Articles & Guides Hub Restyle — Implementation Plan

> **For agentic workers:** REQUIRED SUB-SKILL: Use superpowers:subagent-driven-development (recommended) or superpowers:executing-plans to implement this plan task-by-task. Steps use checkbox (`- [ ]`) syntax for tracking.

Goal: Replace the headless blog landing page's topic-cluster grouping with a flat, client-side searchable / tag-filterable / paginated "Articles & Guides Hub" matching the Figma design.

Architecture: A pure logic module (`src/lib/hub.js`, vitest-tested) assembles the post list + tag vocabulary at build time. The Astro page (`src/pages/blog/index.astro`) renders every card server-side with `data-` attributes. A page-local static script (`public/blog-hub.js`) hides/shows existing cards for search, tag filter, and pagination — it never injects the sole copy of any link. Styles live under `.bkx-hub-*` in the globally-imported `src/styles/blog.css`.

Tech Stack: Astro 4 (static, `@astrojs/cloudflare`), Webflow Data API at build time, Vitest, vanilla DOM JS (no framework).

Global Constraints

- Repo-local only. **No Webflow publish, no CMS writes, no Cloudflare/prod cutover.**
- Page-local only — **no global symbols**; the client script is `public/blog-hub.js`, included solely by the blog index via `<script src="/blog-hub.js" is:inline>`.
- Do **not** modify `src/lib/webflow.js` — reuse `getPublicPosts()`, `getPostBySlug()`, `getTopics()` as-is.
- Keep the **black blog nav** (`bodyClass="theme-blog"`). Do not touch nav CSS.
- Pure logic goes in `src/lib/hub.js` with co-located `src/lib/hub.test.js` (mirror `src/lib/filter.js` / `filter.test.js` style).
- Date byline format: **`MM/DD/YYYY`** (UTC), identical to `src/lib/postView.js`.
- Giclée slug (verbatim): `what-is-a-giclee-print-and-why-theyre-worth-it`.
- Topic IDs (verbatim, priority order): Fine Art Scanning `6a2cb4d782ff4e49bee750e0`, Reproduction `6a2cb4d782ff4e49bee750e4`, Metis Art Scanning `6a2cb4d782ff4e49bee750e2`, Artwork Archiving `6a2cb4d782ff4e49bee750e6`.
- Topic tag-label override field slug: **`tag-label`** (future CMS field; fall back to topic name when absent/empty).
- Tag matching is **case-insensitive**; a chip is rendered only if ≥1 listed post carries it; a free-text tag equal to a topic label is the **same** chip (never duplicated).
- Run all commands from `apps/site/`. Test runner: `npx vitest run <file>`.

File Structure

- **Create** `apps/site/src/lib/hub.js` — pure build-time logic (tag derivation, vocabulary, Giclée gate, card view, sort, client-mirrored match predicate).
- **Create** `apps/site/src/lib/hub.test.js` — vitest unit tests for `hub.js`.
- **Create** `apps/site/public/blog-hub.js` — page-local client script (search / filter / pagination / mobile toggle).
- **Modify** `apps/site/src/pages/blog/index.astro` — replace cluster layout with the flat hub.

- **Modify** `apps/site/src/styles/blog.css` — add `.bkx-hub-*` styles; remove `.bkx-topic*` (keep `.bkx-cluster*`).

Task 1: Tag taxonomy + helpers in `hub.js`

Files:

- Create: `apps/site/src/lib/hub.js`
- Test: `apps/site/src/lib/hub.test.js`

Interfaces:

- Produces:
  - `normalize(v: string): string`
  - `stripHtml(html: string): string`
  - `splitTags(raw: string): string[]`
  - `tagLabelFor(topic: object|null): string|null`
  - `postTagsFor(post, topicsById: Map): {label: string, key: string}[]`
  - `buildVocabulary(posts, topicsById: Map): {label: string, key: string}[]`
  - `TOPIC_PRIORITY: string[]`, `GICLEE_SLUG: string`

- [ ] **Step 1: Write the failing test**

Create `apps/site/src/lib/hub.test.js`:

```
import { describe, it, expect } from 'vitest';
import {
  normalize, stripHtml, splitTags, tagLabelFor, postTagsFor,
buildVocabulary, TOPIC_PRIORITY,
} from './hub.js';

const T = {
  fineArt:  { id: TOPIC_PRIORITY[0], fieldData: { name: 'Fine Art Scanning'
} },
  repro:    { id: TOPIC_PRIORITY[1], fieldData: { name: 'Reproduction' } },
  metis:    { id: TOPIC_PRIORITY[2], fieldData: { name: 'Metis Art
Scanning', 'tag-label': 'Scanning' } },
  archiving:{ id: TOPIC_PRIORITY[3], fieldData: { name: 'Artwork Archiving'
} },
  printing: { id: 'futurePrintingId',  fieldData: { name: 'Printing' } },
};
const byId = new Map(Object.values(T).map((t) => [t.id, t]));
// Fine Art also overridden to Scanning so the two scanning topics
collapse.
T.fineArt.fieldData['tag-label'] = 'Scanning';

const post = (topicId, tag) => ({ fieldData: { topic: topicId, 'new-blog-
post-tag': tag } });

describe('hub tag helpers', () => {
```

```
  it('normalize trims + lowercases', () => {
    expect(normalize('  Scanning ')).toBe('scanning');
  });

  it('stripHtml removes tags and collapses whitespace', () => {
    expect(stripHtml('<p>Hello <b>there</b></p>')).toBe('Hello there');
  });

  it('splitTags splits on ; and , and trims, dropping empties', () => {
    expect(splitTags('scanning; photography,  ')).toEqual(['scanning',
'photography']);
    expect(splitTags('')).toEqual([]);
    expect(splitTags(undefined)).toEqual([]);
  });

  it('tagLabelFor prefers tag-label override, falls back to name', () => {
    expect(tagLabelFor(T.metis)).toBe('Scanning');
    expect(tagLabelFor(T.repro)).toBe('Reproduction');
    expect(tagLabelFor(null)).toBe(null);
  });

  it('postTagsFor merges topic tag + free-text, deduped case-
insensitively', () => {
    // topic Scanning + free-text "scanning; photography" -> Scanning,
Photography (no dup)
    const tags = postTagsFor(post(T.fineArt.id, 'scanning; photography'),
byId);
    expect(tags.map((t) => t.label)).toEqual(['Scanning', 'photography']);
    expect(tags.map((t) => t.key)).toEqual(['scanning', 'photography']);
  });

  it('buildVocabulary orders topic tags by priority, then extras
alphabetically, no dups', () => {
    const posts = [
      post(T.archiving.id, null),          // Artwork Archiving
      post(T.fineArt.id, 'photography'),   // Scanning + photography
      post(T.repro.id, null),              // Reproduction
      post(T.metis.id, null),              // Scanning (collapses)
      post(T.printing.id, null),           // unknown-priority topic
"Printing"
    ];
    const vocab = buildVocabulary(posts, byId);
    expect(vocab.map((c) => c.label)).toEqual([
      'Scanning', 'Reproduction', 'Artwork Archiving', 'Printing',
'photography',
    ]);
  });
});
```

- [ ] **Step 2: Run the test to verify it fails**

Run: `cd apps/site && npx vitest run src/lib/hub.test.js` Expected: FAIL — `Failed to resolve import "./hub.js"`.

- [ ] **Step 3: Write the implementation**

Create `apps/site/src/lib/hub.js`:

```
// Pure build-time logic for the /blog Articles & Guides hub. No I/O, no
DOM —
// mirrors the lib/filter.js pattern (vitest-tested). The client script
// public/blog-hub.js re-implements `cardMatches` (see Task 2) for the
browser.

export const GICLEE_SLUG = 'what-is-a-giclee-print-and-why-theyre-worth-
it';

// Topic IDs in display priority (pillar order). Verbatim from the prior
index.astro.
export const TOPIC_PRIORITY = [
  '6a2cb4d782ff4e49bee750e0', // Fine Art Scanning
  '6a2cb4d782ff4e49bee750e4', // Reproduction
  '6a2cb4d782ff4e49bee750e2', // Metis Art Scanning
  '6a2cb4d782ff4e49bee750e6', // Artwork Archiving
];

export function normalize(v) {
  return String(v ?? '').trim().toLowerCase();
}

export function stripHtml(html) {
  return String(html ?? '').replace(/<[^>]*>/g, ' ').replace(/\s+/g, '
').trim();
}

export function splitTags(raw) {
  return String(raw ?? '')
    .split(/[;,]/)
    .map((s) => s.trim())
    .filter(Boolean);
}

// Display label for a topic: explicit `tag-label` override, else the topic
name.
export function tagLabelFor(topic) {
  if (!topic) return null;
  const override = topic.fieldData?.['tag-label'];
  return (override && override.trim()) || topic.fieldData?.name || null;
}

// All tags on a post: its topic's tag label, plus free-text additional
tags.
// Deduped case-insensitively (first label of a given key wins).
export function postTagsFor(post, topicsById) {
  const f = post.fieldData ?? {};
  const out = [];
  const seen = new Set();
```

```
  const add = (label) => {
    if (!label) return;
    const key = normalize(label);
    if (!key || seen.has(key)) return;
    seen.add(key);
    out.push({ label, key });
  };
  add(tagLabelFor(topicsById.get(f.topic)));
  for (const t of splitTags(f['new-blog-post-tag'])) add(t);
  return out;
}

// Ordered chip list across all listed posts. Topic-derived tags first (by
topic
// priority; unknown-priority topics after known ones), then additional-
only tags
// alphabetically. A chip appears once, only if a listed post carries it.
export function buildVocabulary(posts, topicsById) {
  const KNOWN = TOPIC_PRIORITY.length;
  const UNKNOWN_TOPIC = 900; // after known topics, before additional-only
  const EXTRA = Infinity;    // additional-only tags last
  const chips = new Map();   // key -> { label, rank }

  const rankForTopicId = (id) => {
    const i = TOPIC_PRIORITY.indexOf(id);
    return i >= 0 ? i : (topicsById.has(id) ? UNKNOWN_TOPIC : EXTRA);
  };

  for (const post of posts) {
    const f = post.fieldData ?? {};
    const topicLabel = tagLabelFor(topicsById.get(f.topic));
    const topicKey = topicLabel ? normalize(topicLabel) : null;
    for (const { label, key } of postTagsFor(post, topicsById)) {
      const rank = key === topicKey ? rankForTopicId(f.topic) : EXTRA;
      const prev = chips.get(key);
      if (!prev) chips.set(key, { label, rank });
      else if (rank < prev.rank) chips.set(key, { label, rank });
    }
  }

  return [...chips.entries()]
    .map(([key, v]) => ({ key, label: v.label, rank: v.rank }))
    .sort((a, b) => (a.rank - b.rank) || a.label.localeCompare(b.label))
    .map(({ key, label }) => ({ key, label }));
}
```

- [ ] **Step 4: Run the test to verify it passes**

Run: `cd apps/site && npx vitest run src/lib/hub.test.js` Expected: PASS (5 tests).

- [ ] **Step 5: Commit**

```
cd apps/site && git add src/lib/hub.js src/lib/hub.test.js
git commit -m "feat(blog): tag taxonomy helpers for Articles & Guides hub"
```

Task 2: Post-list assembly, card view, match predicate in `hub.js`

Files:

- Modify: `apps/site/src/lib/hub.js`
- Test: `apps/site/src/lib/hub.test.js`

Interfaces:

- Consumes (Task 1): `normalize`, `stripHtml`, `postTagsFor`, `buildVocabulary`, `TOPIC_PRIORITY`, `GICLEE_SLUG`.
- Produces:
  - `formatPostedLabel(dateStr: string): string` — `MM/DD/YYYY` UTC, `''` when falsy.
  - `includeGiclee(post: object|null): boolean` — public gate for the pass-through.
  - `hubCardView(post, topicsById: Map): object` — `{ name, slug, href, imgUrl, imgAlt, author, postedLabel, descText, featured, tagKeys: string[], searchText }`.
  - `assembleHub({ publicPosts, gicleePost, topics }): { cards: object[], vocabulary: {label,key}[] }` — Giclée appended only if `includeGiclee`, sorted reverse-chron by `date-posted` (tiebreak `name`).
  - `cardMatches(card, { query: string, tagKeys: string[] }): boolean` — search AND (OR-within-tags). Mirrored by `public/blog-hub.js`.

- [ ] **Step 1: Write the failing test**

Append to `apps/site/src/lib/hub.test.js`:

```
import {
  formatPostedLabel, includeGiclee, hubCardView, assembleHub, cardMatches,
GICLEE_SLUG,
} from './hub.js';

const topicsArr = [
  { id: TOPIC_PRIORITY[0], fieldData: { name: 'Fine Art Scanning', 'tag-
label': 'Scanning' } },
  { id: TOPIC_PRIORITY[1], fieldData: { name: 'Reproduction' } },
];
const tById = new Map(topicsArr.map((t) => [t.id, t]));

const mkPost = (over = {}) => ({
  isDraft: false, isArchived: false,
  fieldData: {
    name: 'How to Scan a Large Painting', slug: 'scanning-large-paintings',
```

```
    topic: TOPIC_PRIORITY[0], 'date-posted': '2026-06-13T12:00:00.000Z',
    'post-author-name': 'Brooklyn Editions', 'main-image': { url:
'https://img/x.jpg' },
    'alt-text-for-image': 'A large painting on the scanner',
    'short-description': '<p>Artists working at <b>scale</b>.</p>',
featured: false,
    'new-blog-post-tag': null, ...over,
  },
  ...over._meta,
});

describe('hub assembly', () => {
  it('formatPostedLabel renders MM/DD/YYYY UTC', () => {
    expect(formatPostedLabel('2026-06-
13T12:00:00.000Z')).toBe('06/13/2026');
    expect(formatPostedLabel('')).toBe('');
  });

  it('includeGiclee gates on public flags + topic', () => {
    const base = { isDraft: false, isArchived: false,
      fieldData: { slug: GICLEE_SLUG, name: 'Giclée', 'date-posted': '2023-
06-21', topic: TOPIC_PRIORITY[1] } };
    expect(includeGiclee(base)).toBe(true);
    expect(includeGiclee({ ...base, isDraft: true })).toBe(false);
    expect(includeGiclee({ ...base, isArchived: true })).toBe(false);
    expect(includeGiclee({ ...base, fieldData: { ...base.fieldData, topic:
undefined } })).toBe(false);
    expect(includeGiclee(null)).toBe(false);
  });

  it('hubCardView maps fields + tags + plain-text description', () => {
    const v = hubCardView(mkPost({ 'new-blog-post-tag': 'photography' }),
tById);
    expect(v.name).toBe('How to Scan a Large Painting');
    expect(v.href).toBe('/blog/scanning-large-paintings');
    expect(v.author).toBe('Brooklyn Editions');
    expect(v.postedLabel).toBe('06/13/2026');
    expect(v.descText).toBe('Artists working at scale.');
    expect(v.tagKeys).toEqual(['scanning', 'photography']);
    expect(v.searchText).toContain('how to scan a large painting');
    expect(v.searchText).toContain('scale');
  });

  it('assembleHub appends Giclée only when gated-in and sorts reverse-
chron', () => {
    const a = mkPost({ name: 'Older', slug: 'a', 'date-posted': '2026-01-
01T00:00:00Z' });
    const b = mkPost({ name: 'Newer', slug: 'b', 'date-posted': '2026-06-
13T00:00:00Z' });
    const giclee = { isDraft: false, isArchived: false, fieldData: {
      slug: GICLEE_SLUG, name: 'Giclée', topic: TOPIC_PRIORITY[1],
      'date-posted': '2023-06-21T00:00:00Z', 'post-author-name': 'Brooklyn
Editions',
      'short-description': '<p>Prints.</p>' } };
```

```
    const { cards, vocabulary } = assembleHub({ publicPosts: [a, b],
gicleePost: giclee, topics: topicsArr });
    expect(cards.map((c) => c.slug)).toEqual(['b', 'a', GICLEE_SLUG]);
    expect(vocabulary.map((c) => c.label)).toEqual(['Scanning',
'Reproduction']);
  });

  it('cardMatches: search AND, tags OR', () => {
    const card = { searchText: 'how to scan a large painting scale',
tagKeys: ['scanning', 'photography'] };
    expect(cardMatches(card, { query: 'large', tagKeys: [] })).toBe(true);
    expect(cardMatches(card, { query: 'sculpture', tagKeys: []
})).toBe(false);
    expect(cardMatches(card, { query: '', tagKeys: ['photography']
})).toBe(true);
    expect(cardMatches(card, { query: '', tagKeys: ['printing']
})).toBe(false);
    expect(cardMatches(card, { query: 'large', tagKeys: ['printing']
})).toBe(false);
    expect(cardMatches(card, { query: '', tagKeys: [] })).toBe(true);
  });
});
```

- [ ] **Step 2: Run the test to verify it fails**

Run: `cd apps/site && npx vitest run src/lib/hub.test.js` Expected: FAIL — `formatPostedLabel is not a function` (and the other new exports).

- [ ] **Step 3: Write the implementation**

Append to `apps/site/src/lib/hub.js`:

```
// MM/DD/YYYY UTC — identical formatting to lib/postView.js.
export function formatPostedLabel(dateStr) {
  if (!dateStr) return '';
  const d = new Date(dateStr);
  if (Number.isNaN(d.getTime())) return '';
  return `${String(d.getUTCMonth() + 1).padStart(2,
'0')}/${String(d.getUTCDate()).padStart(2, '0')}/${d.getUTCFullYear()}`;
}

// Public gate for the Giclée pass-through. getPostBySlug() keeps drafts,
so we
// must re-check publish flags explicitly; a topic is required so the post
owns a chip.
export function includeGiclee(post) {
  if (!post || post.isDraft || post.isArchived) return false;
  const f = post.fieldData ?? {};
  return Boolean(f.slug && f.name && f['date-posted'] && f.topic);
}

export function hubCardView(post, topicsById) {
```

```
  const f = post.fieldData ?? {};
  const descText = stripHtml(f['short-description']);
  const tags = postTagsFor(post, topicsById);
  return {
    name: f.name,
    slug: f.slug,
    href: `/blog/${f.slug}`,
    imgUrl: f['main-image']?.url ?? null,
    imgAlt: f['alt-text-for-image'] || f.name || '',
    author: f['post-author-name'] || 'Brooklyn Editions',
    postedLabel: formatPostedLabel(f['date-posted']),
    descText,
    featured: Boolean(f.featured),
    tagKeys: tags.map((t) => t.key),
    searchText: normalize(`${f.name} ${descText}`),
  };
}

// Assemble the full listed set: public posts + the gated Giclée pass-
through,
// sorted newest-first. Vocabulary is built from the SAME listed posts.
export function assembleHub({ publicPosts, gicleePost, topics }) {
  const topicsById = new Map(topics.map((t) => [t.id, t]));
  const listed = [...publicPosts];
  if (includeGiclee(gicleePost)) listed.push(gicleePost);

  const byDate = (a, b) => {
    const da = a.fieldData?.['date-posted'] || '';
    const db = b.fieldData?.['date-posted'] || '';
    return db.localeCompare(da) ||
String(a.fieldData?.name).localeCompare(String(b.fieldData?.name));
  };
  const sorted = [...listed].sort(byDate);

  return {
    cards: sorted.map((p) => hubCardView(p, topicsById)),
    vocabulary: buildVocabulary(sorted, topicsById),
  };
}

// Mirrored by public/blog-hub.js. query is pre-normalized; tagKeys are
normalized.
export function cardMatches(card, { query, tagKeys }) {
  if (query && !card.searchText.includes(query)) return false;
  if (tagKeys.length && !card.tagKeys.some((k) => tagKeys.includes(k)))
return false;
  return true;
}
```

- [ ] **Step 4: Run the test to verify it passes**

Run: `cd apps/site && npx vitest run src/lib/hub.test.js` Expected: PASS (all tests from Tasks 1 + 2).

- [ ] **Step 5: Commit**

```
cd apps/site && git add src/lib/hub.js src/lib/hub.test.js
git commit -m "feat(blog): hub post assembly, card view, match predicate"
```

Task 3: Rebuild `blog/index.astro` as the flat hub

Files:

- Modify: `apps/site/src/pages/blog/index.astro` (full replacement of the body)

Interfaces:

- Consumes (Task 2): `assembleHub` from `../../lib/hub.js`; `getPublicPosts`, `getPostBySlug`, `getTopics`, `GICLEE_SLUG` (re-export not needed — import from hub).
- Produces (consumed by Task 5 `public/blog-hub.js`): DOM contract —
  - root `[data-hub]`; list `[data-hub-list]`; each card `article[data-card]` with `data-tags="key1|key2"` and `data-search="…"`; empty notice `.bkx-hub-empty`; search `#hub-search`; tag checkboxes inside `#hub-tags` (value = tag key); filter toggle `.bkx-hub-filter-toggle` with `aria-controls="hub-tags"`; pager container `[data-hub-pager]`.

- [ ] **Step 1: Replace the file**

Overwrite `apps/site/src/pages/blog/index.astro` with:

```
---
import Base from '../../layouts/Base.astro';
import { getPublicPosts, getPostBySlug, getTopics } from
'../../lib/webflow.js';
import { assembleHub, GICLEE_SLUG } from '../../lib/hub.js';

const [posts, topics, giclee] = await Promise.all([
  getPublicPosts(),
  getTopics(),
  getPostBySlug(GICLEE_SLUG).catch(() => null),
]);

const { cards, vocabulary } = assembleHub({ publicPosts: posts, gicleePost:
giclee, topics });

const INTRO = "Welcome to our Articles & Guides Hub. Here you will find all
articles & guides written by Brooklyn Editions on topics related to fine
art scanning and printing, to help you learn more about these processes and
make important choices regarding your work. Studio announcements will also
be posted here.";
---
<Base
```

```
  title="Articles & Guides | Brooklyn Editions"
  description="Guides on fine art scanning, reproduction, and archiving
from Brooklyn Editions."
  canonical="https://www.brooklyneditions.com/blog"
  ogType="website"
  bodyClass="theme-blog">
  <section class="section-standard has-blog-bg bkx-article">
    <div class="inner-container">
      <h1 class="blog-hero-size bkx-title bkx-hub-h1">Articles &amp;
Guides</h1>
      <div class="bkx-intro bkx-hub-intro"><p>{INTRO}</p></div>

      <div class="bkx-hub-head">
        <h2 class="bkx-hub-head-title">Articles &amp; Guides</h2>
        <div class="bkx-rule"></div>
      </div>

      <noscript><p class="bkx-hub-noscript">All articles are listed below.
</p></noscript>

      <div class="bkx-hub" data-hub>
        <div class="bkx-hub-list" data-hub-list>
          {cards.map((c) => (
            <article class="bkx-hub-card" data-card data-tags=
{c.tagKeys.join('|')} data-search={c.searchText}>
              {c.imgUrl && (
                <a class="bkx-hub-card-fig" href={c.href} tabindex="-1"
aria-hidden="true">
                  <img src={c.imgUrl} alt={c.imgAlt} width="300"
height="174" loading="lazy" />
                </a>
              )}
              <div class="bkx-hub-card-body">
                {c.featured && <span class="bkx-hub-flag">Top
Recommended</span>}
                <h3 class="bkx-hub-card-title"><a href={c.href}>{c.name}
</a></h3>
                <p class="bkx-hub-card-meta">Written by {c.author} on
{c.postedLabel}</p>
                {c.descText && <p class="bkx-hub-card-desc">{c.descText}
</p>}
                <a class="bkx-hub-card-cta" href={c.href}>&#8594; Click
Here to Read</a>
              </div>
            </article>
          ))}
          <p class="bkx-hub-empty" hidden role="status">No articles match
your search.</p>
        </div>

        <aside class="bkx-hub-side">
          <div class="bkx-hub-search">
            <label class="bkx-hub-side-label" for="hub-search">Search
Articles</label>
```

```
            <input id="hub-search" type="search" class="bkx-hub-search-
input" autocomplete="off"
              placeholder="Search using keywords / phrases ie. &ldquo;How
to scan a large painting&rdquo;" />
          </div>

          <div class="bkx-hub-filter" data-hub-filter>
            <button type="button" class="bkx-hub-filter-toggle bkx-hub-
side-label" aria-expanded="false" aria-controls="hub-tags">
              Filter Articles by Tags <span class="bkx-hub-caret" aria-
hidden="true"></span>
            </button>
            <fieldset class="bkx-hub-tags" id="hub-tags">
              <legend class="bkx-hub-vh">Filter Articles by Tags</legend>
              {vocabulary.map((t) => (
                <label class="bkx-hub-tag"><input type="checkbox" value=
{t.key} /> <span>{t.label}</span></label>
              ))}
            </fieldset>
          </div>
        </aside>
      </div>

      <nav class="bkx-hub-pager" data-hub-pager aria-label="Pagination">
</nav>
    </div>
  </section>

  <script src="/blog-hub.js" is:inline></script>
</Base>
```

- ☐ **Step 2: Build to verify it compiles + renders**

Run: `cd apps/site && npm run build` Expected: build succeeds; `dist/blog/index.html` exists and contains `data-hub` and multiple `data-card` elements.

Run: `grep -c 'data-card' dist/blog/index.html` Expected: a number ≥ 20 (all public posts).

- ☐ **Step 3: Commit**

```
cd apps/site && git add src/pages/blog/index.astro
git commit -m "feat(blog): flat Articles & Guides hub markup + data
attributes"
```

Task 4: Hub styles in `blog.css`

Files:

- Modify: `apps/site/src/styles/blog.css` (remove `.bkx-topic*` block at lines ~160-164; append `.bkx-hub-*` styles)

Interfaces:

- Consumes (Task 3): the `.bkx-hub*` class names and the `[data-hub]` root.
- Produces (Task 5): the `.is-js` hook on `[data-hub]` toggles two-column + control visibility; `.is-open` on `[data-hub-filter]` expands tags on mobile; `[hidden]` on cards/pager paginates.
- ☐ **Step 1: Remove the obsolete landing styles**

In `apps/site/src/styles/blog.css`, delete exactly these four lines (the `/blog index: posts grouped by topic` block — **keep** the `.bkx-cluster*` block that follows, it is used by `Cluster.astro`):

```
.bkx-topic { margin-top: 3.2em; }
.bkx-topic-head { font-size: 1.875em; font-weight: 500; margin: 0 0 0.2em;
padding-bottom: 0.5em; border-bottom: 2px solid var(--dark-1, #1d1d1b); }
.bkx-topic-intro { max-width: 760px; color: #4a4a44; margin: 0.8em 0 1.4em;
}
.bkx-topic-intro p { margin: 0; }
```

Also delete the now-orphaned comment line directly above them:

```
/* ---- /blog index: posts grouped by topic ---- */
```

- ☐ **Step 2: Append the hub styles**

Append to `apps/site/src/styles/blog.css`:

```
/* ===== /blog Articles & Guides hub ===== */
.bkx-hub-h1 { text-transform: uppercase; }
.bkx-hub-intro { max-width: none; }
.bkx-hub-vh { position: absolute; width: 1px; height: 1px; padding: 0;
margin: -1px; overflow: hidden; clip: rect(0 0 0 0); white-space: nowrap;
border: 0; }

.bkx-hub-head { margin: 2.4em 0 1.6em; }
.bkx-hub-head-title { font-size: 1.5em; font-weight: 500; margin: 0 0
0.6em; }
.bkx-hub-head .bkx-rule { margin: 0; }
.bkx-hub-noscript { color: #4a4a44; margin: 0 0 1.2em; }

/* layout: single column until JS upgrades it to list + sidebar */
.bkx-hub { display: block; }
.bkx-hub.is-js { display: grid; grid-template-columns: minmax(0, 1fr)
18.5em; gap: 3em; align-items: start; }
```

```
/* article list */
.bkx-hub-list { display: flex; flex-direction: column; gap: 1.5em; }
.bkx-hub-card { display: grid; grid-template-columns: 300px minmax(0, 1fr);
gap: 1.5em;
  border: 1px solid #c9c9bd; border-radius: 4px; padding: 1.25em;
background: #f2f2eb; align-items: start; }
.bkx-hub-card[hidden] { display: none; }
.bkx-hub-card-fig { display: block; }
.bkx-hub-card-fig img { width: 300px; height: 174px; object-fit: cover;
border-radius: 3px; display: block; }
.bkx-hub-card-body { min-width: 0; }
.bkx-hub-flag { display: inline-block; background: #6b6b63; color: #f2f2eb;
font-size: 0.7em; font-weight: 500;
  text-transform: uppercase; letter-spacing: 0.03em; padding: 0.3em 0.7em;
border-radius: 3px; margin-bottom: 0.7em; }
.bkx-hub-card-title { font-size: 1.3em; font-weight: 700; line-height:
1.15; margin: 0 0 0.4em; }
.bkx-hub-card-title a { color: var(--dark-2, #1d1d1b); text-decoration:
none; }
.bkx-hub-card-title a:hover { text-decoration: underline; }
.bkx-hub-card-meta { color: #6b6b63; font-size: 0.85em; margin: 0 0 0.6em;
}
.bkx-hub-card-desc { color: #4a4a44; font-size: 0.95em; line-height: 1.5;
margin: 0 0 0.9em; }
.bkx-hub-card-cta { color: var(--dark-2, #1d1d1b); font-weight: 700; text-
decoration: underline; text-underline-offset: 2px; }
.bkx-hub-empty { color: #4a4a44; padding: 1.5em 0; }

/* sidebar (hidden until JS) */
.bkx-hub-side { display: none; }
.bkx-hub.is-js .bkx-hub-side { display: block; }
.bkx-hub-side-label { display: block; font-weight: 700; font-size: 0.95em;
margin: 0 0 0.6em; color: var(--dark-2, #1d1d1b); }
.bkx-hub-search { margin-bottom: 1.6em; }
.bkx-hub-search-input { width: 100%; box-sizing: border-box; padding: 0.7em
0.85em; border: 1px solid #b6b6a8;
  border-radius: 4px; background: #fff; font: inherit; font-size: 0.9em;
color: var(--dark-2, #1d1d1b); }
.bkx-hub-search-input::placeholder { color: #8a8a7e; font-style: italic; }
.bkx-hub-filter-toggle { background: none; border: none; padding: 0; width:
100%; text-align: left; cursor: pointer;
  display: flex; align-items: center; justify-content: space-between; font:
inherit; }
.bkx-hub-caret { display: none; } /* caret only used on mobile */
.bkx-hub-tags { border: 0; margin: 0; padding: 0; display: flex; flex-
direction: column; gap: 0.55em; }
.bkx-hub-tag { display: flex; align-items: center; gap: 0.6em; font-size:
0.92em; color: var(--dark-2, #1d1d1b); cursor: pointer; }
.bkx-hub-tag input { width: 0.95em; height: 0.95em; accent-color: #6b6b63;
}

/* pagination (hidden until JS builds it) */
.bkx-hub-pager { display: none; gap: 0.4em; align-items: center; justify-
```

```
content: center; margin: 2.4em 0 0.5em; flex-wrap: wrap; }
.bkx-hub-pager.is-active { display: flex; }
.bkx-hub-pager-label { color: #4a4a44; font-size: 0.9em; margin-right:
0.4em; }
.bkx-hub-pager button { font: inherit; font-size: 0.9em; min-width: 2em;
padding: 0.35em 0.6em; border: 1px solid #c9c9bd;
  background: #f2f2eb; color: var(--dark-2, #1d1d1b); border-radius: 3px;
cursor: pointer; }
.bkx-hub-pager button[aria-current="page"] { background: #1b1a18; color:
#f2f2eb; border-color: #1b1a18; }
.bkx-hub-pager button[disabled] { opacity: 0.4; cursor: default; }
.bkx-hub-pager .bkx-hub-pager-gap { color: #6b6b63; padding: 0 0.2em; }

/* focus states */
.bkx-hub-search-input:focus-visible, .bkx-hub-filter-toggle:focus-visible,
.bkx-hub-tag input:focus-visible, .bkx-hub-pager button:focus-visible,
.bkx-hub-card-title a:focus-visible, .bkx-hub-card-cta:focus-visible {
  outline: 2px solid #1b1a18; outline-offset: 2px;
}

@media (max-width: 800px) {
  .bkx-hub.is-js { grid-template-columns: 1fr; gap: 2em; }
  .bkx-hub-side { order: -1; } /* search/filter above the list on mobile */
  .bkx-hub-card { grid-template-columns: 1fr; gap: 0; }
  .bkx-hub-card-fig { display: none; } /* text-only cards on mobile */
  .bkx-hub-caret { display: inline-block; border: solid currentColor;
border-width: 0 2px 2px 0; padding: 0.18em; transform: rotate(45deg);
transition: transform 0.15s; }
  .bkx-hub-filter:not(.is-open) .bkx-hub-tags { display: none; }
  .bkx-hub-filter.is-open .bkx-hub-caret { transform: rotate(-135deg); }
}
```

- [ ] **Step 3: Build + confirm no orphaned references**

Run: `cd apps/site && npm run build && rg "bkx-topic" src/ ; echo "rg-exit:$?"`
Expected: build succeeds; `rg` prints nothing and `rg-exit:1` (no matches).

- [ ] **Step 4: Commit**

```
cd apps/site && git add src/styles/blog.css
git commit -m "feat(blog): hub styles; remove obsolete .bkx-topic landing
styles"
```

Task 5: Client script `public/blog-hub.js`

Files:

- Create: `apps/site/public/blog-hub.js`

Interfaces:

- Consumes (Task 3 DOM contract + Task 4 classes): `[data-hub]`, `[data-card]` (`data-tags`, `data-search`), `#hub-search`, `#hub-tags` checkboxes, `.bkx-hub-filter-toggle`, `[data-hub-filter]`, `[data-hub-pager]`, `.bkx-hub-empty`. Toggles `.is-js`, `.is-open`, `.is-active`, and `[hidden]`. Match logic mirrors `cardMatches` from `hub.js`.

- [ ] **Step 1: Write the script**

Create `apps/site/public/blog-hub.js`:

```
/* Page-local: drives the /blog Articles & Guides hub (search, tag filter,
   pagination, mobile filter toggle). Loaded only by blog/index.astro.
Mirrors
   the cardMatches predicate from src/lib/hub.js. No-op on any other page.
*/
(function () {
  var PER_PAGE = 8;
  var root = document.querySelector('[data-hub]');
  if (!root) return;
  root.classList.add('is-js');

  var listEl = root.querySelector('[data-hub-list]');
  var emptyEl = root.querySelector('.bkx-hub-empty');
  var searchEl = root.querySelector('#hub-search');
  var pager = document.querySelector('[data-hub-pager]');
  var filterBox = root.querySelector('[data-hub-filter]');
  var toggle = root.querySelector('.bkx-hub-filter-toggle');
  var checks = Array.prototype.slice.call(root.querySelectorAll('#hub-tags
input[type=checkbox]'));

  var cards = Array.prototype.slice.call(root.querySelectorAll('[data-
card]')).map(function (el) {
    return {
      el: el,
      search: el.getAttribute('data-search') || '',
      tags: (el.getAttribute('data-tags') ||
'').split('|').filter(Boolean),
    };
  });

  var query = '';
  var activeTags = [];
  var page = 1;

  function matches(card) {
    if (query && card.search.indexOf(query) === -1) return false;
    if (activeTags.length) {
      var hit = card.tags.some(function (t) { return activeTags.indexOf(t)
!== -1; });
      if (!hit) return false;
    }
    return true;
  }
```

```
  function pageNumbers(total, current) {
    // 1 … (current-1) current (current+1) … total, deduped + ordered
    var set = {};
    [1, total, current, current - 1, current + 1].forEach(function (n) {
      if (n >= 1 && n <= total) set[n] = true;
    });
    var nums = Object.keys(set).map(Number).sort(function (a, b) { return a
- b; });
    var out = [];
    for (var i = 0; i < nums.length; i++) {
      if (i > 0 && nums[i] - nums[i - 1] > 1) out.push('gap');
      out.push(nums[i]);
    }
    return out;
  }

  function renderPager(total) {
    pager.innerHTML = '';
    if (total <= 1) { pager.classList.remove('is-active'); return; }
    pager.classList.add('is-active');

    var label = document.createElement('span');
    label.className = 'bkx-hub-pager-label';
    label.textContent = 'Page ' + page;
    pager.appendChild(label);

    function btn(text, target, opts) {
      opts = opts || {};
      var b = document.createElement('button');
      b.type = 'button';
      b.innerHTML = text;
      if (opts.current) b.setAttribute('aria-current', 'page');
      if (opts.disabled) b.disabled = true;
      if (opts.label) b.setAttribute('aria-label', opts.label);
      if (!opts.disabled) b.addEventListener('click', function () {
go(target); });
      return b;
    }

    pager.appendChild(btn('&#8249;', page - 1, { disabled: page === 1,
label: 'Previous page' }));
    pageNumbers(total, page).forEach(function (n) {
      if (n === 'gap') {
        var g = document.createElement('span');
        g.className = 'bkx-hub-pager-gap';
        g.textContent = '…';
        pager.appendChild(g);
      } else {
        pager.appendChild(btn(String(n), n, { current: n === page, label:
'Page ' + n }));
      }
    });
    pager.appendChild(btn('&#8250;', page + 1, { disabled: page === total,
```

```
  label: 'Next page' }));
    }

    function go(target) {
      page = target;
      apply();
      if (listEl && listEl.scrollIntoView) listEl.scrollIntoView({ behavior:
'smooth', block: 'start' });
    }

    function apply() {
      var visible = cards.filter(matches);
      var total = Math.max(1, Math.ceil(visible.length / PER_PAGE));
      if (page > total) page = total;
      var start = (page - 1) * PER_PAGE;
      var end = start + PER_PAGE;

      var shown = 0;
      cards.forEach(function (c) { c.el.hidden = true; });
      visible.forEach(function (c, i) {
        if (i >= start && i < end) { c.el.hidden = false; shown++; }
      });
      if (emptyEl) emptyEl.hidden = visible.length !== 0;
      renderPager(total);
    }

    var t;
    if (searchEl) {
      searchEl.addEventListener('input', function () {
        clearTimeout(t);
        t = setTimeout(function () {
          query = searchEl.value.trim().toLowerCase();
          page = 1;
          apply();
        }, 150);
      });
    }
    checks.forEach(function (cb) {
      cb.addEventListener('change', function () {
        activeTags = checks.filter(function (c) { return c.checked;
}).map(function (c) { return c.value; });
        page = 1;
        apply();
      });
    });
    if (toggle && filterBox) {
      toggle.addEventListener('click', function () {
        var open = filterBox.classList.toggle('is-open');
        toggle.setAttribute('aria-expanded', open ? 'true' : 'false');
      });
    }
```

```
  apply();
})();
```

- [ ] **Step 2: Build + confirm the script ships**

Run: `cd apps/site && npm run build && test -f dist/blog-hub.js && echo "script-present"` Expected: build succeeds; prints `script-present`.

- [ ] **Step 3: Commit**

```
cd apps/site && git add public/blog-hub.js
git commit -m "feat(blog): client-side search, tag filter, pagination for hub"
```

Task 6: Full verification + manual QA

Files: none (verification only)

- [ ] **Step 1: Run the suite + build**

Run: `cd apps/site && npm run test && npm run build` Expected: all vitest tests pass; build succeeds.

- [ ] **Step 2: No orphaned classes**

Run: `cd apps/site && rg "bkx-topic" src/ ; echo "exit:$?"` Expected: no output, `exit:1`.

- [ ] **Step 3: Manual QA in dev preview**

Run: `cd apps/site && npm run dev` then open `http://localhost:4321/blog`.

Verify against the Figma frames (`2022:188` desktop, `2022:126` / `2022:48` mobile):

- Desktop: H1 uppercase; sage intro box; "Articles & Guides" header + rule; two columns (cards left, Search + Filter sidebar right); cards show image (300×174) + Top-Recommended only when featured + title + "Written by … on MM/DD/YYYY" + description + "→ Click Here to Read".
- Search box filters live; checking a tag filters (OR across tags, AND with search); pagination appears when > 8 results, current page highlighted, prev/next + ellipsis work, filtering resets to page 1.
- Mobile (narrow the window < 800px): single column; cards drop the image; search + collapsible "Filter Articles by Tags ▾" sit above the list; toggle expands/collapses the checkbox list.
- Disable JS (or view `view-source`): every article link present in HTML; controls hidden; `<noscript>` note visible.

- [ ] **Step 4: Final commit (if any QA tweaks were needed)**

```
cd apps/site && git add -A
git commit -m "fix(blog): hub QA adjustments" || echo "no changes"
```

Self-Review

Spec coverage:

- Flat hub IA replacing clusters → Task 3. ✓
- Pixel-faithful card (image/flag/title/byline/desc/CTA) → Tasks 3 + 4. ✓
- Tag taxonomy (topic label + override `tag-label`, free-text extras, dedupe, ordering, chip only if matched) → Tasks 1 + 2. ✓
- Giclée gated pass-through → Task 2 (`includeGiclee`) + Task 3 (fetch). ✓
- Client search + tag filter (OR/AND) + pagination, page-local, no-JS fallback, controls hidden until JS, accessibility → Tasks 4 + 5. ✓
- Pagination only hides/shows; all links in HTML → Task 3 markup + Task 5. ✓
- Remove `.bkx-topic*`, keep `.bkx-cluster*` → Task 4. ✓
- No CMS writes / nav unchanged / `rg bkx-topic` + legacy-route caveat → Global Constraints + Task 6. ✓

Placeholder scan: none — every code step is complete.

Type consistency: `assembleHub` returns `{ cards, vocabulary }`; cards carry `tagKeys`/`searchText`/`href`; the page emits `data-tags(tagKeys.join('|'))`/`data-search`; the client splits `data-tags` on `|` and lowercases the query — consistent with `cardMatches`. `tag-label` slug, topic IDs, and `GICLEE_SLUG` match across tasks.

Notes for the executor

- The Giclée pass-through link `/blog/what-is-a-giclee-print-and-why-theyre-worth-it` 404s in local dev (no headless page) — **expected**; the production Worker routes it. Do not "fix" it. Confirm the live/staging route before relying on the Printing chip (per spec).
- The "Printing" chip and the merged "Scanning" label only appear once Maya does the CMS work (add a Printing topic; set `tag-label` overrides). With today's CMS the chips render as raw topic names + "photography" — that is correct behavior, not a bug.