog modal-lg">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Playbook Presets</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
      </div>
      <div class="modal-body">
        <div class="list-group" id="playbookPresetList">
          <div class="text-center text-muted py-4">Loading presets...</div>
        </div>
      </div>
    </div>
  </div>
</div>
```

```
<div id="drawingStudioModal" class="modal fade" tabindex="-1">
  <div class="modal-dialog modal-xl">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Infinite Drawing Studio</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
      </div>
      <div class="modal-body">
        <div class="row g-3">
          <div class="col-lg-9">
            <div class="border rounded-3 position-relative"
style="height:70vh;background:#0b1020;">
              <canvas id="drawingStudioCanvas"
style="width:100%;height:100%;display:block;"></canvas>
            </div>
          </div>
          <div class="col-lg-3">
            <div class="d-flex flex-column gap-2">
              <div>
                <label class="form-label small">Brush color</label>
                <input type="color" id="drawingBrushColor" class="form-control
form-control-color" value="#e5e7eb">
              </div>
              <div>
                <label class="form-label small">Brush width</label>
                <input type="range" id="drawingBrushWidth" class="form-range"
min="1" max="24" value="2">
              </div>
              <div>
                <label class="form-label small">Opacity</label>
                <input type="range" id="drawingBrushOpacity" class="form-range"
min="0.05" max="1" step="0.05" value="1">
              </div>
              <div class="form-check">
                <input class="form-check-input" type="checkbox"
id="drawingEraser">
                <label class="form-check-label" for="drawingEraser">Eraser</label>
              </div>
              <hr class="my-2"/>
              <div class="form-check">
                <input class="form-check-input" type="checkbox"
id="drawingAssistSnap">
                <label class="form-check-label" for="drawingAssistSnap">Snap to
grid</label>
              </div>
```

```
          <div class="form-check">
            <input class="form-check-input" type="checkbox"
id="drawingAssistSmooth" checked>
            <label class="form-check-label" for="drawingAssistSmooth">Smooth
stroke</label>
          </div>
          <div class="form-check">
            <input class="form-check-input" type="checkbox"
id="drawingAssistStabilize" checked>
            <label class="form-check-label"
for="drawingAssistStabilize">Stabilize</label>
          </div>
          <div class="d-flex gap-2">
            <div class="form-check">
              <input class="form-check-input" type="checkbox"
id="drawingMirrorX">
              <label class="form-check-label" for="drawingMirrorX">Mirror X</
label>
            </div>
            <div class="form-check">
              <input class="form-check-input" type="checkbox"
id="drawingMirrorY">
              <label class="form-check-label" for="drawingMirrorY">Mirror Y</
label>
            </div>
          </div>
          <div>
            <label class="form-label small">Snap size</label>
            <input type="number" id="drawingAssistSnapSize" class="form-
control form-control-sm" min="4" max="64" value="12">
          </div>
          <div>
            <label class="form-label small">Stabilize level</label>
            <input type="number" id="drawingAssistLevel" class="form-control
form-control-sm" min="0.1" max="0.9" step="0.05" value="0.35">
          </div>
          <hr class="my-2"/>
          <div class="d-flex gap-2 flex-wrap">
            <button class="btn btn-outline-light btn-sm" type="button"
id="drawingUndo">Undo</button>
            <button class="btn btn-outline-light btn-sm" type="button"
id="drawingRedo">Redo</button>
            <button class="btn btn-outline-danger btn-sm" type="button"
id="drawingClear">Clear</button>
          </div>
          <hr class="my-2"/>
          <div>
```

```
            <label class="form-label small">Send points to code card</label>
            <select id="drawingTargetCard" class="form-select form-select-
sm"></select>
          </div>
          <div class="d-flex gap-2">
            <button class="btn btn-primary btn-sm" type="button"
id="drawingSendPoints">Send</button>
            <button class="btn btn-outline-secondary btn-sm" type="button"
id="drawingCreateCode">New code card</button>
          </div>
          <small class="text-muted d-block">Use Shift+drag to pan. Scroll to
zoom.</small>
        </div>
      </div>
    </div>
  </div>
 </div>
</div>

<input id="fileInput" type="file" accept="application/json" hidden>

<section class="magnetic-bridge container-xl py-4" id="serverBridge">
 <div class="row g-4">
  <div class="col-lg-6">
   <div class="bridge-card h-100">
    <h5>Storage ↔ API Handshake Monitor</h5>
    <p class="text-muted small">Track whether the storage authority on
<code>127.0.0.1:6601</code> issued a link and if the API acknowledged it.</p>
    <button class="btn btn-dark w-100 mb-3"
id="refreshHandshake">Refresh handshake</button>
    <pre class="bridge-console" id="handshakeStatus">Waiting for data…</
pre>
   </div>
  </div>
  <div class="col-lg-6">
   <div class="bridge-card h-100">
    <h5>Secret Note Console</h5>
    <p class="text-muted small">Secret notes are encrypted with RSA (4096-
bit). Toggle the shield to have the server seal the content before storing it.</p>
    <form id="noteForm">
     <input class="form-control" name="title" placeholder="Title" required>
     <textarea class="form-control" name="content" placeholder="Body"
rows="3" required></textarea>
     <input class="form-control" name="tags" placeholder="Tags (comma
separated)">
     <select class="form-select" name="theme">
```

```
          <option value="northern-lights">Northern Lights</option>
          <option value="midnight-grid">Midnight Grid</option>
          <option value="solar-storm">Solar Storm</option>
        </select>
        <label class="secret-toggle mt-2">
          <input type="checkbox" name="secret" id="secretToggle">
          <span>Mark as secret / encrypted</span>
        </label>
        <button class="btn btn-primary mt-3" type="submit">Save note</
button>
      </form>
      <input class="form-control mt-3" type="password"
id="secretPassphrase" placeholder="Passphrase for secret header (optional)">
      <div class="d-flex flex-wrap gap-2 mt-3">
        <button class="btn btn-outline-dark flex-fill" id="fetchSecretNotes"
type="button">Fetch secret notes</button>
        <button class="btn btn-outline-secondary flex-fill"
id="fetchPublicNotes" type="button">Refresh public notes</button>
      </div>
      <pre class="bridge-console mt-3" id="secretNotesList">No notes
loaded.</pre>
    </div>
  </div>
 </div>
</section>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/
bootstrap.bundle.min.js"></script>
<script src="/public/js/lib/composer-export.js"></script>
<script>
(() => {
'use strict';
const $ = (s, r=document)=> r.querySelector(s);
const $$ = (s, r=document)=> Array.from(r.querySelectorAll(s));
const gridEl = $('#grid');
const panel = $('#sidePanel');
const overlay = $('#panelOverlay');
const togglePanelBtn = $('#togglePanelBtn');
const fileInput = $('#fileInput');
let zoom = 1;
let currentUser = null;

const toast = (msg, type='info') => {
  const n = document.createElement('div');
  n.className = `alert alert-${type} shadow-sm position-fixed top-0 end-0
m-3 py-2 px-3 small`;
  n.style.zIndex = 3000;
```

```
  n.textContent = msg;
  document.body.appendChild(n);
  setTimeout(()=> n.remove(), 2200);
};

function openPanel(){ panel.classList.add('open'); panel.setAttribute('aria-
hidden','false'); overlay.style.display='block'; togglePanelBtn.setAttribute('aria-
expanded','true'); }
function closePanel(){ panel.classList.remove('open'); panel.setAttribute('aria-
hidden','true'); overlay.style.display='none'; togglePanelBtn.setAttribute('aria-
expanded','false'); }

togglePanelBtn.addEventListener('click', (e)=>{ e.preventDefault();
panel.classList.contains('open')?closePanel():openPanel(); });
overlay.addEventListener('click', closePanel);

// Data model
let data = [];
let selectedId = null;
let filters = { tag:'', group:'', from:'', to:'' };

// Defaults
function defaultData(){
  return [
    { id:1, field:0, axis:0, rowSpan:1, colSpan:1, type:'text', title:'ART',
subtitle:'PHI', color:'#082880', hoverColor:'#FD765D', tags:['art'],
group:'Gallery' },
    { id:2, field:0, axis:1, rowSpan:1, colSpan:1, type:'text', title:'DESIGN',
subtitle:'SKIT', color:'#7462f9', hoverColor:'#B5A6DC', tags:['design'],
group:'Gallery' },
    { id:3, field:1, axis:0, rowSpan:1, colSpan:1, type:'text', title:'Gallery',
description:'Creative Coding', color:'#ffa943', hoverColor:'#2177f4', tags:
['gallery'], group:'Play' },
    { id:4, field:1, axis:1, rowSpan:1, colSpan:1, type:'text', title:'Portfolio',
subtitle:'Creators', exponent:'Artists', color:'#ce2d42', hoverColor:'#7462f9' },
    { id:5, field:1, axis:2, rowSpan:1, colSpan:1, type:'markdown', title:'Web
Architecture', subtitle:'Application', content:'**Webmastering**\\n\
\n_Substack:_ Dev Design', color:'#06a0ba', hoverColor:'#6ec6ff' },
    { id:6, field:0, axis:2, rowSpan:1, colSpan:2, type:'html', title:'Custom HTML',
content:"<div style='padding:.5rem'><h3>Custom <em>HTML</em></
h3><p>Inline <code>script</code> runs in iframe.</p></div>",
color:'#333333', hoverColor:'#777777' }
  ];
}

function buildPlaybookPresets(){
  const sketchCards = [
```

```
    { title: "Riemann Fields", path: "/public/data-viz/compute/riemann.html",
group: "Compute", tags: ["riemann", "compute"] },
    { title: "Spiky Noise", path: "/public/data-viz/creative-code/spiky.html",
group: "Creative", tags: ["spiky", "creative"] },
    { title: "Bubble Pack 3", path: "/public/data-viz/ecosystem/
bubble_pack3.html", group: "Ecosystem", tags: ["bubble", "ecosystem"] },
    { title: "Exosphere", path: "/public/data-viz/ecosystem/exosphere.html",
group: "Ecosystem", tags: ["exosphere", "ecosystem"] },
    { title: "JS Kand Meta", path: "/public/data-viz/genart/jsKandMeta.html",
group: "GenArt", tags: ["genart", "meta"] },
    { title: "Spectra Grid 3", path: "/public/data-viz/grid/spectra3.html", group:
"Grid", tags: ["grid", "spectra"] },
    { title: "Multi Timeline", path: "/public/data-viz/timeserie/multitimeline.html",
group: "Timeserie", tags: ["timeline", "timeserie"] },
    { title: "Timeline Helix", path: "/public/data-viz/timeserie/
timelinehelix3dsmooth.html", group: "Timeserie", tags: ["helix", "timeserie"] },
    { title: "Spectra 7", path: "/public/data-viz/timeserie/spectra7.html", group:
"Timeserie", tags: ["spectra", "timeserie"] },
    { title: "Ideo View", path: "/public/data-viz/web-view/ideo.html", group:
"Web", tags: ["web", "ideo"] },
    { title: "Uniphi Composer", path: "/public/data-viz/composer/
index_uniphi_v5_combo.html", group: "Composer", tags: ["composer",
"combo"] },
    { title: "Kosmos Flow", path: "/public/data-viz/animation-flow/kosmos.html",
group: "Flow", tags: ["flow", "animation"] },
    { title: "Multitype Analysis", path: "/public/data-viz/analysis/multitype/
index.html", group: "Analysis", tags: ["analysis", "multitype"] },
    { title: "Uniphilab Abstract", path: "/public/data-viz/abstract/uniphilab.html",
group: "Abstract", tags: ["abstract", "lab"] }
  ];

  function makeSketchGrid(startId, cols, rowOffset) {
    return sketchCards.map((s, i) => {
      const row = Math.floor(i / cols) * 2 + rowOffset;
      const col = (i % cols) * 2;
      return {
        id: startId + i,
        field: row,
        axis: col,
        rowSpan: 2,
        colSpan: 2,
        type: "html",
        title: s.title,
        content: s.path,
        color: "#0f172a",
        hoverColor: "#38bdf8",
        group: s.group,
```

```
      tags: s.tags
    };
  });
}

const baseCards = [
  { id: 1, field: 0, axis: 0, rowSpan: 1, colSpan: 2, type: "text", title: "Playbook", subtitle: "Composer Preset", description: "Vibecode + data viz + sequencing", color: "#0b1020", hoverColor: "#5eead4", group: "System", tags: ["playbook", "preset"] },
  { id: 2, field: 0, axis: 2, rowSpan: 1, colSpan: 2, type: "code", title: "Vibe Coding Input", content: vibeSnippet("code", seedFrom("vibe-playbook")), color: "#111827", hoverColor: "#a78bfa", group: "Vibe", tags: ["vibe", "code"] },
  { id: 3, field: 0, axis: 4, rowSpan: 1, colSpan: 2, type: "markdown", title: "Sequencing", content: "1. Frame intent\n2. Load sketches\n3. Annotate metrics\n4. Publish planche\n5. Iterate", color: "#0f172a", hoverColor: "#f59e0b", group: "Ops", tags: ["sequence", "steps"] },
  { id: 4, field: 1, axis: 0, rowSpan: 1, colSpan: 2, type: "code", title: "IDE Block", content: "// Compose here\nconst idea = {\n  name: \"Planche\",\n  mood: \"luminous\",\n  tempo: 0.62\n};\n", color: "#111827", hoverColor: "#60a5fa", group: "IDE", tags: ["ide", "code"] },
  { id: 5, field: 1, axis: 2, rowSpan: 1, colSpan: 2, type: "markdown", title: "Metrics", content: "- Cells: 14\n- Heatmap: 0.78\n- Focus: 0.64\n- Cohesion: 0.71", color: "#0f172a", hoverColor: "#22c55e", group: "Metrics", tags: ["metrics"] },
  { id: 6, field: 1, axis: 4, rowSpan: 1, colSpan: 2, type: "html", title: "Form Input", content: "<div style='padding:12px;font-family:system-ui'><h3>Annotation</h3><input style='width:100%;padding:8px;margin-bottom:8px' placeholder='Label'/><textarea style='width:100%;padding:8px' rows='4' placeholder='Notes'></textarea><button style='margin-top:8px;padding:8px 12px'>Save</button></div>", color: "#0f172a", hoverColor: "#fb7185", group: "Forms", tags: ["form", "input"] }
];

const fullSketches = makeSketchGrid(10, 3, 2);
const timelineSketches = [
  { title: "Multi Timeline", path: "/public/data-viz/timeserie/multitimeline.html", group: "Timeserie", tags: ["timeline", "timeserie"] },
  { title: "Timeline Helix", path: "/public/data-viz/timeserie/timelinehelix3dsmooth.html", group: "Timeserie", tags: ["helix", "timeserie"] },
  { title: "Spectra 7", path: "/public/data-viz/timeserie/spectra7.html", group: "Timeserie", tags: ["spectra", "timeserie"] },
  { title: "Spectra Grid 3", path: "/public/data-viz/grid/spectra3.html", group: "Grid", tags: ["grid", "spectra"] },
  { title: "Riemann Fields", path: "/public/data-viz/compute/riemann.html", group: "Compute", tags: ["riemann", "compute"] },
  { title: "Kosmos Flow", path: "/public/data-viz/animation-flow/kosmos.html",
```

```
group: "Flow", tags: ["flow", "animation"] }
  ];
  const timelineCards = timelineSketches.map((s, i) => ({
    id: 20 + i,
    field: 2 + Math.floor(i / 3) * 2,
    axis: (i % 3) * 2,
    rowSpan: 2,
    colSpan: 2,
    type: "html",
    title: s.title,
    content: s.path,
    color: "#0f172a",
    hoverColor: "#38bdf8",
    group: s.group,
    tags: s.tags
  }));

  return [
    {
      id: "playbook-full",
      name: "Full Composer Playbook",
      description: "All sketches + vibecode + metrics + sequencing",
      cards: [...baseCards, ...fullSketches]
    },
    {
      id: "playbook-timeseries",
      name: "Timeline Suite",
      description: "Timeseries + flow + vibecode blocks",
      cards: [...baseCards.map((c) => ({ ...c, id: c.id + 40 })), ...timelineCards]
    }
  ];
}

const vibePalettes = [
  { base: "#0f172a", hover: "#38bdf8", accent: "#f97316" },
  { base: "#1f2937", hover: "#f472b6", accent: "#22c55e" },
  { base: "#0b1020", hover: "#a78bfa", accent: "#facc15" },
  { base: "#111827", hover: "#fb7185", accent: "#60a5fa" },
  { base: "#0a0f16", hover: "#5eead4", accent: "#f59e0b" }
];

function seedFrom(text){
  const str = String(text || "");
  let h = 0;
  for (let i = 0; i < str.length; i += 1) h = (h * 31 + str.charCodeAt(i)) >>> 0;
  return h || Date.now();
}
```

```
function pickPalette(seed){
  const idx = Math.abs(seed) % vibePalettes.length;
  return vibePalettes[idx];
}

function vibeSnippet(type, seed){
  const tempo = (0.4 + (seed % 60) / 100).toFixed(2);
  const hue = seed % 360;
  const snippet = `// Vibecode preset
const vibe = {
  tempo: ${tempo},
  hue: ${hue},
  chroma: ${(seed % 50) + 20},
  noise: { amp: 0.85, freq: 1.3 },
  waves: { curl: 0.5, divergence: 0.3 }
};

function tick(t){
  return Math.sin(t * vibe.tempo) * 0.5 + 0.5;
}
console.log("vibe", vibe, tick(performance.now() * 0.001));
`;
  if (type === "markdown") return `# Vibecode\n\n${snippet.replace(/\n/g, "\n
")}\n`;
  if (type === "html") return `<div style="padding:12px;font-family:system-
ui"><h3>Vibecode</h3><pre>${escapeHtml(snippet)}</pre></div>`;
  return snippet;
}

function applyAutoFill(it){
  const type = it.type || "text";
  const seed = seedFrom(`${it.title || ""}${it.subtitle || ""}${type}`);
  const palette = pickPalette(seed);
  if (!it.title) it.title = type === "drawing" ? "Sketch" : "New Card";
  if (!it.subtitle) it.subtitle = type === "code" ? "Vibecode" : "Composer";
  if (!it.group) it.group = (it.tags && it.tags[0]) ? it.tags[0] : "Gallery";
  if (!it.tags || !it.tags.length) it.tags = [type, it.group].filter(Boolean);
  if (!it.color) it.color = palette.base;
  if (!it.hoverColor) it.hoverColor = palette.hover;
  if (!it.scale || it.scale === "auto") it.scale = it.scale || "auto";
  if (!it.priority || it.priority === "auto") it.priority = "auto";
  if (!it.description && ["text", "link"].includes(type)) {
    it.description = `Vibecode · ${it.group}`;
  }
  if (["code", "js", "html", "markdown"].includes(type) && !it.content) {
    it.content = vibeSnippet(type, seed);
```

```
  }
}

function applyVibe(it){
  const seed = seedFrom(`${it.title || ""}${it.group || ""}${Date.now()}`);
  const palette = pickPalette(seed);
  it.color = palette.base;
  it.hoverColor = palette.hover;
  it.tags = Array.from(new Set([...(it.tags || []), "vibe", "composer"]));
  if (!it.subtitle) it.subtitle = "Vibe mode";
  if (["code", "js", "html", "markdown"].includes(it.type || "")) {
    it.content = vibeSnippet(it.type, seed);
  } else if (!it.description) {
    it.description = `Vibe ${new Date().toLocaleTimeString()}`;
  }
}

function getSuggestions(){
  const titles = new Set();
  const groups = new Set();
  const tags = new Set();
  data.forEach((it) => {
    if (it.title) titles.add(it.title);
    if (it.group) groups.add(it.group);
    (it.tags || []).forEach((t) => { if (t) tags.add(t); });
  });
  return {
    titles: Array.from(titles).slice(0, 60),
    groups: Array.from(groups).slice(0, 60),
    tags: Array.from(tags).slice(0, 120)
  };
}

function refreshAutocompleteLists(){
  const lists = getSuggestions();
  const titleList = document.getElementById("titleSuggestions");
  const groupList = document.getElementById("groupSuggestions");
  const tagList = document.getElementById("tagSuggestions");
  if (titleList) {
    titleList.innerHTML = "";
    lists.titles.forEach((t) => {
      const opt = document.createElement("option");
      opt.value = t;
      titleList.appendChild(opt);
    });
  }
  if (groupList) {
```

```
    groupList.innerHTML = "";
    lists.groups.forEach((g) => {
      const opt = document.createElement("option");
      opt.value = g;
      groupList.appendChild(opt);
    });
  }
  if (tagList) {
    tagList.innerHTML = "";
    lists.tags.forEach((t) => {
      const opt = document.createElement("option");
      opt.value = t;
      tagList.appendChild(opt);
    });
  }
}

function setupTagAutocomplete(){
  const input = document.getElementById("cell-tags");
  if (!input) return;
  input.addEventListener("keydown", (e) => {
    if (e.key !== "Tab" && !(e.ctrlKey && e.key === " ")) return;
    const lists = getSuggestions();
    const raw = input.value;
    const parts = raw.split(",");
    const last = parts[parts.length - 1].trim();
    if (!last) return;
    const match = lists.tags.find((t) =>
t.toLowerCase().startsWith(last.toLowerCase()));
    if (!match) return;
    e.preventDefault();
    parts[parts.length - 1] = ` ${match}`;
    input.value = parts.map((p) => p.trim()).join(", ");
  });
}

function loadDrawingAssist(){
  try {
    const raw = localStorage.getItem("composer.drawingAssist");
    return raw ? JSON.parse(raw) : {};
  } catch {
    return {};
  }
}

function saveDrawingAssist(opts){
  try { localStorage.setItem("composer.drawingAssist", JSON.stringify(opts)); }
```

```
catch {}
}

function bindDrawingAssistControls(){
  const smooth = document.getElementById("assistSmooth");
  const stabilize = document.getElementById("assistStabilize");
  const snap = document.getElementById("assistSnap");
  const snapSize = document.getElementById("assistSnapSize");
  const stabilizeLevel = document.getElementById("assistStabilizeLevel");
  const mirrorX = document.getElementById("assistMirrorX");
  const mirrorY = document.getElementById("assistMirrorY");
  if (!smooth || !stabilize || !snap || !snapSize || !stabilizeLevel || !mirrorX || !
mirrorY) return;
  const saved = loadDrawingAssist();
  const state = {
    smooth: saved.smooth ?? true,
    stabilize: saved.stabilize ?? true,
    snap: saved.snap ?? false,
    snapSize: saved.snapSize ?? Number(snapSize.value),
    stabilizeLevel: saved.stabilizeLevel ?? Number(stabilizeLevel.value),
    mirrorX: saved.mirrorX ?? false,
    mirrorY: saved.mirrorY ?? false
  };
  smooth.checked = !!state.smooth;
  stabilize.checked = !!state.stabilize;
  snap.checked = !!state.snap;
  snapSize.value = String(state.snapSize);
  stabilizeLevel.value = String(state.stabilizeLevel);
  mirrorX.checked = !!state.mirrorX;
  mirrorY.checked = !!state.mirrorY;

  function apply(){
    const next = {
      smooth: smooth.checked,
      stabilize: stabilize.checked,
      snap: snap.checked,
      snapSize: Number(snapSize.value) || 12,
      stabilizeLevel: Number(stabilizeLevel.value) || 0.35,
      mirrorX: mirrorX.checked,
      mirrorY: mirrorY.checked
    };
    window.__drawingAssist = next;
    if (window.DrawingBoard?.setOptions)
window.DrawingBoard.setOptions(next);
    saveDrawingAssist(next);
  }
```

```
  [smooth, stabilize, snap, snapSize, stabilizeLevel, mirrorX,
mirrorY].forEach((el) => {
    el.addEventListener("change", apply);
    el.addEventListener("input", apply);
  });
  apply();
}

// Local storage
let autoSaveTimer = null;
function triggerAutoSave(){
  clearTimeout(autoSaveTimer);
  if($('#autoSaveSwitch')?.checked){
    autoSaveTimer = setTimeout(() => apiSave(true), 3000);
  }
}

function saveLocal(key='spectra.magnetic'){
  try { localStorage.setItem(key, JSON.stringify(data)); } catch(e)
{ console.warn(e); }
  triggerAutoSave();
}
function loadLocal(key='spectra.magnetic'){ try { const
s=localStorage.getItem(key); if(s){ data = adaptInput(JSON.parse(s));
renderGrid(); } } catch(e){ console.warn(e); } }

// API
async function apiSave(silent=false){
  const token = localStorage.getItem('authToken') ||
localStorage.getItem('ssoToken');
  if (!token) return !silent && toast('Missing SSO token','warning');
  const isPublic = $('#publicSwitch')?.checked || false;
  try{
    const res = await fetch('/api/cms/composer', {
      method:'POST',
      headers:{ 'Content-Type':'application/json', 'Authorization': `Bearer ${token}
` },
      body: JSON.stringify({ data, isPublic })
    });
    if(!res.ok) throw new Error(await res.text());
    if(!silent) toast('Saved to API','success');
    else console.log('Auto-saved to API');
  }catch(e){
    if(!silent) toast(`Save failed: ${e.message}`,'danger');
  }
}
```

```
async function apiLoad(){
  const token = localStorage.getItem('authToken') ||
localStorage.getItem('ssoToken');
  if (!token) return toast('Missing SSO token','warning');
  try{
    const res = await fetch('/api/cms/composer', {
      method:'GET',
      headers:{ 'Authorization': `Bearer ${token}` }
    });
    if(!res.ok) throw new Error(await res.text());
    const j = await res.json();
    data = adaptInput(j.layout?.data || []);
    if(j.layout?.isPublic && $('#publicSwitch')) $('#publicSwitch').checked =
true;
    renderGrid();
    toast('Loaded from API','success');
  }catch(e){
    toast(`Load failed: ${e.message}`,'danger');
  }
}

async function loadCurrentUser(){
  const token = localStorage.getItem('authToken') ||
localStorage.getItem('ssoToken');
  if (!token) return;
  try{
    const res = await fetch('/api/auth/me', { headers:{ 'Authorization': `Bearer $
{token}` } });
    if (!res.ok) return;
    const json = await res.json();
    currentUser = json.user || null;
    window.currentUser = currentUser;
  }catch(e){ console.warn(e); }
}
window.loadCurrentUser = loadCurrentUser;

  function errorOverlayScript(){
    return `(function(){
      const box = document.createElement('div');
      box.style.position='fixed';
      box.style.left='12px';
      box.style.bottom='12px';
      box.style.maxWidth='70%';
      box.style.padding='10px 12px';
      box.style.borderRadius='8px';
      box.style.background='rgba(11,16,32,0.9)';
      box.style.color='#f8dada';
```

```
      box.style.font='12px/1.4 system-ui,Segoe UI,Arial';
      box.style.zIndex='99999';
      box.style.display='none';
      document.body.appendChild(box);
      function show(msg){ box.textContent=msg; box.style.display='block'; }
      window.addEventListener('error', (e)=>{ show('Script error: '+(e.message||
e.error||'unknown')); });
      window.addEventListener('unhandledrejection', (e)=>{ show('Promise error:
'+(e.reason?.message||e.reason||'unknown')); });
    })();`;
  }

  function sandboxForType(type, src){
    const roles = currentUser?.roles || [];
    const isPriv = roles.includes('admin') || roles.includes('creator');
    const isExternal = /^https?:\/\//.test(src || '');
    // Composer is already gated; allow scripts for HTML/JS to render live
sketches.
    if (type === 'js') {
      return isExternal && !isPriv ? 'allow-scripts' : 'allow-scripts allow-same-
origin';
    }
    if (type === 'html') {
      return isExternal && !isPriv ? 'allow-scripts' : 'allow-scripts allow-same-
origin';
    }
    if (type === 'code') return 'allow-scripts allow-same-origin';
    return 'allow-same-origin';
  }

async function loadMediaLibrary(){
  const token = localStorage.getItem('authToken') ||
localStorage.getItem('ssoToken');
  const container = $('#mediaLibrary');
  if (!container) return;
  try{
    const res = await fetch('/api/cms/media/public', { headers: token ?
{ 'Authorization': `Bearer ${token}` } : {} });
    const json = await res.json();
    const list = json.media || [];
    container.innerHTML = '';
    list.slice(0, 24).forEach(item=>{
      const btn = document.createElement('button');
      btn.className = 'btn btn-outline-light btn-sm text-start';
      btn.textContent = `${item.type || 'media'} · ${item.title || item.file?.filename ||
'Untitled'}`;
      btn.addEventListener('click', ()=>{
```

```
        if (selectedId == null) {
          const newId = data.length ? Math.max(...data.map(d=>d.id||0)) + 1 : 1;
          data.push({ id:newId, field:0, axis:0, rowSpan:1, colSpan:1, type:'image',
title:item.title||'Media', color:'#1e90ff', hoverColor:'#6ec6ff', group:'Media',
tags: item.tags||[] });
          selectedId = newId;
        }
        const it = data.find(d=>d.id===selectedId);
        if (!it) return;
        const url = item.url || (item.file?.filename ? `/media/${item.file.filename}` :
'');
        if (item.type?.includes('video')) { it.type='video'; it.image = url; }
        else if (item.type?.includes('audio') || item.type?.includes('podcast'))
{ it.type='audio'; it.audio = url; }
        else if (item.type?.includes('html')) { it.type='html'; it.content = url; }
        else if (item.type?.includes('js')) { it.type='js'; it.content = url; }
        else if (item.type?.includes('css')) { it.type='css'; it.content = url; }
        else { it.type='image'; it.image = url; }
        it.title = item.title || it.title;
        saveLocal(); renderGrid(); selectCell(it.id); openPanel();
      });
      container.appendChild(btn);
    });
  }catch(e){ console.warn(e); }
}
window.loadMediaLibrary = loadMediaLibrary;

// Geometry
function dims(){
  let maxRow = 0, maxCol = 0;
  data.forEach(it=>{
    const r = (it.field|0) + Math.max(1, it.rowSpan|0) - 1;
    const c = (it.axis|0) + Math.max(1, it.colSpan|0) - 1;
    if (r > maxRow) maxRow = r;
    if (c > maxCol) maxCol = c;
  });
  return { rows: maxRow+1, cols: maxCol+1 };
}

function getCellSize(){ return
parseInt(getComputedStyle(document.documentElement).getPropertyValue('--
cell-size')) || 140; }
function getGap(){ return
parseInt(getComputedStyle(document.documentElement).getPropertyValue('--
gap')) || 6; }

function sizeGrid(){
```

```
  const { cols } = dims();
  gridEl.style.gridTemplateColumns = `repeat(${cols}, var(--cell-size))`;
  gridEl.style.transform = `scale(${zoom})`;
  gridEl.style.transformOrigin = 'top left';
}

// Occupancy matrix and placeholder fill
function buildMatrix(){
  const { rows, cols } = dims();
  const mat = Array.from({length: rows}, ()=> Array(cols).fill(null));
  data.forEach(it=>{
    for(let r=it.field; r<it.field+it.rowSpan; r++){
      for(let c=it.axis; c<it.axis+it.colSpan; c++){
        if(mat[r] && typeof mat[r][c] !== 'undefined') mat[r][c] = it.id;
      }
    }
  });
  return mat;
}

function isAreaFree(row, col, rowSpan, colSpan, ignoreId=null){
  // allow placement beyond current matrix bounds (we treat out-of-range cells
as free)
  if (row < 0 || col < 0) return false;
  const rMax = row + rowSpan - 1, cMax = col + colSpan - 1;
  const mat = buildMatrix();
  for(let r=row; r<=rMax; r++){
    if (!mat[r]) continue; // out-of-bounds rows are free
    for(let c=col; c<=cMax; c++){
      // out-of-bounds cols are treated as free since mat[r][c] may be undefined
      const occ = (typeof mat[r][c] !== 'undefined') ? mat[r][c] : null;
      if (occ !== null && occ !== ignoreId) return false;
    }
  }
  return true;
}

// Rendering
function renderGrid(){
  const hasFilters = filters.tag || filters.group || filters.from || filters.to;
  const filtered = hasFilters ? data.filter(c=>{
    let ok=true;
    if(filters.tag) ok = ok && (c.tags||
[]).some(t=>t.toLowerCase().includes(filters.tag.toLowerCase()));
    if(filters.group) ok = ok &&
((c.group||'').toLowerCase().includes(filters.group.toLowerCase()));
    if(filters.from) ok = ok && (c.timestamp ? (new Date(c.timestamp) >= new
```

```
Date(filters.from)) : false);
    if(filters.to) ok = ok && (c.timestamp ? (new Date(c.timestamp) <= new
Date(filters.to)) : false);
    return ok;
  }) : data;

  gridEl.innerHTML = '';
  sizeGrid();

  const { rows, cols } = dims();
  // render real cells
  filtered.forEach(item=> gridEl.appendChild(createCellEl(item)));

  // placeholders
  gridEl.classList.add('placeholder-on');
  const mat = buildMatrix();
  for(let r=0;r<rows;r++){
    for(let c=0;c<cols;c++){
      if(mat[r][c] === null){
        const ph = document.createElement('div');
        ph.className = 'placeholder';
        ph.style.gridRowStart = r+1;
        ph.style.gridColumnStart = c+1;
        ph.style.gridRowEnd = 'span 1';
        ph.style.gridColumnEnd = 'span 1';
        gridEl.appendChild(ph);
      }
    }
  }

  updateNotesStack();
  refreshAutocompleteLists();
  if (window.Phenotype && typeof window.Phenotype.scan === 'function') {
    window.Phenotype.scan();
  }
  if (window.DrawingBoard && typeof window.DrawingBoard.scan ===
'function') {
    window.DrawingBoard.scan({ data, saveLocal, renderGrid, selectCell, assist:
window.__drawingAssist });
  }
}

function encodeBaseHoverGrad(base, hov){ return `linear-gradient(135deg, $
{base} 0%, ${hov} 100%)`; }

function createCellEl(item){
  const el = document.createElement('div');
```

```
  el.className = 'cell';
  el.dataset.id = item.id;
  el.tabIndex = 0;
  if (item.locked) el.classList.add('locked');
  const base = item.color || '#334155', hov = item.hoverColor || base;

  el.style.setProperty('--baseColor', base);
  el.style.setProperty('--hoverGrad', encodeBaseHoverGrad(base, hov));
  el.style.gridRowStart = (item.field|0) + 1;
  el.style.gridColumnStart = (item.axis|0) + 1;
  el.style.gridRowEnd = `span ${Math.max(1, item.rowSpan|0)}`;
  el.style.gridColumnEnd = `span ${Math.max(1, item.colSpan|0)}`;

  // Cover image as background if image in non-image types
  if (item.type !== 'image' && item.image){
    const img = document.createElement('img');
    img.className = 'cover'; img.loading = 'lazy';
    img.src = item.image; img.alt = item.title || '';
    el.appendChild(img);
  }

  // lock badge
  if (item.locked){
    const b = document.createElement('span');
    b.className = 'badge text-bg-warning badge-lock'; b.textContent = '🔒';
el.appendChild(b);
  }

  // content container
  const c = document.createElement('div');
  c.className = 'cell-content';
  c.setAttribute('role','gridcell');
  c.setAttribute('aria-label', item.title || 'cell');
  el.appendChild(c);

  renderCellContent(item, c);

  // mini toolbar
  const mt = document.createElement('div');
  mt.className = 'mini-toolbar';
  mt.innerHTML = `
    <button class="btn btn-outline-light btn-sm" data-cmd="edit" data-
title="Edit (double-click)">✎</button>
    <button class="btn btn-outline-light btn-sm" data-cmd="vibe" data-
title="Vibe">✷</button>
    <div class="btn-group">
      <button class="btn btn-outline-light btn-sm dropdown-toggle" data-bs-
```

```
toggle="dropdown">Type</button>
      <ul class="dropdown-menu dropdown-menu-end">
        <li><a class="dropdown-item" data-type="text">Text</a></li>
        <li><a class="dropdown-item" data-type="markdown">Markdown</a></
li>
        <li><a class="dropdown-item" data-type="html">HTML</a></li>
        <li><a class="dropdown-item" data-type="js">JS</a></li>
        <li><a class="dropdown-item" data-type="css">CSS</a></li>
        <li><a class="dropdown-item" data-type="code">Code</a></li>
        <li><a class="dropdown-item" data-type="phenotype">Phenotype 3D</
a></li>
        <li><a class="dropdown-item" data-type="exosphere">Exosphere</a></
li>
        <li><a class="dropdown-item" data-type="drawing">Drawing</a></li>
        <li><a class="dropdown-item" data-type="link">Link</a></li>
        <li><a class="dropdown-item" data-type="image">Image</a></li>
        <li><a class="dropdown-item" data-type="video">Video</a></li>
        <li><a class="dropdown-item" data-type="audio">Audio</a></li>
      </ul>
    </div>`;
  el.appendChild(mt);

  mt.addEventListener('click', (e)=>{
    const t = e.target.closest('[data-type],[data-cmd]'); if(!t) return;
    e.preventDefault();
    if (t.dataset.cmd === 'edit'){ selectCell(item.id); openPanel(); return; }
    if (t.dataset.cmd === 'vibe'){
      const it = data.find(d=>d.id===item.id); if(!it) return;
      applyVibe(it); saveLocal(); renderGrid(); selectCell(item.id); openPanel();
return;
    }
    if (t.dataset.type){ const it = data.find(d=>d.id===item.id); if(!it) return;
      it.type = t.dataset.type; saveLocal(); renderGrid(); selectCell(item.id);
openPanel();
    }
  });

  // drag/resize handles
  if (!item.locked){
    const drag = document.createElement('div'); drag.className='handle-drag';
drag.textContent='≡'; drag.title='Drag';
    const res = document.createElement('div'); res.className='handle-resize';
res.title='Resize';
    const resR = document.createElement('div'); resR.className='handle-
resize-r'; const resB = document.createElement('div');
resB.className='handle-resize-b';
    el.appendChild(drag); el.appendChild(res); el.appendChild(resR);
```

```
el.appendChild(resB);
    enableDrag(el, drag, item);
    enableResize(el, {corner:res, right:resR, bottom:resB}, item);
  }

  // open preview or edit
  el.addEventListener('click', (e)=>{
    if (e.target.closest('a')) return; // allow link click
    if (item.locked) return;
    selectCell(item.id);
    if (e.shiftKey) openPanel();
    else openPreview(el);
  });
  el.addEventListener('dblclick', ()=>{ if(item.locked) return; selectCell(item.id);
openPanel(); });

  return el;
}

// Content rendering
function renderCellContent(item, container){
  container.innerHTML = '';
  const htmlDoc = (body = '', css = '', js = '') => {
    return `<!doctype html><html><head><meta
charset="utf-8"><style>html,body{margin:0;padding:0}body{font-
family:system-ui,Segoe UI,Arial,sans-
serif;background:transparent;color:#0f172a}img,video,canvas{max-
width:100%}</style>${css ? `<style>${css}</style>` : ''}</head><body>$
{body}${js ? `<script>${js}<\/script>` : ''}</body></html>`;
  };
  const htmlFromUrl = (url) => {
    const safe = String(url).replace(/"/g, '&quot;');
    return `<!doctype html><html><head><meta charset="utf-8"></
head><body><script>
      fetch("${safe}").then(r=>r.text()).then(txt=>{
        const base = document.createElement('base');
        base.href = "${safe}";
        const parser = new DOMParser();
        const doc = parser.parseFromString(txt, 'text/html');
        if (doc.head) doc.head.prepend(base);
        const err = doc.createElement('script');
        err.textContent = ${JSON.stringify(errorOverlayScript())};
        (doc.body || doc.head || doc.documentElement).appendChild(err);
        document.open();
        document.write(doc.documentElement.outerHTML);
        document.close();
      }).catch(err=>{
```

```
      document.body.innerHTML = '<pre
style="color:#cbd5f5;background:#0b1020;padding:12px;border-
radius:8px">HTML load failed\\n'+err+'</pre>';
    });
  <\/script></body></html>`;
 };
 const htmlFromContent = (rawHtml) => {
  const safe = JSON.stringify(String(rawHtml || ""));
  return `<!doctype html><html><head><meta charset="utf-8"></
head><body><script>
    const txt = ${safe};
    const parser = new DOMParser();
    const doc = parser.parseFromString(txt, 'text/html');
    const err = doc.createElement('script');
    err.textContent = ${JSON.stringify(errorOverlayScript())};
    (doc.body || doc.head || doc.documentElement).appendChild(err);
    document.open();
    document.write(doc.documentElement.outerHTML);
    document.close();
  <\/script></body></html>`;
 };
 const blobFromContent = (content, type = 'text/html') => {
  try {
   const blob = new Blob([content], { type });
   return URL.createObjectURL(blob);
  } catch {
   return null;
  }
 };
 const iframeFor = (srcdoc) => {
  const iframe = document.createElement('iframe');
  iframe.style.width='100%';
  iframe.style.height='100%';
  iframe.style.border='0';
  iframe.style.background='transparent';
  iframe.srcdoc = srcdoc;
  return iframe;
 };
 if (item.type === 'image'){
  if (item.images && item.images.length > 1) {
   const grid = document.createElement('div');
   grid.style.display = 'grid';
   grid.style.gridTemplateColumns = 'repeat(auto-fit, minmax(60px, 1fr))';
   grid.style.gap = '2px';
   grid.style.width = '100%';
   grid.style.height = '100%';
   grid.style.overflow = 'hidden';
```

```
      item.images.slice(0, 4).forEach((src, i) => {
        const img = document.createElement('img');
        img.src = src;
        img.loading = 'lazy';
        img.style.width = '100%';
        img.style.height = '100%';
        img.style.objectFit = 'cover';
        img.style.borderRadius = '2px';
        grid.appendChild(img);
      });
      container.appendChild(grid);
    } else {
      const img = document.createElement('img');
      img.src = item.image || item.src || (item.images && item.images[0]) || '';
img.alt = item.title || ''; img.loading = 'lazy';
      img.style.maxWidth='100%'; img.style.maxHeight='100%';
      container.appendChild(img);
    }
    if (item.title){ const t=document.createElement('div'); t.className='subtitle
mt-1'; t.textContent=item.title; container.appendChild(t); }
    return;
  }
  if (item.type === 'video'){
    container.classList.add('has-media');
    const url = item.url || item.image || primaryUrl(item);
    const isEmbed = /youtube\.com|youtu\.be|vimeo\.com/.test(url);
    if (url && !isEmbed) {
      const video = document.createElement('video');
      video.className = 'cell-video';
      if (/\.mov(\?|$)/i.test(url)) {
        const s = document.createElement('source');
        s.src = url;
        s.type = 'video/mp4';
        video.appendChild(s);
      } else {
        video.src = url;
      }
      video.muted = true;
      video.loop = true;
      video.autoplay = true;
      video.playsInline = true;
      video.preload = 'auto';
      video.setAttribute('playsinline','');
      video.setAttribute('muted','');
      video.setAttribute('autoplay','');
      container.appendChild(video);
```

```
      const label = document.createElement('div');
      label.innerHTML = `<div>▶ ${escapeHtml(item.title||'Video')}</div>`;
      label.style.position = 'relative';
      label.style.zIndex = '2';
      container.appendChild(label);
      const sine = document.createElement('div');
      sine.className = 'cell-sine';
      sine.innerHTML = `<svg viewBox="0 0 120 18"
preserveAspectRatio="none" aria-hidden="true">
        <path d="M0 9 Q 8 2 16 9 T 32 9 T 48 9 T 64 9 T 80 9 T 96 9 T 120 9"></
path>
      </svg>`;
      container.appendChild(sine);
      video.play().catch(() => {});
    } else {
      const embed = resolveVideoEmbed(item);
      const iframe = document.createElement('iframe');
      iframe.style.width='100%'; iframe.style.height='100%';
iframe.style.border='0';
      iframe.setAttribute('allow', 'autoplay; fullscreen');
      if (typeof embed === 'string') iframe.src = embed;
      else if (embed.srcdoc) iframe.srcdoc = embed.srcdoc;
      container.appendChild(iframe);
    }
    return;
  }
  if (item.type === 'audio'){
    const snap = document.createElement('div');
    snap.innerHTML = `<div>♪ ${escapeHtml(item.title||'Audio')}</div>`;
    container.appendChild(snap);
    const src = item.audio || item.src || item.image || primaryUrl(item);
    if (src){
      const audio = document.createElement('audio');
      audio.controls = true; audio.src = src; audio.style.width='100%';
      container.appendChild(audio);
    }
    return;
  }
  if (item.type === 'link'){
    const url = primaryUrl(item);
    const u = new URL(url||'http://example.com', location.origin);
    const card = document.createElement('div');
    card.innerHTML = `<div><strong>${escapeHtml(item.title||u.hostname)}</
strong></div>
      <div class="small">${escapeHtml(item.subtitle||url||'')}</div>`;
    container.appendChild(card);
    attachHoverIframe(container, url);
```

```
    // also render links badges
    renderLinks(item, container);
    return;
  }
  if (item.type === 'html'){
    container.classList.add('has-media');
    const htmlSrc = item.content || item.htmlContent || '';
    const iframe = document.createElement('iframe');
    iframe.sandbox = sandboxForType('html', htmlSrc);
    iframe.style.width='100%'; iframe.style.height='100%';
iframe.style.border='0';
    if (/^(https?:)?\/\//.test(htmlSrc) || htmlSrc.startsWith('/') ||
htmlSrc.startsWith('data:')) {
      iframe.src = htmlSrc;
    } else {
      iframe.srcdoc = htmlFromContent(htmlSrc);
    }
    container.appendChild(iframe);
    if (item.title){ const t=document.createElement('div'); t.className='subtitle
mt-1'; t.textContent=item.title; container.appendChild(t); }
    return;
  }
  if (item.type === 'js'){
    container.classList.add('has-media');
    const jsSrc = item.content || '';
    const iframe = document.createElement('iframe');
    iframe.sandbox = sandboxForType('js', jsSrc);
    iframe.style.width='100%'; iframe.style.height='100%';
iframe.style.border='0';
    if (/^(https?:)?\/\//.test(jsSrc) || jsSrc.startsWith('/')) iframe.srcdoc =
htmlDoc("", "", `const s=document.createElement('script');s.src='$
{jsSrc}';document.body.appendChild(s);`);
    else iframe.srcdoc = htmlDoc("", "", jsSrc);
    container.appendChild(iframe);
    if (item.title){ const t=document.createElement('div'); t.className='subtitle
mt-1'; t.textContent=item.title; container.appendChild(t); }
    return;
  }
  if (item.type === 'css'){
    container.classList.add('has-media');
    const cssSrc = item.content || '';
    const iframe = document.createElement('iframe');
    iframe.sandbox = sandboxForType('code', cssSrc);
    iframe.style.width='100%'; iframe.style.height='100%';
iframe.style.border='0';
    if (/^(https?:)?\/\//.test(cssSrc) || cssSrc.startsWith('/')) {
      iframe.srcdoc = htmlDoc(`<div class="app">CSS preset loaded</div>`,
```

```
`@import url('${cssSrc}');`);
    } else {
      iframe.srcdoc = htmlDoc(`<div class="app">CSS preview</div>`, cssSrc);
    }
    container.appendChild(iframe);
    if (item.title){ const t=document.createElement('div'); t.className='subtitle mt-1'; t.textContent=item.title; container.appendChild(t); }
    return;
  }
  if (item.type === 'code'){
    const src = item.content || '';
    container.classList.add('has-media');
    const wrapped = src.trim().startsWith('<') ? src : htmlDoc(`<pre style="margin:0;padding:12px;white-space:pre-wrap">${escapeHtml(src)}</pre>`);
    const iframe = document.createElement('iframe');
    iframe.sandbox = sandboxForType('code', src);
    iframe.style.width='100%'; iframe.style.height='100%'; iframe.style.border='0';
    const blobUrl = blobFromContent(wrapped);
    if (blobUrl) {
      iframe.src = blobUrl;
      iframe.addEventListener('load', () => URL.revokeObjectURL(blobUrl), { once: true });
    } else {
      iframe.srcdoc = wrapped;
    }
    container.appendChild(iframe);
    if (item.title){ const t=document.createElement('div'); t.className='subtitle mt-1'; t.textContent=item.title; container.appendChild(t); }
    return;
  }
  if (item.type === 'phenotype'){
    const wrap = document.createElement('div');
    wrap.className = 'phenotype-widget';
    wrap.dataset.phenotype = 'cell';
    wrap.dataset.model = item.model || 'neurobio';
    wrap.dataset.genome = JSON.stringify(item.genome || {});
    container.appendChild(wrap);
    if (item.title){ const t=document.createElement('div'); t.className='subtitle mt-1'; t.textContent=item.title; container.appendChild(t); }
    return;
  }
  if (item.type === 'exosphere'){
    const iframe = document.createElement('iframe');
    iframe.sandbox = 'allow-scripts allow-same-origin';
    iframe.style.width='100%'; iframe.style.height='60vh';
```

```
    iframe.src = '/public/sims/exosphere.html';
    container.appendChild(iframe);
    if (item.title){ const t=document.createElement('div'); t.className='subtitle
mt-1'; t.textContent=item.title; container.appendChild(t); }
    return;
  }
  if (item.type === 'drawing'){
    const canvas = document.createElement('canvas');
    canvas.className = 'drawing-board';
    canvas.dataset.drawingId = String(item.id);
    container.appendChild(canvas);
    if (item.title){ const t=document.createElement('div'); t.className='subtitle
mt-1'; t.textContent=item.title; container.appendChild(t); }
    return;
  }
  if (item.type === 'markdown'){
    const html = md2html(item.content || item.description || '');
    const div = document.createElement('div'); div.style.textAlign='left';
div.innerHTML = html;
    container.appendChild(div);
    // links badges
    renderLinks(item, container);
    return;
  }
  // default text
  if (item.exponent){ const ex=document.createElement('div');
ex.className='subtext'; ex.textContent=item.exponent;
container.appendChild(ex); }
  const h3=document.createElement('h3'); h3.textContent=item.title||'';
container.appendChild(h3);
  if (item.subtext){ const st=document.createElement('div');
st.className='subtext'; st.textContent=item.subtext;
container.appendChild(st); }
  const sub=document.createElement('div'); sub.className='subtitle';
sub.textContent=item.subtitle||''; container.appendChild(sub);
  if (item.description){ const desc=document.createElement('div');
desc.className='small mt-1'; desc.textContent=item.description;
container.appendChild(desc); }
  renderLinks(item, container);
}

function primaryUrl(item){
  if (item.links && item.links.length){
    const l = item.links[0];
    return (typeof l === 'string') ? l : (l.url||'');
  }
  if (item.url) return item.url;
```

```
  if (item.content && /^https?:\/\//.test(item.content.trim())) return
item.content.trim();
  return '';
}

function renderLinks(item, container){
  const links=document.createElement('div'); links.className='links';
  (item.links||[]).forEach(l=>{
    const a=document.createElement('a');
    const url=(typeof l==='string')? l : (l.url||'#');
    const label=(typeof l==='string')? (new URL(url, location.origin)).hostname :
(l.label||'Link');
    a.href=url; a.target='_blank'; a.rel='noopener'; a.textContent=label;
links.appendChild(a);
  });
  if (links.children.length) container.appendChild(links);
}

function resolveVideoEmbed(item){
  const u = primaryUrl(item) || item.src || '';
  if (/youtube\.com\/watch\?v=|youtu\.be\//.test(u)){
    const id = (u.match(/[?&]v=([^&]+)/) || [null,null])[1] || u.split('/').pop();
    return `https://www.youtube.com/embed/${id}?autoplay=1&mute=1`;
  }
  if (/vimeo\.com\/(\d+)/.test(u)){
    const id = (u.match(/vimeo\.com\/(\d+)/)||[null,null])[1];
    return `https://player.vimeo.com/video/${id}?autoplay=1&muted=1`;
  }
  // direct video
  if (item.type === 'video' || /\.(mp4|webm|ogg|mov|m4v)(\?|$)/i.test(u)){
    const type = /\.mov(\?|$)/i.test(u) ? ' type="video/mp4"' : '';
    return {srcdoc:`<video controls autoplay muted loop playsinline
preload="auto" style="width:100%;height:100%"><source src="${u}"${type}
></video>`};
  }
  return u;
}

function attachHoverIframe(container, srcOrObj){
  let iframe=null, timer=null;
  container.addEventListener('mouseenter', ()=>{
    timer = setTimeout(()=>{
      if (iframe) return;
      iframe = document.createElement('iframe');
      iframe.setAttribute('loading','lazy');
      iframe.setAttribute('referrerpolicy','no-referrer');
      iframe.setAttribute('sandbox','allow-scripts allow-same-origin allow-popups
```

```
allow-forms');
      iframe.style.width='100%'; iframe.style.height='100%';
iframe.style.border='0'; iframe.style.background='#fff';
      if (typeof srcOrObj === 'string'){ iframe.src = srcOrObj; }
      else if (srcOrObj && typeof srcOrObj === 'object'){
        if (srcOrObj.srcdoc !== undefined){ iframe.srcdoc = srcOrObj.srcdoc; }
        else if (srcOrObj.src){ iframe.src = srcOrObj.src; }
      }
      container.appendChild(iframe);
    }, 250);
  });
  container.addEventListener('mouseleave', ()=>{
    clearTimeout(timer); if (iframe){ iframe.remove(); iframe=null; }
  }, {passive:true});
}

// Markdown (minimal)
function md2html(md){
  if (!md) return '';
  let s = md.replace(/&/g,'&amp;').replace(/</g,'&lt;').replace(/>/g,'&gt;');
  s = s.replace(/^###### (.*)$/gm, '<h6>$1</h6>')
       .replace(/^##### (.*)$/gm, '<h5>$1</h5>')
       .replace(/^#### (.*)$/gm, '<h4>$1</h4>')
       .replace(/^### (.*)$/gm, '<h3>$1</h3>')
       .replace(/^## (.*)$/gm, '<h2>$1</h2>')
       .replace(/^# (.*)$/gm, '<h1>$1</h1>');
  s = s.replace(/\*\*(.+?)\*\*/g, '<strong>$1</strong>')
       .replace(/\*(.+?)\*/g, '<em>$1</em>')
       .replace(/`([^`]+?)`/g, '<code class="inline">$1</code>')
       .replace(/!\[([^\]]*)\]\(([^)]+)\)/g, '<img alt="$1" src="$2" style="max-
width:100%"/>')
       .replace(/\[([^\]]+)\]\(([^)]+)\)/g, '<a href="$2" target="_blank"
rel="noopener">$1</a>');
  s = s.replace(/^- (.*)$/gm, '<li>$1</li>').replace(/(<li>.*<\/li>)/gs, '<ul>$1</
ul>');
  s = s.split(/\n{2,}/).map(p=> /^<h\d|<ul|<img|<p|<blockquote|<pre|
<code/.test(p) ? p : `<p>${p}</p>`).join('\n');
  return s;
}

function escapeHtml(s){ return String(s??'').replace(/[&<>"']/g,
m=>({'&':'&amp;','<':'&lt;','>':'&gt;','"':'&quot;'}[m])); }

// Selection + panel form
function selectCell(id){
  selectedId = id;
  $$('.cell.selected').forEach(x=>x.classList.remove('selected'));
```

```
  const el = $(`.cell[data-id="${id}"]`); if (el) el.classList.add('selected');
  fillForm();
}

function fillForm(){
  const it = data.find(d=>d.id===selectedId); if(!it) return;
  $('#cell-id').value = it.id||'';
  $('#cell-type').value = it.type||'text';
  $('#cell-title').value = it.title||'';
  $('#cell-subtitle').value = it.subtitle||'';
  $('#cell-exponent').value = it.exponent||'';
  $('#cell-subtext').value = it.subtext||'';
  $('#cell-description').value = it.content || it.description || it.htmlContent || '';
  $('#cell-link').value = (it.links && it.links[0]) ? (typeof it.links[0]==='string' ?
it.links[0] : `${it.links[0].label||''}|${it.links[0].url||''}`) : (it.url||'');
  $('#cell-image').value = it.audio || it.image || it.src || '';
  $('#cell-color').value = it.color || '#1e90ff';
  $('#cell-hoverColor').value = it.hoverColor || '#6ec6ff';
  $('#cell-group').value = it.group || '';
  $('#cell-tags').value = (it.tags||[]).join(', ');
  $('#cell-timestamp').value = it.timestamp ? toLocalDatetime(it.timestamp) : '';
  $('#cell-locked').checked = !!it.locked;
  $('#cell-row').value = it.field|0;
  $('#cell-col').value = it.axis|0;
  $('#cell-rowspan').value = Math.max(1, it.rowSpan|0);
  $('#cell-colspan').value = Math.max(1, it.colSpan|0);
  $('#cell-priority').value = it.priority || 'auto';
  $('#cell-scale').value = it.scale || 'auto';
}

function toLocalDatetime(iso){ const d=new Date(iso); const
pad=n=>String(n).padStart(2,'0'); return `${d.getFullYear()}-$
{pad(d.getMonth()+1)}-${pad(d.getDate())}T${pad(d.getHours())}:$
{pad(d.getMinutes())}`; }

// Form handlers
$('#cellForm').addEventListener('submit', (e)=>{
  e.preventDefault();
  if (selectedId==null) return toast('Select a cell','warning');
  const idx = data.findIndex(d=>d.id===selectedId); if(idx<0) return;
  const it = data[idx];
  it.type = $('#cell-type').value;
  it.title = $('#cell-title').value;
  it.subtitle = $('#cell-subtitle').value;
  it.exponent = $('#cell-exponent').value;
  it.subtext = $('#cell-subtext').value;
  const desc = $('#cell-description').value;
```

```
  if (it.type === 'markdown') it.content = desc;
  else if (it.type === 'html') it.content = desc;
  else if (it.type === 'js') it.content = desc;
  else if (it.type === 'code') it.content = desc;
  else if (it.type === 'link') it.url = desc || $('#cell-link').value;
  else it.description = desc;
const lnk = $('#cell-link').value.trim();
it.links = lnk ? (lnk.includes('|') ? [{label:lnk.split('|')[0].trim(), url:lnk.split('|')
[1].trim()}] : [lnk]) : [];
const img = $('#cell-image').value.trim();
if (it.type === 'image') it.image = img;
else if (it.type === 'video') it.url = img;
else if (it.type === 'audio') it.audio = img;
else it.image = img || it.image;
  it.color = $('#cell-color').value;
  it.hoverColor = $('#cell-hoverColor').value;
  it.group = $('#cell-group').value;
  it.tags = $('#cell-tags').value.split(',').map(s=>s.trim()).filter(Boolean);
  it.timestamp = $('#cell-timestamp').value ? new Date($('#cell-
timestamp').value).toISOString() : '';
  it.locked = $('#cell-locked').checked;
  it.priority = $('#cell-priority').value || 'auto';
  it.scale = $('#cell-scale').value || 'auto';
  const nRow = Math.max(0, parseInt($('#cell-row').value||'0',10));
  const nCol = Math.max(0, parseInt($('#cell-col').value||'0',10));
  const nRS = Math.max(1, parseInt($('#cell-rowspan').value||'1',10));
  const nCS = Math.max(1, parseInt($('#cell-colspan').value||'1',10));
  // check area free or ignore itself
  if (isAreaFree(nRow, nCol, nRS, nCS, it.id)){
    it.field = nRow; it.axis = nCol; it.rowSpan = nRS; it.colSpan = nCS;
  } else {
    toast('Space not free; position unchanged','warning');
  }
  saveLocal(); renderGrid(); selectCell(it.id);
  if ($('#livePreviewSwitch').checked){ /* live already reflected */ }
});

$('#cell-upload-apply').addEventListener('click', async ()=>{
  const input = $('#cell-upload');
  if (!input?.files?.length) return toast('No file selected','warning');

  const it = data.find(d=>d.id===selectedId);
  if (!it) return toast('Select a cell first','warning');

  const token = localStorage.getItem('authToken') ||
localStorage.getItem('ssoToken');
  const persist = async (file, type) => {
```

```
    return new Promise((resolve) => {
      const reader = new FileReader();
      reader.onload = async () => {
        const result = reader.result;
        if (!token) return resolve({ result });
        try {
          const res = await fetch('/api/cms/media', {
            method:'POST',
            headers:{ 'Content-Type':'application/json', 'Authorization': `Bearer ${token}` },
            body: JSON.stringify({ title: it.title || file.name, type, dataUrl: result, isPublic: true })
          });
          if (!res.ok) throw new Error(await res.text());
          const json = await res.json();
          resolve({ result, media: json.media });
        } catch(e) {
          console.warn(e);
          resolve({ result });
        }
      };
      if (type === 'image' || type === 'video' || type === 'audio') reader.readAsDataURL(file);
      else reader.readAsText(file);
    });
  };

  const files = Array.from(input.files);
  const imageFiles = files.filter(f => f.type.startsWith('image/'));

  if (imageFiles.length > 0) {
    it.type = 'image';
    it.images = it.images || [];
    // Reset images if not appending (simple logic for now: replace if upload)
    if (!it.images.length || confirm('Replace existing images? Cancel to append.')) {
      it.images = [];
    }

    for (const file of imageFiles) {
      const { result, media } = await persist(file, 'image');
      const url = (media?.file?.filename) ? `/media/${media.file.filename}` : result;
      it.images.push(url);
    }
    it.image = it.images[0]; // Set cover
    saveLocal(); renderGrid(); selectCell(it.id); openPanel(); toast('Images uploaded','success');
```

```
    return;
  }

  // Fallback for single other file types (take first)
  const file = files[0];
  const name = file.name.toLowerCase();

  // Helper to get type string
  let type = 'code';
  if (file.type.startsWith('video/') || /\.(mp4|webm|ogg|mov|m4v)$/.test(name))
type = 'video';
  else if (file.type.startsWith('audio/') || /\.(mp3|aiff|wav|ogg)$/.test(name)) type
= 'audio';
  else if (name.endsWith('.html')) type = 'html';
  else if (name.endsWith('.js')) type = 'js';
  else if (name.endsWith('.css')) type = 'css';
  else if (name.endsWith('.json')) type = 'json';
  else if (name.endsWith('.md')) type = 'markdown';

  const { result, media } = await persist(file, type);
  const url = (media?.file?.filename) ? `/media/${media.file.filename}` : result;

  if (type === 'video') { it.type = 'video'; it.url = url; }
  else if (type === 'audio') { it.type = 'audio'; it.audio = url; }
  else if (type === 'html') { it.type = 'html'; it.content = String(result); }
  else if (type === 'js') { it.type = 'js'; it.content = String(result); }
  else if (type === 'css') { it.type = 'css'; it.content = String(result); }
  else if (type === 'json') { it.type = 'json'; it.content = String(result); }
  else if (type === 'markdown') { it.type = 'markdown'; it.content =
String(result); }
  else { it.type = 'code'; it.content = String(result); }

  $('#cell-description').value = it.content || '';
  saveLocal(); renderGrid(); selectCell(it.id); openPanel();
toast('Uploaded','success');
});

$('[data-action="add-cell"]').addEventListener('click', ()=>{
  const newId = data.length ? Math.max(...data.map(d=>d.id||0)) + 1 : 1;
  const it = {
    id:newId, field:0, axis:0, rowSpan:1, colSpan:1, type:'text',
    title:'New', subtitle:'', description:'', color:'#1e90ff', hoverColor:'#6ec6ff',
    group:'', tags:[], locked:false, priority:'auto', scale:'auto'
  };
  data.push(it); saveLocal(); renderGrid(); selectCell(it.id); openPanel();
toast('Added','success');
});
```

```
$('[data-action="autocomplete-cell"]').addEventListener('click', ()=>{
  if (selectedId==null) return toast('Select a cell','warning');
  const it = data.find(d=>d.id===selectedId); if(!it) return;
  applyAutoFill(it);
  saveLocal(); renderGrid(); selectCell(it.id); openPanel(); toast('Auto-
filled','success');
});

$('[data-action="vibe-cell"]').addEventListener('click', ()=>{
  if (selectedId==null) return toast('Select a cell','warning');
  const it = data.find(d=>d.id===selectedId); if(!it) return;
  applyVibe(it);
  saveLocal(); renderGrid(); selectCell(it.id); openPanel(); toast('Vibe
applied','success');
});

$('[data-action="delete-cell"]').addEventListener('click', ()=>{
  if (selectedId==null) return;
  const idx = data.findIndex(d=>d.id===selectedId); if(idx<0) return;
  if (!confirm('Delete #'+selectedId+'?')) return;
  data.splice(idx,1); selectedId=null; saveLocal(); renderGrid();
toast('Deleted','info');
});

// Filters
$('[data-action="apply-filters"]').addEventListener('click', ()=>{
  filters.tag = $('#filter-tag').value.trim();
  filters.group = $('#filter-group').value.trim();
  filters.from = $('#filter-from').value;
  filters.to = $('#filter-to').value;
  renderGrid();
});
$('[data-action="clear-filters"]').addEventListener('click', ()=>{
  filters = {tag:'',group:'',from:'',to:''};
  $('#filter-tag').value=''; $('#filter-group').value=''; $('#filter-from').value=''; $
('#filter-to').value='';
  renderGrid();
});

// View
$('[data-action="toggle-theme"]').addEventListener('click', ()=>{
  const cur = document.documentElement.getAttribute('data-
theme')==='dark' ? 'light' : 'dark';
  document.documentElement.setAttribute('data-theme', cur);
  localStorage.setItem('spectra.theme', cur);
});
```

```
$('[data-action="zoom-in"]').addEventListener('click', ()=>{ zoom =
Math.min(2, zoom + 0.1); sizeGrid(); });
$('[data-action="zoom-out"]').addEventListener('click', ()=>{ zoom =
Math.max(0.5, zoom - 0.1); sizeGrid(); });
$('[data-action="reset-zoom"]').addEventListener('click', ()=>{ zoom = 1;
sizeGrid(); });

// Data menu
$('[data-action="save-local"]').addEventListener('click', ()=>{ saveLocal();
toast('Saved','success'); });
$('[data-action="load-local"]').addEventListener('click', ()=>{ loadLocal();
toast('Loaded','success'); });
$('[data-action="clear-local"]').addEventListener('click',
()=>{ localStorage.removeItem('spectra.magnetic'); toast('Cleared','info'); });
$('[data-action="download-json"]').addEventListener('click', ()=>{
  const blob = new Blob([JSON.stringify(data,null,2)], {type:'application/json'});
  const a = document.createElement('a'); a.href=URL.createObjectURL(blob);
a.download='spectra-magnetic.json'; a.click(); URL.revokeObjectURL(a.href);
});
$('[data-action="import-file"]').addEventListener('click', ()=> fileInput.click());
fileInput.addEventListener('change', async (e)=>{
  const f=e.target.files[0]; if(!f) return; const text=await f.text();
  try{ data = adaptInput(JSON.parse(text)); renderGrid();
toast('Imported','success'); }catch(err){ toast('Invalid JSON','danger'); }
  e.target.value='';
});
$('[data-action="import-url"]').addEventListener('click', async ()=>{
  const url = $('#jsonUrl').value.trim(); if(!url) return toast('Enter URL','warning');
  try{ const res=await fetch(url); if(!res.ok) throw new Error(res.statusText);
const json=await res.json(); data = adaptInput(json); renderGrid();
toast('Loaded','success'); }
  catch(e){ toast('Load failed','danger'); }
});

// Exports
$('[data-action="export-html"]').addEventListener('click', ()=>{
  const snap = buildSnapshotHtml();
  const blob = new Blob([snap], {type:'text/html'});
  const a = document.createElement('a'); a.href=URL.createObjectURL(blob);
a.download='spectra-magnetic.html'; a.click(); URL.revokeObjectURL(a.href);
});
$('[data-action="export-md"]').addEventListener('click', ()=>{
  const lines = data.map(d=>{
    const pos = `Row ${d.field}, Col ${d.axis}${(d.colSpan>1||d.rowSpan>1)?`
(span ${d.colSpan||1}×${d.rowSpan||1})`:''}`;
    const title = d.title || (d.type==='image'?'Image':
d.type==='video'?'Video':'Cell '+d.id);
```

```
    const desc = (d.type==='markdown') ? (d.content||'') : (d.description||'');
    return `## ${title}\n- Position: ${pos}\n- Type: ${d.type}\n${desc?
('\n'+desc+'\n'):''}`;
  }).join('\n\n');
  const blob = new Blob([lines], {type:'text/markdown'});
  const a = document.createElement('a'); a.href=URL.createObjectURL(blob);
a.download='spectra-magnetic.md'; a.click(); URL.revokeObjectURL(a.href);
});
$('[data-action="export-csv"]').addEventListener('click', ()=>{
  const {rows, cols} = dims();
  const mat = Array.from({length:rows},()=> Array(cols).fill(''));
  data.forEach(d=>{ mat[d.field][d.axis] = (d.title||'') + (d.subtitle?('
'+d.subtitle):''); });
  const csv = mat.map(r=> r.map(v=>`"${String(v??'').replace(/"/
g,'""')}"`).join(',')).join('\n');
  const blob = new Blob([csv], {type:'text/csv'});
  const a = document.createElement('a'); a.href=URL.createObjectURL(blob);
a.download='spectra-magnetic.csv'; a.click(); URL.revokeObjectURL(a.href);
});
$('[data-action="export-txt"]').addEventListener('click', ()=>{
  const txt = data.map(d=>`[${d.field},${d.axis}] ${d.title||''} ${d.subtitle||''}
`).join('\n');
  const blob = new Blob([txt], {type:'text/plain'}); const a =
document.createElement('a'); a.href=URL.createObjectURL(blob);
a.download='spectra-magnetic.txt'; a.click(); URL.revokeObjectURL(a.href);
});
$('[data-action="export-svg"]').addEventListener('click', ()=>{
  const {rows, cols} = dims();
  const size = getCellSize(), gap = getGap();
  const w = cols*(size+gap)-gap, h = rows*(size+gap)-gap;
  const xmlns='http://www.w3.org/2000/svg';
  const svg = document.createElementNS(xmlns,'svg');
  svg.setAttribute('xmlns',xmlns); svg.setAttribute('width',w);
svg.setAttribute('height',h); svg.setAttribute('viewBox',`0 0 ${w} ${h}`);
  const bg = document.createElementNS(xmlns,'rect'); bg.setAttribute('x',0);
bg.setAttribute('y',0); bg.setAttribute('width',w); bg.setAttribute('height',h);
bg.setAttribute('fill','#ffffff'); svg.appendChild(bg);
  data.forEach(d=>{
    const x = d.axis*(size+gap), y = d.field*(size+gap), w0 = (d.colSpan||1)*size +
((d.colSpan||1)-1)*gap, h0 = (d.rowSpan||1)*size + ((d.rowSpan||1)-1)*gap;
    const rect = document.createElementNS(xmlns,'rect');
rect.setAttribute('x',x); rect.setAttribute('y',y); rect.setAttribute('width',w0);
rect.setAttribute('height',h0);
    rect.setAttribute('fill', d.color||'#999'); rect.setAttribute('rx','8');
rect.setAttribute('ry','8'); svg.appendChild(rect);
    const text = document.createElementNS(xmlns,'text'); text.setAttribute('x', x
+ w0/2); text.setAttribute('y', y + h0/2); text.setAttribute('text-anchor','middle');
```

```
text.setAttribute('font-family','Inter, Arial'); text.setAttribute('font-size','14');
text.setAttribute('fill','#fff'); text.textContent=d.title||''; svg.appendChild(text);
  });
  const s = new XMLSerializer().serializeToString(svg);
  const blob = new Blob([s], {type:'image/svg+xml'}); const
a=document.createElement('a'); a.href=URL.createObjectURL(blob);
a.download='spectra-magnetic.svg'; a.click(); URL.revokeObjectURL(a.href);
});
$('[data-action="export-pdf"]').addEventListener('click', async ()=>{
  if (!window.ComposerPdfExport?.exportComposerPdf){
    toast('PDF exporter not loaded','danger');
    return;
  }
  let cfg = {};
  try {
    const res = await fetch('/api/config');
    if (res.ok) cfg = await res.json();
  } catch {}
  try {
    await window.ComposerPdfExport.exportComposerPdf(data, {
      title: 'Composer Print Planche',
      subtitle: 'Golden ratio hierarchy and minimal narrative grid',
      config: cfg?.composerExport || {},
      seed: cfg?.meta?.seed || String(Date.now())
    });
    toast('Print dialog opened. Choose Save as PDF.','success');
  } catch (err){
    toast(`PDF export failed: ${err?.message || err}`,'danger');
  }
});

// API stub menu
$('[data-action="api-save"]').addEventListener('click', apiSave);
$('[data-action="api-load"]').addEventListener('click', apiLoad);

// Group n apply
$('[data-action="group-apply"]').addEventListener('click', ()=>{
  if (selectedId==null) return toast('Select a cell','warning');
  const it = data.find(d=>d.id===selectedId); if(!it) return;
  const n = Math.max(1, parseInt($('#group-n').value||'1',10));
  // attempt col-span change if free
  if (isAreaFree(it.field, it.axis, it.rowSpan||1, n, it.id)){
    it.colSpan = n; saveLocal(); renderGrid(); selectCell(it.id);
toast('Applied','success');
  } else { toast('Area not free','warning'); }
});
```

```
// Reorder rows (basic)
$('[data-action="reorder-rows"]').addEventListener('click', ()=>{
  const currentRows = Math.max(...data.map(d=>d.field))+1;
  const list = document.createElement('div');
  list.className='p-3';
  for(let r=0;r<currentRows;r++){
    const row = document.createElement('div'); row.className='border rounded
p-2 mb-2 d-flex align-items-center justify-content-between';
    row.draggable = true; row.dataset.row = r;
    row.innerHTML = `<strong>Row ${r}</strong><span class="text-muted
small">drag</span>`;
    row.addEventListener('dragstart', (e)=> e.dataTransfer.setData('text/plain',
String(r)));
    row.addEventListener('dragover', (e)=> e.preventDefault());
    row.addEventListener('drop', (e)=>{
      e.preventDefault();
      const from = parseInt(e.dataTransfer.getData('text/plain'),10);
      const to = parseInt(row.dataset.row,10);
      if (from===to) return;
      data.forEach(c=>{
        if(c.field===from) c.field=-1;
        else if(from<to && c.field>from && c.field<=to) c.field--;
        else if(from>to && c.field<from && c.field>=to) c.field++;
      });
      data.forEach(c=>{ if(c.field===-1) c.field=to; });
      saveLocal(); renderGrid();
    });
    list.appendChild(row);
  }
  const dlg = document.createElement('div');
  dlg.className='modal fade'; dlg.tabIndex=-1; dlg.innerHTML = `
  <div class="modal-dialog"><div class="modal-content">
    <div class="modal-header"><h5 class="modal-title">Reorder Rows</h5>
      <button type="button" class="btn-close" data-bs-dismiss="modal"></
button>
    </div>
    <div class="modal-body"></div>
    <div class="modal-footer"><button type="button" class="btn btn-
secondary" data-bs-dismiss="modal">Close</button></div>
  </div></div>`;
  dlg.querySelector('.modal-body').appendChild(list);
  document.body.appendChild(dlg);
  const modal = new bootstrap.Modal(dlg); modal.show();
  dlg.addEventListener('hidden.bs.modal', ()=> dlg.remove());
});
```

```
// Bulk edit (JSON)
$('[data-action="bulk-edit"]').addEventListener('click', ()=>{
  const input=prompt('Paste JSON array of cells to merge/replace. Use [] to
clear.', JSON.stringify(data,null,2));
  if (input==null) return;
  try{ const arr=JSON.parse(input); data = adaptInput(arr); saveLocal();
renderGrid(); toast('Bulk updated','success'); }
  catch(e){ console.error(e); toast('Invalid JSON','danger'); }
});

// Preview lightbox
const lightbox = $('#lightbox'), lbBody = $('#lightboxBody');
lightbox.addEventListener('click', (e)=>{ if (e.target.dataset.close==='1')
closePreview(); });
$('.close', lightbox).addEventListener('click', closePreview);
function openPreview(cellEl){
  const id = Number(cellEl?.dataset?.id);
  const it = data.find(d=>d.id===id) || {};
  $('#lb-id').textContent = '#'+(it.id??'?');
  $('#lb-title').textContent = it.title || '';
  $('#lb-subtitle').textContent = it.subtitle ? '— '+it.subtitle : '';
  const meta=[]; if(it.group) meta.push('Group: '+it.group); if (it.tags &&
it.tags.length) meta.push('Tags: '+it.tags.join(', ')); if (it.timestamp)
meta.push(new Date(it.timestamp).toLocaleString());
  $('#lb-meta').textContent = meta.join(' • ');
  lbBody.innerHTML='';
  const hero = document.createElement('div');
  hero.className = 'p-3 rounded text-white mb-3';
  hero.style.backgroundImage = `linear-gradient(135deg, ${it.color||'#334155'}
0%, ${it.hoverColor||it.color||'#6b7280'} 100%)`;
  hero.innerHTML = `<h3 class="m-0">${escapeHtml(it.title||'')}</h3><div
class="small">${escapeHtml(it.subtext||'')}</div>`;
  lbBody.appendChild(hero);
  const cont = document.createElement('div'); cont.className='mb-2';
  if (it.type==='markdown'){ cont.innerHTML = md2html(it.content||''); }
  else if (it.type==='html'){
    const iframe=document.createElement('iframe');
    const htmlSrc = it.content || '';
    iframe.sandbox=sandboxForType('html', htmlSrc);
    iframe.style.width='100%'; iframe.style.height='60vh';
    if (/^(https?:)?\/\//.test(htmlSrc) || htmlSrc.startsWith('/') ||
htmlSrc.startsWith('data:')) {
      iframe.src = htmlSrc;
    } else {
      iframe.srcdoc = `<!doctype html><html><head><meta charset="utf-8"></
head><body><script>
```

```
        const txt = ${JSON.stringify(String(it.content || ''))};
        const parser = new DOMParser();
        const doc = parser.parseFromString(txt, 'text/html');
        const err = doc.createElement('script');
        err.textContent = ${JSON.stringify(errorOverlayScript())};
        (doc.body || doc.head || doc.documentElement).appendChild(err);
        document.open();
        document.write(doc.documentElement.outerHTML);
        document.close();
      <\/script></body></html>`;
    }
    cont.appendChild(iframe);
  }
  else if (it.type==='js'){
    const iframe=document.createElement('iframe');
    const jsSrc = it.content || '';
    iframe.sandbox=sandboxForType('js', jsSrc);
    iframe.style.width='100%'; iframe.style.height='60vh';
    iframe.srcdoc = (/^(https?:)?\/\//.test(jsSrc) || jsSrc.startsWith('/'))
      ? `<!doctype html><html><head><meta charset="utf-8"></
head><body><script src="${jsSrc}"><\/script></body></html>`
      : `<!doctype html><html><head><meta charset="utf-8"></
head><body><script>${jsSrc}<\/script></body></html>`;
    cont.appendChild(iframe);
  }
  else if (it.type==='css'){
    const iframe=document.createElement('iframe');
    const cssSrc = it.content || '';
    iframe.sandbox=sandboxForType('code', cssSrc);
    iframe.style.width='100%'; iframe.style.height='60vh';
    iframe.srcdoc = (/^(https?:)?\/\//.test(cssSrc) || cssSrc.startsWith('/'))
      ? `<!doctype html><html><head><meta charset="utf-8"><link
rel="stylesheet" href="${cssSrc}"></head><body><div class="app">CSS
preview</div></body></html>`
      : `<!doctype html><html><head><meta charset="utf-8"><style>${cssSrc}
<\/style></head><body><div class="app">CSS preview</div></body></html>`;
    cont.appendChild(iframe);
  }
  else if (it.type==='image'){ const img=document.createElement('img');
img.src=it.image||it.src||''; img.style.maxWidth='100%';
cont.appendChild(img); }
  else if (it.type==='video'){
    const url = it.url || it.image || primaryUrl(it);
    if (url && /\.(mp4|webm|ogg|mov|m4v|quicktime)(\?|$)/i.test(url)) {
      const video=document.createElement('video');
      video.controls=true;
      if (/\.mov(\?|$)/i.test(url)) {
```

```
        const s = document.createElement('source');
        s.src = url;
        s.type = 'video/mp4';
        video.appendChild(s);
      } else {
        video.src=url;
      }
      video.style.width='100%';
      video.autoplay=true;
      video.loop=true;
      video.muted=true;
      video.playsInline=true;
      video.setAttribute('playsinline','');
      video.setAttribute('muted','');
      video.setAttribute('autoplay','');
      cont.appendChild(video);
      video.play().catch(() => {});
    } else {
      attachHoverIframe(cont, resolveVideoEmbed(it));
    }
  }
  else if (it.type==='audio'){ const audio=document.createElement('audio');
audio.controls=true; audio.src=it.audio||it.src||it.image||primaryUrl(it);
audio.style.width='100%'; cont.appendChild(audio); }
  else if (it.type==='code'){
    const iframe=document.createElement('iframe');
    iframe.sandbox=sandboxForType('code', it.content || '');
    iframe.style.width='100%'; iframe.style.height='60vh';
    const src = it.content || '';
    const wrapped = src.trim().startsWith('<') ? src : `<!doctype
html><html><head><meta charset="utf-8"></head><body><pre
style="margin:0;padding:12px;white-space:pre-wrap">${escapeHtml(src)}</
pre></body></html>`;
    const blob = new Blob([wrapped], { type: "text/html" });
    const url = URL.createObjectURL(blob);
    iframe.src = url;
    iframe.addEventListener('load', ()=> URL.revokeObjectURL(url), { once:
true });
    cont.appendChild(iframe);
  }
  else if (it.type==='drawing'){
    const canvas=document.createElement('canvas');
    canvas.className='drawing-preview';
    cont.appendChild(canvas);
    if (window.DrawingBoard?.renderStatic) {
      let strokes = [];
      try { strokes = JSON.parse(it.content || '[]'); } catch {}
```

```
      window.DrawingBoard.renderStatic(canvas, strokes);
    }
  }
  else if (it.type==='link'){ attachHoverIframe(cont, primaryUrl(it)); }
  else { cont.textContent = it.description||''; }
  lbBody.appendChild(cont);
  lightbox.classList.add('show'); lightbox.setAttribute('aria-hidden','false');
document.body.style.overflow='hidden';
}
function closePreview(){ lightbox.classList.remove('show');
lightbox.setAttribute('aria-hidden','true'); document.body.style.overflow=''; }

// Drag & resize
let dragState = null, resizeState = null, dropIndicator = null;

function enableDrag(el, handle, item){
  handle.addEventListener('mousedown', (e)=>{
    e.preventDefault();
    const rect = el.getBoundingClientRect();
    dragState = {
      id: item.id,
      startX: e.clientX, startY: e.clientY,
      offsetX: e.clientX - rect.left, offsetY: e.clientY - rect.top,
      w: rect.width, h: rect.height
    };
    createDropIndicator(rect);
    document.addEventListener('mousemove', onDragMove);
    document.addEventListener('mouseup', onDragEnd, {once:true});
  });
}

function onDragMove(e){
  if(!dragState) return;
  const gRect = gridEl.getBoundingClientRect();
  const size = getCellSize()*zoom, gap = getGap()*zoom;
  const x = e.clientX - gRect.left - dragState.offsetX;
  const y = e.clientY - gRect.top - dragState.offsetY;
  const col = Math.max(0, Math.round(x / (size+gap)));
  const row = Math.max(0, Math.round(y / (size+gap)));
  const it = data.find(d=>d.id===dragState.id);
  const rs = it.rowSpan||1, cs = it.colSpan||1;
  setDropIndicator(row, col, rs, cs);
}

function onDragEnd(e){
  const gRect = gridEl.getBoundingClientRect();
  const size = getCellSize()*zoom, gap = getGap()*zoom;
```

```
  const it = data.find(d=>d.id===dragState.id);
  const x = e.clientX - gRect.left - dragState.offsetX;
  const y = e.clientY - gRect.top - dragState.offsetY;
  const col = Math.max(0, Math.round(x / (size+gap)));
  const row = Math.max(0, Math.round(y / (size+gap)));
  const rs = it.rowSpan||1, cs = it.colSpan||1;
  if (isAreaFree(row, col, rs, cs, it.id)){
    it.field = row; it.axis = col; saveLocal(); renderGrid(); selectCell(it.id);
  } else {
    toast('Target occupied','warning');
  }
  destroyDropIndicator();
  document.removeEventListener('mousemove', onDragMove);
  dragState = null;
}

function enableResize(el, handles, item){
  const start = (mode)=>(e)=>{
    e.preventDefault();
    const rect = el.getBoundingClientRect();
    resizeState = {
      id: item.id, mode,
      startX: e.clientX, startY: e.clientY,
      rectW: rect.width, rectH: rect.height,
      row: item.field|0, col: item.axis|0, rs: item.rowSpan||1, cs: item.colSpan||1
    };
    createDropIndicator(rect);
    document.addEventListener('mousemove', onResizeMove);
    document.addEventListener('mouseup', onResizeEnd, {once:true});
  };
  handles.corner.addEventListener('mousedown', start('corner'));
  handles.right.addEventListener('mousedown', start('right'));
  handles.bottom.addEventListener('mousedown', start('bottom'));
}

function onResizeMove(e){
  if(!resizeState) return;
  const it = data.find(d=>d.id===resizeState.id);
  const size = getCellSize()*zoom, gap = getGap()*zoom;
  const dx = e.clientX - resizeState.startX;
  const dy = e.clientY - resizeState.startY;
  let cs = resizeState.cs, rs = resizeState.rs;
  if (resizeState.mode==='right' || resizeState.mode==='corner'){
    const colsDelta = Math.round(dx / (size+gap));
    cs = Math.max(1, resizeState.cs + colsDelta);
  }
  if (resizeState.mode==='bottom' || resizeState.mode==='corner'){
```

```
    const rowsDelta = Math.round(dy / (size+gap));
    rs = Math.max(1, resizeState.rs + rowsDelta);
  }
  // visual indicator at same origin, new span
  setDropIndicator(resizeState.row, resizeState.col, rs, cs);
}

function onResizeEnd(e){
  const it = data.find(d=>d.id===resizeState.id);
  const size = getCellSize()*zoom, gap = getGap()*zoom;
  const dx = e.clientX - resizeState.startX;
  const dy = e.clientY - resizeState.startY;
  let cs = resizeState.cs, rs = resizeState.rs;
  if (resizeState.mode==='right' || resizeState.mode==='corner'){
    const colsDelta = Math.round(dx / (size+gap));
    cs = Math.max(1, resizeState.cs + colsDelta);
  }
  if (resizeState.mode==='bottom' || resizeState.mode==='corner'){
    const rowsDelta = Math.round(dy / (size+gap));
    rs = Math.max(1, resizeState.rs + rowsDelta);
  }
  if (isAreaFree(resizeState.row, resizeState.col, rs, cs, it.id)){
    it.rowSpan = rs; it.colSpan = cs; saveLocal(); renderGrid(); selectCell(it.id);
  } else {
    toast('Area not free','warning');
  }
  destroyDropIndicator();
  document.removeEventListener('mousemove', onResizeMove);
  resizeState = null;
}

function createDropIndicator(rect){
  destroyDropIndicator();
  dropIndicator = document.createElement('div');
  dropIndicator.className='drop-indicator';
  document.body.appendChild(dropIndicator);
  positionIndicator(rect);
}
function positionIndicator(rect){
  dropIndicator.style.left = rect.left+'px';
  dropIndicator.style.top = rect.top+'px';
  dropIndicator.style.width = rect.width+'px';
  dropIndicator.style.height = rect.height+'px';
}
function setDropIndicator(row, col, rs, cs){
  // If dragging beyond current columns, temporarily expand the visual grid so
the indicator sits on-grid
```

```
  const size = getCellSize()*zoom, gap = getGap()*zoom;
  const { cols: curCols } = dims();
  const neededCols = Math.max(curCols, col + cs);
  if (neededCols > curCols){
    gridEl.style.gridTemplateColumns = `repeat(${neededCols}, var(--cell-
size))`;
  }
  const gRect = gridEl.getBoundingClientRect();
  const x = gRect.left + col*(size+gap);
  const y = gRect.top + row*(size+gap);
  const w = cs*size + (cs-1)*gap;
  const h = rs*size + (rs-1)*gap;
  dropIndicator.style.left = x+'px';
  dropIndicator.style.top = y+'px';
  dropIndicator.style.width = w+'px';
  dropIndicator.style.height = h+'px';
}
function destroyDropIndicator(){ if (dropIndicator){ dropIndicator.remove();
dropIndicator=null; } sizeGrid(); }


// I18n (basic)
$$('[data-lang]').forEach(btn=> btn.addEventListener('click', ()=>{
  const l = btn.getAttribute('data-lang'); localStorage.setItem('spectra.lang', l);
  toast('Language: '+l.toUpperCase(),'info');
}));


// Adapt input
function adaptInput(input){
  let arr=[];
  const base = (input && input.data) ? input.data : input;
  if (Array.isArray(base)){
    base.forEach((cell,i)=>{
      const it = Object.assign({}, cell);
      it.field = (typeof cell.field==='number') ? cell.field : (cell.row ?? 0);
      it.axis = (typeof cell.axis==='number') ? cell.axis : (cell.col ?? i);
      it.rowSpan = Math.max(1, cell.rowSpan || cell.ratio || 1);
      it.colSpan = Math.max(1, cell.colSpan || 1);
      it.type = cell.type || (cell.htmlContent ? 'html' : (cell.audio ? 'audio' :
(cell.image ? 'image' : (cell.content && /\.(mp3|aiff|wav|ogg)(\?|$)/
i.test(cell.content) ? 'audio' : (cell.content && /^https?:\/\//.test(cell.content)?
'link' : 'text')))));
      it.priority = it.priority || 'auto';
      it.scale = it.scale || 'auto';
      if (!it.id) it.id = i+1;
      arr.push(it);
```

```
    });
  } else if (base && typeof base==='object' && Array.isArray(base.rows)){
    base.rows.forEach((row,r)=> ((row.cells||[]).forEach((cell,c)=>{
      const it = Object.assign({}, cell); it.field = (row.index ?? r); it.axis =
(cell.axis ?? c);
      it.rowSpan = Math.max(1, cell.rowSpan || cell.ratio || 1); it.colSpan =
Math.max(1, cell.colSpan || 1);
      it.priority = it.priority || 'auto';
      it.scale = it.scale || 'auto';
      if (!it.id) it.id = arr.length+1; arr.push(it);
    })));
  } else {
    console.warn('Unknown input shape');
  }
  return arr;
}

// Notes stack
function updateNotesStack(){
  const container = $('#notesStack');
  const notes = data.filter(d=> (d.description||d.content||'').trim().length>0);
  container.innerHTML = '';
  if (!notes.length){ const p=document.createElement('p'); p.className='text-
muted m-0'; p.textContent='Notes added to cells will appear here.';
container.appendChild(p); return; }
  notes.sort((a,b)=> (b.timestamp||'').localeCompare(a.timestamp||''));
  notes.forEach(n=>{
    const div = document.createElement('div'); div.className='border
rounded-2 p-2 mb-2';
    const title=n.title||`#${n.id}`; const ts=n.timestamp?new
Date(n.timestamp).toLocaleString():'';
    div.innerHTML = `<div class="d-flex justify-content-between"><strong>$
{escapeHtml(title)}</strong><small class="text-muted">${ts}</small></div>
    <div class="small mt-1">${escapeHtml((n.description||
n.content||'').slice(0,280))}${(n.description||n.content||'').length>280?'...':''}</
div>`;
    div.addEventListener('click', ()=> selectCell(n.id));
    container.appendChild(div);
  });
}

// Snapshot HTML export
function buildSnapshotHtml(){
  const { cols } = dims();
  const css = `body{background:#f8f9fa;font-family:Inter,Arial,sans-
serif} .grid{display:grid;gap:6px;grid-auto-rows:140px;justify-content:start}
  .cell{position:relative;border:1px solid #e5e7eb;border-
```

```
radius:.6rem;overflow:hidden;background:#334155;color:#fff}
  .cell-content{height:100%;padding:.5rem;display:flex;flex-
direction:column;align-items:center;justify-content:center;text-align:center}`;
  const esc = (s)=> String(s??'').replace(/&/g,'&amp;').replace(/</
g,'&lt;').replace(/>/g,'&gt;');
  const cells = data.map(d=>{
    const style = `grid-row:${d.field+1} / span ${d.rowSpan||1}; grid-column:$
{d.axis+1} / span ${d.colSpan||1}; background:${d.color||'#334155'};`;
    let inner='';
    if (d.type==='markdown'){ inner = md2html(d.content||''); }
    else if (d.type==='html'){ inner = esc(d.content||''); }
    else if (d.type==='image'){ inner = `<img src="${esc(d.image||d.src||'')}"
style="max-width:100%;max-height:100%"/>`; }
    else if (d.type==='link'){ inner = `<a href="${esc(primaryUrl(d))}"
target="_blank" rel="noopener">${esc(d.title||primaryUrl(d)||'link')}</a>`; }
    else if (d.type==='video'){ inner = `<div>▶ ${esc(d.title||'Video')}</div>`; }
    else { inner = `<h3>${esc(d.title||'')}</h3><div class="subtitle">$
{esc(d.subtitle||'')}</div>`; }
    return `<div class="cell" style="${style}"><div class="cell-content">${inner}
</div></div>`;
  }).join('');
  return `<!doctype html><html><head><meta charset="utf-8"><meta
name="viewport" content="width=device-width,initial-scale=1">
  <title>Spectra Magnetic Snapshot</title><style>${css}</style></
head><body><div class="container py-3">
  <div class="grid" style="grid-template-columns:repeat(${cols},140px)">$
{cells}</div></div></body></html>`;
}

// Keyboard
document.addEventListener('keydown', (e)=>{
  if(e.key==='Escape'){ closePanel(); closePreview(); return; }
  if(e.target && ['INPUT','TEXTAREA','SELECT'].includes(e.target.tagName))
return;
  const idx = data.findIndex(d=>d.id===selectedId);
  if(e.key==='Enter'){ const el = $(`.cell[data-id="${selectedId}"]`); if(el)
openPreview(el); }
  if(e.key.toLowerCase()==='e'){ if(selectedId!=null) openPanel(); }
  if(['ArrowLeft','ArrowRight','ArrowUp','ArrowDown'].includes(e.key)){
    e.preventDefault();
    let f=0,a=0;
    if(idx>=0){ f=data[idx].field; a=data[idx].axis; }
    if(e.key==='ArrowLeft') a=Math.max(0,a-1);
    if(e.key==='ArrowRight') a=a+1;
    if(e.key==='ArrowUp') f=Math.max(0,f-1);
    if(e.key==='ArrowDown') f=f+1;
    const candidate = data.find(d=>d.field===f && d.axis===a) || data[0];
```

```
    if(candidate) selectCell(candidate.id);
  }
});

// Init
async function init(){
  const savedTheme = localStorage.getItem('spectra.theme'); if(savedTheme)
document.documentElement.setAttribute('data-theme', savedTheme);
  const saved = localStorage.getItem('spectra.magnetic'); data = saved ?
adaptInput(JSON.parse(saved)) : defaultData();
  try { await loadCurrentUser(); } catch {}
  renderGrid();
  if (data.length) selectCell(data[0].id);
  const nav = document.getElementById('topNav'); if(nav)
document.documentElement.style.setProperty('--navbar-h',
nav.offsetHeight+'px');
  window.addEventListener('resize', sizeGrid);
}
init();

// Planches Manager
const planchesModal = new
bootstrap.Modal(document.getElementById('planchesModal'));
const planchesList = document.getElementById('planchesList');
const plancheSaveForm = document.getElementById('plancheSaveForm');
const playbookPresetsModal = new
bootstrap.Modal(document.getElementById('playbookPresetsModal'));
const playbookPresetList = document.getElementById('playbookPresetList');
const drawingStudioModal = new
bootstrap.Modal(document.getElementById('drawingStudioModal'));
const drawingStudioCanvas =
document.getElementById('drawingStudioCanvas');
const openDrawingStudioBtn =
document.getElementById('openDrawingStudio');

function readPlancheTargetsFromForm(){
  const targets = [];
  if ($('#plancheTargetDisplay').checked) targets.push('display');
  if ($('#plancheTargetHome').checked) targets.push('home');
  if ($('#plancheTargetLive').checked) targets.push('live');
  return targets.length ? targets : ['display'];
}

function writePlancheTargetsToForm(targets){
  const set = new Set((Array.isArray(targets) ? targets :
[]).map((v)=>String(v||'').toLowerCase()));
  $('#plancheTargetDisplay').checked = set.has('display') || !set.size;
```

```
  $('#plancheTargetHome').checked = set.has('home');
  $('#plancheTargetLive').checked = set.has('live');
}

$('[data-action="manage-planches"]').addEventListener('click', ()=>{
  planchesModal.show();
  loadPlanchesList();
  // Reset save form
  $('#plancheId').value = '';
  $('#plancheName').value = '';
  $('#planchePublic').checked = false;
  writePlancheTargetsToForm(['display']);
});

$('[data-action="open-playbook-presets"]').addEventListener('click', ()=>{
  playbookPresetsModal.show();
  const presets = buildPlaybookPresets();
  if (!playbookPresetList) return;
  playbookPresetList.innerHTML = '';
  presets.forEach((preset) => {
    const item = document.createElement('div');
    item.className = 'list-group-item d-flex flex-column gap-2';
    const meta = document.createElement('div');
    meta.innerHTML = `<strong>${escapeHtml(preset.name)}</strong><div
class="text-muted small">${escapeHtml(preset.description || '')}</div>`;
    const actions = document.createElement('div');
    actions.className = 'd-flex gap-2 flex-wrap';
    const loadBtn = document.createElement('button');
    loadBtn.className = 'btn btn-primary btn-sm';
    loadBtn.textContent = 'Load (replace)';
    loadBtn.addEventListener('click', ()=>{
      if (!confirm('Load preset? Current changes will be replaced.')) return;
      data = adaptInput(preset.cards || []);
      renderGrid();
      playbookPresetsModal.hide();
      toast('Preset loaded','success');
    });
    const appendBtn = document.createElement('button');
    appendBtn.className = 'btn btn-outline-secondary btn-sm';
    appendBtn.textContent = 'Append';
    appendBtn.addEventListener('click', ()=>{
      const maxId = data.length ? Math.max(...data.map(d=>d.id||0)) : 0;
      const incoming = (preset.cards || []).map((c, idx)=> ({ ...c, id: maxId + idx +
1 }));
      data = data.concat(adaptInput(incoming));
      renderGrid();
      toast('Preset appended','success');
```

```
    });
    actions.appendChild(loadBtn);
    actions.appendChild(appendBtn);
    item.appendChild(meta);
    item.appendChild(actions);
    playbookPresetList.appendChild(item);
  });
});

function listCodeCards(){
  return data.filter((it) => ["code", "js", "html", "markdown"].includes(it.type));
}

function refreshDrawingTargets(){
  const select = document.getElementById("drawingTargetCard");
  if (!select) return;
  select.innerHTML = "";
  const items = listCodeCards();
  if (!items.length) {
    const opt = document.createElement("option");
    opt.value = "";
    opt.textContent = "No code cards yet";
    select.appendChild(opt);
    return;
  }
  items.forEach((it) => {
    const opt = document.createElement("option");
    opt.value = String(it.id);
    opt.textContent = `#${it.id} · ${it.title || it.type}`;
    select.appendChild(opt);
  });
}

function openDrawingStudio(){
  if (selectedId == null) return toast('Select a drawing card','warning');
  const it = data.find(d=>d.id===selectedId);
  if (!it || it.type !== "drawing") return toast('Select a drawing card','warning');
  if (!drawingStudioCanvas) return;
  drawingStudioModal.show();
  const assist = {
    smooth: document.getElementById("drawingAssistSmooth")?.checked ??
true,
    stabilize: document.getElementById("drawingAssistStabilize")?.checked ??
true,
    snap: document.getElementById("drawingAssistSnap")?.checked ?? false,
    snapSize:
Number(document.getElementById("drawingAssistSnapSize")?.value) || 12,
```

```
    stabilizeLevel:
Number(document.getElementById("drawingAssistLevel")?.value) || 0.35,
    mirrorX: document.getElementById("drawingMirrorX")?.checked ?? false,
    mirrorY: document.getElementById("drawingMirrorY")?.checked ?? false
  };
  DrawingBoard?.attachBoard?.(drawingStudioCanvas, it, { data, saveLocal,
renderGrid, assist });
  const brush = {
    color: document.getElementById("drawingBrushColor")?.value || "#e5e7eb",
    width: Number(document.getElementById("drawingBrushWidth")?.value) || 2,
    opacity: Number(document.getElementById("drawingBrushOpacity")?.value)
|| 1,
    mode: document.getElementById("drawingEraser")?.checked ? "erase" :
"draw"
  };
  DrawingBoard?.setBrush?.(drawingStudioCanvas, brush);
  refreshDrawingTargets();
}

openDrawingStudioBtn?.addEventListener("click", openDrawingStudio);

document.getElementById("drawingBrushColor")?.addEventListener("input",
(e) => {
  DrawingBoard?.setBrush?.(drawingStudioCanvas, { color: e.target.value });
});
document.getElementById("drawingBrushWidth")?.addEventListener("input",
(e) => {
  DrawingBoard?.setBrush?.(drawingStudioCanvas, { width:
Number(e.target.value) || 2 });
});
document.getElementById("drawingBrushOpacity")?.addEventListener("input",
(e) => {
  DrawingBoard?.setBrush?.(drawingStudioCanvas, { opacity:
Number(e.target.value) || 1 });
});
document.getElementById("drawingEraser")?.addEventListener("change", (e)
=> {
  DrawingBoard?.setBrush?.(drawingStudioCanvas, { mode: e.target.checked ?
"erase" : "draw" });
});
document.getElementById("drawingUndo")?.addEventListener("click", () =>
DrawingBoard?.undo?.(drawingStudioCanvas));
document.getElementById("drawingRedo")?.addEventListener("click", () =>
DrawingBoard?.redo?.(drawingStudioCanvas));
document.getElementById("drawingClear")?.addEventListener("click", () =>
DrawingBoard?.clear?.(drawingStudioCanvas));
```

```
document.getElementById("drawingAssistSnap")?.addEventListener("change", openDrawingStudio);
document.getElementById("drawingAssistSmooth")?.addEventListener("change", openDrawingStudio);
document.getElementById("drawingAssistStabilize")?.addEventListener("change", openDrawingStudio);
document.getElementById("drawingAssistSnapSize")?.addEventListener("input", openDrawingStudio);
document.getElementById("drawingAssistLevel")?.addEventListener("input", openDrawingStudio);
document.getElementById("drawingMirrorX")?.addEventListener("change", openDrawingStudio);
document.getElementById("drawingMirrorY")?.addEventListener("change", openDrawingStudio);

document.getElementById("drawingCreateCode")?.addEventListener("click", () => {
  const newId = data.length ? Math.max(...data.map(d=>d.id||0)) + 1 : 1;
  const it = {
    id: newId,
    field: 0,
    axis: 0,
    rowSpan: 1,
    colSpan: 2,
    type: "code",
    title: "Drawing Points",
    content: "const points = [];\n",
    color: "#111827",
    hoverColor: "#60a5fa",
    group: "Data",
    tags: ["points", "drawing"]
  };
  data.push(it);
  saveLocal();
  renderGrid();
  refreshDrawingTargets();
  toast("Code card created","success");
});

document.getElementById("drawingSendPoints")?.addEventListener("click", () => {
  const select = document.getElementById("drawingTargetCard");
  if (!select || !drawingStudioCanvas) return;
  const targetId = Number(select.value);
  const target = data.find((d) => d.id === targetId);
  if (!target) return toast("Select a code card","warning");
```

```
  const points = DrawingBoard?.exportPoints?.(drawingStudioCanvas) || [];
  const payload = `const points = ${JSON.stringify(points, null, 2)};\n`;
  if (["code", "js", "html", "markdown"].includes(target.type)) {
    target.content = payload + (target.content || "");
    saveLocal();
    renderGrid();
    toast("Points sent to code card","success");
  }
});

async function loadPlanchesList(){
  planchesList.innerHTML = '<div class="text-center text-muted
py-4">Loading...</div>';
  const token = localStorage.getItem('authToken') ||
localStorage.getItem('ssoToken');
  if (!token) return planchesList.innerHTML = '<div class="text-center text-
danger py-4">SSO Token required.</div>';

  try {
    const res = await fetch('/api/cms/planches', { headers: { 'Authorization':
`Bearer ${token}` } });
    if (!res.ok) throw new Error(await res.text());
    const json = await res.json();
    const list = json.planches || [];

    if (list.length === 0) {
      planchesList.innerHTML = '<div class="text-center text-muted py-4">No
planches found. Switch tab to save one.</div>';
      return;
    }

    planchesList.innerHTML = '';
    list.sort((a,b) => new Date(b.updatedAt) - new Date(a.updatedAt)).forEach(p
=> {
      const item = document.createElement('div');
      item.className = 'list-group-item list-group-item-action d-flex justify-
content-between align-items-center';
      const targets = Array.isArray(p.publishTargets) ? p.publishTargets :
['display'];
      item.innerHTML = `
        <div>
          <div class="fw-bold">${escapeHtml(p.name)} ${p.isPublic ? '<span
class="badge text-bg-success" style="font-size:0.6em">PUBLIC</span>' : ''}
</div>
          <small class="text-muted">${new
Date(p.updatedAt).toLocaleDateString()} · ${p.cards?.length||0} cells · $
{escapeHtml(targets.join(', '))}</small>
```

```
      </div>
      <div class="btn-group btn-group-sm">
        <button class="btn btn-outline-primary btn-load">Load</button>
        <button class="btn btn-outline-secondary btn-edit">Edit</button>
        <button class="btn btn-outline-danger btn-del">×</button>
      </div>
    `;

    item.querySelector('.btn-load').addEventListener('click', () => {
      if(confirm('Load planche "'+p.name+'"? Current changes will be lost.')){
        data = adaptInput(p.cards || []);
        saveLocal(); renderGrid(); toast('Loaded','success');
        planchesModal.hide();
      }
    });

    item.querySelector('.btn-edit').addEventListener('click', () => {
      // Pre-fill save form
      $('#plancheId').value = p.id;
      $('#plancheName').value = p.name;
      $('#planchePublic').checked = !!p.isPublic;
      writePlancheTargetsToForm(p.publishTargets || ['display']);
      // Switch tab
      const tab = new bootstrap.Tab(document.getElementById('p-save-tab'));
      tab.show();
    });

    item.querySelector('.btn-del').addEventListener('click', async () => {
      if(!confirm('Delete planche "'+p.name+'"?')) return;
      try {
        const res = await fetch('/api/cms/planches/'+p.id, {
          method: 'DELETE',
          headers: { 'Authorization': `Bearer ${token}` }
        });
        if(!res.ok) throw new Error('Failed');
        loadPlanchesList();
        toast('Deleted','info');
      } catch(e) { toast('Error deleting','danger'); }
    });

    planchesList.appendChild(item);
  });

 } catch(e) {
   planchesList.innerHTML = `<div class="text-center text-danger py-4">Error:
${escapeHtml(e.message)}</div>`;
 }
```

```
}

plancheSaveForm.addEventListener('submit', async (e)=>{
  e.preventDefault();
  const token = localStorage.getItem('authToken') ||
localStorage.getItem('ssoToken');
  if (!token) return toast('SSO Token required','warning');

  const id = $('#plancheId').value;
  const name = $('#plancheName').value.trim();
  const isPublic = $('#planchePublic').checked;
  const publishTargets = readPlancheTargetsFromForm();

  if (!name) return toast('Name required','warning');

  try {
    const payload = {
      id: id || undefined,
      name,
      isPublic,
      publishTargets,
      cards: data
    };

    const res = await fetch('/api/cms/planches', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json', 'Authorization': `Bearer $
{token}` },
      body: JSON.stringify(payload)
    });

    if(!res.ok) throw new Error(await res.text());
    toast('Planche saved','success');

    // Refresh list and switch tab
    loadPlanchesList();
    const tab = new bootstrap.Tab(document.getElementById('p-list-tab'));
    tab.show();

  } catch(e) {
    toast(`Save failed: ${e.message}`, 'danger');
  }
});

// Simple i18n label toggle kept minimal (EN/FR labels are static for brevity)

})();
```

```
</script>
<script>
  window.__MAGNETIC_STATE__ = (() => {
    try {
      return __MAGNETIC_STATE_PLACEHOLDER__;
    } catch (error) {
      return null;
    }
  })();
</script>
<script>
(function(){
  const $ = (sel, root = document) => root.querySelector(sel);
  const $$ = (sel, root = document) => Array.from(root.querySelectorAll(sel));
  const state = window.__MAGNETIC_STATE__ || { notes: [] };
  const secretList = document.getElementById('secretNotesList');
  const handshakeWidget = document.getElementById('handshakeStatus');

  function renderNotes(notes){
    if(!secretList) return;
    if(!notes || !notes.length){
      secretList.textContent = 'No notes available from the API.';
      return;
    }
    secretList.textContent = notes.map((note)=> {
      const lock = note.secret ? '🔐' : '📝';
      return `${lock} ${note.title || 'untitled'} — ${note.content || '[sealed]'}`;
    }).join('\n');
  }

  renderNotes(state.notes || []);

  async function fetchJson(url, options = {}){
    const res = await fetch(url, options);
    if(!res.ok){
      throw new Error(`Request failed: ${res.status}`);
    }
    return res.json();
  }

  async function refreshHandshake(){
    if(!handshakeWidget) return;
    handshakeWidget.textContent = 'Refreshing…';
    try{
      const payload = await fetchJson('/api/handshake');
      handshakeWidget.textContent = JSON.stringify(payload, null, 2);
    }catch(error){
```

```
    handshakeWidget.textContent = error.message;
   }
  }

  async function refreshNotes(includeSecrets){
   const secretInput = document.getElementById('secretPassphrase');
   const headers = { 'Content-Type': 'application/json' };
   if(includeSecrets && secretInput && secretInput.value){
    headers['x-magnetic-secret'] = secretInput.value;
   }
   const payload = await fetchJson(`/api/notes?includeSecrets=$
{includeSecrets}`, { headers });
   renderNotes(payload.notes);
  }

  const form = document.getElementById('noteForm');
  if(form){
   form.addEventListener('submit', async (event)=>{
    event.preventDefault();
    const formData = new FormData(form);
    const body = {
     title: formData.get('title'),
     content: formData.get('content'),
     tags: (formData.get('tags') ||
'').split(',').map((tag)=>tag.trim()).filter(Boolean),
     theme: formData.get('theme'),
     secret: formData.get('secret') === 'on'
    };
    try{
     await fetchJson('/api/notes', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(body)
     });
     form.reset();
     await refreshNotes(false);
    }catch(error){
     secretList.textContent = `Failed to save: ${error.message}`;
    }
   });
  }

  document.getElementById('refreshHandshake')?.addEventListener('click',
refreshHandshake);
  document.getElementById('fetchSecretNotes')?.addEventListener('click', ()=>
refreshNotes(true));
  document.getElementById('fetchPublicNotes')?.addEventListener('click', ()=>
```

```
refreshNotes(false));

  const codeInsert = $('#codeInsert');
  const codeNew = $('#codeNew');
  async function loadCodePresets(){
    const sel = $('#codePreset');
    if (!sel) return;
    try{
      const res = await fetch('/api/cms/code-presets');
      const json = await res.json();
      sel.innerHTML = '';
      (json.presets || []).forEach(p=>{
        const opt = document.createElement('option');
        opt.value = p.path;
        opt.textContent = `${p.name} (${p.ext})`;
        sel.appendChild(opt);
      });
    }catch(e){ console.warn(e); }
  }

  $('#codePresetLoad')?.addEventListener('click', async ()=>{
    const path = $('#codePreset').value;
    if (!path) return;
    try{
      const res = await fetch(`/api/cms/code-presets/file?path=$
{encodeURIComponent(path)}`);
      const json = await res.json();
      $('#codeMode').value = json.ext === 'html' ? 'html' : (json.ext === 'js' ? 'js' :
'code');
      $('#codeTitle').value = path.split('/').pop();
      $('#codeInput').value = json.html || json.raw || '';
    }catch(e){ console.warn(e); }
  });

  function codeValue(){
    if (window.__codeMirrorInstance) return
window.__codeMirrorInstance.getValue();
    return $('#codeInput').value;
  }

  function applyCodeToCell(makeNew=false){
    const mode = $('#codeMode').value;
    const title = $('#codeTitle').value.trim() || 'Code';
    const content = codeValue();
    if (!content) return toast('Code is empty','warning');
    if (makeNew || selectedId == null){
      const newId = data.length ? Math.max(...data.map(d=>d.id||0)) + 1 : 1;
```

```
      const type = (mode === 'code') ? 'code' : mode;
      data.push({ id:newId, field:0, axis:0, rowSpan:1, colSpan:1, type, title,
content, color:'#1e90ff', hoverColor:'#6ec6ff', group:'Code', tags:['code'] });
      selectedId = newId;
    } else {
      const it = data.find(d=>d.id===selectedId);
      if (!it) return;
      it.type = (mode === 'code') ? 'code' : mode;
      it.title = title;
      it.content = content;
    }
    saveLocal(); renderGrid(); selectCell(selectedId); openPanel();
  }
  codeInsert?.addEventListener('click', ()=> applyCodeToCell(false));
  codeNew?.addEventListener('click', ()=> applyCodeToCell(true));
  document.getElementById('codeVibe')?.addEventListener('click', ()=>{
    const vibe = `// Vibe coding preset\nconst vibe = {\n  field: 'exosphere',\n
tempo: 0.62,\n  chroma: 38,\n  hue: 210,\n  noise: { amp: 0.9, freq: 1.4 },\n
waves: { curl: 0.5, divergence: 0.35 }\n};\n\nfunction pulse(t){\n  return
Math.sin(t * vibe.tempo) * 0.5 + 0.5;\n}\n\nconsole.log('vibe', vibe,
pulse(performance.now() * 0.001));\n`;
    if (window.__codeMirrorInstance)
window.__codeMirrorInstance.setValue(vibe);
    else $('#codeInput').value = vibe;
    $('#codeMode').value = 'js';
    $('#codeTitle').value = 'Vibe preset';
  });

  async function seedPhenotype(mode){
    const model = document.getElementById('phenotypeModel')?.value ||
'neurobio';
    try{
      const res = await fetch('/api/phenotype/seed', { method:'POST', headers:
{ 'Content-Type':'application/json' }, body: JSON.stringify({ model }) });
      const json = await res.json();
      if (!json.ok) throw new Error(json.error || 'Seed failed');
      const el = document.getElementById('phenotypeWidget');
      if (el) {
        el.dataset.model = model;
        el.dataset.genome = JSON.stringify(json.genome || {});
        window.Phenotype?.scan?.();
      }
    }catch(e){ console.warn(e); }
  }

  async function mutatePhenotype(){
    const el = document.getElementById('phenotypeWidget');
```

```
    if (!el) return;
    let base = {};
    try { base = JSON.parse(el.dataset.genome || '{}'); } catch {}
    const model = document.getElementById('phenotypeModel')?.value ||
el.dataset.model || 'neurobio';
    try{
      const res = await fetch('/api/phenotype/mutate', { method:'POST', headers:
{ 'Content-Type':'application/json' }, body: JSON.stringify({ genome: base }) });
      const json = await res.json();
      if (!json.ok) throw new Error(json.error || 'Mutate failed');
      el.dataset.model = model;
      el.dataset.genome = JSON.stringify(json.genome || {});
      window.Phenotype?.scan?.();
    }catch(e){ console.warn(e); }
  }

  document.getElementById('phenotypeSeed')?.addEventListener('click', ()=>
seedPhenotype('seed'));
  document.getElementById('phenotypeMutate')?.addEventListener('click',
mutatePhenotype);

  function setupComposerChat(){
    const log = document.getElementById('composerChatLog');
    const input = document.getElementById('composerChatInput');
    const sendBtn = document.getElementById('composerChatSend');
    if (!log || !input || !sendBtn) return;
    const render = (entry)=>{
      const line = document.createElement('div');
      line.className = 'chat-line';
      line.textContent = `${entry.user || 'anon'}: ${entry.text || ''}`;
      log.appendChild(line);
      log.scrollTop = log.scrollHeight;
    };
    fetch('/api/cms/chat').then(r=>r.json()).then(j=>{
      log.innerHTML = '';
      (j.chat||[]).forEach(render);
    }).catch(()=>{});
    const wsUrl = `${location.protocol === 'https:' ? 'wss' : 'ws'}://$
{location.host}/ws`;
    const ws = new WebSocket(wsUrl);
    ws.addEventListener('message', (e)=>{
      try{
        const msg = JSON.parse(e.data);
        if (msg.type === 'chat') render(msg);
      }catch{}
    });
    sendBtn.addEventListener('click', ()=>{
```

```
    const text = input.value.trim();
    if (!text) return;
    ws.send(JSON.stringify({ type:'chat', text, user: (currentUser?.handle ||
'anon') }));
    input.value = '';
   });
  }

  window.addEventListener('load', ()=>{
   refreshHandshake();
   loadCurrentUser().then(loadMediaLibrary);
   loadCodePresets();
   setupComposerChat();
   setupTagAutocomplete();
   bindDrawingAssistControls();
   refreshAutocompleteLists();
   setupFiboMenu();
   if (window.CodeMirror){
    const textarea = document.getElementById('codeInput');
    if (textarea){
     window.__codeMirrorInstance = CodeMirror.fromTextArea(textarea, {
      lineNumbers: true,
      mode: 'javascript',
      theme: 'material-darker',
      viewportMargin: Infinity
     });
     document.getElementById('codeMode')?.addEventListener('change',
(e)=>{
      const mode = e.target.value === 'html' ? 'htmlmixed' : 'javascript';
      window.__codeMirrorInstance.setOption('mode', mode);
     });
    }
   }
  });

  function setupFiboMenu(){
   const menu = document.getElementById('fiboMenu');
   if (!menu) return;
   let dragEl = null;
   menu.querySelectorAll('.fibo-item').forEach((item)=>{
    item.addEventListener('dragstart', (e)=>{
     dragEl = item;
     item.classList.add('dragging');
     e.dataTransfer.effectAllowed = 'move';
    });
    item.addEventListener('dragend', ()=>{
     item.classList.remove('dragging');
```

```
      dragEl = null;
    });
    item.addEventListener('click', (e)=>{
      const action = item.dataset.action;
      if (action) {
        const target = document.querySelector(`[data-action="${action}"]`);
        if (target) target.click();
      } else if (item.id === 'fiboOpenDrawing') {
        document.getElementById('openDrawingStudio')?.click();
      }
    });
    item.addEventListener('dragover', (e)=>{
      e.preventDefault();
      if (!dragEl || dragEl === item) return;
      const rect = item.getBoundingClientRect();
      const next = (e.clientY - rect.top) / rect.height > 0.5;
      menu.insertBefore(dragEl, next ? item.nextSibling : item);
    });
  });
 }
})();
</script>
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.65.16/lib/
codemirror.js"></script>
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.65.16/mode/javascript/
javascript.js"></script>
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.65.16/mode/xml/
xml.js"></script>
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.65.16/mode/css/
css.js"></script>
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.65.16/mode/htmlmixed/
htmlmixed.js"></script>
<script src="/public/js/drawing-board.js"></script>
<script type="module" src="/public/app.js"></script>
<script type="module" src="/public/js/composer-app.js"></script>
<script type="module" src="/public/js/phenotype.js"></script>
</body>
</html>
```

with full implementation accross database type, user gated features in order to build storayboard and as a tool of the media intraction design paradigm lab dev envi