The Definitive WordPress Handover Protocol

> A Comprehensive Due Diligence Questionnaire

Introduction

The handover of a WordPress project is a critical inflection point in the lifecycle of a digital asset. It is frequently misconstrued as a simple, transactional event involving the exchange of files & passwords. This perspective is dangerously incomplete. A successful handover is not an event but a comprehensive process of knowledge transfer, moving far beyond the delivery of technical components to encompass the full strategic, legal, & operational context of the website. The ultimate objective is to de-risk the transition by eliminating dependency on the departing developer, thereby empowering the new owner with complete & unencumbered autonomy over their digital property.

The stakes of a poorly executed handover are exceptionally high. An incomplete transfer can precipitate a cascade of negative consequences that manifest long after the original developer has departed. These risks include:

- **Severe security vulnerabilities** stemming from lingering, unauthorized access to critical systems.
- **Operational downtime** caused by unknown dependencies or undocumented, fragile workflows.
- **Significant legal liabilities** arising from ambiguous ownership of intellectual property or non-compliance with data privacy regulations.
- **Escalating future costs** as the new team grapples with undocumented **technical debt** that is difficult & expensive to remediate.

This guide provides a structured, exhaustive set of questions designed to serve as a definitive due diligence protocol for any WordPress project handover. It is engineered to be a practical tool, enabling the incoming owner or team to proactively identify, assess, & mitigate the myriad risks inherent in the transition process. By systematically addressing each area—from high-level strategy to the most granular technical detail—the new owner can ensure a truly complete & seamless transfer of control, establishing a firm foundation for the future management, maintenance, & growth of the website.

Part I: Strategic & Commercial Foundation

Before any technical examination of the website can be meaningful, it is imperative to establish a deep understanding of its business context. The technical architecture, features, & code are merely instruments designed to serve a strategic purpose. Their quality & appropriateness can only be judged against the business goals they were intended to achieve. This section provides the questions necessary to uncover the **why** behind the website, clarifying its purpose, history, stakeholders, & the legal framework that governs it.

A. Project Purpose & History

- □ **What was the original business problem this website was built to solve? What were the primary goals & objectives defined at the project's inception?**
- □ **Who was the target audience defined at the project's outset, & has that audience evolved over time?**
- □ **Can you provide a brief history of the project, including key milestones, major feature additions, or significant strategic pivots?**
- □ **What were the key performance indicators (KPIs) used to measure the website's success?**

B. Stakeholder & Audience Landscape

- □ **Who are the key internal & external stakeholders involved with the website? Who has the final authority for approving changes?**
- □ **Which stakeholders were involved in the original 'Discovery' or research phase of the project?**

C. Commercial & Legal Framework

- □ **Can you confirm in writing that upon final payment, our organization will have 100% ownership of the website, its design, all custom-developed code, & all content created specifically for the project?**
- □ **Are there any third-party or premium software licenses (for themes, plugins) that need to be transferred or repurchased? Please provide a comprehensive list of all such software, including license keys, original purchase dates, renewal dates, & annual costs.**
- □ **Are there any ongoing service level agreements (SLAs), hosting contracts, or other contractual obligations with your company or external third parties that we will be inheriting?**
- □ **What data privacy agreements or policies are in place to comply with regulations such as GDPR or CCPA? Where is user data stored, & what measures are in place to protect it?**

Part II: The Keys to the Kingdom: Access, Credentials, & Third-Party Services

This section constitutes a non-negotiable inventory of all access points required to control the website & its surrounding ecosystem. The primary goal is to achieve full, exclusive administrative control over every component. A handover cannot be considered complete until every credential has been securely transferred & the previous developer's access has been systematically & verifiably revoked from every platform. Failure to do so represents a critical & ongoing security risk.

A. Core Platform Credentials

- □ **Please provide credentials for a new, unique Administrator-level user for the WordPress dashboard.**
- □ **What are the login details for the website's hosting control panel (e.g., cPanel, Plesk, or a custom provider dashboard)?**
- □ **What are the login details for the domain registrar (e.g., GoDaddy, Namecheap) where the site's domain name is managed?**

B. Infrastructure & Development Access

- □ **Please provide SFTP (Secure File Transfer Protocol) or SSH (Secure Shell) credentials for direct file access to the server.**

- ☐ **What are the credentials for direct database access, including the Database Name, Username, Password, & Hostname?**

C. Integrated Services & APIs

- ☐ **Please provide a comprehensive list of all third-party services integrated with the site. This must include, but is not limited to: Payment Gateways (e.g., Stripe, PayPal), Content Delivery Networks (CDNs) like Cloudflare, Email Marketing Platforms (e.g., Mailchimp, ConvertKit), CRM systems, Analytics platforms (e.g., Google Analytics, Google Tag Manager), & any other services connected via an API key.**
- ☐ **For each service identified, what are the login credentials or API keys, & what is the process for transferring ownership of these accounts to us?**

D. The Secure Transfer Process

ACTION ITEMS:

- ☐ **Insist on Secure Credential Sharing.** Under no circumstances should passwords or API keys be sent in plain text via email. Use a secure sharing method, such as a one-time, self-destructing note service (e.g., Privnote) or a shared vault within a reputable password manager (e.g., 1Password, Bitwarden).
- ☐ **Immediate Credential Rotation & 2FA Activation.** Upon receiving all credentials, the new owner's first action should be to log in to every single service, change the password to a new, strong, & unique one, & enable two-factor authentication (2FA) wherever possible.
- ☐ **Systematic Revocation of Old Access.** The final & most crucial step of the access transfer is to systematically remove the previous developer's user accounts from every platform: WordPress, hosting, domain registrar, CDN, & all other third-party services. The handover is not complete until this de-provisioning process is finished.

Master Credentials & Services Inventory

Service/Platform	Purpose	Login URL	Username / Account Email	Password Transfer Method	API Key (if applicable)	License/Subsc Cost
WordPress Admin	Content Management	https://yourdomain.com/wp-admin	`new_admin_user`	Shared via Bitwarden	--	--
GoDaddy	Domain Registrar	https://dcc.godaddy.com/	owner@yourcompany.com	To be reset by new owner	--	$19.99/year
SiteGround Hosting	Web Hosting	https://login.siteground.com/	owner@yourcompany.com	To be reset by new owner	--	$299/year
Stripe	Payment Processing	https://dashboard.stripe.com/	finance@yourcompany.com	Existing account	`pk_live_...`	Transaction fees
Cloudflare	CDN / Firewall	https://dash.cloudflare.com/	tech@yourcompany.com	To be reset by new owner	`[Global API Key]`	$20/month
Mailchimp	Email Marketing	https://login.mailchimp.com/	marketing@yourcompany.com	Existing account	`[API Key]`	$50/month
Advanced Custom Fields PRO	Plugin License	https://www.advancedcustomfields.com/my-account/	developer@oldcompany.com	Transfer license	`[License Key]`	$49/year

Part III: Deconstructing the Technical Architecture

Understanding how a website was built is a prerequisite for being able to effectively maintain, troubleshoot, & extend it. This section focuses on creating a comprehensive **blueprint** of the site's technical construction. The questions are designed to uncover the development practices, tools, & structural decisions that define the website's functionality and, by extension, its future maintainability.

A. The Code Repository & Version Control

- ☐ **Is the website's code managed in a version control system like Git? If so, please provide access to the repository (e.g., on GitHub, Bitbucket, or GitLab).**
- ☐ **What is the branching strategy used in the repository (e.g., GitFlow, feature branches)? Which branch represents the code currently running on the live production site?**
- ☐ **Can you confirm that the repository contains the complete & unabridged history of the project's development?**

B. Theme & Parent/Child Theme Structure

- ☐ **What theme is the site built on? Is it a pre-built commercial theme from a marketplace, or is it a fully custom theme built from scratch?**
- ☐ **If it's a commercial theme, is a child theme being used for all customizations? Can you provide explicit confirmation that no modifications have been made directly to the parent theme's files?**
- ☐ **Where are the theme's original purchase files & official documentation located?**

C. Plugin Ecosystem Audit

- □ **Please provide a complete list of all installed plugins. For each plugin, can you briefly explain its purpose & the rationale for why it was chosen over alternatives?**
- □ **Which of these plugins are premium (paid) versions? Please provide the associated license keys & account information required to receive future updates & technical support.**
- □ **Are there any custom-built plugins developed specifically for this site? If so, where is their source code managed, & is there documentation for their functionality?**
- □ **Are there any known conflicts between plugins in the current stack, or are there any plugins known to cause performance issues?**