

Migrating from WordPress to a managed Astro site, the operator's guide

This guide is written from the operator's seat, the person who actually carries out the migration, debugs the launch-day failures at 9pm, and answers the phone the morning after. It is not a sales document. It does not promise a frictionless move. It is honest about which parts of WordPress translate cleanly and which parts do not.

A GUIDE FOR UK OPERATORS

Author

Jordan Gilbert, Founder · UK Web Marketing

Edition

v1.0 · June 2026

Audience

UK SMEs sitting on an ageing WordPress build, evaluating a move to a managed static stack.

TABLE OF CONTENTS

1. Why Astro, and the four trade-offs you accept
 2. The seven-stage migration plan
 3. Stage 1, Content audit and export
 4. Stage 2, Redirect map
 5. Stage 3, SEO carry-over
 6. Stage 4, Rebuild on Astro
 7. Stage 5, Hosting cut-over
 8. Stage 6, Launch day, what breaks and what doesn't
 9. Stage 7, The 30-day post-launch sweep
 10. The awkward bits, honestly
 11. Sources + further reading
-

01 Why Astro, and the four trade-offs you accept

The case for moving from WordPress to a managed Astro site is straightforward:

- **Static output by default.** Pages are built to plain HTML files at deploy time. There is no PHP processing each request, no database to compromise, no plugin layer to update.
- **Sub-100ms TTFB on a CDN.** Astro builds to flat files served from edge POPs. Performance is structural, not optimisation theatre.
- **Drastically smaller attack surface.** No `wp-admin` to brute-force. No plugin auto-updates that break at 3am. No PHP-version end-of-life cycles. Sentry tells me the median UK WordPress install has 23 plugins; the median Astro deploy has zero.
- **A predictable cost curve.** Hosting on Vercel (Hobby/Pro), Cloudflare Pages (free), or a Hetzner VPS (£5/mo) replaces the typical UK WordPress hosting bill (£20-£60/mo) and the recurring patch-and-pray cost.

You accept four trade-offs:

1. **Content editing is different.** WordPress's WYSIWYG editor goes. In its place: a headless CMS (Sanity, Tina, Decap, Strapi) or MDX files in the repo. The CMS option keeps the visual editor; the MDX option is git-friendly but technical.
2. **Plugins disappear.** Every plugin needs an equivalent or a build-time replacement. Most have one. A handful, membership areas, complex e-commerce, deeply forum-style, are awkward (see Section 10).
3. **Live preview is staged.** WordPress shows the next page draft instantly. Astro builds the site; a draft preview requires a Vercel/Cloudflare preview deploy (90 seconds typically). Fast for most teams, glacial for editors used to instant publishing.
4. **You need a developer for structural changes.** Adding a new content type, rewiring the navigation, building a custom landing page, these are now git commits, not admin-screen toggles. For a small UK SME, this is usually delegated to the same retainer that hosts and maintains the site.

If those four trade-offs are deal-breakers, do not migrate. If they are acceptable, read on.

02 The seven-stage migration plan

STAGE	OUTPUT	TYPICAL HOURS (SME, < 200 PAGES)
1. Content audit + export	inventory CSV, exported XML, exported media	4–8
2. Redirect map	<code>redirects.json</code> , validated 1:1 against the audit	2–4
3. SEO carry-over	titles, descriptions, OG, schema, sitemap, robots	3–6
4. Rebuild on Astro	new Astro project, content imported, design rebuilt	20–60 (depends on scope)
5. Hosting cut-over	DNS, SSL, CDN, email, monitoring	2–4
6. Launch day	go-live, smoke tests, redirect verification	1–3 + standby
7. 30-day post-launch sweep	crawl errors, broken links, search-console health	4–8

Total: roughly 35–90 hours for a typical small UK business site. The variance is mostly in stage 4 (rebuild).

03 Stage 1, Content audit and export

The first job is to know what you have. WordPress sites accrue cruft, orphaned pages, draft posts from 2018, plugin-generated CPTs you forgot existed.

3.1, Crawl the live site

Use **Screaming Frog SEO Spider** (free up to 500 URLs; paid licence for larger sites) to crawl every page on the live domain. Export the URL list to CSV. This is your starting inventory.

3.2, Cross-check against WordPress

Log into `/wp-admin` . Export Posts → All Content. The XML export covers posts, pages, custom post types, comments, tags, categories, and media references.

3.3, Reconcile

The crawl finds what Google can see. The WP export finds what the database holds. Reconcile in a spreadsheet:

URL	STATUS	IN WP DB?	LINKED FROM?	KEEP?	NEW URL
/about	200	yes	menu	keep	/about
/services/old-thing	200	yes	none (orphan)	drop	410
/?p=1234	200	yes	sitemap	keep	/blog/post-slug

The "Keep?" column is where 20–40% of WordPress sites lose pages they had forgotten about. The "New URL" column drives stage 2.

3.4, Export the media

`/wp-content/uploads/` is the media root. SFTP it down, or use a plugin like **WP All Export** to bundle it. Note: most WordPress sites have 5–20x more media than they reference (cropped variants, retina copies, plugin-generated thumbnails). You only need the originals plus actively-referenced variants.

3.5, Export the data layer

For non-content data, form submissions, user accounts, e-commerce orders, separate exports. The WordPress XML covers content, not data. Use the relevant plugin (WPForms → CSV export; WooCommerce → Tools → Export). Decide *during the audit* whether each data class moves to the new system or is retained in WordPress for archival.

04 Stage 2, Redirect map

A redirect map is the single most important deliverable in a migration. Get it wrong and you lose rankings. Get it right and Google preserves the value of every link.

4.1, One row per old URL

The map has one row for every URL on the old site. Three columns:

- Old URL (without the domain, start with `/`).
- New URL (or `410 Gone` if you are dropping the page deliberately).
- Status (301 permanent for content moves; 410 for retired content; 302 only for short-term experiments).

4.2, Slug changes are common, that's fine

The new IA usually reorganises content. Old: `/?p=1234` . New: `/blog/dspt-checklist-for-clinics` . The 301 covers it.

4.3, Validate the map before launch

Run the map through a script that checks every "new URL" actually exists on the rebuild. Empty 301 chains (a redirect to a 404) are worse than no redirect.

4.4, Implement at the edge, not in WordPress

The redirects live in Vercel's `vercel.json` , Cloudflare's `_redirects` , or Netlify's `_redirects` , at the edge, evaluated before the request reaches the application. WordPress plugins like "Redirection" are fine for the old site, but the new site implements at the edge for speed.

```
{
  "redirects": [
    { "source": "/?p=1234", "destination": "/blog/dspt-checklist-for-clinics", "permanent": true },
    { "source": "/services/old-thing", "destination": "/410", "permanent": true }
  ]
}
```

05 Stage 3, SEO carry-over

This is where migrations succeed or fail in Google's eyes. The mechanical work is small. The judgement work is everything.

5.1, Page titles and meta descriptions

Export from the old site (Yoast / Rank Math both export to CSV). Carry over verbatim unless the content has substantively changed. Do not "improve" titles during a migration, change one variable at a time.

5.2, Schema markup

WordPress sites typically have Yoast / Rank Math autogenerating Article, Organization, LocalBusiness, BreadcrumbList. Re-emit each in the Astro build. Validate via `search.google.com/test/rich-results` before launch.

5.3, Open Graph and Twitter cards

OG image, OG title, OG description per page. Twitter card type (`summary_large_image` is the default for SME blogs). The OG image is a common breakage on migration, old sites reference `/wp-content/uploads/2020/03/og-default.jpg` which 404s after launch. Audit and remap.

5.4, Sitemap

The new sitemap is auto-generated by Astro's `@astrojs/sitemap` integration. Validate the XML at launch and submit to Google Search Console immediately.

5.5, robots.txt

The new `robots.txt` lives in `public/robots.txt`. Carry over the disallow rules from the old site. Add the new sitemap URL. Remove WordPress-specific disallows (`/wp-admin/`) since they no longer apply.

5.6, Canonicals

Every page should declare a canonical pointing at itself. Pagination, tag pages, and search-results pages should `noindex, follow`. Astro doesn't do this automatically, write a small helper to emit the right tag per page type.

5.7, Internal links

Internal links in posts are typically full URLs (`https://yourdomain.com/about`), these survive a migration. But links to `/?p=1234` -style permalinks break, these need updating in content. Find/replace during the import.

06 Stage 4, Rebuild on Astro

The rebuild itself. This is where most of the budget goes, and where the value of choosing Astro shows up.

6.1, Project skeleton

```
npm create astro@latest
```

For a UK SME site, the typical setup is:

- `@astrojs/mdx`, MDX content with components in markdown.
- `@astrojs/sitemap`, auto-generated XML sitemap.
- `@astrojs/tailwind` or hand-rolled CSS, the design system.
- Content Collections (`src/content/`), typed content schemas.
- A small `src/lib/` for site-config, redirects, schema emitters.

6.2, Content import

For MDX-as-source: convert each WordPress post to a markdown file with frontmatter. Tools like `wordpress-export-to-markdown` (Node) handle the bulk. Inspect 10% by hand to catch encoding errors, broken shortcodes, and embedded HTML.

For headless-CMS-as-source: import via the CMS's content API. Sanity has a `sanity-import` CLI. Tina syncs to git. Decap reads markdown directly.

6.3, Design rebuild

The temptation is to rebuild the WordPress theme pixel-for-pixel. Resist it. A migration is the right moment to:

- Update the visual design to current conventions.
- Fix accessibility (contrast, focus rings, alt text, heading hierarchy).
- Move to a system font or self-hosted webfont (Google Fonts is a Schrems II issue, see the *Healthcare website checklist* guide).
- Drop the dark/light-mode toggle if it's never used; keep it if your audience uses it.

6.4, Forms

WordPress forms (Contact Form 7, WPForms, Gravity Forms) do not translate. Replacements:

- **Plain HTML form to a serverless function** (cheap, owned by you).
- **Formspree, Basin, Web3Forms**, third-party endpoints.
- **A headless form vendor** with EU-sovereign data residency (see the *Healthcare website checklist* for named alternatives).

Whichever, the form data has a new destination. Update the privacy notice.

6.5, Analytics

Migrate the analytics in parallel. Move from Google Analytics 4 (cookie-banner liability) to a cookieless EU-hosted option (Plausible, Fathom) at the same time as the rebuild. The pageview counts will look slightly different, different sampling, different bot-filtering, but trend over time is what matters.

6.6, Search

WordPress's built-in search is database-driven. Astro is static, you need a different search.

- **Pagefind**, static-site search; runs entirely in the browser; works offline; no server.
- **Algolia**, hosted search; faster, more features; £/mo.
- **A meilisearch instance**, self-hosted; free; small VPS.

For most UK SME sites, Pagefind is the right answer.

07 Stage 5, Hosting cut-over

Cut-over is two days of work compressed into 20 minutes of DNS propagation.

7.1, Pre-flight (24 hours before cut-over)

- The new site is built and deployed to a staging URL (e.g. `staging.yourdomain.com` or `yourdomain.vercel.app`).
- Every page renders. Every link resolves. Every form submits. Run a full Screaming Frog crawl of the staging URL, zero 4xx, zero 5xx, zero unexpected redirects.
- SSL is valid on the staging URL.
- All redirects are in place at the edge.
- Schema validates.

7.2, Lower the TTL on the live DNS

48 hours before cut-over, lower the TTL on the live A / AAAA records to 300 seconds. This means when you change the records, propagation is fast.

7.3, The cut-over itself

In one sitting:

1. Point the apex A record (or CNAME, depending on host) to the new infrastructure.
2. Update the `www` CNAME similarly.
3. Wait for propagation (the TTL you set in 7.2).
4. Confirm the new site serves at the apex domain.
5. Confirm SSL is valid (some hosts auto-issue Let's Encrypt; others require manual).
6. Run the redirect smoke test, 10 old URLs hit, all 301 to the right new URLs.

7.4, Email, leave alone, deliberately

Do not touch MX records during the website cut-over. Email runs on its own infrastructure (the email host) and should be migrated in a separate, calmer cycle. Conflating the two is the most common cause of migration disasters.

7.5, Monitoring

Set uptime monitoring (UptimeRobot, Better Stack, Pingdom) before the cut-over. Set Search Console and Bing Webmaster Tools to monitor the new site.

08 Stage 6, Launch day, what breaks and what doesn't

The honest list, from a decade of migrations.

Usually breaks (be ready)

- **The favicon.** Browsers cache aggressively. Force a hard refresh.
- **One or two redirects.** The audit missed a trailing-slash variant or a deep CPT URL. Patch on the day.
- **An embedded YouTube / Vimeo video** that referenced a width parameter only the old theme set. Quick CSS fix.
- **A form notification email** going to the old admin account. Update the destination.

Rarely breaks

- **Search rankings.** A clean redirect map + verbatim title/description carry-over preserves rankings. Expect a 10–20% temporary dip in week 1 as Google re-crawls; recovered by week 4.
- **The sitemap submission.** Modern sitemaps are forgiving.
- **The SSL certificate.** Vercel / Cloudflare Pages issue automatically.
- **The CDN.** Cache invalidation is automatic on deploy.

What you must do on launch day

- **Force a re-crawl** via Search Console (Submit Sitemap → URL Inspection → Request Indexing for the homepage).
- **Run the redirect smoke test** within 15 minutes of go-live.
- **Watch error logs** for the first hour. Sentry / Better Stack will flag 4xx and 5xx rates.
- **Be available by phone for 24 hours.** Migrations rarely fall over, but when they do, the SLA is "the operator answers the phone".

09 Stage 7, The 30-day post-launch sweep

The work after launch, usually folded into the first month of the managed-infrastructure subscription.

- **Week 1, daily.** Check Search Console for crawl errors. Patch any redirect misses. Watch the form-submission stream, confirm every form is reaching its destination.
 - **Week 2, alternate days.** Audit page-speed. The new build should hit a 90+ Lighthouse score on every page. If it doesn't, fix.
 - **Week 3, twice weekly.** Watch the Search Console "Coverage" report. Confirm every page is indexed. Submit URLs for any that aren't.
 - **Week 4, end-of-month review.** Compare analytics pre/post, pageview parity $\pm 10\%$ is healthy. Below that, investigate (often a tracking-snippet issue rather than a traffic issue).
-

10 The awkward bits, honestly

Some WordPress features do not translate cleanly. Be honest about them in the quote.

10.1, Plugins that don't translate

- **Membership / paywall plugins** (Restrict Content Pro, MemberPress), the auth model is fundamentally different. Static sites don't have logged-in state by default. The fix is either (a) keep the membership area on WordPress as a subdomain (`members.yourdomain.com`), or (b) rebuild it as a separate Auth.js + serverless app. Neither is a 2-hour job.
- **WooCommerce**, possible, but a project in itself. For most SME sites with simple e-commerce, **Shopify Buy Buttons embedded in the Astro site** is the right answer. For larger catalogs, headless Shopify or headless Commerce.js.
- **BuddyPress / forums**, static sites can't host forums. Migrate to **Discourse, Circle, or Discord** depending on the community.
- **LMS plugins** (LearnDash, LifterLMS), the same auth-model problem as membership. Migrate to **Teachable, Thinkific, or a Kajabi tenancy** unless the LMS is core to the business model.

10.2, Comment threads

WordPress comments do not export cleanly. Three options:

- **Disqus or Commento**, third-party hosted; embed in Astro pages. Easy.
- **Static comments via giscus**, GitHub-discussions-backed; nice for technical blogs; requires a GitHub account to comment.
- **Drop the comments**. For most SME blogs, the comments are 90% spam. Audit; if value is low, drop them with intent.

10.3, Membership / login areas

Repeated from 10.1 because it's the most-asked question. Static sites are not the right architecture for logged-in user experiences. If a substantial fraction of your traffic is logged-in users, either keep WordPress for that section or commit to a rebuild as a separate app. The right architecture is rarely "Astro for everything".

10.4, Custom post types with heavy filtering

A WordPress CPT with a filterable archive (e.g. "events by city by date") is doable in Astro, but it is a custom-built filter, not a plugin. Budget for it.

10.5, Highly visual page builders

If the WordPress site is built on Elementor, Divi, Beaver Builder, or WPBakery, the content export is technically possible but visually unfaithful, the page-builder shortcodes do not translate. The rebuild is genuinely a rebuild, page by page. This is the largest variance in stage-4 hours.

11 Sources + further reading

- Astro docs · `docs.astro.build`
- Astro Content Collections · `docs.astro.build/en/guides/content-collections/`

- Google Search Central, Site moves · developers.google.com/search/docs/crawling-indexing/site-move-with-url-changes
 - Schema.org, Article, Organization, LocalBusiness · schema.org
 - WordPress XML export reference · wordpress.org/support/article/tools-export-screen/
 - Screaming Frog SEO Spider · screamingfrog.co.uk/seo-spider/
 - Pagefind · pagefind.app
 - ICO, Cookie guidance (relevant if migrating analytics) · ico.org.uk/for-organisations/direct-marketing-and-privacy-and-electronic-communications/guide-to-pecr/cookies-and-similar-technologies/
-

A note on the long-form version

This guide is the operator's view of a single migration. The long-form (planned late 2026) will include three worked examples, a healthcare clinic with light WooCommerce, a professional-services firm with a heavy Elementor build, and a community membership site that migrated half to Astro and kept half on WordPress. If you'd like to be notified when it ships, message me.