

AffStackDev Suite Complete Customer Reference

AffStackDev | Release 1.0.2 | Documentation revision 2026-10-13.5

Edition date: October 12, 2026.

Audience: Store owners, administrators and agency staff.

About this reference

AffStackDev Suite brings compatible AffStackDev products into one WordPress administration workspace. You can organize your own menus and categories, arrange dashboard cards, inspect product connections and use the shared Commerce Core administration tools. The Suite needs no license key of its own: it unlocks while one of your licensed AffStackDev plugins is active on the same site.

This reference covers every current Suite capability and configuration for store owners, administrators and agency staff. Use the contents and feature index to find the relevant setup, controls, result and recovery steps. The companion Master Technical Reference gives developers the exact registration, storage and integration contracts.

What's new in 1.0.2

Release 1.0.2 follows the launch release 1.0.1.6a. One correction: with WordPress debugging switched on, the Suite made WordPress log "translations were loaded too early" for the other AffStackDev plugins on every request; it no longer does. Version numbers have three parts from this release on. Everything else in this section came with 1.0.1.6a.

Release 1.0.1.6a was the AffStackDev launch release of the Suite; WordPress showed it as version 1.0.1.6.1. It contains every change since 1.0.1.5.

No Suite key needed. The Suite's licensed features — the enhanced interface, faster page navigation, and layout and dashboard customization — unlock while any AffStackDev plugin on the same site holds an active license: AffiliPartnerCS, B2B, Dynamic Pricing, Store Credit & Gift Cards, Free Gifts & BOGO, Support Desk, Theme Manager, Image Manager, Gate Manager, Unified Email Manager or AffStackDev AI. Install and activate one of your plugins with its own key; the Suite unlocks automatically. The header badge, the Suite license card and **AffStackDev → License** then read **Unlocked by your licensed AffStackDev plugin** with the product codes found on this site. With no licensed plugin they read **No Suite key needed** and ask you to install and activate one. A Suite key from an earlier purchase keeps working, but you do not need one.

Get the Suite and its updates from your account. The Suite is included with your AffStackDev licenses. **My Account → Downloads** on affstackdev.com lists the current Suite as included with your licenses and needing no key of its own. Without a Suite key, WordPress is not offered Suite updates on your site: when a new Suite release is announced, download it from My Account and upload it under **Plugins → Add New → Upload Plugin**, choosing to replace the installed version. Your Suite settings, layouts and preferences are kept. Keep the Suite current, because it carries the shared Commerce Core that your other AffStackDev plugins use.

Your licenses. Each AffStackDev license covers one live (Production) site and one test copy of it. An installation set to Staging, Development or Local uses the license's staging place. AffStackDev's current plans do not require a Suite license.

Contents

- [What's new in 1.0.2](#)
- [Feature index](#)
- [Install and open your Suite workspace](#)
- [Find and open tools](#)

- [Customize your toolbar, categories and WordPress menu](#)
- [Use and arrange the dashboard](#)
- [Suite Settings and PHP mode](#)
- [Search Discover and interpret product status](#)
- [Integration boundaries for site administrators](#)
- [Enhanced navigation and saved preferences](#)
- [How preference saves are authorized](#)
- [Download a support report](#)
- [Common problems and recovery](#)
- [How the Suite unlocks and the optional Suite key](#)
- [Commerce Core in Suite Settings](#)
- [Read and refresh plugin connections](#)
- [Retire a B2B audience policy](#)
- [Recover from a connection or retirement problem](#)
- [What Commerce Core does](#)
- [Read shared records and technical details](#)
- [Download reports and individual order records](#)
- [Customer eligibility and pricing integrations](#)
- [Shared cart lines and protected merchandise](#)
- [Accepted order history and publication recovery](#)
- [Diagnostic reports and integration interfaces](#)
- [Retained commerce records and safe updates](#)
- [Choosing a supported feature provider](#)
- [Shared configuration, pauses and ownership](#)
- [Fresh evaluation and short-lived permission](#)
- [One business operation, one retained identity](#)
- [Reconciling and recovering unfinished work](#)
- [Retiring a business policy safely](#)
- [Discovering checkout credit participation](#)
- [Shared operation troubleshooting](#)
- [Find the complete Suite references](#)
- [Glossary](#)

Feature index

Capability or reference	Section
Accepted order history and publication recovery	<u>Open reference</u>

Capability or reference	Section
Activate a new installation	Open reference
AffiliPartnerCS	Open reference
B2B	Open reference
Checks, retries, and routine browsing	Open reference
Choosing a supported feature provider	Open reference
Commerce Bridge	Open reference
Commerce Core in Suite Settings	Open reference
Commercial facts without price changes	Open reference
Common problems and recovery	Open reference
Customer eligibility and pricing integrations	Open reference
Customize your toolbar, categories and WordPress menu	Open reference
Diagnostic reports and integration interfaces	Open reference
Discovering checkout credit participation	Open reference
Download a support report	Open reference
Download reports and individual order records	Open reference
Draft, publishing and committed	Open reference
Dynamic Pricing	Open reference
Dynamic Pricing + Store Credit	Open reference
Enhanced navigation and saved preferences	Open reference
Find and open tools	Open reference
Find the complete Suite references	Open reference
Free Gifts & BOGO	Open reference
Fresh evaluation and short-lived permission	Open reference
How preference saves are authorized	Open reference
How the Suite unlocks and the optional Suite key	Open reference
Information shared and developer integrations	Open reference
Install and open your Suite workspace	Open reference
Integration boundaries for site administrators	Open reference
Legacy B2B and Dynamic Pricing	Open reference
One business operation, one retained identity	Open reference
Protected update readiness	Open reference

Capability or reference	Section
Read and refresh plugin connections	Open reference
Read shared records and technical details	Open reference
Read Suite through AffStackDev AI	Open reference
Read the Suite License screen	Open reference
Reconciling and recovering unfinished work	Open reference
Recover a manual download	Open reference
Recover from a connection or retirement problem	Open reference
Release the remote activation	Open reference
Replace a key or recover an interrupted activation	Open reference
Respond to connection messages	Open reference
Retained commerce records and safe updates	Open reference
Retire a B2B audience policy	Open reference
Retiring a business policy safely	Open reference
Search Discover and interpret product status	Open reference
Shared cart lines and protected merchandise	Open reference
Shared configuration, pauses and ownership	Open reference
Shared operation troubleshooting	Open reference
Store Credit	Open reference
Suite Settings and PHP mode	Open reference
Understand each license state	Open reference
Update verification and recovery	Open reference
Use and arrange the dashboard	Open reference
What Commerce Core does	Open reference
What's new in 1.0.2	Open reference

Install and open your Suite workspace

AffStackDev Suite brings compatible active products into one WordPress administration workspace. Each product continues to manage its own settings, permissions and business records. Installing Suite does not install the products listed in Discover.

Install and activate the Suite plugin through WordPress, then open **AffStackDev** in the administration menu. The shell requires WordPress 6.5 or later and PHP 7.4 or later. Its optional Commerce Core has separate

requirements and can be unavailable while the normal Suite workspace remains usable. WooCommerce is not required for the shell itself.

Administrators and shop managers receive Suite access automatically. Other accounts need the Suite capability assigned by the site's role administrator. Product screens and dashboard cards can require additional permissions. Seeing a product in a status report does not necessarily mean your account can open it. Suite's normal administration initialization reapplies access to the administrator and shop-manager roles; use appropriate role configuration for other staff accounts.

Deactivating Suite preserves its saved layout and local license identity, while stopping its license-check schedule. Explicitly uninstalling it removes Suite-owned preferences, local license/update state, its setup markers and its default-role capability. It preserves shared commerce records, order evidence and product-owned data. Uninstalling does not contact the license server or release a remote activation slot; use the License workflow when a remote deactivation is intended. On multisite, Suite's user preferences belong to the WordPress account, while site settings use the current site. Uninstall is not an automatic sweep of every site's Suite options.

Find and open tools

The **AffStackDev** menu offers Dashboard, permitted registered products, Discover, License and Suite Settings. Product entries appear when an active product registers its Suite integration and your account has the required product access. An inactive plugin does not acquire an operational menu merely because it appears in Discover.

The header displays the AffStackDev logo, a Suite badge and a **License** status link, which reads **Unlocked by your licensed AffStackDev plugin** while a licensed AffStackDev plugin on the site unlocks the Suite. If the Suite needs attention — no licensed plugin is active, or a Suite key has a problem — a banner explains the condition and provides **Manage license**. A license condition can turn enhanced navigation off or prevent layout changes while normal navigation, license recovery and independently authorized product screens remain available.

The default toolbar shows **Dashboard** followed by these categories:

Category	Normal automatic assignments
Commerce	AffiliPartnerCS, Free Gifts & BOGO, Dynamic Pricing, Store Credit and a registered Commerce Core product.
B2B	B2B.
Performance	Performance Manager and Image Manager.
Site & Content	Image Manager, Theme Manager and Gate Manager; also the fallback for an unrecognized product.
Customers	Unified Email Manager, Email Manager/UEM and Support Desk.
Tools & Testing	Products whose integration or your personal arrangement assigns this category.

The actual category can come from the product's integration, and your own saved assignment takes priority. The bundled Commerce Core is managed in Suite Settings; the category mapping does not create a separate Core product tab.

Open a category to see its accessible tools, descriptions, compatibility warnings and **Open** buttons. An empty category says that no active tools are available to your account there. **All tools** lists every permitted product plus

Dashboard, Discover, License and Suite settings, even when you have removed a shortcut from the main toolbar. **More** holds toolbar links that do not fit when enhanced navigation is running. Dashboard stays first.

A product panel shows its name, optional description and reported version. A **Compatibility hold** means the product's supported Suite integration range does not match this Suite. Update to compatible product/Suite versions; changing a category or layout does not bypass the hold. **Screen unavailable** means the requested panel has no usable in-Suite renderer. Some products deliberately use a separate destination that opens through a full WordPress page load.

With Enhanced UI, supported tools open in the main panel without replacing the whole administration page. Products that require a full load normally ask for confirmation. Moving into or out of a product placed on its own WordPress menu also reloads, usually without the generic reload prompt. Products may protect unsaved changes through their own confirmation integration; save important forms before leaving because Suite does not automatically detect every product or workspace form's unsaved fields.

If a panel fails to load, the existing screen remains available and an error offers **Retry** or **Reload this screen**. Retry attempts to load it again in place. Reload performs a normal page request after confirmation. A product renderer error stays inside its panel; a widget error stays inside its card. Persistent errors should be investigated through the owning product and Suite diagnostics.

Narrow screens and keyboard use

Toolbar menus use normal links and disclosure controls. Use Tab and Enter to reach/open them. Escape closes an open More/All tools menu and returns focus to its heading; moving focus outside the toolbar or clicking a link also closes it. Resizing can move a focused link into More and return focus to More's heading. In PHP mode, links wrap instead of relying on the overflow script.

After an enhanced page change, focus moves to the new screen heading and a status announcement identifies completion. Visible focus outlines help track keyboard position. Layout controls offer earlier/later buttons as an alternative to dragging. Wide product screens and the Plugin arrangement table can be scrolled horizontally inside their panel. Cards stack on narrow screens; the saved width still applies when space is available again. Reduced-motion preferences suppress animated loading/transition effects. The Suite does not provide a logo or color customization control.

Customize your toolbar, categories and WordPress menu

Open **Customize my workspace** beneath the toolbar. These preferences belong to your WordPress account, so another administrator can use a different arrangement. The editor is unavailable when the Suite license policy prevents layout changes. It works as an ordinary form even with Enhanced UI off, provided changes are permitted.

Choose a toolbar layout, select what it should show, arrange the items and select **Save my workspace**. The page reloads to apply the changes. Dashboard always remains first, and All tools remains available.

Setting	What to choose and what happens
Categories	Shows the selected category tabs. This is the default; the six built-in categories start selected.
Individual plugins	Shows selected shortcuts instead of category tabs. Select the desired product or core-screen shortcuts; none are pinned by default.

Setting	What to choose and what happens
Categories + pinned plugins	Shows both selected category tabs and selected shortcuts.
Category/Shortcut checkbox	Includes that item in its applicable toolbar mode. Turning it off removes only the shortcut.
Position beside each toolbar choice	Use 1–999; lower numbers come first. Equal toolbar numbers are resolved by internal destination ID, so use distinct numbers for a precise order.
Earlier/later arrows	Move a choice and update position numbers. Available with the enhancement script; otherwise edit the number fields.
Save my workspace	Commits the toolbar, category, plugin arrangement and menu placement changes together for your account.
Cancel	Returns fields to their values when the page loaded. With enhanced controls it also restores row order and closes the editor. It cannot undo an earlier save.

Switching toolbar modes keeps the selections for the other mode. New products remain in All tools without automatically becoming pinned shortcuts. The editor is part of the header; after an in-place product change, saving it can return to the section where the full page originally loaded.

Create, rename and delete categories

Under **My custom categories**, enter a **New category** name and save. The field allows up to 80 characters. You can have up to 20 custom categories, with one added per save. The new category is selected for the category toolbar layouts. Reopen the editor after saving to assign products to it.

Change **Category name** and save to rename an existing category while keeping its assignments. A blank rename leaves the existing name. To delete one, select its **Delete** checkbox and save. Deletion removes that category's toolbar entry and assignments, including references saved for inactive products. Products with other personal category assignments keep them; products with none remaining return to their normal default categories. The six built-in categories cannot be renamed or deleted here. Different custom categories can share a display name, so use distinct names for clarity.

At the 20-category limit, the New category field is replaced by an explanation. Delete a category and save before adding another through the normal form.

Arrange products and menu placement

Under **Plugin arrangement**, each accessible registered product has its own row. This product order is separate from toolbar-shortcut order.

Column	Effect
Position	Number 1–999, with lower numbers earlier in the workspace's product lists and category pages. Equal positions retain the current order. Earlier/later arrows update these positions when JavaScript is available.
Use default categories	Uses the categories declared by the product or Suite's normal mapping.
Choose categories	Select one or more built-in or saved custom categories from the multiple-selection list. An empty selection restores defaults.

Column	Effect
Dock in Suite	Keeps the product's Suite entry under the AffStackDev WordPress menu. Default placement.
Own WordPress menu	Places its Suite-compatible shortcut separately in the main WordPress menu, normally after AffStackDev, without changing the product's underlying screen or permissions.

Save to apply placement on the next page load. If a matching top-level entry already exists, Suite uses it rather than adding another. The separate entry does not recreate a product's older native submenu tree. Other plugins' native menus remain their responsibility. All tools still lists the product regardless of docking.

When a product is temporarily inactive, its personal placement, category assignment and order are retained within Suite's bounded preference storage and can resume when its permitted registration returns. You can restore all personal navigation defaults using the layout reset described below. Developer-supplied categories and advanced limits are documented in the technical workspace contract.

Use and arrange the dashboard

The Dashboard includes four Suite cards and any cards registered by compatible products for your permissions:

Card	What it tells you
Suite license	Whether the Suite is unlocked (by a licensed AffStackDev plugin or a Suite key), its message, and Manage Suite license. Starts small.
Suite overview	Number of accessible registered products, Suite contract and product compatibility indicators. Starts wide. Ready here means the integration version range matches.
Quick access	Links to permitted registered products. Starts medium.
Suite runtime	Wrapper release, contract, effective Enhanced UI, license status and the fact that preferences are per user. Starts small.

Every card identifies its source. A missing product card can mean the product did not register one or your account lacks its required capability; it does not necessarily indicate a Suite fault.

Select **Customize dashboard** to enter edit mode. Drag a card, or use its Move earlier/Move later arrows. **Change width** cycles small, medium, wide and full. **Hide** removes a card from the normal dashboard; **Show** restores it. Hidden cards remain visible with faded/dashed styling while you edit. Select **Finish customizing** to leave edit mode.

Each movement, width change and Hide/Show action saves immediately. Finish customizing is not a separate Save button, and there is no Cancel that reverses these saves. Reverse a change using the same controls. A **Saved** message confirms the request; if you see **Could not save your layout**, reload to see which state persisted and retry the change. Avoid making competing changes from multiple tabs because later saves can replace earlier ones.

If all cards are hidden, select Customize dashboard to show them again. With Enhanced UI off, Suite displays all permitted cards while retaining their hidden preferences; saved order and widths still apply. Re-enabling

Enhanced UI restores the saved hidden behavior. Dashboard saves retain currently visible/registered card IDs, so saving while a product's cards are absent can replace their former layout preferences.

Reset your personal layout

Reset my layout on Dashboard and **Reset my Suite layout** in Suite Settings perform the same reset. Confirming removes your personal menu placement, custom categories, product order, toolbar selections and dashboard arrangement. It does not reset product settings or the global Enhanced UI setting.

The reset may then ask for a full page reload. The saved preferences have already been cleared by that point: canceling the second reload prompt leaves the current display temporarily in place but does not undo the reset. Reload when ready to see defaults. To preserve your custom categories, use individual card or workspace controls instead of the full reset.

Suite Settings and PHP mode

Open **Suite Settings** to inspect the current shell and change its site-wide navigation preference.

Section	How to use it
Runtime contract	Compare Suite version, Contract version, Registered products, Effective Enhanced UI, Saved UI preference and Suite license. Registered products counts all reported integrations, including ones your account cannot open.
Enhanced UI	Enable enhanced Suite navigation is on by default. Select or clear it, then Save & reload Suite. The choice affects Suite and compatible products' effective enhancements without overwriting their own local preference.
My layout	Opens the explanation of personal arrangement and provides Reset my Suite layout when enhanced controls are available. Customize my workspace remains the editor above the page.
Protected updates	Read Updater state, whether installation-token update authentication is advertised, trusted release-key count and the explanation. Complete root key stored displays No. WordPress's plugin auto-update preference determines automatic installation.
System status	Check Registration issues, Catalog status provider, WordPress/PHP versions and download diagnostics.
Active product registry	Review each product's version, contract range, navigation mode, reported Product UI, number of dashboard cards and Ready/Needs attention explanation.
Registration diagnostics	Read the severity, stable code, affected entity and explanation for malformed or duplicate integration records. No registry issues detected means none were collected in this request.

Saved UI preference and Effective Enhanced UI can differ. A license condition can force PHP mode while preserving your enabled preference. Recovering the license can restore effective enhancements without another preference change. Conversely, a product can choose its own PHP mode even when the Suite permits enhancements.

Turning Enhanced UI off removes Suite's optional JavaScript navigation/dashboard editing layer. Normal links, product forms, category pages, Discover's submitted filters and permitted workspace customization continue through full page requests. If scripts are blocked or fail while the saved setting remains on, open Suite Settings,

disable it and save/reload for the deliberate PHP presentation. This also makes saved-hidden dashboard cards visible.

The registry's **Ready** means Suite can use the product's declared presentation contract. It does not validate a payout, price rule, credit balance, external connection or the product's own license. **Not reported** under Product UI means the adapter did not provide that flag. A reported number of dashboard cards includes cards that your account may not be permitted to see.

The Commerce Core and connection controls in Suite Settings have their own chapters. The documentation download controls are covered by the documentation chapter; neither changes your product settings.

Search Discover and interpret product status

Discover is a catalog of products that can share the workspace. The built-in entries are AffiliPartnerCS, Free Gifts & BOGO, Store Credit and Unified Email Manager; integrations can add or refine entries. An active product is not automatically a catalog entry. Cards show the product category, description, status/detail, optional badge/version and available action.

Use **Search** to match product name, description or catalog category. **Status** offers All statuses, Active, Owned, Needs attention, Installed, Inactive and Available. **Category** uses catalog categories, which are separate from your personal workspace categories. **Sort** offers Recommended, Name and Status.

With Enhanced UI, typing or changing a filter immediately updates cards and the shown count; Apply filters also applies them. These local changes are not saved as preferences or written into the URL. With Enhanced UI off, select **Apply filters** to submit and reload the page. **Clear** resets the fields to their values when that page loaded. If you arrived with filters in its URL, Clear restores those filters; choose blank Search, All statuses, All categories and Recommended to explicitly return to the complete catalog, then apply in PHP mode. A no-results message means no catalog cards match the current combination.

Display/action	Meaning
Active on this site	Suite found a compatible active product registration. Open is offered when your account has the product capability.
Needs attention	The active product's declared Suite contract range does not match. Read its explanation and use compatible versions.
Active but no Open button	Your account lacks the required product capability, or a provider supplied a different presentation action. Ask the site's role administrator when access is appropriate.
Not active on this site / Available filter	Suite did not match an active registration. It does not determine purchase ownership from this alone. View product opens a public product page in a new tab.
Owned, Installed, Inactive or other provider status	A registered status provider can supply these labels and details. Check that provider's explanation and its own authorized workflow.
Entitlement/status provider detected	A local WordPress status callback is present; this label alone does not confirm a live remote connection or license entitlement.

Discover supplies links and presentation status. It does not itself install or activate plugins, issue a license, purchase a product or perform checkout. External links open a new tab; a provider destination owns any

subsequent operation. Status order can differ between enhanced and PHP mode because one sorts the state key and the other the displayed status label.

Integration boundaries for site administrators

Products join the workspace through their active plugin's registration code. That code supplies names, destinations or embedded renderers, permissions, supported Suite contract versions, categories and optional dashboard cards. Duplicate product/widget IDs do not replace the first registration; malformed registrations appear in diagnostics. A catalog entry is a separate presentation record and does not grant access.

Product adapters also decide whether their screens support enhanced navigation and how to protect unsaved forms. Suite's public availability helper reports an integration-contract version, while license permission and protected-update readiness have separate checks. These are developer interfaces rather than settings to enter in a customer form. For exact fields, hooks, limits and helper meanings, see the technical registration reference.

Enhanced navigation and saved preferences

When products support Suite's navigation guard, they can refuse leaving a screen before a request starts and again before its replacement arrives. This allows a product to protect edits made while another screen was loading. Only the newest valid navigation response can replace the current panel. Browser Back/Forward uses the same integrated protection and restores the last stable location when navigation is canceled.

A failed in-place load does not automatically force a full reload. Use Retry or the explicitly confirmed Reload this screen. Product-specific forms, warnings and autosave behavior remain the product's responsibility. Developer details are in the browser navigation contract.

How preference saves are authorized

Workspace and dashboard changes apply to the signed-in account and require Suite access plus permission to change the layout. Normal form saves and enhanced requests carry WordPress security tokens. An expired token, changed role or license condition can prevent a save; refresh the page after resolving it and try again. Changing the global Enhanced UI checkbox is a separate Suite setting and remains available for recovery.

Use the actual controls for current permitted products/cards. The saved workspace retains inactive product placement and category choices, but dashboard saves use the cards currently available to your account. A message accepting a save cannot independently prove database persistence; reload when the result is uncertain. Developers integrating preference requests must follow the exact request schemas.

Download a support report

In **Suite Settings** → **System status**, choose **Download diagnostics JSON**. The download captures current Suite/WordPress/PHP information, product compatibility and health, catalog/provider status and registration issues. It also includes local license status, check times, update readiness, installation ID, site URL and environment. It identifies products that your account cannot open with a visibility flag.

The report excludes complete license keys, installation tokens, encrypted credential envelopes, update grants, callbacks, product settings, orders and customer records. It still identifies the site and installation, so share it with the support recipient appropriate for the issue. The filename includes the GMT date and time. Generating it

does not change a layout, create an order or exercise a commerce operation. Commerce Core offers its own separate status/report controls.

Common problems and recovery

What you see	What to do
AffStackDev or a product is missing	Check active plugin registration and your Suite/product permissions. Discover presence alone does not create a menu.
Empty category	Check your category assignment, product activation and permissions; use All tools to find permitted destinations.
Compatibility hold / Check version	Read the stated supported contract and install compatible Suite/product versions.
Needs attention in registry	Read the row's message: it can identify a contract mismatch or missing renderer/destination. Give its owning product's support the diagnostics report.
Duplicate or invalid registration diagnostic	The adapter needs a unique ID and valid fields. Do not try to fix it by changing layout positions.
This widget could not be rendered	Other cards remain usable. Reload and investigate the source product if it persists.
That screen did not load	Retry. If needed, choose Reload this screen and confirm the full load.
Could not save/reset your layout	Refresh after checking login, permissions and license state; verify the persisted layout before repeating the change.
Customize my workspace is absent	The Suite is not unlocked, so layout changes are held. Activate one of your licensed AffStackDev plugins on this site, or recover a Suite key on License; existing destinations remain available.
Enhanced UI is off despite saved On	Compare the effective runtime and license status in Settings. The preference is preserved during a hold.
Enhanced controls are inert or expected cards are hidden	Disable Enhanced UI and Save & reload Suite to obtain the normal PHP presentation, then investigate blocked scripts.
Clear leaves Discover filtered	Clear restores page-load defaults. Explicitly select the unfiltered values and apply them.
Reset appeared canceled but defaults return later	The second confirmation concerns reloading after preferences were deleted; canceling it does not undo the reset.

The technical error reference supplies exact diagnostic codes, request conditions and recovery boundaries for developers and support staff.

How the Suite unlocks and the optional Suite key

The Suite needs no license key of its own. Its licensed features — the enhanced interface, faster page navigation, and layout/dashboard customization — are allowed while any AffStackDev plugin on the same site holds an active license, or while a Suite key from an earlier purchase is licensed. If neither applies, you can still use server-

rendered navigation, open product routes, view diagnostics, and use the License screen. The Suite only reads the other plugins' own license state on this site; it never licenses them and sends nothing anywhere to do so. Each companion product continues to make its own licensing decisions, and installing Suite does not make an independently licensed product dependent on Suite.

A saved Enhanced UI preference remains saved while licensing forces the basic PHP interface. Once permission is restored, that preference can take effect again. A server policy restriction can also prevent enhanced behavior even when the accepted license state appears current. The Suite License screen reports the license state; it does not override restrictions attached to the entitlement.

Read the Suite License screen

Open **AffStackDev → License**. Your account needs Suite-management permission. The separate plugin-update control needs WordPress permission to update plugins. These forms work in the basic PHP interface.

The heading badge shows what the Suite screens show: **Unlocked by your licensed AffStackDev plugin, No Suite key needed**, or the state of a Suite key. The first panel names the licensed AffStackDev plugins found on this site (**No Suite key is needed. Licensed on this site: ...**) or asks you to install and activate one. The second panel describes the Suite's own key: **Suite key (optional)** on a site without one, otherwise that key's state. The cards below describe a Suite key's installation, checks and protected updates; on a site without a Suite key they show dashes and **No Suite key**.

Screen section	What to review
Status summary	Whether a licensed plugin unlocks the Suite, then the Suite key's own state and explanation (Suite key (optional) when there is none). A normal Licensed key is shown as healthy; permitted grace/pending states show a warning; unavailable permission shows a hold.
Licensed installation	Product code, Plan, Environment, Installation ID, Site identity, and License service. Confirm that the site/environment are the installation you intend to license. A blank plan appears as a dash.
Validation & grace	Current wrapper state, last accepted server status, last accepted check, Last attempted check, next automatic check, retry due, Connection grace deadline, Term deadline used locally, effective hard deadline, its reason, pending operation, and last error.
Time and error details	Times are UTC; missing times appear as a dash. The last error can include a code, category, HTTP status, and time. A due check may still be waiting for an eligible visit or scheduled run.
Enhanced UI during licensing	Your saved On/Off preference and the license-state explanation. A license hold does not erase the saved preference.
Protected updates	Whether token-based update access is advertised, how many release-verification keys are available, confirmation that the complete root key is not retained, readiness guidance, and the last plugin-update check.

The effective hard deadline is the earliest applicable limit. **License term / server grace** means the server's term limit wins. **Connection-validation grace** means another accepted check is needed before the connection allowance ends. The screen identifies when both deadlines are equal. Remaining time is approximate; all permission remains subject to an earlier explicit server denial.

Activate a new installation

You need these steps only if you have a Suite key from an earlier purchase. New AffStackDev customers do not receive a Suite key and do not need one; on a site without a key this panel is titled **Activate a Suite key (optional)**.

1. Confirm the **Site identity** and choose the correct **Environment**: Production, Staging, Development, or Local. The initial choice follows the site's WordPress environment setting when available, otherwise Production.
2. Enter the complete Suite license key in the blank password field. It accepts up to 128 characters in the form; the submitted key must also fit the 128-byte server-side limit. Copy the whole key without adding internal spaces or control characters.
3. Select **Activate Suite**. On acceptance, the screen confirms activation and stores an encrypted credential for this installation. The complete root key is not retained.
4. Review the accepted status, site/environment, next check, and protected-update readiness.

The server decides which environment bucket and activation limit apply. If no activation slot is available, release an old activation or use another independent Suite key. Do not repeatedly submit the same key to work around a limit or an installation conflict.

The environment is locked while an active or recoverable pending credential exists. There is no ordinary license-screen control to replace the stored site identity or license-service address. For a legitimate environment change, first release the existing active binding through the deactivation control, then choose the environment when activating again. An unresolved pending activation must be recovered first.

Activation requires secure credential encryption on the host. Suite uses Sodium when available, otherwise supported OpenSSL encryption. If neither is available, ask the hosting administrator to restore supported cryptography before activating. A secure-randomness failure also needs hosting attention.

Replace a key or recover an interrupted activation

For an active installation, the panel becomes **Replace / recover activation**, and the button becomes **Activate with replacement key**. A replacement is prepared while the older active credential is retained. The new credential becomes active only after a response is accepted. This protects against losing a working binding just because a replacement request could not finish.

An interrupted request may have reached the server even if the browser did not receive confirmation. Suite keeps a protected pending credential so it can check that exact attempt. While this candidate exists, the key field and activation submission are disabled. Use **Check now** instead of entering another complete key. A valid candidate is recovered and becomes active. If the candidate is definitively rejected and an older active credential remains, a manual check can validate that older credential; automatic recovery schedules that second check separately.

A server “license unavailable” response can leave a pending candidate in place when there is no older active credential. It does not mean the server has disclosed whether the license expired, was suspended, was revoked, or has another restriction. Review the entitlement with the license provider before retrying.

Changing the site's address, copying its database to another site, or changing WordPress security salts can affect recovery. The stored installation identity remains bound to its original site; it is not silently regenerated. Salt or identity changes can make the encrypted credential unreadable. Resolve the migration or restore the correct credential-storage context with your administrator; do not create a new identity merely to hide the warning. The screen does not offer a reset that promises to free an old server activation.

Understand each license state

Status shown	Meaning and next action
Unlocked by your licensed AffStackDev plugin	A licensed AffStackDev plugin on this site unlocks the Suite. No Suite key is needed.
No Suite key needed	The site has no Suite key and no licensed AffStackDev plugin was found. Install and activate one of your AffStackDev plugins with its license.
Site deactivated	A remote deactivation of the Suite key was confirmed. The Suite still unlocks from a licensed AffStackDev plugin; enter a Suite key only if you want to use it again.
Site identity changed	The current site differs from the stored licensed site. Resolve the migration/binding before activation or validation.
License credential needs recovery	The encrypted credential cannot be read in the current site context. Ask the administrator to investigate salts, identity, and cryptography.
Activation recovery pending	An activation candidate exists without an active credential. Use Check now before another key submission.
Local license state needs recovery	The local accepted-state cache failed its integrity check, but the credential remains. Use Check now to obtain a new accepted state.
Validation required	A credential exists without a current accepted validation. Use Check now.
System clock needs attention	The clock moved backward by more than five minutes from the last acceptance. Correct the clock and validate again.
Activation missing	The server did not find this exact active installation. Review site identity and server activation state before replacement.
License could not be authenticated	The server rejected the installation credential or product scope. Check the Suite key/product mapping and use the recovery controls.
License unavailable / License needs attention	The server explicitly denied current permission. Review the entitlement; the unavailable response does not reveal its private underlying reason.
License validation expired	The accepted term or connection deadline has passed. Restore the connection/entitlement and check again.
Licensed	A current validation is accepted.
Licensed — deactivation pending	Release was requested but not confirmed. Existing permission lasts only to its current deadline. Retry the pending deactivation.
Licensed — term grace	The server has allowed a term-grace period. Review the license before that period ends.
Licensed — connection grace	The latest check could not reach the service. Previously accepted permission remains only until the fixed deadline.
Licensed — term and connection grace	Both conditions apply; the earlier deadline wins.
Licensed — validation overdue	A scheduled check is overdue, but the accepted permission is still within its deadline. Use Check now or allow an eligible visit/scheduled run to retry.

A fresh accepted response normally schedules the next check around 12 hours later, with up to 15 minutes of variation. Connection allowance lasts at most 72 hours after acceptance; an earlier license-term or server-grace deadline wins. Repeated connection failures do not start a new 72-hour period. For example, a Monday noon acceptance with a Wednesday noon term deadline permits at most another 48 hours without a newer accepted response.

Checks, retries, and routine browsing

Check now appears while an active or pending credential can be read. It validates the pending candidate first, then the active credential as appropriate. When a remote deactivation is pending, its label becomes **Retry pending deactivation** and the action retries that release request.

Automatic validation uses WordPress scheduled events and eligible ordinary visits. Failures back off: approximately 15–16½ minutes after the first failure, 60–65 minutes after the second, and six hours after later failures. A server request to wait can extend a shorter retry up to six hours. An explicit denial follows the normal next-check schedule. Another operation already in progress may produce a “wait briefly” message rather than issuing overlapping requests.

The browsing fallback runs only on suitable read-only pages. It avoids checkout, cart, account actions, WooCommerce endpoints, APIs, form submissions, AJAX, command-line requests, feeds, and trackbacks. Administrator activity recovery requires WordPress settings-management permission. Several participating AffStackDev products share a one-operation budget per eligible request, so they can recover over successive visits. On supported FastCGI hosting the response is finished before front-end recovery; other hosts can add up to the automatic request's four-second network timeout to that visit. Viewing a status value itself does not trigger a license-server request.

Scheduled work depends on WordPress cron and site activity. A missed check does not automatically mean an expired license; compare the next-check time with the effective hard deadline. Use the manual control when you need an immediate answer, and avoid repeatedly submitting while rate-limited.

Release the remote activation

1. Open **Deactivate this Suite installation** on the License screen.
2. Check the acknowledgment that this releases the remote Suite activation slot for this site.
3. Select **Deactivate license on this site**.
4. Wait for the confirmed deactivation message. If the request is pending, use **Retry deactivation** or **Retry pending deactivation**; Suite retries the same release intent.

The local credential is removed only after an acceptable server acknowledgment. A connection failure does not mean the remote slot has been released. While release is pending, previously accepted permission can remain usable only within its existing deadline. There is no cancel-pending-deactivation button; the displayed recovery action continues that intent.

Confirmed deactivation keeps the installation ID, stored site details, and Suite settings. It clears the active/pending credentials and automatic license-check schedule. Disabling the plugin in WordPress only stops its local scheduled validation; it does not release the remote license. Uninstall removes Suite-owned local license data without contacting the server. Release the remote activation first if that is your intention.

On a migrated site, **Check now** is blocked until identity is resolved. The explicit deactivation operation can request release using the stored original binding when its active credential is still readable. This does not automatically license the new address.

Protected update readiness

Protected Suite updates through WordPress need a Suite key that is active on the site. Without one, the **Protected updates** panel shows **Suite license required** and the note: download the current Suite from **My Account** → **Downloads** on affstackdev.com and upload it under **Plugins** → **Add New** → **Upload Plugin**; your Suite settings and layouts are kept.

Suite uses its installation credential for protected updates, keeping the complete root key out of update requests and stored settings. The **Protected updates** panel distinguishes these states:

Readiness label	What it means
Waiting for token-authenticated updates	The accepted license response has not advertised this update method. Validate the license and review server support; Suite will not store the root key as a workaround.
Release trust key required	Token-based updates are advertised, but a trusted release-verification key, PHP Sodium signature verification, or PHP ZipArchive support is unavailable. Ask the administrator to check those requirements.
Suite license required	Protected updates need a Suite key that is active on this site. Without a Suite key, get Suite updates from My Account → Downloads instead.
Protected updates ready	WordPress can check for signed Suite releases and request an authorized package. Installation still follows WordPress controls and scheduling.

With plugin-update permission, choose **Check for plugin updates** for an explicit release check. The screen reports the last check as **Up to date**, **Update available**, or **Update check needs attention**, with a UTC timestamp. Normal release checks reuse results for up to 15 minutes, including a short cooldown after a failed check; this explicit control requests a fresh check. Only newer stable-channel releases are offered through this updater.

Install available releases from **WordPress** → **Plugins**. Enable or disable automatic updates there as well. Suite does not turn automatic installation on for you. Both manual and automatic protected updates verify the release signature and package contents. A license check and a plugin-update check are separate actions: a healthy license does not guarantee the host can download or install a particular release.

Update verification and recovery

Before protected installation, Suite requires an authorized offer for the same site and installed product, supported WordPress/PHP/WooCommerce versions, an acceptable signed package, and unchanged authorization immediately before replacement. The download selector shown to WordPress is not a reusable download authorization. Download authorization credentials remain in the request and are not placed in the stored update notice.

The package must have the expected size and checksum, a trusted signature, exactly the signed files, and matching embedded proof. Unsafe paths, conflicting file names, links identified by the ZIP checker, oversized archives, missing/unexpected files, and modified contents are rejected. The extracted source is checked again before installation, including its version header. Developers can find the exact limits in the technical archive reference.

Suite also checks the signed files already installed before replacing them. If you or a hosting process altered a signed file, the protected update stops for review. Preserve and assess intentional changes with the responsible administrator before replacing the plugin. An existing unsigned installation can receive its first signed update when neither proof file is present; one missing proof file on an otherwise signed installation is an integrity problem. The check protects the signed file inventory and does not classify every extra local file as a modification.

Update notice	Recovery
Filesystem problem	Ask the host to check file permissions, then retry from Plugins.
Verification failure	Open Suite License, check again, and retry only after resolving the reported problem.
Installed files changed	Review those changes before replacing the plugin.
Authorization failure	Review the license connection and protected-update readiness.
Connection failure	Restore access to the release service, then retry from Plugins.
Incompatible platform	Review the release's WordPress, PHP, and WooCommerce requirements before retrying.
WordPress restored the prior version	Contact support before retrying the failed release.
WordPress could not restore the prior version	Check Plugins and contact support to recover the installation.
Other automatic-update failure	Review the installed version and update controls; use the available details when seeking support.

These notices are shown to users who can update plugins on the Dashboard, Plugins, Updates, and Suite screens, including applicable network-admin screens. A stored failure takes priority over an available-update message. Buttons lead to **Open plugin updates** and **Review Suite License**. Rendering the notice does not contact the server.

Suite retains a small automatic-failure record containing the time, target version, prior version, and reason category. A successful automatic update clears it. A later successful manual replacement normally clears an older failure once Suite recognizes the target version was reached; a report that WordPress could not roll back is not cleared merely by that version inference. There is no separate notice-dismissal control or time-based expiry. Installed-plugin and license data are not replaced by a rejected package.

Information shared and developer integrations

Licensing requests identify the Suite product, installation, stored site URL, selected environment, and primary activation type. Activation also sends the complete key and a newly generated installation credential. Later validation, deactivation, and release checks use the installation credential. Release checks also identify the installed version and stable channel. Requests verify HTTPS and do not follow redirects.

Local configuration preserves the installation identity; encrypted tokens preserve access without keeping the complete root key. Accepted state records plan, feature/capability information, server term dates and limits, check/retry times, and bounded error details. Update cache and notices retain release/status information rather than tokens or complete error bodies. Avoid sharing root keys or decrypted credentials in support requests; the screen's status, times, and error code are usually the relevant starting details.

Integrators can query a small local Suite permission result without contacting the server or reading secrets. That result describes the Suite key only: the unlock by a licensed AffStackDev plugin applies to the Suite's own features, so a product plan that explicitly requires a Suite license still needs a Suite key. AffStackDev's current plans do not require one. A product that explicitly requires Suite must still verify its own entitlement, the supported contract, and the current permission/deadline when performing its operation. Suite does not confer another product's license merely by being installed. Administrators do not need to configure this API on the License screen; its exact fields and fail-closed rules are in the public entitlement contract.

License/service problems with unfamiliar codes have a technical error lookup. A malformed or mismatched response is not accepted as new trusted state. Rate-limit and temporary-service messages call for waiting until the retry time; installation conflicts call for resolving the original binding, not generating another identity.

Commerce Core in Suite Settings

Open **AffStackDev Suite** → **Suite settings** → **Commerce Core** to see which shared runtime is running, how commerce products connect and which shared records are available. You need access to Suite and permission to manage WooCommerce. Retirement controls additionally require Suite's standard management permission. If a colleague sees fewer controls, an administrator should review those permissions; changing a screen preference does not grant access.

Commerce Core is included with Suite. It starts automatically when the environment supports it: 64-bit PHP 8.1 or newer, WordPress 6.7 or newer, and active WooCommerce 10.0 or newer. It declares support for WooCommerce's high-performance order storage and Cart/Checkout Blocks. There is no separate Core enable switch or setup wizard on this card. Each commerce product still owns its prices, credit, gifts and business policy.

Suite includes Core release 0.1.0.2c. Several compatible products can include Core, and the site selects the highest eligible runtime version. **Running inside Suite** means Suite supplies that copy. **Running from a connected product** means another active product supplies it. The **Core runtime** line identifies the version and host. Hosting shared infrastructure does not give that product ownership of another product's business features. See [How the shared runtime is hosted](#).

Message or situation	What to do
PHP, WordPress or WooCommerce requirement is not met	Ask the site administrator to update that component. WooCommerce must be active.
Standalone Commerce Core is still active	Deactivate the old standalone Core in Plugins. Reload Settings so the available bundled copy can be selected. Existing shared records are retained.
Unsupported standalone Commerce Core	Deactivate the unsupported copy so a compatible bundled runtime can load.
Same-version Core copies differ	Install matching, complete product releases. A version label alone does not resolve differing runtime files.
Included files cannot be verified	Reinstall a complete valid Suite build, then reload Settings.
Core is unavailable on this request	Review the runtime/host information and the relevant requirement above. Connection controls appear once a usable runtime is available.

Schema setup is automatic on an eligible request when Suite hosts Core. A failed setup pauses this wrapper's next attempt for five minutes, then another eligible request can retry. It does not stop WordPress or delete existing records. If Core still reports unavailable records, ask the administrator to investigate storage using the technical status; repeatedly clicking refresh does not bypass the backoff.

Older direct Core links redirect authorized GET visits to this Commerce Core card. The card and its forms work through ordinary page requests when enhanced navigation is unavailable.

Expand **Shared records and technical details** to see runtime status, provider and shared-owner details. Under **Records and support**, **Download status report** saves the diagnostic JSON. To retrieve a retained order record, enter its numeric **Order ID** and select **Download order record**. Both downloads use the filename `commerce-core-report.json`; choose a useful local filename when keeping several reports. A retained order record can include an unfinished draft or publishing record, which is useful for resolving an interrupted operation. Detailed download permissions, accepted IDs and messages are covered in [Core status and records](#).

Read and refresh plugin connections

The **Plugin connections** list shows each product's name, role, available version, status and explanation. Select **Refresh connection check** after activating a product, correcting an integration setting or restoring a compatible runtime. It opens a fresh Settings request. The list describes registration and interfaces visible during that request; it does not place an order, redeem credit or authorize a financial operation.

Status	Meaning and next step
Connected	Required registration and callable integration points are present. Customer eligibility and each transaction are still evaluated by the responsible product.
Interface available	A supported interface is detectable, but the row does not confirm a complete audience or checkout connection. Read its explanation before interpreting it as readiness.
Connection blocked	Expected ownership, required registration, callback evidence or Core records prevent a usable connection. Follow the row's product/settings guidance.
Not verified	The report, permission or integration evidence is incomplete or unrecognized. Review the product's integration status, then refresh.
Not installed	The required product is absent from this site. Install it only if you intend to use that connection.
Installed but inactive	The product is installed but inactive. Activate it in Plugins if this connection is required.
Connection unavailable	The product can be active without making this integration available. Check its integration and license settings.
Policy retired	Core confirms accepted removal of B2B's audience policy. Each consuming product must separately support that proof and an appropriate continuation policy.

Missing or inactive wording is used only when a reader reports no loaded connection. It does not replace a more specific blocked or unverified explanation. For **Dynamic Pricing + Store Credit**, both products are checked; a missing product takes precedence over an inactive product. A renamed plugin directory can still be recognized through its text domain.

If Core records are unavailable for the current site, otherwise Connected or Interface available rows become Connection blocked. Fix that shared runtime/storage problem before treating product-level registration as usable. A reader failure generally affects that product's row, allowing the remaining product rows to remain informative.

B2B

B2B supplies the audience and business-data connection. **Connected** means Core accepted B2B's supported audience registration. It does not say that a particular shopper is a business customer or eligible for a discount.

If a previously configured B2B provider disappears, Suite retains a blocked status. Restore B2B or complete the supported retirement process below. Simply deactivating it does not remove the site's expectation of a business policy. Conflicting audience providers also block the connection; inspect B2B Tools and Core status. If B2B is loaded but its audience provider is not registered, B2B Tools is the place to resolve its readiness.

When B2B is absent and Core confirms an accepted audience-retirement proof, the row says **Policy retired**. That proof preserves the historical records; it does not transfer B2B prices or turn customers into consumers.

Store Credit

The Store Credit row covers its checkout participant. **Connected** means Core accepted that participant and both required checkout callbacks are available. It does not spend a balance. Duplicate participant registrations block the connection; inspect active integration adapters. If Store Credit is active but has not registered, inspect its runtime and license status.

A runtime unable to report checkout registration gives **Not verified**. An old standalone Core may need to be deactivated so a compatible bundled copy can provide that reporting. The underlying participant and settlement behavior is described in [Shared checkout participation](#).

Dynamic Pricing

The Dynamic Pricing row covers its audience and cart-line integration. **Connected** requires accepted B2B consumer registration, the active pricing callbacks and Core's audience callback. A competing audience callback blocks confirmation; the administrator should inspect the active adapters rather than adding another writer.

With no B2B audience configured, the row can say **Interface available** because pricing's integration is present without an audience connection. A missing or unrecognized product report gives **Not verified**; a failed companion connection gives **Connection blocked**. Review Dynamic Pricing's integration settings and B2B Tools as the explanation directs.

After accepted B2B retirement, Dynamic Pricing can report **Interface available** when it recognizes that proof. Its explicitly selected consumer-only policy can continue under its own rules. Verified-audience mode still needs an audience provider. Suite does not select or change either policy, and retirement does not clear Core's remembered owners.

Dynamic Pricing + Store Credit

This separate row covers pricing's coupon compatibility reader. It is distinct from Store Credit's checkout participant. **Interface available** means Dynamic Pricing recognizes the credit reader and required validation callbacks. Your product settings for combining pricing with credit and the actual coupon's eligibility still apply.

A missing reader leads to the appropriate unavailable, missing or inactive explanation. An unsupported credit contract or missing validation callback blocks the pair. Install compatible integrations and review Store Credit's

runtime before refreshing. An unknown or unreadable report gives **Not verified**. This check does not redeem credit or change coupon settings.

Free Gifts & BOGO

Free Gifts & BOGO's row covers its audience and gift-line connection. Its reported integration hold gives **Connection blocked**. A declared B2B connection with an active cart-validation callback and Connected B2B gives **Connected**. Its Core interface with an active cart callback and no B2B audience configured gives **Interface available**. Other incomplete combinations give **Not verified**.

Suite does not use Free Gifts & BOGO's separate retirement field in this row. Consequently a retired B2B policy can leave the Free Gifts & BOGO row **Not verified** even when Free Gifts & BOGO has its own retirement-aware decision path. Consult Free Gifts & BOGO's integration settings for its actual promotion decision; Suite does not select gifts or override line validation.

AffiliPartnerCS

AffiliPartnerCS reports its partner-audience and protected-cart-line integration. Suite checks that the report identifies the installed product version, matches the current Core runtime and has the required active registration, coupon and cart callbacks. You also need permission to view AffiliPartnerCS's own status. Without that product permission, Suite can show **Not verified** even though you can open this card.

Connected confirms the B2B consumer registration and active cart hooks. With no B2B audience configured or expected, **Interface available** can describe the available Core and cart interfaces. A rejected registration, missing expected B2B connection or mismatched Core runtime blocks the connection. Review AffiliPartnerCS Infrastructure and B2B Tools. A report that claims automatic referral succession is not accepted as verified.

This status does not certify financial readiness, send payouts or transfer customer/referral ownership. A product can report that its shared executor exists without making those operations available through Suite's connection card.

Commerce Bridge

Commerce Bridge provides a protected licensing-line interface. Suite shows **Interface available** when its reader and supported contract are present. Actual protection still depends on the particular cart line and the owner's response.

If protection was previously expected and the owner disappears, Suite shows **Connection blocked**. Restore Commerce Bridge to resolve that missing-owner condition. If the product is loaded without its supported interface, the row is **Not verified**. Removing the plugin does not erase protection expectations.

Retire a B2B audience policy

Use **Retire a B2B audience policy** only when intentionally removing that policy. The disclosure appears within Commerce Core for users with both Suite's standard management permission and WooCommerce management permission. It manages B2B's audience retirement only; it cannot retire arbitrary products, transfer prices or release B2B's write freeze.

1. In B2B's own workflow, establish that no live business policy remains and freeze changes. If Suite cannot prepare a proof, inspect B2B Tools; Suite cannot manufacture that evidence.
2. Open Suite Settings and expand **Retire a B2B audience policy**. Select **Prepare retirement** when offered.

3. Deactivate B2B in Plugins. Return to Suite Settings on a fresh request and select **Accept after deactivation** within five minutes of preparation.
4. Read the resulting state. **Accepted for this site** confirms usable accepted evidence on that site. Each consuming product still decides whether it supports a continuation policy.

Preparation is bound to the current site, owner, scope and runtime evidence. Changes to that evidence can make the stored proof unusable. Expiration does not accept or cancel a retirement. Accepted evidence preserves existing records and ownership expectations; ordinary missing-owner checks can still hold until a compatible consumer applies its separate policy.

Displayed state	Available action and meaning
No retirement prepared	Prepare retirement starts the supported proof workflow. Cancel retirement can also record cancellation even when no proof exists.
Prepared — waiting for B2B deactivation	Accept after deactivation uses the current preparation. Cancel retirement is also available.
Accepted for this site	Cancel retirement records revocation if you change your mind.
Cancellation recorded	Prepare retirement can start a new workflow. There is no repeat Cancel button for this stored state.
Preparation expired — cancellation is still available	Use Cancel retirement to record revocation, then prepare again through the supported process if required. The expired acceptance is unavailable.
Stored proof is not currently usable	The stored record is stale or its validity is unavailable. Cancellation remains available unless the record is already revoked. Keep business changes frozen.
Cancellation could not be confirmed — keep B2B changes frozen	Retain the freeze, refresh the current record and resolve the persistence problem. An uncertain cancellation is not a completed cancellation.
Retirement management unavailable — keep B2B changes frozen	No usable revision is available, so the form is hidden. Restore the compatible runtime/status before continuing.

There is no additional confirmation dialog after selecting these buttons. The form carries the current record revision; if another request changes the record, refresh Settings before choosing the appropriate operation again. Do not reuse an old submission.

After a successful operation, the notice says the record was updated and B2B must confirm its own safe resume. A blocked notice says to keep B2B changes frozen, refresh and retry using the current record. These notices are brief, per-user feedback and are consumed when displayed; the current state is the durable reference.

Cancel retirement writes cancellation rather than deleting the old evidence. It can cancel expired or stale preparation as well as accepted retirement. An active returning owner attempts to revoke accepted proof, but a failed persistence attempt is still unconfirmed. B2B must verify cancellation of both its audience and protected-line retirement records under its own safeguards before allowing business changes. This Suite screen covers the audience record only. See [Retirement records and safeguards](#).

Shared-operation recovery is separate from audience retirement. This card has no force, reset, financial retry or takeover control. A compatible product or authorized integration must use the original retained operation and the recovery contract described in [Shared operation recovery](#).

Recover from a connection or retirement problem

Follow the explanation beside the affected connection. Restore the missing product or compatible adapter, confirm the product's own permission/runtime/license requirements, and use **Refresh connection check**. Activation alone does not prove registration, and refreshing does not clear a policy hold.

For a retirement authorization error, reload Settings while signed in with both required management permissions and submit the current form. For a stale revision, expired proof, unavailable management contract or uncertain save, keep B2B changes frozen and reread the current state. Resolve the underlying runtime, storage or owner-evidence problem before repeating the relevant operation. A retirement action must use the Settings POST form; opening the action as a direct link is rejected.

Suite has no button here to delete shared records, clear remembered owners or reset retirement evidence. Technical maintainers can find exact input validation and error codes in the connection and retirement reference.

What Commerce Core does

Commerce Core connects compatible products to shared customer context, cart-line identities and accepted order records. Suite includes the runtime and starts it when its requirements are met. It needs 64-bit PHP 8.1 or newer, WordPress 6.7 or newer, and active WooCommerce 10.0 or newer. The included Core release is 0.1.0.2c; its technical version is 0.1.0.2.3. See [Commerce Core in Suite](#) for the outer setup card and unavailable-runtime messages.

When several compatible products include Core, the highest eligible runtime version supplies the code for that request. Matching versions must contain matching listed runtime files; conflicting copies are held until the affected products are updated to a consistent release. An already loaded standalone Core is not loaded a second time. Suite identifies whether the selected runtime runs inside Suite, from a connected product or from a standalone copy. The selected host is not automatically the owner of your prices, gifts, business policies or credit balances.

Core coordinates information and records. Your connected products still control their business features and licensing. Installing Core alone does not create a discount, assign a customer to a business program, issue credit, transfer a gift rule or add a new payment action. Features that need a product adapter stay unavailable until that compatible adapter is present.

Read shared records and technical details

Open **Suite** → **Settings** → **Commerce Core**, then expand **Shared records and technical details**. You need permission to manage WooCommerce to view this panel or download its records. Suite access remains subject to the shell's normal permissions. The panel works with ordinary forms; its downloads do not require JavaScript.

Display	How to read it
Commerce Core version	The version of the runtime selected for this request. A compatible connected product may supply a newer runtime than Suite's included copy.
Foundation — Available	Core can read its local runtime configuration. Connected products still have their own readiness checks.

Display	How to read it
Foundation — Needs review	Core cannot read the required local record. Resolve the database or stored-record problem before relying on affected shared features.
Audience connection — Configured	Core remembers at least one product responsible for determining consumer-promotion eligibility. This is a retained requirement, not proof that the product is currently working.
Audience connection — Not configured	No audience owner has been configured in Core's retained record. Each product continues to apply its own ordinary consumer behavior and other conditions.
Order records	Count of accepted Core order snapshots. If the database count fails, the card can display 0; the downloaded status report distinguishes that failure from an actual empty history.
Audience providers	Currently registered owner names, their declared versions and registration state. This list does not itself run their eligibility checks.
Shared feature owners	The last saved feature assignment, with its feature ID, capability, owner and current recorded state. Products verify permission and availability again before applying a change.

The shared-feature state reads **Operation needs reconciliation or is in progress** when an operation is pending, **Enabled** for an enabled unarchived definition, or **Paused** otherwise. **No assigned owner** means no owner is recorded; **A shared record needs review** means the returned entry cannot be presented normally. If no shared adapters are available, the panel explains that none are registered for this request. The list displays the first 25 shared records. An integration can obtain additional records through the paginated status interface; there is no Next button in this panel. The [shared-feature chapters](#) explain ownership, editing and operations.

Respond to connection messages

Connections need attention can show more than one issue. Resolve each applicable condition rather than treating the Foundation card as approval for every feature.

Message concerns	What to do
Local records unavailable	Restore database access and readable Core storage. Reload the panel after repair. Existing records remain the basis for recovery.
B2B needs its Commerce Core reader update	Use a B2B integration that supplies the supported fresh reader. The older reader can retain business exclusions but cannot approve uncertain consumer promotions.
Dynamic Pricing has multiple audience callbacks	Have the owning integrations remove the competing adapter as part of their coordinated update. One audience decision path is required.
More than one expected audience owner	Review the competing integrations and their retained requirements. Core does not choose an owner by loading order or silently discard one.
Previously configured audience provider missing	Restore that product's supported integration or follow the explicitly supported policy-retirement procedure for the affected consumer. Simply deactivating a provider does not erase Core's expectation.
Previously configured licensing protection unavailable	Restore the expected protection integration. A missing reader does not turn protected or unknown merchandise into ordinary merchandise.

The connection report provides the broader product-by-product view. Accepted retirement is a separate proof that integrations may explicitly consume. This bundled Core's ordinary audience and line-protection readers do not automatically clear their remembered requirements when a retirement proof is accepted.

Download reports and individual order records

Under **Records and support**:

1. Select **Download status report** to download `commerce-core-report.json` with current runtime, provider, connection, snapshot-count and shared-feature information, plus the latest retained diagnostic sample.
2. To investigate a specific order, enter its positive numeric **Order ID** and select **Download order record**. The server accepts an existing order ID of 1–13 digits, without a sign, decimal point, spaces or leading zero.
3. Keep the file with the relevant support or integration investigation. Downloading changes neither the order nor its journal state.

Both downloads use the same filename. Rename saved copies if you need to retain more than one report. An individual order download contains the journal's revision, state, fingerprint and recorded data. It can include a draft or an interrupted publication as well as a committed record. It does not convert an unfinished record into an accepted one.

Enter a valid order ID means the input format or order lookup failed. Correct the ID and confirm the order exists. **This order has no readable Core record** means there is no readable Core journal snapshot for that order. A real WooCommerce order can exist without a Core snapshot, for example when no compatible historical-audience provider was configured. The export does not substitute today's customer classification or an older B2B record. **Access denied** requires a user with WooCommerce-management permission. If a form's security verification expires, reload the Settings screen and use its fresh form.

Reports are administrative records. Status reports identify integrations and saved shared definitions; order reports contain the recorded order subject and line evidence. Provide the relevant file to the person authorized to investigate the issue.

Customer eligibility and pricing integrations

An audience owner determines whether the current shopper is eligible for consumer promotions. Core ties its decision to the store, shopper and current checkout context. Changes in login, cart contents or quantities invalidate the request's previous context. Currency, shipping choice, tax scope, variation attributes and applied coupons also contribute to that context. Core passes a digest rather than storing the shopper's address in these context fields.

Fresh decisions have a limited validity period of at most five minutes and must match the same request. Consumer eligibility requires a consumer classification. Invalid, missing, conflicting or expired evidence leaves eligibility unresolved; it is not a permission to apply a consumer promotion. Connected products may also have their own finality and business-rule requirements.

The **Configured** state is deliberately retained when an audience provider disappears. Removing the plugin does not make all shoppers consumers. If no owner has ever been configured, Core reports that no audience provider applies; the connected product decides how its own standalone behavior uses that result. Core does not expose a customer selector, manual audience override or force-eligible button.

Legacy B2B and Dynamic Pricing

The older B2B reader can preserve a known ineligible outcome. It cannot establish fresh eligibility for a consumer, even when its earlier result said eligible. Its historical reader also cannot prove that an order's business context was atomically finalized. Install the supported reader integration when the affected product requires fresh or accepted historical evidence.

Core's Dynamic Pricing adapter runs only when an audience dependency already exists, B2B is present, or the runtime configuration cannot be verified. It requires exactly one audience callback. It delivers fresh supported owner evidence to Dynamic Pricing; it does not calculate the discount itself. Competing callbacks or uncertain B2B evidence hold the affected consumer decision until the integration is corrected. Installing Core by itself does not turn a previously independent Dynamic Pricing setup into a new audience-dependent setup.

Commercial facts without price changes

A compatible product can attach a commercial fact to a cart line, including currency, quantity, tax basis, price source, lock state, an optional unit amount and the kinds of adjustments its contract recognizes. Refreshing that fact replaces the prior fact in the same owner/slot. Core records the information; the owning product must actually apply any permitted price or adjustment.

Stored facts can remain on an order after their current-request validity expires. They describe what was recorded and do not approve a later price change. Missing amounts are distinct from zero. For integration authors, value rules and commercial-fact fields give the exact decimal, field and size requirements.

Shared cart lines and protected merchandise

Compatible products can opt a cart line into Core identity. An untouched ordinary line does not receive an identity merely because Core reads it. Once a product claims or links a line, Core preserves its UUID, the declaring product's reference and any declared relationships or facts. This allows connected products to describe a line consistently without changing the product itself.

A line can contain up to 16 provenance entries, 32 relationship entries and 16 commercial facts; Core cart operations support at most 200 lines. A product updating its own owner/slot replaces that entry rather than adding another identity. A relationship creates matching outgoing and incoming records on two different lines. Removing that particular relationship keeps the lines' identities and other records. These are integration operations, with no claim/link/unlink buttons in Suite Settings.

If Commerce Bridge supplies the existing line identity, Core reads it as Bridge-owned and does not migrate it. While the Bridge cart reader is present, Core's own line writers stay read-only, including for a line that Bridge describes as unprotected. Core also refuses an unknown protection result. An integration must establish one supported writer rather than let two systems alter the same identity graph.

Core reports relationships pointing to a cart line that is no longer present; it does not silently remove them. Complete order snapshot creation requires every related UUID to exist in the order. Duplicate IDs, mixed Core/Bridge identities, malformed records or unresolved protection need correction by the owning integration. If checkout displays **A shared cart line needs review. Please refresh the cart before checkout.**, refresh and review the cart. If the same message returns, investigate the line-owning and protection integrations before repeating checkout.

Core's classic-checkout line hook stores the validated line record in WooCommerce order-item metadata. Blocks snapshot capture reads whichever Core line records actually exist on its order items; the bundle does not

register a separate Blocks line-record writer. Normal formatted order-item metadata and normal WooCommerce REST order responses omit raw `_afsd_cc_` records. Authorized Core readers and exports remain the way to inspect their recorded evidence.

Accepted order history and publication recovery

An accepted Core snapshot preserves the owning provider's committed audience decision together with the recorded order and line context. It includes the recorded customer, currency, totals, payment-method identifier and line evidence. The order fingerprint binds those fields to a specific order state. A snapshot does not itself mean the order was paid or authorize a commission, reward or credit.

Automatic capture is available on classic and Blocks checkout only when exactly one expected audience owner has a registered historical-order reader. A current eligibility reader alone is insufficient. If no such owner is available, Core does not manufacture a snapshot. This means the Order records count need not match your total WooCommerce order count.

Once accepted, history remains readable even if the customer changes, the order is refunded, the provider is removed or the order's current totals change. Those later events do not rewrite the original snapshot. Its purpose is retained historical context, not a fresh entitlement for today's shopper. An older B2B record can be returned only as unverified legacy history; it does not become a committed Core record.

Draft, publishing and committed

State	Meaning and next action
Draft	Core has prepared matching order and owner evidence but has not finalized publication. A supported integration can continue using the current revision and original order fingerprint.
Publishing	Core saved the durable publication intent before confirming the corresponding order metadata. Preserve that intent and investigate the original record.
Committed	The exact order projection was confirmed and the journal finalized. The accepted record remains available for historical reads.

If the panel reports **An interrupted order publication needs review**, download the original order record. A user with WooCommerce-management permission can have the supporting integration invoke Core's reconciliation API with that exact order, revision and fingerprint. There is no reconciliation button in this panel. Reconciliation confirms that the intended order projection already exists and then finalizes the journal; it does not write a second projection or replace the original request.

If that projection is missing or differs, the operation remains pending. Passage of time does not authorize a takeover or discard the intent. Resolve the saved-record discrepancy through the owning integration. Do not create another order or use a different operation identity as a substitute for confirming the original result.

At checkout, **The order context needs review before payment** means Core could not verify the order fingerprint. **The accepted order context could not be saved. Please contact the store before retrying payment** means preparation or publication failed. Inspect the existing order and its retained Core state before retrying payment. A late failure can leave a publishing record that requires reconciliation rather than a fresh attempt. Technical procedures are in snapshot publication and snapshot recovery.

Diagnostic reports and integration interfaces

The status report includes diagnostic counters from the current request and the latest retained affected request. Core retains up to 32 distinct bounded code names per request, with each counter capped at 999. It saves one latest sample rather than an accumulating activity log. An ordinary “no audience provider” outcome is not stored as a persistent warning. A later request with no new relevant codes leaves the earlier sample intact, so compare its recorded time with the current problem.

Diagnostic samples contain code names, counts, Core version and recording time rather than raw provider error text or shopper details. The broader status and order reports contain their respective integration and order records. A report code identifies the failed boundary; it does not mean every previous step was rolled back. Runtime error recovery and [shared-operation recovery](#) explain how to interpret the retained state.

Developers use Core's local PHP interfaces to register readers, read current audiences, manage line declarations, prepare and reconcile order snapshots, and call the separate shared-definition, operation, retirement and checkout-participant protocols. There is no public HTTP audience endpoint. Most local interfaces rely on the calling integration to verify its own user permissions and request security. Core adds WooCommerce-management checks to its administrative downloads and specified management/reconciliation operations. The complete function reference identifies every facade, argument order and owning contract.

The selected runtime supplies the `affstackdev-commerce-core` translation domain. The included catalog is a template that translation packages can use. Core's status layout wraps long identifiers, stacks cards and download fields on smaller screens, and provides visible keyboard focus for its controls. Its included GPL text is available with the runtime files.

Retained commerce records and safe updates

Commerce Core keeps the records that let compatible products agree on a feature's current configuration, responsible product and unfinished operation. It does not become the owner of product prices, customer balances or payout ledgers. Orders continue to use WooCommerce's normal storage interfaces.

Suite includes Core 0.1.0.2c. Another compatible active host can provide the selected shared runtime. The available operation depends on that selected runtime and the owning products' supported interfaces, rather than on a release number alone.

Records belong to the original WordPress site. Core requires a supported 64-bit PHP environment and rejects attempts to reuse its active runtime after switching to another site. Its installer creates missing local records without rewriting an existing database schema. Network-wide installation through the Core installer is refused; activation must be handled separately for each site. An unavailable or unfamiliar record is preserved and held for recovery.

Each record has an identity, a revision, a state, its data and a fingerprint. A successful update must name the revision it actually read. If someone else saves first, the stale update fails instead of overwriting their work. A new record uses expected revision zero. For example, if two operators read revision 7 and one saves revision 8, the second must reload revision 8 before making a reviewed change. Saving at revision 7 again does not overwrite revision 8.

A fingerprint confirms the expected data content; it is not a payment receipt. Core can coordinate its records without guaranteeing that a separate financial service completed a payment. If a result is uncertain, keep the original reference and reconcile the owning service before taking another value-changing action.

Core also remembers which audience and line-protection services it has already depended on. Adding a requirement is persistent. Removing the product does not erase that requirement or make a previously protected benefit safe. There can be at most 16 expected audience owners. Repeated conflicting configuration writes eventually return a busy result; they do not silently drop an owner. Snapshot status counts report retained records by state, not the amount of money owed or paid.

These records have bounded, validated data. The journal accepts at most 128 KiB of canonical data per record; individual text values are at most 2,048 bytes, lists/maps at most 400 entries per level and nesting at most 12 levels. The responsible integration supplies supported fields and types. Merchants should use its current form or supported service instead of editing database rows or changing revision numbers manually. Exact storage contracts are in the technical journal reference.

Choosing a supported feature provider

A shared feature can run only through a product that explicitly supports its full configuration and is currently ready to perform the requested task. Products register their local adapters during normal initialization. This happens automatically through code supplied by those products; there is no screen for entering a PHP callback.

Provider registration identifies the owner, provider version, supported feature families and schema versions, preference priority, and separate availability, evaluation, execution and reconciliation functions. Up to 16 owners can register; an owner describes up to 16 feature families with up to eight schema versions each. These limits are integration boundaries, not a reason to divide a financial operation across several invented owners.

Core distinguishes checking or editing a feature, executing its effect and reconciling an earlier effect. A product may remain able to read an old receipt while it cannot perform new business. Both full feature support and current task readiness must succeed. Missing callbacks, exceptions, a wrong site or conflicting registration keep the feature unavailable. A duplicate accepted owner can make the shared registry unavailable for the whole request; resolve the conflicting integration and begin a fresh request.

The ready preferred owner wins. If the preferred product is unavailable, Core considers other products that explicitly support the same definition. Among those, a smaller provider priority wins, with owner-name order breaking ties. For example, priority 10 precedes 200, but a ready preferred owner takes precedence over both. This is a technical adapter setting; it is not a ranking of merchant product discounts. A feature's own stored priority is separate from provider priority.

The provider overview reports registered or unavailable and its declared feature support. Registered does not mean licensed for this transaction, eligible to serve this customer or safe to issue money. Those checks remain with the product.

An authorized integration can retire a shared executor, or restore its availability, using the exact saved retirement-record revision and a reason. This requires WooCommerce management permission. Retirement prevents that owner from beginning a new authorized effect; it does not delete shared definitions, undo earlier effects or remove its ability to supply supported old-receipt reconciliation. A compatible alternative may become responsible after current authority is resolved. A pending operation is not forcibly transferred by this action. This executor control is separate from the [business-policy retirement procedure](#).

The technical provider contract defines the trusted local integration fields. It does not add a general provider editor to Suite.

Shared configuration, pauses and ownership

Shared configuration keeps one original feature identity while compatible products can edit or perform that feature under its supported contract. A product must understand the complete saved terms. Installing another product does not convert old terms or turn on a paused feature.

Configuration or record	Effect and restriction
Feature identity	The original unique ID is retained across edits, ownership changes and recovery.
Capability and semantics	Identify the feature family and exact meaning of its terms; existing records cannot change these identities.
Definition schema and payload	The owning product's complete supported field set. A replacement schema must be understood explicitly. Core does not infer missing settings.
Enabled	Allows execution when true and every other check succeeds. Turning it off retains the definition.
Archived	Holds execution while preserving configuration and history. It does not cancel a customer's accepted obligation.
Feature priority	Integer 0–9999 retained for the feature's own meaning; not the provider selection priority.
Preferred owner	Empty for normal supported selection, or the exact preferred supported product.
Creator and last editor	Preserve the original creator and identify the most recent changed definition.
Definition revision	Advances when the complete configuration changes.
Ownership generation	Advances when configuration or selected ownership changes.
Record revision	Used to protect every saved replacement from a stale request. It is separate from definition revision and generation.
Pending operation	Holds the original operation reference, attempt token, expiry and evidence; blocks another feature operation or edit until resolved.

These fields are supplied through an owning product's form or an authorized local integration. Suite does not provide an unrestricted editor for arbitrary feature payloads. Configuration data must fit 48 KiB and the common text/depth limits.

An editor must currently support editing that exact definition. Saving a changed definition requires its exact record revision and no pending operation. It preserves the creator, advances definition revision and generation, clears the old owner and records the last editor. A save with identical terms leaves the row unchanged, but still checks edit authority and current revision. There is no shared-feature delete operation in this interface; disabling or archiving preserves the record.

When a compatible product asks who should act, Core checks whether the feature is enabled, unarchived and free of an unfinished operation, then selects a ready supported owner. If none is available, the saved feature remains held, potentially with an empty current owner. A missing owner never authorizes another product to guess the terms. Simultaneous ownership changes may return busy after three conflicting attempts; reload the current result.

The shared status view is read-only and requires WooCommerce management permission. It shows owner, generation, definition revision, enabled/archived state and whether work is pending. An unreadable feature is identified as unreadable rather than hidden as successful. Reporting is paginated: 25 records by default and at most 100 per request. A displayed owner is a stored assignment, not permission to execute now. For exact output fields see shared definitions and reporting.

Fresh evaluation and short-lived permission

A product evaluates a shared feature using its current configuration, current owner and exact transaction context. Core does not retain an old cart result as permission to apply it later. The selected product supplies the result, and Core checks authority again after that callback completes.

The returned authority claim lasts at most 60 seconds. It binds the site, feature, owner, ownership generation, definition revision and complete definition/context fingerprints. A changed customer or transaction context, changed owner, edited definition or expired claim invalidates the earlier evaluation. Obtain a fresh evaluation instead of editing or extending the old claim.

An unfinished operation has its own reserved authority and prevents ordinary resolution from supplying a competing owner. A local claim is not a public authentication token, product license, customer eligibility decision or financial receipt. The relevant product must still enforce each of those requirements.

Evaluation results are bounded to 16 KiB; context is at most 8 KiB. The owning product controls the supported fields. Integrators can use the evaluation and claim reference to preserve the complete envelope and correct data types.

One business operation, one retained identity

Every delegated operation uses a stable business reference and exact context. Core derives a stable operation and effect identity from that original reference and site. Keep it when retrying, moving between supported owners or investigating a failure. Creating a new reference to get around a pending result can create a separate effect.

Operation state	Meaning and next step
Pending	An attempt was reserved and its outcome is not yet confirmed. Reconcile the original operation.
Committed	The owner supplied a confirmed effect receipt. Repeating the original request returns the retained result without executing again.
Not applied	The owner supplied a confirmed no-effect receipt. Ordinary repeats return that result. Another attempt requires the separate explicit recovery conditions.
Unknown callback answer	The journal remains pending; unknown is not a fourth terminal state or proof that nothing happened.

Core reserves the feature before asking a product to perform its effect. The feature permits one pending operation at a time. The operation freezes the feature definition, context, owner and short-lived claim, with a unique attempt token and empty receipt. Other operations and edits remain held while the reservation is unfinished.

Immediately at its financial action boundary, the product must confirm that the original pending record, token, lease, owner, generation and definition revision still match. Core checks this before and again after callback-based readiness. Retiring the owner or losing its current availability can stop a stale attempt. The product still owns its transaction, money lock, durable deduplication and authoritative receipt; a Core check cannot replace those controls.

A product returns a confirmed applied or no-effect result with its receipt. Core saves that terminal result before releasing the feature reservation. If reservation release fails, replaying the terminal original result can retry that release safely. A thrown callback, malformed answer or unknown outcome preserves pending evidence. A storage error can happen after the owning product applied an effect, so do not equate an error with “nothing was paid.”

Passive lookup retrieves the original record without executing anything. It is useful for support and reconciliation, but its existence alone does not make a pending effect safe to repeat. The technical operation contract defines the exact callback and receipt fields.

Reconciling and recovering unfinished work

Reconciliation asks a supported authoritative reader what happened to the original operation. It never invokes the effect callback again. Core waits until the original reservation's lease expires before reconciling pending work. If the process stopped after reserving the feature but before inserting its operation row, Core can reconstruct that original pending row from the reservation, keeping the same claim and token.

The reconciler may be another supported product that understands the frozen feature and can verify its authoritative original receipt. It must report confirmed applied, confirmed not applied, or unknown. Unknown keeps the hold. Expiry by itself does not prove absence: an earlier writer could still have applied the effect before the response was lost.

A new attempt is supported only through the explicit no-effect recovery API, when both the selected Core and the owning integration support it. Suite does not create an automatic financial retry schedule or unrestricted Retry button for this API. The operator uses the original product's supported recovery workflow.

For explicit recovery, all of these conditions matter:

1. The original operation is confirmed **not applied**, its old claim has expired, and the request names its exact current operation revision, receipt and unchanged context.
2. The current feature's full definition and definition revision match the frozen original. Editing terms and then restoring their old values still changes the revision and blocks this recovery route.
3. An authoritative reconciler freshly confirms the same no-effect receipt and proves that every old writer is prevented from applying the old attempt. An empty table, elapsed timeout or manually typed receipt is insufficient.
4. A currently ready supported owner receives a different attempt token and bounded authority. The original business reference, effect identity, site and terms remain.
5. The old terminal attempt is archived before the original operation is advanced to pending again. Only the winning current revision can reach the effect callback, which must still enforce its own financial protections.

For example, after a provider confirms that `reference reward:order-42` was not applied, recovery can keep that same reference and its original frozen terms. Submitting `reward:order-42-retry` would be a different operation and is not the supported recovery procedure.

If recovery is interrupted before advancing the operation, the same request can resume an existing matching reservation without extending its expiry. An expired reserved attempt cannot be executed merely by resuming it; it must be reconciled. A failure after the new pending operation is saved leaves pending evidence. Reread that operation after an ambiguous result or conflict rather than repeatedly pressing retry or clearing another attempt's reservation.

Archived attempts retain their exact original receipts and payloads, linked by previous-attempt revisions. They are not overwritten or embedded recursively. Keep the archives, current row and financial owner's evidence together. If the current host lacks the required recovery interface or the original terms changed, leave the original case held for supported owner review; there is no force override. Exact requirements are in same-operation recovery.

Retiring a business policy safely

Business-policy retirement is a specific proof that an owner has no live policy left in an exact scope. It differs from retiring a shared executor. The two supported scopes concern current audience policy and protected-line policy. Suite's native [B2B retirement controls](#) expose the bounded audience workflow; they do not expose every owner/scope or release the B2B product's business freeze.

Use the owning product's preparation procedure before deactivating it. The product must establish a durable freeze and confirm no live policy remains. Core records the exact owner, scope, site, request identity, policy revision, evidence reference and freeze reference. Preparation also records the current runtime configuration fingerprint and a five-minute acceptance deadline. Core checks the proof's shape; the owner is responsible for the actual freeze.

Step	Requirement and effect
Prepare	Owner and its retirement guard are present. An authorized WooCommerce manager uses the current revision. The guard confirms no live policy and frozen state. Core writes Prepared evidence.
Deactivate owner	Follow the owner's supported procedure while keeping its freeze intact. Preparation alone does not remove the product.
Accept	Use a fresh request after deactivation, before five minutes expire. The guard and both site/network-active plugin entry must be absent, with the exact Prepared revision and unchanged runtime fingerprint.
Read accepted proof	A compatible consumer freshly verifies the accepted original record. It does not substitute the management display or an error.
Cancel	An authorized exact-revision write revokes the record. The owner separately decides when its own resume conditions are satisfied.

An attestation can freeze owner-side work before Core's final save succeeds. If preparation fails, keep the freeze and check the actual stored result; do not assume the owner can resume. Likewise, an acceptance notification can fail after the Accepted record was saved. Reread persisted state instead of repeating a different operation on the assumption of failure.

Accepted proof does not expire simply because the original five-minute preparation deadline passes. It remains subject to its site, runtime fingerprint, absent owner and unrevoked state. Changed runtime expectations, returning owner or invalid data make it unusable. Retirement does not erase old audience/protection

requirements, transfer business prices, change historical order evidence or grant consumer discounts. Ordinary Core readers still hold when their required provider is absent unless a compatible consumer explicitly uses the separate retirement proof under its own rules.

The management view separates **stored state** from **validity**. State can be None, Prepared, Accepted or Revoked. Validity can be none, current, expired, stale, owner present, revoked or unavailable. Prepared can be current while the owner is present; that does not make acceptance possible yet. The owner-present flag reflects the collected guard or retained plugin basename, not an unrestricted scan of every possible owner.

Revocation reporting also distinguishes not attempted, confirmed and unconfirmed. This is what the current request observed; a later stored revision may differ. A read/write error must not be shown as confirmed revocation.

Cancellation always writes and advances the revision, even if the record was already revoked. Expired or stale proof can be cancelled, but corrupt or wrong-site records remain held for recovery. If no record exists, cancelling at revision zero creates a cancellation-only marker. That marker is not retirement proof; it prevents a delayed earlier preparation from appearing after cancellation. A later legitimate preparation requires the newer revision and new authentic owner evidence.

When an owner returns, Core attempts to revoke its previously accepted audience and protection records. It preserves Prepared records so the normal prepare → deactivate → accept sequence can work. Failed revocation remains unconfirmed: the owner's durable freeze must stay until all relevant cancellations succeed under its own write lock. A saved cancellation does not itself release that lock or resume business policy. Do not delete the journal or reset a revision to bypass a failed cancellation. The technical retirement contract contains exact fields and permission boundaries.

Discovering checkout credit participation

Core can advertise the currently registered Store Credit reservation participant to a compatible checkout coordinator. This discovery is independent of ordinary audience providers and shared feature owners. It does not reserve or spend credit.

The supported descriptor has exactly six values: schema 1, owner `affstackdev-store-credit`, capability `store_credit_reservation_participant_v1`, integer revision 1, and the owner's callable `enlist` and `order-verification` functions. Compatible plugins supply this descriptor in code during checkout-participant collection at initialization priority 40. There is no customer field for choosing a callback or entering an alternate protocol revision.

Registration must finish before another plugin resolves the participant. A duplicate attempt for the supported owner makes the participant conflicted for the request, even if the second submission is identical or follows a malformed first attempt. A thrown collection callback discards partial registration. Records are not saved across requests, and stale unavailable callbacks are discarded rather than reused.

Connection status	Meaning and response
Registered	Descriptor and callable functions are available. The actual transaction still needs its owning products' checks.
Not ready	Collection has not finished; the integration must resolve later in initialization.
Missing	No accepted supported descriptor was registered; restore the owning integration.

Connection status	Meaning and response
Unavailable	Required function is not callable; restore the compatible implementation.
Conflict	Competing supported-owner registrations occurred; correct them before a fresh request.
Failed	Collection could not complete; investigate the registering service.
Site mismatch	The active Core belongs to another site; use the original supported site context.

Reading status does not invoke either financial function. Resolving a participant checks the exact owner, capability and integer revision, then returns a detached descriptor for the trusted caller. It does not validate an order's financial proof, verify payment, mark an order paid or provide a substitute participant. B2B owns coordination, Store Credit owns its ledger and receipt evidence, and both products' actual transaction contracts remain necessary. Use [Store Credit connection](#), [B2B connection](#) and the technical participant contract for their respective boundaries.

Shared operation troubleshooting

Symptom	Appropriate response
Record or runtime cannot be verified	Restore the original site's compatible runtime/storage. Preserve unfamiliar schema and corrupt evidence for recovery; do not overwrite it as empty.
Save conflicts or reports busy	Reload the current revision and review concurrent changes before saving.
No supported feature owner	Check the exact definition, product availability, enabled/archive state, adapter registration and current permissions. A displayed old owner is not a grant.
Evaluation or claim expired	Obtain a fresh current evaluation; do not edit the token or extend its expiry.
Operation pending or outcome unknown	Keep the original reference and context. Reconcile through its authoritative owner after the lease boundary; do not submit a new financial operation.
Confirmed not applied but retry held	Review exact operation revision/receipt, unchanged definition and fresh no-effect proof. Explicit supported recovery is required.
Archived attempt cannot be read	Keep the original record/history intact and restore the compatible reader; do not manufacture replacement history.
Retirement preparation fails	Keep the owner's freeze, inspect current proof/revision and complete the owning product's supported preparation.
Acceptance is unavailable	Confirm owner deactivation, five-minute preparation window, unchanged runtime and exact revision. Reread actual state after ambiguous results.
Cancellation or automatic revocation is unconfirmed	Keep the freeze and inspect persisted state under the owner lock. Cancel at the current revision; do not assume it succeeded.
Retirement was accepted but benefits remain held	Acceptance is separate evidence, not a customer eligibility grant or an automatic clearing of provider requirements. Follow the consuming product's supported policy.

Symptom	Appropriate response
Checkout participant is missing/conflicted	Restore the exact compatible owner and correct registration timing/duplicates. Discovery failure does not authorize bypassing credit checks.

Provide support with the original site, feature/business reference and current error code. Keep the original journal contents intact while the owning integration resolves the case. The technical error lookup maps every returned code to its specific recovery boundary.

Find the complete Suite references

Open **AffStackDev → Suite Settings → Documentation**. Choose **Download customer reference Markdown** for the complete task reference or **Download technical reference Markdown** for the detailed developer and agency reference. Both files belong to the installed Suite release. Word and PDF editions of the customer reference are on affstackdev.com.

Your account needs access to Suite. The downloads remain available when Enhanced UI is off or a Suite license restriction has forced the workspace into PHP mode. Downloading a reference changes no settings and performs no commerce operation.

Recover a manual download

If a saved download link is rejected, reopen Suite Settings and use a fresh link. If the file is unavailable, restore the matching Suite package so both canonical files are present in its docs directory. A permission message means the signed-in account lacks the Suite capability; a file-read error means the installed document or its filesystem permissions need attention. The Documentation card downloads Markdown. Use the Word or PDF edition on affstackdev.com when you prefer those formats.

Read Suite through AffStackDev AI

When the separate AffStackDev AI plugin is installed, its plugin documentation area can identify Suite's exact installed version, feature reference and two current manuals. Suite includes a declaration that points to the same Markdown files downloaded from Suite Settings. It does not give an AI assistant permission to change your store.

After installing an updated Suite package in the existing plugin folder, open Suite in the AI plugin documentation area and choose **Check this plugin again**. Check **Installed plugin version**, then select the references labeled **Installed package**. That check rereads the installed files; it does not install Suite or fetch replacement documentation. If Suite is excluded from owner AI, use its **Include this plugin for owner AI** control when you want those owner-facing reference tools to read it.

Under **Previous manuals & automatic checks**, enable **Continue using previous documentation when current manuals are unavailable** and choose **Save documentation preference** if you want eligible retained references to fill missing documentation. This preference applies across installed plugins. A **Previous manual** records its earlier documentation version; compatibility with the installed version is unverified. A readable current customer or technical manual takes precedence over previous manuals for that audience. Use the **Installed package** pair for the current Suite release. A missing or mismatched feature reference is recovered by reinstalling the matching installer from **My Account → Downloads** and checking it again. A large feature reference may require opening

Suite's individual documentation screen to complete inspection, and its full feature list is retrieved in pages. After a completed check, the inventory can reuse its saved summary.

Plugin manuals and reviewed site knowledge have separate refresh and sharing controls. Updating Suite replaces its installed references; it does not change an earlier AI reply or automatically approve a site knowledge source for a customer-facing channel.

Glossary

Term	Meaning
Suite workspace	The shared WordPress administration area that hosts compatible product screens and Suite controls.
Product registration	A compatible plugin's current-request declaration of its screen, permissions and presentation capabilities.
Catalog entry	Presentation metadata shown in Discover, separate from product runtime registration and license authority.
Personal navigation	Menu placement, categories, assignments, order and shortcuts saved for one WordPress user.
Enhanced UI	Optional JavaScript navigation and dashboard controls permitted by the saved preference and current Suite policy.
PHP mode	Server-rendered administration with ordinary page requests.
Contract	A versioned public interface and its accepted inputs, outputs and rules.
Installation identity	The persistent local UUID and site/environment binding used by Suite licensing.
Installation token	The encrypted local credential used after activation for permitted licensing and update requests.
Protected update	A plugin release checked against its product identity, signature and exact file inventory.
Commerce Core	The shared runtime coordinating registered commerce owners, accepted facts and retained operations.
Owner	The registered product that has authority over a defined business decision or shared feature.
Snapshot	Facts saved at a defined point to preserve the original decision context.
Shared definition	A validated, versioned configuration published through a supported shared feature contract.
Revision	A saved record version used to detect stale or competing changes.
Retirement	An explicit transition releasing an owner's authority while retaining the required historical evidence.
Reconciliation	Establishing the original operation's outcome from its retained evidence before retry or recovery.
Nonce	A WordPress action-bound request token checked in addition to the user's capabilities.