

WordPress Migration Checklist

Moving a website to WordPress is mostly planning. The content import is the easy part. What protects your search traffic, your editors and your launch date is everything around it: knowing what you have, deciding where every URL goes, rebuilding the features people rely on and testing before anyone flips DNS.

This is the checklist AlphaWeb works through on platform migrations, from Schoolwires and Drupal to ExpressionEngine, ASP.NET and older WordPress sites. Use it to plan your own move, to check a vendor's proposal or as the agenda for a kickoff call with us.

How to use this checklist

1. Work the phases in order. Each one feeds the next, and skipping the inventory is the most common reason a migration goes sideways.
2. Give every item an owner. Some belong to your team (decisions, content cleanup, approvals), some to the developer.
3. Keep the checklist next to the redirect map and the content inventory. Those two spreadsheets are the real deliverables of the planning phases.

1 Inventory and content audit

You cannot migrate, redirect or rebuild what you do not know about. Start with a complete picture of the current site.

- ☐ Crawl the live site and export every URL, page title and status code
- ☐ Pull URLs from the XML sitemaps, Search Console, analytics landing pages and server logs, since no single source is complete
- ☐ List every content type, template, taxonomy, custom field and menu on the source platform
- ☐ Count pages, posts, documents, images and users so you have numbers to check against after the move
- ☐ Mark each item keep, merge, rewrite or retire, and name an owner for anything that needs a decision
- ☐ Record every integration: forms, CRM, email lists, calendars, search, analytics, single sign on and third party embeds
- ☐ Note who edits what today, so roles and permissions can be planned instead of guessed

2 URL mapping and 301 redirects

A redirect map is really a content map. Every old address needs a decision before launch, not after the 404 reports start.

- ☐ Map every old URL to its new URL in one spreadsheet (old URL, new URL, type, status)
- ☐ Keep good URLs exactly as they are where the new structure allows it
- ☐ Point merged pages to the page that absorbs them and retired pages to the closest relevant page, not the home page
- ☐ Use permanent 301 redirects with a single hop, no chains
- ☐ Cover predictable URL families with pattern rules and handle the rest with an explicit list
- ☐ Include old file and document paths, not just pages
- ☐ Test the full redirect list automatically on staging before launch

3 SEO preservation

Rankings depend on more than URLs. Carry the signals search engines already trust into the new site.

- ☐ Migrate title tags and meta descriptions, and write them where the old site had none
- ☐ Keep heading structure and body content intact for pages that rank today
- ☐ Carry over canonical URLs, plus hreflang for multilingual sites
- ☐ Add structured data such as Organization, BreadcrumbList, Article and FAQPage where it fits
- ☐ Carry Open Graph and social images so shared links still look right
- ☐ Keep image alt text and descriptive file names where practical
- ☐ Benchmark top pages (traffic, rankings, clicks) before launch so you have something to compare against
- ☐ Block staging from search engines and plan the moment production is opened

4 Media, documents and users

Files and accounts are where migrations quietly lose data. Treat them as content, with their own counts and checks.

- ☐ Copy images, PDFs and office documents into the WordPress media library with alt text and captions
- ☐ Rewrite every file reference in content so nothing still points at the old server
- ☐ Match the old platform's image sizes to WordPress image sizes
- ☐ Move user accounts and map old roles to WordPress roles with the least access each person needs
- ☐ Choose a password plan: legacy password support on first login, single sign on, or a reset email at launch
- ☐ Remove accounts nobody uses anymore before they move
- ☐ Compare file and user counts against the inventory

5 Forms and integrations

Forms are often the reason the website exists. Rebuild them on purpose and test every notification.

- ☐ List every form, its fields, who receives submissions and where the data goes
- ☐ Rebuild forms in Gravity Forms or a similar tool with accessible labels and spam protection
- ☐ Reconnect CRM, email marketing and payment integrations, and test them end to end
- ☐ Decide whether old form entries need to be exported and kept
- ☐ Confirm notification emails come from an authenticated sending domain so they do not land in spam
- ☐ Reconnect analytics and tag manager, and record the date of the switch

6 Plugin and feature parity

Do not copy modules and plugins one for one. Look at what each feature does for your team, then rebuild it the WordPress way.

- ☐ List every feature people use today: search, calendars, directories, member areas, news, alerts
- ☐ For each one, choose a well maintained plugin or custom code, and write down why
- ☐ Convert page builder and layout content into editable Gutenberg blocks, not frozen HTML
- ☐ Plan editor roles and workflows (drafts, reviews, scheduled posts) around how your team actually works
- ☐ Vet every plugin for maintenance history, security record and performance cost before it goes on the site
- ☐ Retire features nobody uses instead of rebuilding them

7 Staging QA at five screen sizes

Review real migrated content, not samples, on a staging site that matches production.

- ☐ Run the migration into staging until page, file and user counts match the inventory
- ☐ Check every template at 393, 768, 1024, 1280 and 1440 pixels wide
- ☐ Look for horizontal scrolling, overlapping text, broken grids and images that fail to load
- ☐ Test the header, menus and footer at every width, including keyboard use of the mega menu
- ☐ Click through internal links and fix any that still point at old URLs
- ☐ Have your editors review their own sections and sign off before launch
- ☐ Submit every form on staging and confirm each notification arrives

8 Accessibility

A migration is the cheapest moment to fix accessibility. Build Section 508 and WCAG 2.1 AA into the plan instead of bolting them on later.

- ☐ Run automated WCAG 2.1 AA checks (we use axe) on every template at mobile and desktop widths
- ☐ Check color contrast on text, buttons and links against their backgrounds
- ☐ Make sure every image has useful alt text, or empty alt text when it is purely decorative
- ☐ Confirm form fields have labels and error messages that screen readers announce
- ☐ Tab through each template: visible focus, logical order, a working skip link
- ☐ Use a logical heading order and real lists, tables and buttons in the markup
- ☐ Fix what the checks find before launch, not on a later wish list

9 Performance

A new platform should be faster than the old one. Measure it before launch, while there is still time to fix it.

- ☐ Serve modern image formats such as WebP, sized for how they are displayed
- ☐ Lazy load images below the fold and give every image width and height to prevent layout shift
- ☐ Self host web fonts and preload only the ones the first screen needs
- ☐ Remove scripts and styles the new site does not use
- ☐ Check Core Web Vitals (LCP, CLS, INP) on key templates and fix the worst first
- ☐ Set up page caching and confirm logged out visitors get cached pages

10 Launch day

Launch should be the boring part. Everything risky was rehearsed on staging.

- ☐ Announce a short content freeze, then run the final incremental migration
- ☐ Take a full backup of the old site and the new site
- ☐ Lower DNS TTL ahead of time, then switch DNS or hosting
- ☐ Confirm HTTPS, the redirect rules and the canonical domain are live
- ☐ Remove staging noindex settings and check robots.txt
- ☐ Submit the new XML sitemaps in Search Console
- ☐ Smoke test top pages, forms, search, logins and any checkout or payment flow
- ☐ Keep the old site reachable privately for a few weeks in case something needs to be checked

11 Post-launch monitoring

The weeks after launch are when missed redirects and small bugs show up. Watch closely and fix fast.

- ☐ Check 404 logs daily at first and add redirects for anything that was missed
- ☐ Watch Search Console coverage, crawl errors and top pages against your benchmark
- ☐ Expect some short term movement while search engines recrawl, and investigate a sustained drop on one page
- ☐ Confirm form submissions, notifications and integrations are still flowing
- ☐ Set up uptime monitoring, backups and a schedule for core, plugin and theme updates
- ☐ Train editors on their real content and give them short written guides
- ☐ Schedule a review a few weeks out to close remaining issues

Frequently asked questions

How long does a WordPress migration take?

It depends on size and complexity. A simple site can move in weeks, while large platforms with many sites take longer. Repeatable, incremental migration scripts keep the final content freeze short, and a written plan should include the timeline.

Will we lose our search rankings when we move to WordPress?

Not if the move is planned. Map every old URL, cover each change with a single hop 301 redirect, and carry over titles, meta descriptions, canonical URLs and structured data. With that in place rankings usually hold, though short term movement is normal while search engines recrawl.

What if our current platform has no export?

Then the published website becomes the source. Custom scripts can capture pages, menus and structure from the live site and rehost documents and images in WordPress. That is how the Newark Public Schools sites were rescued from Schoolwires. If an export or API exists, use it first and fill the gaps with crawling.

Can our users keep their passwords?

Usually, yes. Platforms store passwords differently, so the options are legacy password support on first login, single sign on, or a password reset email at launch. For public sites with few logged in users, the reset email is often simplest.

Do we need a content freeze?

A short one. Content is migrated to staging early and refreshed as often as needed, so the freeze only covers the final incremental run before launch. Editors can usually keep publishing on the old site until a day or two before cutover.

Should we change our domain at the same time?

Only if you have to. Combining a domain change with a platform change adds risk, so if both are required, plan the redirects and Search Console steps for each one carefully.

Can we use this checklist on our own?

Yes. It is free to use, print and share with your team. If you would rather have AlphaWeb run it for you, request a migration quote and we will come back with a written plan.

Want AlphaWeb to run this checklist for you?

We are veteran owned WordPress specialists. Tell us what platform you are on and when you need to launch, and we will come back with a migration plan and a free quote.

Request a quote: alphaweb.com/custom-development-quote

Contact us: alphaweb.com/contact

Notes