

AffStackDev Image Manager — Complete Customer Reference

Version 1.8.0 · AffStackDev · Documentation revision 2026-10-22.1

Edition date: October 22, 2026.

This reference covers installation, every workspace and control, all media and optimization workflows, local and remote recovery, optional protected updates, and the complete reviewed AI workflow. Image Manager manages native WordPress media, creates separately stored image alternatives, and helps the administrator organize, inspect, edit, replace, back up and recover media. The companion Master Technical Reference describes the exact source interfaces and all action parameters.

Contents

[What's new in 1.8.0](#)

[Automatic folders and protected Community photos \(since 1.7.0.4\)](#)

[Installation and access](#)

[What each file and result means](#)

[Media library browsing](#)

[Folders and collections](#)

[Website galleries from a folder](#)

[Import folders from another plugin](#)

[Folders for pages and posts](#)

[Uploads and metadata](#)

[Copy-first image editor](#)

[Convert one PDF page to an image](#)

[Replace a file while retaining the attachment ID](#)

[Inspect where media is used](#)

[Files with no use found](#)

[Optimization setup and public source selection](#)

[Compression controls](#)

[Start, pause, resume and stop work](#)

[Optimization results and per-image controls](#)

[Comparison and additional display sizes](#)

[Bulk optimization and frozen selections](#)

[Automatic discovery and future uploads](#)

[Public image delivery](#)

[Lazy loading](#)

[Pictures sized to fit the page](#)

[Delivery through your own CDN](#)

[Public-page and browser delivery checks](#)

[Tools and background processing](#)

[Local portable backups](#)

[Import and restore media](#)

[Remote backup settings](#)

[Google Drive](#)

[S3-compatible storage](#)

[Remote transfer, retention and recovery](#)

[Optional protected software updates](#)

[Complete reviewed AI workflow](#)

[Stage files and secrets privately](#)

[Recovery of an uncertain reviewed operation](#)

[Deactivation, uninstall and managed-copy cleanup](#)

[Troubleshooting by saved state](#)

[Local command-line controls](#)

[Glossary](#)

[Complete AI action index](#)

[All screens actions](#)

[Settings actions](#)

[Optimize actions](#)

[Optimize / Tools actions](#)

[Tools actions](#)

[Images / Compare compression actions](#)

[Images / selected actions actions](#)

[Images / Bulk editor actions](#)

[Media / Organization actions](#)

[Media / selection and native attachment fields actions](#)

[Media / Edit selected details actions](#)

[Media / Upload to this folder actions](#)

[Media / Editor actions](#)

[Media / Convert actions](#)

[Media / Replace actions](#)

[Backups / Create actions](#)

[Backups actions](#)

[Backups / Local history actions](#)

[Backups / Inspect and restore actions](#)

[Backups / Bring in a backup actions](#)

[Backups / Backup destination actions](#)

[Backups / Google Drive actions](#)

[Backups / S3 connection actions](#)

[Backups / Remote transfer actions](#)

[Backups / Remote history actions](#)

[Media / Where is this used actions](#)

[Images / Compare / Delivery actions](#)

[License & Updates actions](#)

[Media / Library / Website galleries actions](#)

[Media / Library / Import folders actions](#)

[Media / Library / Files with no use found actions](#)

[Admin lists / folder bar actions](#)

[Optimize / Sizes for the page actions](#)

[Settings / Deliver pictures through your CDN actions](#)

[AffStackDev AI / Native recovery review actions](#)

What's new in 1.8.0

Release 1.8.0 follows 1.7.5 and adds three things. **Folders for pages and posts** (Settings, off at the start) gives the admin lists of posts, pages, products and other types folders of their own, with a folder bar, a Folder column, Move to folder and drag and drop; the folders stay in the admin and change nothing on your website. **Pictures sized to fit the page** (Optimize) measures how wide each picture is shown on your pages, at every screen width, and with **Fit pictures to the page** (Settings, off at the start) tells the browser, so phones download small files; it can also make the sizes those widths need. **Deliver pictures through your CDN** (Settings) sends your pictures through a CDN you already have, after checking that it returns your files unchanged, and checks it every day. Extra display sizes are now also made for JPEG files that carry a comment or a print resolution (as GD writes them), in WebP or AVIF without those two notes. AffStackDev AI can do the new things too, as proposals you approve in WordPress. Everything else in this section came with earlier releases.

Release 1.7.5 follows 1.7.4 and adds five things. **Lazy-load pictures that WordPress does not reach** (Settings, off at the start) lazy-loads the pictures a theme or plugin prints without WordPress; WordPress's own choices stay. **Largest picture to make lighter** (Settings) lets pictures up to $2,560 \times 2,560$ pixels be made lighter from the start where the server has the memory, or up to $8,192 \times 8,192$ pixels; up to 2 megapixels was the limit before. A folder can become a gallery on a page (the block Image Manager folder gallery or a shortcode) once it is switched on under **Website galleries**. **Import folders from another plugin** copies the folders of FileBird, Real Media Library, HappyFiles, Enhanced Media Library or Folders (Premio). **Files with no use found** checks the whole library at once; it deletes nothing. Library forms now say what is missing when a confirmation or a folder was not chosen. JPEG files with a comment or a print resolution in their header, as GD writes them, now get WebP and AVIF copies. AffStackDev AI can do the new things too, as proposals you approve in WordPress. Optional: Image Manager works fully without a license. A free license from your AffStackDev account adds signed updates from affstackdev.com and counts for the full mode of AffStackDev Suite on this site (its enhanced screens and layout choices). Everything you set up keeps working whatever happens to the license. Release 1.7.3 made License & Updates a tab of Image Manager's own page and 1.7.4 made every screen text translatable. Everything else in this section came with the launch release 1.7.0.5a.

Release 1.7.0.5a was the AffStackDev launch release of Image Manager; WordPress showed it as version 1.7.0.5.1. It contains every change since 1.7.0.4.

Your site stays up while Image Manager is busy. Image Manager allows only one of its own jobs to change media at a time. Image batches, backups and saves each take that turn for a few seconds. In 1.7.0.4, a page that arrived during that window could stop with "There has been a critical error on this website". That could be a storefront page, an admin page, a WP-CLI command or a plugin update screen. Release 1.7.0.5a includes the fix. Two Image Manager jobs still never run at the same time.

What happens now while Image Manager is busy:

- **Ordinary page views, admin screens and scheduled runs** load normally. Background scheduling for that one request is skipped; the job that is already running schedules the next turn itself.
- **Uploading, editing or deleting an image** waits up to eight seconds for the running batch to finish, then records the change. An unrelated save, such as an order, a product price or an ordinary post, never waits.
- **Folder or collection fields in the media window** are not saved if Image Manager stays busy for longer than eight seconds. The window says so; save again a moment later. The rest of the attachment still saves.
- **Deleting an image** never waits for the batch to finish. If an unsaved editor draft belonged to that image, its private copy is removed a minute later instead.

- **A scheduled remote backup** that finds Image Manager busy tries again after fifteen minutes instead of skipping the whole day or week.
- **Image Manager Updates** shows the reference `im_license_worker_busy` when a license request arrives while Image Manager is busy. A protected Image Manager update says "Image Manager is finishing background work. Try the update again in a minute." The installed files are not changed.
- **Deactivating Image Manager** waits up to fifteen seconds for the running batch to finish, then completes. If the batch is still running after that, deactivation still succeeds. It stops all future runs, but your Image Manager settings are left unchanged. **Activating** while an earlier batch is still finishing shows a message asking you to try again in a minute.

The header of every Image Manager screen, WordPress Plugins and the update system all show the same release number, 1.8.0.

A plain refusal instead of an error. A signed-in account without Image Manager access that submits the AI inputs form now gets the message "You do not have permission to change this media item." instead of a critical error, and the administrator no longer receives a Technical Issue email. Nothing is staged or shown.

Free media tools; a free key for updates. Image Manager's media tools work without any key. The free Image Manager license from your AffStackDev account enables protected software updates; it covers one live (Production) site and one test copy of it. AffStackDev Suite is optional and needs no key of its own. Download the plugin from **My Account → Downloads** on affstackdev.com.

Automatic folders and protected Community photos (since 1.7.0.4)

New WordPress image attachments are automatically placed in Website images / Uploads, Pages, Posts, Products or Courses according to their current parent. This works with automatic optimization switched off. A new upload remains available through the normal Media Library and Image Manager. Updating the attachment parent updates its automatic category. These are virtual folders; original files and their URLs stay in place.

Folders chosen through native or reviewed organization controls take priority over the automatic fallback. An importer can supply more useful category paths through the organization API; its assigned context survives subsequent image saves. Importer categories preserve any manually chosen folders. This is context-based organization, without visual recognition or an external AI service. Files uploaded outside WordPress and Community need an owning integration before Image Manager can catalog them.

Open Images > Protected Community photos to see current Community photos grouped as Posts, Groups, Grow-offs, Profile pictures or Profile covers. Community 0.1.0.12 or newer is required. The gallery is for authorized site administrators and applies Community's current visibility checks to each item. New photos appear on refresh without synchronization or an import job. Deleting a photo in Community removes it from this listing. Use Community's own controls to edit or remove protected photos. Its private storage, URL checks, photo processing and size/count limits remain authoritative; these photos are never copied to public WordPress uploads or sent through the public optimization pipeline.

Automatic folder assignment never prevents a successful native upload. A temporary failure schedules up to three retries (approximately one, two and four minutes, subject to WordPress cron). A notice identifies an outstanding failure; save the attachment's Media details to retry. Its original bytes and existing folders remain intact. Imported public resources may also include PDFs and video; they appear in the normal library under their importer categories.

Installation and access

Download the Image Manager ZIP from **My Account → Downloads** on affstackdev.com, install it through WordPress Plugins > Add New > Upload Plugin, then activate it. This release requires a single WordPress site, WordPress 6.5 or later and PHP 8.1 or later. Network activation and multisite are rejected. The media tools work without an AffStackDev account, update key, Suite, WooCommerce or AI connector. Optional integrations appear when their owning plugin and host capability are present.

The administrator needs both `manage_options` and `upload_files` for the Image Manager workspace. Editing a particular attachment also requires WordPress permission to edit that attachment. The separate native update-license screen requires `manage_options`; checking for software updates additionally requires `update_plugins`.

Open Image Manager under Media when it runs on its own. A compatible AffStackDev Suite can place the product in its shared product navigation. Its tabs are Library, Optimize, Backups, Settings, Tools and License & Updates (in the WordPress menu also Media > Image Manager Updates). Library contains the media library workspace and the results per image; Optimize has Run optimization and Optimization results. The image detail view adds the controls appropriate to the selected file. A separate Media > Image Manager AI inputs screen provides private staging for reviewed AI actions.

Activation installs the image index and saved defaults. It does not start a scan, recompress existing images, replace media, authorize remote storage, or turn optimized public delivery on. On an update, saved settings, organization labels, source records, prepared files and active or paused work are retained. Reload an open workspace after changing the plugin package so that its forms and scripts match the installed release.

The normal PHP forms work without the enhanced JavaScript interface. Settings > Enable enhanced interface and live progress enables the interactive workspace. Suite's global PHP mode takes precedence. Turning the enhanced interface off preserves the saved media rules and scheduled work.

What each file and result means

WordPress owns attachment IDs, uploaded source files, native size metadata and the pages or products that reference attachments. Image Manager owns its optimization index, recipes, verified image alternatives, virtual folders and collections, private drafts, portable snapshots and cloud-transfer records.

Optimization writes separate files in the managed image directory under WordPress uploads. The source and native crops remain available. The copy-first editor creates a new attachment. PDF conversion creates a new PNG attachment. Replacement keeps the selected attachment ID but gives it a new file family and URLs; it creates a verified recovery snapshot first and retains the former files.

An image can have prepared optimized files while public delivery is off. A queued run is work awaiting processing. A completed snapshot is a private recovery archive. A verified remote upload is a confirmed archive at the configured destination. These states have different controls and are shown separately.

The optimization savings figures compare prepared image files with their source files. Keeping originals and additional copies means that these figures are not reclaimed disk space. The public-page and browser delivery checks describe separate observations of the selected image.

Media library browsing

Open Images and choose the media library view. Switch between Grid and List. Each page contains up to 30 media items; Previous and Next move through the matching set. Search matches native media title, filename or caption. Changing filters starts at the first page. Clear filters resets search, folder, collection and type while retaining the selected layout and sort preference.

Folder and Collection filters select existing organization labels. All folders includes unfiled items; a selected hierarchical folder can include its children through the native taxonomy query. Collection membership is independent of folder membership.

File type filter	Matching saved WordPress media
All file types	All attachments in the permitted library query
All images	Image MIME types
JPEG, PNG, WebP, AVIF	The corresponding image MIME type; JPEG also accepts the legacy progressive JPEG MIME label
GIF, SVG, HEIC / HEIF, TIFF, BMP	Their corresponding saved image MIME labels
PDF documents	application/pdf
Documents	PDF, plain text, CSV, RTF, ZIP, Word, Excel, PowerPoint and OpenDocument types
Video or Audio	Native video or audio MIME families
Archives	ZIP, RAR, 7z, gzip and tar MIME labels

The filter describes what can be browsed. Editing, optimization and conversion appear only for files their owning service can safely process. A document appearing in Documents does not gain image-editing controls.

Sort control	Order across the matching set before pagination
Newest first	Upload date descending, then attachment ID descending
Oldest first	Upload date ascending, then attachment ID ascending
Title: A to Z	Title ascending, then ID ascending
Title: Z to A	Title descending, then ID descending
File type, then title	MIME type, title and ID ascending
Recently updated	Modification time descending, then ID descending

Library cards and list rows show native attachment information and the permitted file controls. Selecting an item opens its details. Image previews use native WordPress image sizes. PDF cards reuse a saved raster preview no larger than 320 pixels on its longest edge; the details view allows a saved raster preview up to 1200 pixels. If no suitable saved preview exists, a PDF badge and Open PDF link remain available. Browsing does not render new PDF pages or start an optimization run.

The detail view exposes title, caption, alternative text for images, native file information, folders, collections, usage inspection and the available edit, conversion or replacement workflow. Open the original file or the native attachment editor through its supplied link when the task belongs to WordPress.

Folders and collections

Folders are hierarchical virtual organization labels. Collections are flat labels. They organize native attachments without moving files, changing public URLs or changing attachment IDs. An item can belong to several folders and collections at the data level; the native WordPress attachment Folder field offers a single folder choice, while the Image Manager bulk tools can assign multiple memberships.

To create a folder, enter New folder, choose a Parent or Top level, and select Create folder. To create a collection, enter New collection and select Create collection. Names are plain text and must be nonempty and no longer than 120 bytes after cleaning. A same-name folder in the same parent is resolved through WordPress's native term rules; select the existing label instead of creating a duplicate organization entry.

Open Rename or delete a folder / collection, select the existing label and choose Rename or Delete organization label. Rename requires a new name. Confirm the operation before applying it. Deleting a label removes its organization relationship; it does not delete any attachment or media file. Folder child handling follows the native WordPress taxonomy owner.

For bulk organization, check the media items and choose the target folder or collection. Add preserves existing memberships and adds the selected labels. Remove detaches only the selected labels. Set replaces the membership list for that taxonomy; an explicitly empty list clears all memberships of that kind. Folder changes do not clear collections, and collection changes do not clear folders. Bulk metadata and organization selection is bounded to 100 permitted attachments per operation.

In the enhanced workspace, drag selected media to a folder. Review the displayed selection and destination before applying. The drag interaction is a proposal; the confirmed native organization operation performs the change. A compatible native WordPress media dialog also exposes the Folder selector and Collection checkboxes, allowing organization during ordinary media selection and attachment editing.

Website galleries from a folder

A page can show the pictures of one folder as a gallery that keeps itself current: pictures added to the folder later appear by themselves, and pictures taken out of it disappear. Use the block Image Manager folder gallery (in the block editor's Media group) or the shortcode `[afsd_im_folder_gallery folder="ID"]`. The output is WordPress's own gallery markup, so the theme styles it, WordPress lazy-loads it and Image Manager's optimized delivery applies to it.

Visitors see a folder's pictures only after someone who manages media switches it on: open Folders and collections > Website galleries, choose the folder and Show in website galleries, tick the confirmation that every picture in the folder may be shown to visitors, and select Save website gallery choice. Stop showing in website galleries switches it off again; the pictures and the folder stay as they are, and the gallery shows visitors nothing. The panel lists the folders that are on, each with its shortcode. A gallery of a folder that is off, does not exist or has no pictures shows visitors nothing; people who manage media see a short note saying why.

Block setting	Shortcode attribute	Choices and initial value
Folder	folder	The folder's ID (the block lists the folders by name)
Include sub-folders that are shown on the website	subfolders	Off (no); yes adds the pictures of sub-folders that are switched on too
Columns	columns	1 to 9; 3
Picture size	size	The image sizes of the site and full; thumbnail
Link each picture to	link	The picture file (file), its attachment page (attachment) or no link (none); the picture file
Order	order	Newest first (newest), oldest first (oldest) or by title (title); newest first
Most pictures shown	max	1 to 100; 50

Only pictures are shown, never other files in the folder. The gallery reads the folder when the page is built, so a page cache shows a change once it is renewed.

Import folders from another plugin

Folders and collections > Import folders from another plugin copies the folders of FileBird, Real Media Library, HappyFiles, Enhanced Media Library or Folders (by Premio) into Image Manager folders. The other plugin may be active or not. The panel lists each plugin it finds with its number of folders and file placements; tick the confirmation and select Import folders from that plugin.

- The tree stays the same: each folder becomes an Image Manager folder under the same parent, at the top level for the other plugin's top level. A folder with the same name under the same parent is reused.
- Files are added to their folders and never taken out of a folder they are already in, including Image Manager's automatic folders.
- FileBird folders that belong to one user (FileBird's per-user mode) go under a folder named after that user, for example "Jane (FileBird)".
- Real Media Library's folders, collections and galleries all become folders. A Real Media Library shortcut puts the original file into that folder; the shortcut's own entry is left alone.
- Enhanced Media Library is imported from its own Media Categories (media_category), not from other taxonomies it may be set to use.
- A placement whose file or folder no longer exists is counted and skipped.

Each step works for about eight seconds and saves its place; a large import says More remains: choose Continue import. Running the same import again adds nothing twice and reports the folders that were already there. The other plugin's tables, folders and settings are only read, never changed, so it keeps working as before; deactivate it yourself once you no longer need it. Imports of more than 5,000 folders from one plugin are refused. WP Media Folder and WordPress Media Library Folders are not imported.

Folders for pages and posts

Settings > **Folders for pages and posts** gives the admin lists of the types you tick folders of their own: Posts, Pages, Products, Coupons and any other type with an admin list. It is off at the start. Media keeps its own folders (above).

Each list you tick shows a folder bar under its heading: **All** (every item, the trash not counted), **Not in a folder**, and your folders as a tree, each with the number of items in it (drafts, scheduled and private items count). Choose one to show only its items; the list's own filters, search and pages keep the choice. A **Folder** column shows each item's folder as a link to it.

To put items into a folder, tick them, choose the folder under **Bulk actions > Move to folder**, and Apply. **Take out of its folder** leaves the ticked items outside any folder. You can also drag a row, or several ticked rows, onto a folder in the bar. An item is in one folder at a time: moving it into another takes it out of the first.

Manage folders (in the bar) creates a folder at the top or inside another one, renames it, moves it into another folder or to the top, and deletes it. Names are plain text of up to 120 characters. Two folders with the same name in the same place, and a folder inside itself or one of its own sub-folders, are refused. Deleting a folder deletes nothing else: its items are then outside any folder and its sub-folders move up one level. A list keeps up to 500 folders.

Anyone who may edit an item may move it (an Author only their own posts). Creating, renaming, moving and deleting folders needs the right to edit other people's items of that type, which Editors and Administrators have.

Folders only sort the admin list. They stay in the admin: nothing appears on your website, no address changes, and WordPress's REST API does not list them. Switching a list off hides its bar and column and keeps its folders; switching it on again brings them back with their items.

Uploads and metadata

Use the library's **Upload media to a folder** workflow: choose an existing destination folder and one to eight files, then select **Upload to selected folder**. WordPress's own allowed MIME types and upload limit apply. The folder is assigned after WordPress creates each attachment and its native metadata. WordPress remains responsible for the upload name, path, file validation and image sizes. An upload does not prove that an image belongs on a public page; optimization eligibility is evaluated from its supported public associations.

The bulk metadata form has separate Set title, Set caption and Set alternative text choices. Check only the fields to change. The entered value replaces that field on every selected attachment. An empty value with its Set checkbox checked clears that field. A field whose Set checkbox is unchecked is retained, even if its input box is empty. Title and alternative text are cleaned as plain text; caption permits multiline plain text. Alternative text is applied to image attachments, and ignored for other MIME families. Page-specific alt text already written into content remains the page owner's value.

The attachment's title and caption are native WordPress fields. Image alt text is the native attachment alt metadata. Metadata editing does not edit a page's content, rename the stored file or change its URL. Review a shared title or caption carefully before applying it to a group.

Copy-first image editor

The basic editor opens a safe local JPEG, PNG, WebP or AVIF attachment when the host's WordPress image editor supports it. The source must be readable, within the uploads tree, no larger than 64 MiB, no larger than 24 million pixels, and no more than 16384 pixels on either edge. The editor does not fetch a missing source from a remote URL.

Choose the operations, confirm preparation and select **Prepare edit preview**. The order is rotate, flip, crop and resize. At least one change is required.

Editor control	Choices and behavior
Rotate	None, 90° clockwise, 90° counterclockwise or 180°
Flip	None, horizontal or vertical
Crop	None, square 1:1, 4:3, 3:2, 16:9 or Custom
Custom crop	Supply x, y, width and height within the image after rotation and flip; the rectangle must remain inside that image
Resize width / height	Zero leaves that constraint unset; positive values are bounded to 16384 and must not upscale the edited image

Review the private draft before saving. Save as new media creates a new attachment and native sizes from the prepared file. It inherits the source's caption, description, alternative text, parent, folders and collections, and adds edit lineage and the normalized recipe. The title identifies the edited copy. The source attachment, its URL and existing page or product assignments remain available. Select the new item in the page, product or other native owner where it should be used.

Discard removes the active edit draft. Drafts belong to the current actor and expire after 24 hours. If the source or draft file changes, prepare a fresh draft rather than saving an obsolete preview. Editing JPEG, WebP or AVIF uses the native editor's image output with quality 95 where that setting is supported; this workflow is an image edit, independent of the optimization recipe.

Convert one PDF page to an image

Select a local PDF and open Convert PDF page. The host must provide Imagick with a working PDF/Ghostscript policy. The ordinary saved PDF preview used for browsing is independent of this conversion capability.

Choose one page from 1 through 500 and a maximum output edge from 320 through 2560 pixels. Prepare the preview. The service renders the chosen page locally at 144 DPI, flattens it on white, converts to sRGB, strips image metadata and produces PNG within its resource limits. The chosen page must actually exist. Oversized or unsupported pages are rejected.

Review the private draft, then confirm Save as new media. The result is a separate PNG attachment titled for the source PDF and page number. Its caption, folders and collections are inherited, and its recipe and source attachment are recorded. The PDF itself and its existing links remain intact. Select the new PNG in the native page or product owner when you want to display it. Discard removes the prepared conversion draft; preparing a new conversion replaces the actor's prior conversion draft for that PDF.

Replace a file while retaining the attachment ID

Open an image's replacement panel and select exactly one replacement image of the same supported MIME type. Prepare and review the replacement preview. Replacement uses a native uploaded file, not an AI-selected filesystem path or remote URL. The source and candidate must pass the image checks before the prepared draft is available.

Applying the replacement first creates and verifies a local recovery ZIP for the old attachment family. It then creates a new file and native sizes, updates the existing attachment's file metadata, retains its ID, and records a replacement journal. The old files stay on disk. WordPress-rendered uses of the attachment ID can follow the new file. Direct URLs saved in content, CSS, external pages or caches still refer to the old files and need their owning content or cache control.

The replacement workflow bounds the old family to 128 files and 512 MiB and bounds the newly generated family to 64 files. Its journal identifies the old and new source families and the safety snapshot. A source, metadata or revision mismatch stops application before it can overwrite an unrelated edit.

Undo is available for the latest matching applied revision. It verifies the retained old files and restores that attachment's former file and metadata while retaining the replacement files. It does not revert arbitrary later media edits or delete caches. Recovery handles an interrupted applying or undoing journal only when the current fields match the recorded old/new states. Review and confirm the supplied recovery control. An unrelated change requires inspection through the native attachment and backup controls rather than forcing the old journal over it.

Discard removes a prepared replacement that has not been applied. A prepared draft is private, actor-bound and expires after 24 hours. An applied replacement revision is recovery history, separate from an unconsumed draft.

Inspect where media is used

Open the Usage panel for the selected attachment and start its inspection. The service freezes an upper content ID and reads a bounded batch of permitted native content. The enhanced interface continues while the report has more work; in the PHP interface use Continue. The report can be restarted for a fresh content range.

Usage inspection checks native featured-image relationships, WooCommerce product gallery and variation relationships, core image/gallery block references, gallery shortcodes, supported content image URLs, excerpts, relevant native metadata, the file URL, PDF preview references and supported embedded links. Results show the owner that can be opened and edited. Private content is included only when the current administrator can edit that owner. Permissions are checked as the report is read.

The report is specific to one attachment and the current user, lasts up to 24 hours, and uses a run identity and cursor so a continued batch is not accidentally counted twice. Each batch reads up to 40 content items within a 1.5-second soft budget. It accepts up to 256 KiB of content/excerpt per item, up to 16 KiB of checked metadata and up to 100 findings. It stops with More, Done or Limit according to the saved result. Closing or hiding the enhanced workspace stops its client-side continuation; the PHP batch remains bounded.

Usage inspection reads stored native relationships and text. It does not execute arbitrary shortcodes or builders, inspect every custom field, run theme CSS/JavaScript or search external websites. A report with no findings therefore supports inspection of the checked owners, and is not an instruction to delete the file.

Files with no use found

Files with no use found, in the Library, checks the whole library in one go. It reads saved pages, posts, products and their variations, templates, template parts, navigation menus of the Site Editor, reusable blocks, global styles and Additional CSS, comments and reviews, featured images, product galleries, category pictures, picture addresses in custom fields and page-builder data, the site icon, the logo, the header and background image, WooCommerce's placeholder picture and email header image, and the saved widgets. A file is listed when none of these refer to it by its attachment ID or by any of its file names (the file, each size WordPress made, and a kept original). A gallery shortcode without IDs counts the pictures attached to its page. A Folder gallery block or `[afsd_im_folder_gallery]` shortcode counts every file in the folder it names (and in its sub-folders when the gallery includes them), whether that folder is switched on for website galleries or not.

Select Check the library. Each step works for about eight seconds and saves its place; choose Continue checking until it says the check is finished. The report shows how many content items were read and files compared, lists the first 60 files with no use found (each opens its Quick details, where Check usage looks again for that one file) and can put the whole list into a new collection, named "No use found" with the date and time of the check: select Put these files into a new collection, then Show the collection to see every one in the library. The list keeps the first 5,000 files. Start a new check replaces the report.

Listed does not mean unused. The check does not read theme and plugin files, bare numbers in custom fields (an ID stored without a picture address), CSS files, emails already sent, other websites or links to your files from elsewhere, and a content item or setting larger than its read limit (256 KiB per content item, 64 KiB per custom field or comment) is counted as not read; the report says how many. If a site refers to its files in more places than one check can keep track of (50,000 IDs or 50,000 file names), the check lists nothing rather than a list that could name files in use. Image Manager deletes nothing and changes no file, page or setting; delete a file with WordPress only after checking it yourself.

Optimization setup and public source selection

Open Settings and choose the current defaults before starting a new run. Public-image approval is required before enabling delivery, automatic uploads or automatic public-page rechecks, and before queuing supported optimization. Confirm that the selected images are intended for public use.

Saved setting	Initial value	Effect
Public-image approval	Off	Records the administrator's public-use choice required by publication and automatic processing
Offer optimized images on supported website output	Off	Enables managed alternatives on newly generated supported public markup
Automatically optimize eligible future uploads	Off	Queues future uploaded images after a supported public association exists
Automatically recheck changed public pages and theme updates	Off	Rechecks Home, Shop and known linked public pages after supported changes
Include images supplied by the active theme	On	Includes supported active-theme assets observed on scanned public pages
Include public product images	On	Adds supported public WooCommerce product relationships during a scan
Include local images in approved comments and reviews	On	Adds eligible local image associations from approved comments on public content
Enable enhanced interface and live progress	On	Enables the interactive workspace unless Suite enforces PHP mode

Saved setting	Initial value	Effect
Allow local copies for exactly matched Jetpack image sizes	Off	Allows delivery for a verified local attachment and exactly supported CDN size mapping
Lazy-load pictures that WordPress does not reach	Off	Adds loading="lazy" to pictures a theme or plugin prints outside WordPress's image functions (see Lazy loading)
Largest picture to make lighter	Up to 2,560 × 2,560 pixels	Sets the size and memory limits of compression (see Compression controls)

Source choices take effect in later scanning; saving them does not remove existing prepared files. Changing the defaults applies to newly queued, unprepared work. Existing ready copies and per-image recipes are retained until an explicit comparison or bulk change is applied.

The scanner starts at the public Home page and the configured Shop page and follows one level of allowed links. The page fetch budget is 30 attempts for a scan; product and approved-comment discovery use their own bounded native batches. It sends anonymous read-only page requests, with no signed-in cookies or authorization header. It does not fetch checkout, cart, account, member, password, API or administrative routes. Redirects and network destinations are rechecked.

Supported images come from actual public published post/page/product associations, approved comments attached to public content, and supported active-theme images observed in public output. A private or password-protected parent can veto publication. An unattached upload, arbitrary local file, external image, unmapped direct URL or unknown offload relationship does not become a public source merely because the administrator can see it in the library.

Compression controls

Control	Choices	Initial value
Compression	Lossless, High Quality, Maximum Savings	Lossless
Content hint	Auto, Photo, Graphics	Auto
Effort	Fast, Balanced	Balanced
Output formats	Auto, Native, WebP, AVIF	Auto
Metadata	Preserve, Remove GPS	Preserve

Lossless verifies the accepted output against the source with the appropriate native-data or decoded-pixel check. High Quality and Maximum Savings use photographic lossy output with sampled quality checks. Confirm the photographic pixel-change acknowledgement before choosing a lossy recipe. Graphics and transparency force the lossless fidelity path even when a photographic mode is selected. Auto treats PNG, alpha-bearing files and names suggesting graphics such as logos, screenshots, diagrams or artwork conservatively.

Auto chooses among available native and supported alternative formats. Native requests the source format. WebP and AVIF require a verified encoder for the current recipe. AVIF cannot be selected with Lossless. A missing encoder, unsupported metadata profile or failed quality check keeps the existing source path rather than publishing an incomplete candidate.

Preserve retains supported metadata and color information or holds the file when the round trip cannot be verified. Remove GPS removes supported GPS data; ambiguous XMP or structural metadata requires review and may be held. A JPEG's comment and the print resolution in its header (dots per inch; GD, which makes many of WordPress's sizes, writes both into every JPEG it saves) do not hold a WebP or AVIF copy: the copy leaves them out and the original keeps them. The lossless JPEG optimization (jpegtran, where the server has it) keeps both, as before; a JPEG copy made by GD or Imagick of such a file is still held. Since 1.8.0 the same holds for extra display sizes: a WebP or AVIF display size leaves the two notes out, and a JPEG display copy of such a file is not made. This is separate from title, caption and alt metadata stored by WordPress.

The size limits follow Settings > Largest picture to make lighter:

Choice	Largest file processed	Memory ceiling for one file
Up to 2 megapixels (the limit before 1.7.5)	2 million pixels, 12 MiB	256 MiB
Up to 2,560 × 2,560 pixels (WordPress's own upload limit; the initial value)	6,553,600 pixels, 24 MiB	768 MiB
Up to 8,192 × 8,192 pixels (camera originals)	67,108,864 pixels, 64 MiB	2 GiB

No side may be longer than 8,192 pixels. Before each file, Image Manager asks WordPress for its image memory, as WordPress does when it makes the sizes of an upload (WP_MAX_MEMORY_LIMIT: the server's PHP memory limit when that is more than 256 MB, otherwise 256 MB, unless the site sets its own), and checks that the file fits in the memory left, after a reserve of 32 MiB, up to the ceiling above. A file that does not fit, or that is larger than the choice allows, keeps its native family and its result says why; the smaller sizes WordPress made of it are processed as usual. The Settings tab shows how much memory image work has on this server at that moment and about how large a picture that allows. Larger files take longer to process. To try files that were skipped for

their size after choosing a larger limit, use Retry using each image's saved policy in the Bulk editor (for example on All matching results).

Compression also needs a 128 MiB minimum of free space plus six times the largest file size, and keeps a copy only when it saves at least 5 percent. A candidate that is larger, insufficiently smaller, stale or unverifiable is discarded or held. Original uploaded files remain available.

Start, pause, resume and stop work

In Optimize choose the recipe and scope controls, then select Optimize site images. Scan only creates a scan run and does not request recompression. A new optimization run scans supported public sources and saves its bounded processing selection. A still-active saved image run must be paused, resumed or stopped before another new run is started.

The enhanced interface shows progress and can assist one bounded batch at a time while the workspace is open. The PHP interface exposes the saved status and manual step control. Background processing continues through Action Scheduler when its required APIs are present, otherwise through WP-Cron. The recurring rescue interval is two minutes and queued work can receive a near-term wakeup. A browser-assisted batch is separate from proof that a background callback executed.

Pause records the pause choice and advances the publication generation. It prevents later batches and stale publication but does not forcibly kill an encoder already executing its native bounded step. Resume retains the saved job and continues through the normal worker. Stop / Cancel marks the run cancelled, stops its remaining selection and leaves retained original and prepared files available. It also pauses further image processing. Stopping a run is not the managed-copy cleanup operation.

The progress display identifies the current run and its last durable checkpoint, the work stage, selection and outcome counts, activity and background status. Site-wide inventory counts and current-run counts describe different sets. A slow-stage message is advisory; it does not declare the job dead or silently restart it.

Optimization results and per-image controls

Open Images > Optimization results. Search filenames, choose a result filter and move through pages of 20 indexed sources. The list groups a source with its supported native sizes and managed alternatives. Select an image to view the particular size and output format, dimensions, byte comparison, recipe, reason and delivery observations.

Result label	Meaning and next control
Optimized files ready	Accepted managed files are available; use delivery controls or compare another recipe
No useful saving	No candidate met the useful-saving requirement; retain the source or deliberately compare another eligible recipe
Retained by policy	Resource, metadata, privacy or other native policy retained the source; inspect the reason
Unsupported format or integration	The format or ownership mapping is unsupported; keep native output or use the appropriate native owner
Excluded by you	Explicitly excluded from optimization; Include permits a later reviewed optimization request
Waiting or in progress	Discovered or requested work has not reached a terminal result; follow the saved run
Needs attention	A native processing failure needs inspection; Retry queues an eligible fresh attempt

Optimize requests selected eligible sources. Retry resets the eligible failed source into the native retry path. Exclude records exclusion, cancels its pending request and retains any managed files. Include clears that exclusion and makes existing verified copies or discovery state available. Use original turns that source's delivery switch off; Use optimized turns it on, subject to the global delivery switch and current eligibility. Neither per-image switch enables global delivery.

Remove managed copies is a separate cleanup control. First turn global optimized delivery off, clear pages or CDN caches that can still reference managed URLs, then review the selected cleanup. The native service verifies its own managed paths and attempts removal; original files are retained. A cleanup mismatch or filesystem failure is shown through the owner result rather than treated as successful reclamation.

Comparison and additional display sizes

Open Compare compression on an eligible public source, choose a different recipe, acknowledge photographic pixel changes if applicable and prepare the comparison. This queues a private-to-the-workflow comparison draft while keeping current published delivery unchanged. Wait for the saved draft to complete, then review its sizes and alternatives.

Apply comparison requires a completed draft with a qualifying verified smaller copy, the exact draft identity, unchanged source and metadata, and the current publication policy. Applying adopts the reviewed recipe and publishes its accepted copy family. Former copies are retained for old markup and caches within the native history bounds. Discard removes the active comparison choice; it does not restore arbitrary other settings or delete the source. If the source, recipe, revision or draft files changed, discard and prepare again.

The sizing control separates existing-size compression from Generate useful missing display sizes. Additional widths use supported GD resampling, retain the source aspect ratio, avoid cropping and upscaling, and fill significant native size gaps. Candidate widths are 320, 480, 640, 960, 1280 and 1600 pixels, with at most four useful additions per source. Existing native candidates are reused. Metadata, transparency, unsupported format or insufficient saving can hold a proposed width without removing the native family.

A third choice, **Sizes measured on your pages**, makes the widths the measured pages need instead: up to six per source, where no existing size is within 20 % above a needed width (see Pictures sized to fit the page). A source that was not measured gets no display size from it.

These are resized display copies, not claims that resized pixels equal the original image. Their compression is validated against their same-size resized reference. Public markup still follows its owning layout and supported size relationships.

Bulk optimization and frozen selections

Choose one of Checked images, This page or All matching results. Checked selection is limited to 1000 native source IDs; reviewed AI selection accepts at most 100 natural references per project. This page freezes the current 20-result page. All matching results freezes the current filter/search set, including later pages, rather than repeatedly following a moving list.

Choose Optimize, Retry, Use original, Use optimized, Exclude, Include or Check delivery. Confirm the action and scope. Optimization additionally chooses compression, Apply now or Prepare for review, Existing sizes, Useful missing display sizes or Sizes measured on your pages, and All, Missing or Verify targets. The photographic and resize acknowledgements apply when those effects are selected.

The saved batch records the source IDs once. Later discoveries do not silently join that selection. A source whose permission or eligibility changes can be skipped. The selection records Waiting, Working, Checking, Complete, Review, Skipped or Failed per source, so a prepared review draft is distinct from a published result. Stopping the batch ends unfinished entries while preserving accepted files and histories.

Automatic discovery and future uploads

Automatic upload processing waits for a supported public association; saving an upload alone does not authorize a public optimized copy. Automatic page rechecks are separately optional. Turning them on queues one recheck of Home, Shop and already-known linked public pages. Supported native post/product/comment saves, public-state changes and active-theme update events can request later rechecks. Repeated events for the same page are combined and delayed by 60 seconds to keep saving bounded.

The recheck queue holds up to 64 pages. It advances one due known page through the image worker after explicit runs and scanner work yield. Public scope is checked again after fetching. Newly observed eligible sources in Discovered or Held state can be requested using current defaults. Already-ready, failed, excluded, per-image-original or comparison-draft sources are not silently reset into a new optimization intent.

Transient HTTP/DNS errors use the bounded retry schedule, with at most three attempts. The workspace shows pending pages, retrying pages, the last checked result and recent page issues. Queue full, failed or capped rechecks remain visible. Run a deliberate scan or optimization to inspect a wider selection. A supported save event is needed to trigger this queue; an FTP file edit, direct database edit or unrelated builder/CSS change is not an automatic public-page discovery event.

Public image delivery

Enable global delivery only after the public-use choice is saved and ready results exist. Newly rendered supported native attachment or content image markup can select verified Image Manager alternatives. The plugin does not compress an image in a visitor request and does not replace the original files.

Surface	Current behavior
Native WordPress attachment image HTML	Supported when current source, recipe and public eligibility are valid
Native content images	Supported when their markup and attachment/size mapping are safe
Observed active-theme images	Supported when indexed and native markup is safe
WooCommerce simple-product gallery	Display image and responsive candidates can use managed files; native zoom/lightbox source stays original
WooCommerce classic variable-product gallery	Supported through explicit native variation-attribute coordination; unknown swaps retain native output
WooCommerce blocks or custom galleries	Native output unless their owner integrates the public rendering helper
Existing picture elements, scripted lazy markup or ambiguous candidates	Native output retained
Protected or unknown offloaded media	Native output retained
Jetpack CDN image size	Local alternatives only when the advanced setting and exact supported mapping permit it

Surface	Current behavior
Your own pull CDN	Picture addresses in the uploads folder point to the CDN when Deliver pictures through your CDN is on and its check passed

WordPress, the theme and WooCommerce own layout, image assignment, gallery interactions and native size/crop requests. Image Manager supplies verified alternatives for recognized requests. Turning global delivery off restores original markup on newly generated pages. Clear existing HTML/CDN caches through their native owner before assuming old managed URLs are no longer in use.

Lazy loading

WordPress itself adds `loading="lazy"` to the pictures it builds: attachment images, pictures in content and widgets, avatars and the header image, when they have a width and height. It leaves out the pictures it expects on the first screen (the page header and the first three pictures of the main content). Settings > Lazy-load pictures that WordPress does not reach adds the same attribute to the other pictures of a public page: those a theme, page builder or plugin prints straight into the page without WordPress's image functions. It is off at the start.

A picture gets `loading="lazy"` only when all of these hold: WordPress did not already decide about it in the same page view (its address went through WordPress's own loading choice); it has no `loading` attribute; it has a real address (not a data placeholder) and a numeric width and height; it is not inside the page's header, a `noscript` or a `template`; it has no `fetchpriority="high"`, no lazy-loading script markup such as `data-src`, and not the class `skip-lazy`; and it is not one of the first pictures of the page (WordPress's own number, 3 unless a plugin or theme changes it, counting every picture in page order). Nothing else in the page changes. Feeds, the admin, previews in the Customizer, AMP pages, REST and AJAX replies and pages larger than 2 MB are left alone; a page with more than 20,000 tags is changed only up to that point.

Pictures in CSS backgrounds, videos and frames are not lazy-loaded by this setting. Add the class `skip-lazy` to a picture that must load at once.

Pictures sized to fit the page

A picture fits the page when the browser downloads a file about as wide as the picture is shown. WordPress gives each picture a list of files of different widths (`srcset`) and a note that tells the browser how wide the picture will be (`sizes`); the browser picks the smallest file that covers that width at the screen's pixel density. WordPress's own note says the picture can be as wide as its file (for a 1,024-pixel size: up to 1,024 pixels, or the whole screen on smaller screens), so a picture shown 380 pixels wide in a three-column grid can still download the 1,024-pixel file. For pictures that load later (lazy), WordPress adds `auto` in front, and browsers that know `auto` then pick by the real width; the pictures of the first screen keep WordPress's note. Some themes write their own notes; the AffStackDev theme does for its own pictures.

Image Manager measures instead of guessing:

1. **Measure pages** (Optimize tab, part **Pictures sized to fit the page**). Image Manager opens Home and the public pages its scan found (Home, Shop and the pages they link to), one at a time, in a hidden frame of that screen in your browser. The frame takes every screen width from 320 to 1,920 pixels, every 16 pixels and every single pixel where the layout changes, and Image Manager reads how wide each picture is shown. Scroll bars are hidden while it measures, so the widths are those of a phone and never smaller than a computer's. It takes a few seconds per page; keep the screen open until it reloads with the report. Nothing leaves your site and no visitor is measured. A theme or plugin can add pages to measure with the filter `afsd_im_fit_pages`.
2. **The report** lists, per page, the pictures fitted, the pictures whose theme writes its own note (left as the theme chose) and the pictures left as they are: a picture with only one file to choose from, a WooCommerce product gallery (WooCommerce's scripts choose its files) and a picture whose theme shows different pictures at different screen widths.
3. **Settings > Fit pictures to the page** (off at the start; it needs the "public images" confirmation) then gives each fitted picture on a measured page the measured note, on the picture and on Image Manager's WebP and AVIF sources. Where WordPress put `auto` in front, `auto` stays. A measured note is never smaller than the measured width (widths are rounded up, by at most a few pixels), so a picture never looks blurred because of it.
4. **Make the sizes these pages need** (optional). If none of a picture's files is within 20 % above a width the measured pages need (on phones at 2x and 3x, tablets at 2x, computer screens at 1x and 2x), Image Manager can make one: up to six per picture, never larger than the original and never cropped, with the same checks as its other display sizes. Tick the confirmation and choose the button; it starts one run with your compression defaults (High Quality and Maximum Savings also need their confirmation). The Bulk editor offers the same as the size choice **Sizes measured on your pages**. Like the other display sizes, the new files are offered when optimized delivery is on; a picture that was not measured gets none.

Measure again after a theme change or a new page layout. **Forget the measurements** deletes every saved measurement: pictures go back to the note their theme gives them, and files already made stay. Pages that were not measured, the admin, feeds and the cart, checkout and account pages are never changed. Clear page caches after switching Fit pictures to the page on or off. **Check in browser** (on a picture's card under Images) shows which file the browser then chose.

Delivery through your own CDN

A CDN keeps copies of your files on servers near your visitors. Settings > **Deliver pictures through your CDN** works with a CDN you already have that has an address of its own and fetches the files from your site (a "pull zone", for example a Bunny CDN pull zone such as `https://example.b-cdn.net`, KeyCDN or an Amazon CloudFront distribution). There is no AffStackDev service involved and nothing to sign up for with us. A CDN that sits in front of your whole site at your own address, such as Cloudflare, needs nothing here: your pictures already go through it.

1. In your CDN account, set the CDN's origin to your site's address, and turn off any picture optimization or conversion the CDN offers for this site: Image Manager makes the lighter files itself, and the check below needs the files unchanged.
2. Enter the **CDN address** (https, optionally with a path, nothing after it), tick **Deliver pictures through this CDN** and choose **Check and save**. The "public images" confirmation on the Settings tab must be saved first.
3. Image Manager writes a new small test picture into its own folder, asks for it at the CDN address and compares it byte for byte with the file on your site. Only when it comes back unchanged is delivery through the CDN switched on. The test picture is then deleted. The result and the time of the check stay on the card.

On the public pages Image Manager delivers on, the addresses of the pictures in your uploads folder then point to the CDN: originals, WordPress's sizes and Image Manager's lighter copies, in `src` and `srcset`, lazy-loading data attributes, WooCommerce's gallery data, links to a picture file, preload links and `style` backgrounds. The CDN mirrors your site's paths: `/wp-content/uploads/2026/10/photo.jpg` becomes `https://your-cdn/wp-content/uploads/2026/10/photo.jpg`. Your theme's own files, scripts, styles and fonts, SVG files, WooCommerce download files, and the cart, checkout, account and admin pages keep your site's address.

The check runs again every day. If the CDN no longer returns the file unchanged (it is down, answers with an error or changes the picture), Image Manager switches delivery through the CDN off at once, pictures come from your site again, and the card and Tools > Recent activity say so. Check and save again once the CDN works. To switch it off yourself, untick it and save. After switching on or off, clear your page cache so cached pages follow.

A replaced picture (Replace a file while retaining the attachment ID) keeps its address, so the CDN can show the old picture until its own cache runs out; purge that picture in your CDN account. Delivery through your CDN cannot be used together with Jetpack's image CDN. The check needs your server to reach the CDN address over HTTPS.

Message of the check	What to do
The CDN address did not answer	Check the address; the CDN may be down, or your server may not be allowed to connect to it. Pictures come from your site.
The CDN answered with status ...	The CDN could not fetch the test picture from your site: check that its origin is your site's address.
The CDN returned a different file	Switch off the CDN's own picture optimization or conversion for this site, or check its origin.
Jetpack's image CDN is active	Use one CDN for pictures: switch Jetpack's image CDN off to use your own.
Enter the CDN address as https://cdn.example.com	Use https and the CDN's host name, with an optional path and nothing after it.
That is this site's own address	Enter the CDN's own address; a CDN in front of your whole site needs no setting.

Public-page and browser delivery checks

Check public page fetches a known permitted page anonymously and compares its HTML with the current source/delivery signature. The saved report records what that HTML offers. It can show that a managed URL appears without proving a visitor browser downloaded it.

Check in browser opens the explicit signed-in inspection route for the chosen public page and current signature. The inspection measures real image elements, viewport/display dimensions, device pixel ratio and currentSrc; its file check can compare the selected managed file. Submit or save the observation through the provided native control. A changed source, recipe or delivery choice makes an old observation stale. Open a fresh inspection for the current state.

The browser inspector is a human browser workflow. AI can inspect the saved result or prepare the public HTML check, but cannot supply invented browser measurements as an action. The two results stay separate in the image detail view.

Tools and background processing

Tools shows the native preflight result, uploads/storage readiness, processing memory budget, codec report, background backend and its independent probe, saved activity, scan pages and release health. It also shows sampled Media Manager server timing: total PHP time, attachment query, organization lookup, card rendering, query count and visible/matching media counts. Those timings describe the sampled server request.

Use Check background processing to schedule an independent token-bound probe on the selected backend. Queued means the probe has been saved; only the callback's own execution changes that evidence to executed. If the site relies on WP-Cron, normal visits or the host's real cron runner must cause WordPress cron callbacks to run. Review the host's cron/loopback settings if progress advances only while Optimize is open.

Use Check available image formats to run the actual native codec capability checks and save their result. An installed PHP extension alone does not establish that a particular recipe can produce and verify its output. PDF

rendering, portable ZIP support, encrypted ZIP support, private storage and image compression are separate capabilities.

Download integration contract exports the passive public product descriptor. Download release health report exports current owner health and compatibility observations. Download diagnostics exports the supplied native diagnostic summary. These are authenticated native downloads. They do not sign or publish a plugin release, upload media to AffStackDev, or authorize a reviewed AI mutation.

Local portable backups

Open Backups, choose a profile, scope and destination, and select **Create media backup**. All media means media the current actor can edit, bounded by the attachment upper ID frozen when selection starts. Folder and Collection scope use the selected existing label. The snapshot records media rather than a complete WordPress site.

Backup profile	Contents
Essential recovery	Source files, title, caption, description, alt text, bounded native metadata, edit/source data, organization and optimization preferences
Published media recovery	Essential data plus current native sizes and Image Manager managed-copy files available to the native exporter

The portable format is `affstackdev-image-manager/media`, version 1. It contains a manifest, newline-delimited media records, checksums and standard files identified by portable record IDs. It does not include WordPress source code, site settings, passwords, update keys, orders, customers or the page/product relationships that use the media.

Local snapshots are stored in private host storage outside the web root, selected by the configured `AFSD_IM_BACKUP_DIR` or the native private temporary storage owner. The directory must pass the private-storage checks. Download completed recovery archives to the device or send them to the configured remote destination so the recovery copy is independent of temporary host storage.

If the host's encrypted-ZIP capability check passes, choose Encrypt archive entries with a password, enter at least eight characters and confirm the same password. Keep that password independently; the archive needs it for import and restore. Scheduled encryption uses its separately saved protected password.

The job progresses through selecting, packaging, finalizing and complete. Pause retains it; Resume continues; Cancel stops unfinished processing. One bounded batch advances the current saved job and is not the same as a complete archive. The native exporter checks files while copying and verifies packaged entries before making a completed snapshot downloadable. A single file is bounded to 1 GiB, a media family to 2 GiB, and the archive to 100000 media records and 500000 entries.

Completed local snapshots show their scope, profile, count, size, encryption and integrity information. Inspect records to select a particular portable media item. Download to this device streams the verified local ZIP through the authenticated native form. Send to the connected provider queues its remote transfer. Delete local copy requires confirmation and removes that archive and its local index, not the Media Library items.

Import and restore media

Bring in a portable backup accepts a local ZIP from this format. Supply its password when encrypted, then Import backup. The native validator checks format identity, record shape, bounded entry count, safe entry names, duplicate/case-colliding paths, link/special entries and checksums before accepting local recovery data. It does not extract arbitrary ZIP paths into the installation.

Inspect the imported snapshot and choose a specific portable record. **As a new Media Library item** creates a local attachment ID and rebuilds native metadata. **In place on this same-site attachment** appears only for a matching existing permitted attachment on the same recorded site identity, with the matching MIME family. In-place restoration creates a safety ZIP before replacing the current source state. A snapshot from another site can be restored as new media, with its portable IDs mapped to new local IDs.

Restore preserves the record's text metadata, organization and supported edit/optimization preferences. Folder hierarchy and collections are matched or created by their organization names on the destination. Original files are restored, native sizes are rebuilt by the owning WordPress service, and old derivatives are not blindly adopted as current published delivery. Restore does not reassign pages or products to a newly created attachment; make that selection through the content owner. An encrypted record needs the archive password for its source extraction.

Remote backup settings

Remote backup is optional. Select Local / download only, Google Drive or S3-compatible storage. The configured provider receives the portable archive directly from this WordPress installation. The product does not route the archive through AffStackDev storage.

Control	Choices / initial value	Behavior
Destination	Local / download only initially; Google Drive; S3	One current remote destination
Automatic schedule	Manual initially; Daily; Weekly	Creates and uploads the selected native snapshot on the saved schedule
Scheduled profile	Essential initially; Published	Profile used for scheduled snapshots
Scheduled scope	All initially; Folder; Collection	Selects a saved organization label when applicable
Remote retention	1–30, initially 3	Prunes old completed copies in the current destination only after a new verified upload
Keep local snapshot	On initially	Off permits removal of that completed local archive only after verified remote transfer
Password-encrypt scheduled archives	Off initially	Needs encrypted-ZIP support and a saved matching password

Control	Choices / initial value	Behavior
Set / change scheduled password	Blank retains saved password	New password needs at least eight characters and exact confirmation

Save settings before connecting or testing. Blank credential inputs retain existing encrypted credentials. A changed destination identity can invalidate connection readiness; test or reconnect the current owner. Changing the website salts can make protected stored credentials or automatic passwords unreadable, requiring secure re-entry.

Google Drive

Provide a Google OAuth web-app client ID and client secret, or have the administrator configure the supported server constants. Register the exact redirect URI shown in Backups: the site's `wp-admin/admin-post.php?action=afsd_im_google_callback`. Save Google as the destination, then select Connect / reconnect Google Drive and complete Google's consent screen with the account that should own the archive folder.

The connector requests `drive.file` and creates its own folder. Its callback uses the saved actor/site and expiring state. A successful connection stores the protected refresh credential and folder identity. Changing client credentials invalidates the old connected state. Disconnect Google Drive clears this site's saved connection credential and folder choice; it does not delete stored Google archive objects or perform Google's account-wide permission revocation.

Upload uses Google's resumable session in bounded 5 MiB chunks. It follows the confirmed offset when a transient response is ambiguous, then verifies provider size and MD5 before recording a verified completed copy. Pause retains the current session. Resume continues the saved transfer rather than starting an unrelated archive.

S3-compatible storage

Enter an HTTPS endpoint hostname, region, bucket, private prefix, access key ID and secret key. Server constants can provide the keys. The endpoint must not contain a username, password, path, query, fragment, localhost name or literal IP address. Region is a 2–32 character lowercase region label; an invalid region is normalized to `us-east-1`. Bucket uses the bounded native lowercase bucket syntax. Prefix is normalized to safe segments and a trailing slash.

Use path-style bucket addressing is initially on. Turn it off only when the provider uses virtual-hosted bucket URLs. Save the current settings, then select Test S3 connection. The native test writes a small probe, reads its length/hash metadata and confirms removal of the probe before marking the connection ready. A failed cleanup remains a failed connection test.

The account needs the provider's permissions for object writes, reads, multipart initialization/completion/abort and removal in the selected bucket/prefix. Image Manager uses Signature V4, 5 MiB multipart parts, saved ETags, completion, HEAD metadata verification and a full private readback of the uploaded archive's SHA-256 and length. Only then is the remote copy verified.

Cancel stops the local transfer. When an S3 multipart upload exists, it attempts the provider's abort and records Confirmed or Unconfirmed; a missing upload is Not required. An unconfirmed abort is not reported as a verified reviewed cancellation. Inspect the provider and saved native transfer before approving another recovery step.

Remote transfer, retention and recovery

Create a backup with the ready remote destination or select Send to provider on an existing completed local snapshot. Follow the remote job's initialization, upload, completion, verification and final result. Pause, Resume and Cancel act on the saved job. Transient failures use up to three retries with the native 60/300/900-second

schedule; permanent failures remain visible. Image work, active local backup work and active remote work have their own priorities and coordination.

Daily and Weekly schedules depend on the host running WordPress's scheduled callbacks. The first scheduled check is delayed by five minutes; busy image or backup work can defer creation by fifteen minutes. Scheduled password encryption uses the saved protected password, without interactive entry during the job.

Retention never prunes a known-good archive because an upload failed. After a verified new upload, only older completed entries for the current destination fingerprint are eligible. Keep local off removes the completed local snapshot after verification, not attachments or source files. The remote history can retain entries for old destination fingerprints; reconnect the corresponding owner before those entries become manageable here.

For a manageable remote copy, Download to this device uses the native authenticated download. Bring into local recovery downloads it privately, verifies it and imports its portable records; encrypted copies require their password. Inspect the local imported records and restore selected items. Delete remote copy requires confirmation, accepts the provider's successful/already-absent removal response and removes the local remote-history entry. This affects the backup object, not live media.

Optional protected software updates

Optional: Image Manager works fully without a license. A free license from your AffStackDev account adds signed updates from affstackdev.com and counts for the full mode of AffStackDev Suite on this site (its enhanced screens and layout choices). Everything you set up keeps working whatever happens to the license. The License & Updates tab says so at its top; no other screen shows a license notice, and nothing is locked. The native license panel manages the separate protected software-update credential for product `affstackdev-image-manager`.

Enter the root update key in the native key field and select Activate. The client binds its request to this product, installation ID, canonical site URL, environment and activation type. A successful activation stores a protected installation token rather than retaining the root key as the ordinary validation credential. Validate now refreshes the server's accepted current status. Check for updates asks the same update owner for an authorized current software release; it does not install that release merely because it was checked.

Active and server-issued Grace states are accepted only within their server-bound deadlines and features. A transport interruption retains previously accepted permission only until its existing deadline; it does not create new grace. Invalid, expired, blocked, mismatched or unavailable responses show their native result. The panel shows the current site identity, credential readiness, validation state, error and accepted update status as available.

Deactivate license explicitly asks the pinned license owner to release this installation before clearing its local credential. Forget on this site clears this site's local credential and cached authorization without releasing a remote slot; it retains the site identity. Establish new site identity is the recovery for a copied installation whose stored identity belongs to another site; it creates a separate installation identity and clears copied credentials without touching the original site's remote activation. The identity also records the WordPress environment type, so changing `WP_ENVIRONMENT_TYPE` (for example when a staging copy becomes the live site) counts as a new site: deactivate the license first while the identity still matches, then establish the new site identity after the change and activate again, so the released place is free. Use the corresponding native panel confirmation rather than treating these three actions as interchangeable.

Protected updates verify the release signature, product/slug/file identity, stable channel, supported requirements, monotonic version, archive size/hash and exact runtime file inventory. A modified signed

installation cannot silently be overwritten by the protected updater. WordPress's own update and auto-update choices remain the native software-update owner. Deactivating the plugin retains its credential and remote activation unless the explicit license deactivation action has completed.

Complete reviewed AI workflow

AffStackDev AI 0.1.0.12 discovers this installed product's passive capability manifest, current manuals and 81 trusted native action definitions. The passive documents explain behavior; only the registered native definitions can execute a reviewed action. The connected WordPress actor must currently be an administrator with media capability, and the connection needs Read, Propose and Apply scopes for their corresponding operations.

Ask to inspect settings, jobs, sources, library, drafts, backups, remote configuration, update state, host readiness, folder imports, the files with no use found or a saved intent. Inspection and preparation do not change native business state. Use bounded pages when examining large sets. Names can identify a source, media item, folder, collection or snapshot. If several permitted items match, select from the returned candidates; an exact #ID can identify an attachment or organization label, and source : ID identifies an indexed source.

Ask for a specific change and review the exact prepared project in WordPress's AffStackDev AI screen. It lists the resolved target, current values, proposed values, file/secret readiness and material effects such as publication, a new attachment, a recovery snapshot, a scheduled job, remote upload, retention or removal. Only the native human review can approve that project. Explicit Apply then performs that exact approved revision through Image Manager's native owning service.

Changing the selected file, metadata, membership, settings, draft, job, provider version, schema, active dependency or actor permission invalidates old approval. Prepare a fresh review after resolving the current state. A job-start receipt confirms the saved queued job; use job inspection and separately approved bounded advancement to follow processing. Completed encoding, remote integrity and browser download observations have their respective native evidence.

Stage files and secrets privately

Open Media > Image Manager AI inputs using the same WordPress account. Choose the purpose, supply the corresponding native file or secret, then Stage secure input. The success page returns a 32-character reference. Give the assistant that reference; the actual secret value or local file path stays out of the project parameters.

Purpose	Human input	Reference lifetime
Media upload	One to eight WordPress-allowed files, each within the host limit and 64 MiB staging bound	24 hours
Replacement image	One supported image of the appropriate MIME type	24 hours
Portable backup ZIP	One valid ZIP within the host limit and 2 GiB staging bound	24 hours
Backup password	At least eight characters, up to 1024 bytes, exact confirmation	10 minutes
Cloud credentials / scheduled password	The supplied Google/S3 secrets; optional scheduled password with matching confirmation; each field at most 1024 bytes	10 minutes
Image Manager update key	Nonempty native root key, at most 128 bytes	10 minutes

The reference is bound to that user, site, purpose and file hashes. Staging itself does not publish media, connect an account or change settings. It becomes consumed by one exact approved execution marker. A new intent needs a fresh reference once the prior one has been consumed. Expired private staging is removed by the bounded cleanup callback. Downloads, Google account consent and actual browser-delivery measurements remain their native human controls.

Recovery of an uncertain reviewed operation

An interrupted apply can be Unknown. Open that exact project, inspect its native evidence and request Reconcile. Reconciliation reads the saved execution identity and authoritative owner result; it never runs the operation again. A completed durable receipt remains historical evidence even after later native edits.

If the exact recorded native result can be confirmed, the shared project records its verified result. If it cannot, the outcome remains unknown and the target reservation prevents a different automatic write from hiding the uncertainty. Use the applicable separately reviewed native replacement recovery, cancellation, draft discard or credential recovery when that is the intended repair.

After inspecting an unresolved outcome, the administrator can explicitly review Release an uncertain target reservation for its exact old marker. The old consumer project must already be Unknown and belong to the same actor/site. The release changes only the native reservation record; it does not confirm, repeat, undo or erase the old business operation. The old outcome stays unknown. Any subsequent business change requires a new separately approved project.

Deactivation, uninstall and managed-copy cleanup

Deactivation stops Image Manager's own scheduled workers, backup schedule, remote schedule, license validation and secure-input cleanup. It turns optimized delivery, automatic uploads, automatic discovery and delivery through your CDN off (the CDN address stays; check and save again after reactivating) and stops the daily CDN check, while retaining original files, managed alternatives, media organization, backup/recovery data and protected credentials. The retained files allow cached pages with old managed URLs to keep working. If a batch is still running fifteen seconds after you deactivate, the scheduled work stops but the delivery, upload and discovery switches keep their current values. Review them in Settings after reactivating.

Uninstall clears the plugin's worker callbacks but retains owned media records, organization, settings, original media, generated copies, snapshots and credentials. There is no automatic delete-everything choice in this release. To remove selected managed image copies, reactivate the owner, keep public delivery off, clear referencing caches and use its explicit selected cleanup operation. WordPress's native media deletion is a separate attachment-owner action.

Troubleshooting by saved state

Situation	Native control and resolution
Library has media but optimization results are empty	Run Scan only; inspect the public associations and source settings
Public approval is missing	Review the intended public images, save the approval and prepare/start the desired operation again
Ready files appear but public output remains original	Check global delivery, per-source delivery, current eligibility and the supported markup/integration; run the separate delivery checks
Work advances only while the workspace is open	Run the independent background probe and inspect host cron/Action Scheduler callbacks
Processing paused	Read the native pause reason and Resume after its storage, permission or tracking condition is resolved
Native image held by resource or metadata policy	Inspect the per-size reason; retain that original or deliberately choose an eligible new recipe
Requested codec unavailable	Recheck formats in Tools and choose a recipe supported by the actual host
Old comparison, edit or replacement draft rejected	Discard that draft and prepare against the current source

Situation	Native control and resolution
Interrupted replacement	Use the matching native journal Recovery/Undo controls and inspect its safety snapshot
Private backup storage unavailable	Configure the native private directory outside the web root with the required ownership/permissions
ZIP encryption unavailable	Use the ordinary portable archive or enable the host's supported AES ZIP capability; encryption and ordinary ZIP are separate checks
Cloud entry cannot be managed	Reconnect its recorded destination fingerprint before fetching, deleting or downloading it here
S3 test cleanup or multipart abort unconfirmed	Inspect the native error and provider object/upload; prepare an explicit recovery once the outcome is understood
Review exceeds the shared row/state limit	Select a smaller media group or separately review individual large field changes
AI project became stale	Inspect current values, prepare a new exact project and approve that new revision

Situation	Native control and resolution
AI project unknown	Reconcile the same project read-only; inspect native evidence before a separately reviewed recovery or reservation release
Update credential identity belongs to a cloned site	Establish the new site identity through the native license recovery control, then activate this installation
A list has no folder bar	Tick that list under Settings > Folders for pages and posts; Authors see the bar of Posts only
Fit pictures to the page changes nothing on a page	Measure that page (Optimize tab, Pictures sized to fit the page), switch the setting on and clear the page cache; pictures whose theme writes its own note stay as the theme chose
Pictures still come from your site with the CDN on	Read the card's last check (the daily check may have switched it off) and clear the page cache; cart, checkout, account and admin pages always use your site's address

Local command-line controls

Administrators with a native WP-CLI environment can use the following product commands. They use the saved owner state and do not create a new public unauthenticated endpoint.

```
wp affstackdev-image-manager contract
wp affstackdev-image-manager status
wp affstackdev-image-manager work --batches=1
wp affstackdev-image-manager test-background
```

Contract prints the passive Free descriptor. Status prints saved workflow state. Work advances already-authorized native work in 1–20 bounded batches, defaulting to one, and refuses to override Pause. Test-background queues the independent scheduler probe and prints its saved status. File assignment, licenses, remote account consent and review approval keep their native owner controls.

Glossary

Term	Meaning
Attachment	Native WordPress media record and local ID
Source	Site-local Image Manager index record for a supported attachment or observed theme asset
Managed alternative	A verified separately stored optimized or display-size file
Recipe	The explicit compression or edit settings for an operation
Draft	Prepared comparison, edit, PDF or replacement state awaiting its separate apply/save
Folder / collection	Virtual organization membership, independent of the stored file path
Snapshot	Portable recovery archive with checksums and media records
Portable ID	Backup-record identity mapped to the destination site's native attachment ID
Receipt	Exact approved-operation result retained by the owning review engine and Image Manager
Reconcile	Read the exact execution evidence without repeating a business write

Term	Meaning
Reservation	Native target fence preventing another reviewed operation from obscuring an uncertain result
Folder of an admin list	Private sorting label of one admin list (Posts, Pages, Products ...), set under Folders for pages and posts
Measured note (sizes)	The width a picture is shown at on a measured page, written for the browser so it picks a file of about that width
Pull CDN	A CDN with an address of its own that fetches each file from your site the first time it is asked for it

Complete AI action index

Every row is an installed current action, with its exact native owner effect. Parameter details appear in the technical reference. Files, secrets and external consent use the secure/native controls described above. Read state uses bounded inspection; every mutation uses a separately approved exact project.

All screens actions

Action	What it does
<code>image_manager.inspect</code>	Read current permitted state and readiness. No media, settings or job changes.

Settings actions

Action	What it does
<code>image_manager.settings_update</code>	Save the exact changed settings; advance publication generation. Delivery and lazy loading can change newly rendered public output; automatic processing can schedule future work; Largest picture to make lighter changes which originals later compression reads; Fit pictures to the page changes the sizes attribute of measured pictures on measured public pages; Folders for pages and posts adds private folders to the chosen admin lists (nothing public).

Optimize actions

Action	What it does
<code>image_manager.scan_start</code>	Start a bounded public site scan and save progress. Public page HTTP requests and local source indexing follow; scan-only leaves delivery switches unchanged.
<code>image_manager.optimization_start</code>	Scan the enabled public sources, freeze the selected images and queue the exact compression/size recipe. Apply mode publishes validated copies; review mode leaves comparisons for later approval. Originals remain unchanged.

Optimize / Tools actions

Action	What it does
<code>image_manager.workflow_pause</code>	Pause before the next image operation and invalidate stale publication; prepared copies remain.
<code>image_manager.workflow_resume</code>	Resume saved image work and scheduling.
<code>image_manager.workflow_cancel</code>	Cancel the saved run and hold queued work; prepared copies and original files remain.
<code>image_manager.workflow_advance</code>	Perform one native bounded batch; local files, indexing and public page checks can advance. Saved work and pause state govern eligibility.

Tools actions

Action	What it does
<code>image_manager.background_check</code>	Queue an independent native scheduling probe. Queued is distinct from independently executed.
<code>image_manager.codec_check</code>	Run native local encoder self-checks and save host capability results.

Images / Compare compression actions

Action	What it does
<code>image_manager.comparison_prepare</code>	Queue comparison files without changing delivery.
<code>image_manager.comparison_apply</code>	Publish the exact ready copies after source and output checksum validation; delivery switches remain.
<code>image_manager.comparison_discard</code>	Retire the selected candidate files while retaining approved copies and originals.

Images / selected actions actions

Action	What it does
<code>image_manager.source_optimize</code>	Queue image optimization through the source repository; originals are retained.
<code>image_manager.source_retry</code>	Retry saved image policy through the source repository; originals are retained.
<code>image_manager.source_exclude</code>	Exclude images through the source repository; originals are retained.
<code>image_manager.source_include</code>	Include images again through the source repository; originals are retained.
<code>image_manager.source_original</code>	Disable each selected source delivery switch. Cached pages retain their prior output until cleared.
<code>image_manager.source_optimized</code>	Enable each source delivery switch; global delivery and public eligibility still govern output.
<code>image_manager.source_cleanup</code>	Delete managed alternatives only with global delivery disabled. Clear page/CDN caches before removing files; originals are retained.

Images / Bulk editor actions

Action	What it does
<code>image_manager.bulk_run</code>	Freeze the native selected/page/all-matching source set and queue the chosen operation. The approved selection fingerprint is checked again under the owner lock; later discoveries are not silently added.

Media / Organization actions

Action	What it does
<code>image_manager.folder_create</code>	Create a virtual organization label. File paths, URLs and media IDs remain.
<code>image_manager.folder_rename</code>	Rename a virtual label.
<code>image_manager.folder_delete</code>	Delete the organization label and its relationships; attachments and files remain. Child folders follow native WordPress term deletion behavior.
<code>image_manager.collection_create</code>	Create a virtual organization label. File paths, URLs and media IDs remain.
<code>image_manager.collection_rename</code>	Rename a virtual label.
<code>image_manager.collection_delete</code>	Delete the organization label and its relationships; attachments and files remain. Child folders follow native WordPress term deletion behavior.

Media / selection and native attachment fields actions

Action	What it does
<code>image_manager.folder_add</code>	Change the exact virtual membership; set replaces the selected taxonomy memberships and an empty set clears them. Paths and attachment IDs remain.
<code>image_manager.folder_remove</code>	Change the exact virtual membership; set replaces the selected taxonomy memberships and an empty set clears them. Paths and attachment IDs remain.
<code>image_manager.folder_set</code>	Change the exact virtual membership; set replaces the selected taxonomy memberships and an empty set clears them. Paths and attachment IDs remain.
<code>image_manager.collection_add</code>	Change the exact virtual membership; set replaces the selected taxonomy memberships and an empty set clears them. Paths and attachment IDs remain.
<code>image_manager.collection_remove</code>	Change the exact virtual membership; set replaces the selected taxonomy memberships and an empty set clears them. Paths and attachment IDs remain.
<code>image_manager.collection_set</code>	Change the exact virtual membership; set replaces the selected taxonomy memberships and an empty set clears them. Paths and attachment IDs remain.

Media / Edit selected details actions

Action	What it does
<code>image_manager.metadata_update</code>	Save supplied title, caption and image alternative text, including explicit empty values. Omitted fields remain. Existing page-specific alternative text is not rewritten; native WordPress hooks run.

Media / Upload to this folder actions

Action	What it does
<code>image_manager.media_upload</code>	Create WordPress attachments from one to eight human-staged files, generate native metadata and assign the approved folder. Normal upload hooks and enabled future-upload processing may run. New images without an explicit folder receive their automatic Website images category. Explicitly approved folders take precedence.

Media / Editor actions

Action	What it does
<code>image_manager.edit_prepare</code>	Render a private draft using rotation, flip, crop then bounded downsize. Replace the prior draft; the original and public uses remain unchanged.
<code>image_manager.edit_save</code>	Create a separate attachment with inherited text and organization, native sizes and edit lineage; remove the private draft.
<code>image_manager.edit_discard</code>	Delete only the selected private edit draft.

Media / Convert actions

Action	What it does
<code>image_manager.pdf_prepare</code>	Render one selected PDF page locally to a private white-background PNG draft. Host Imagick/PDF policy governs readiness; PDF remains unchanged.
<code>image_manager.pdf_save</code>	Create one PNG attachment with native sizes and organization from the exact draft; the source PDF remains unchanged.
<code>image_manager.pdf_discard</code>	Delete the private page conversion preview.

Media / Replace actions

Action	What it does
<code>image_manager.replace_prepare</code>	Stage the human-supplied same-type image as a private replacement draft after source validation; the live attachment remains unchanged.
<code>image_manager.replace_apply</code>	Create a verified recovery ZIP and new native file family, then point the same attachment ID at the new files. New URLs affect attachment-based uses; hardcoded URLs and caches require separate review.
<code>image_manager.replace_undo</code>	Restore the latest verified previous file family and metadata. The replacement files are retained; hardcoded content URLs are not rewritten.
<code>image_manager.replace_recover</code>	Explicitly recover a partially applied replacement to the verified before family. This is a new reviewed compensating intent; automatic reconciliation never invokes it.
<code>image_manager.replace_discard</code>	Delete only the selected private replacement draft; applying drafts cannot be discarded.

Backups / Create actions

Action	What it does
<code>image_manager.backup_start</code>	Freeze eligible attachment IDs and package portable media, text, organization and checksums. Published profile includes approved derivative files. Remote destination queues a transfer after verification; retention and local-copy settings can remove older backup copies.

Backups actions

Action	What it does
<code>image_manager.backup_pause</code>	Pause the current backup while retaining saved staging.
<code>image_manager.backup_resume</code>	Resume the saved backup.
<code>image_manager.backup_cancel</code>	Stop the incomplete job and remove its staging; existing media and completed snapshots remain.
<code>image_manager.backup_advance</code>	Perform one bounded native backup batch and verify completed archive entries; completion may queue the configured remote transfer.

Backups / Local history actions

Action	What it does
<code>image_manager.backup_delete</code>	Delete that local archive and sidecars plus its index record; attachments remain.
<code>image_manager.remote_upload</code>	Queue a transfer of the verified local ZIP to the current ready destination. Transfer completion, retention pruning and configured local-copy removal happen through the native worker.

Backups / Inspect and restore actions

Action	What it does
<code>image_manager.backup_restore</code>	Restore the exact verified original, native metadata and organization. New mode allocates a new attachment; same-site in-place creates a safety recovery point before changing the editable existing attachment. Page/product relationships are not rewritten.

Backups / Bring in a backup actions

Action	What it does
<code>image_manager.backup_import</code>	Validate archive names, manifest, counts and checksums, then add a private local snapshot for inspection. This action does not restore attachments.

Backups / Backup destination actions

Action	What it does
<code>image_manager.remote_settings_update</code>	Preserve omitted settings and credentials; save supplied fields and reschedule. Changing Google app identity invalidates its saved connection; changing S3 identity invalidates its verification. Retention prunes older destination copies after future verified uploads; keep-local false removes completed local copies after verified transfers.

Backups / Google Drive actions

Action	What it does
<code>image_manager.remote_google_disconnect</code>	Clear the local Drive refresh token, folder identity and connection time; remote files remain.

Backups / S3 connection actions

Action	What it does
<code>image_manager.remote_s3_test</code>	PUT a temporary private object, HEAD its size/checksum and attempt its deletion. Record readiness only after the owner check; report incomplete cleanup instead of verified success.

Backups / Remote transfer actions

Action	What it does
<code>image_manager.remote_pause</code>	Pause saved transfer progress.
<code>image_manager.remote_resume</code>	Resume the saved transfer and schedule.
<code>image_manager.remote_cancel</code>	Cancel the transfer and attempt S3 multipart abort when relevant; local media and verified remote snapshots remain.
<code>image_manager.remote_advance</code>	Perform the native bounded transfer batch; can upload chunks, verify the remote object, prune retention and delete the local copy when configured.

Backups / Remote history actions

Action	What it does
<code>image_manager.remote_fetch</code>	Download and checksum-verify the same destination object before importing it privately for local inspection; no attachment restore.
<code>image_manager.remote_delete</code>	Delete the exact current-destination backup object and remove its native index record after provider acceptance.

Media / Where is this used actions

Action	What it does
<code>image_manager.usage_start</code>	Check at most 40 saved content items and save the per-actor report. Supported references only, up to 100 findings; native content permissions apply.
<code>image_manager.usage_continue</code>	Check at most 40 saved content items and save the per-actor report. Supported references only, up to 100 findings; native content permissions apply.

Images / Compare / Delivery actions

Action	What it does
<code>image_manager.delivery_check</code>	Fetch mapped public page HTML and record the native signature-bound delivery result. HTML matches do not prove the browser downloaded a derivative.

License & Updates actions

Action	What it does
<code>image_manager.license_activate</code>	Submit the securely staged Image Manager key and installation identity to the pinned license server; store only encrypted installation credentials on accepted activation.
<code>image_manager.license_validate</code>	Perform the native authenticated validation or pending-request recovery against the pinned license server; cache deadlines govern protected updates.
<code>image_manager.license_deactivate</code>	Request release of the remote update activation and clear credentials only after accepted native response. Media features remain free.
<code>image_manager.license_forget</code>	Clear local credentials and accepted update state while retaining the installation ID; this does not release a remote activation.
<code>image_manager.license_new_site</code>	Clear copied credentials and establish a new site/environment installation identity; original remote activation remains.
<code>image_manager.license_updates</code>	Request the signed release catalog using the native installation credential and WordPress update checks; this does not install a release.

Media / Library / Website galleries actions

Action	What it does
<code>image_manager.folder_gallery_show</code>	Turn this folder on for website galleries: a Folder gallery block or <code>[afsd_im_folder_gallery]</code> shortcode that names it then shows the pictures directly in it to visitors on public pages. Files, folders and the other folders' setting stay as they are.
<code>image_manager.folder_gallery_hide</code>	Turn this folder off for website galleries: a gallery that names it shows visitors nothing (people who may upload files see a short note in its place). Files and folders stay as they are.

Media / Library / Import folders actions

Action	What it does
<code>image_manager.folder_import_start</code>	Make an Image Manager folder for each folder of the chosen media folder plugin (parents first; folders an earlier import made are used again), then put its files into them for about 8 seconds and save the position. Files are only added to folders, never taken out of one. The other plugin's folders and tables are only read.
<code>image_manager.folder_import_continue</code>	Continue the saved import from that plugin for about 8 seconds and save the position. Files are only added to folders; the other plugin's folders and tables are only read.

Media / Library / Files with no use found actions

Action	What it does
<code>image_manager.unused_check_start</code>	Start (or start again) the library-wide check: read saved content, comments, site settings and the Media Library for about 8 seconds and save the position and report. Reads only: no file, post or folder changes. A listed file is one that none of the places checked refers to.
<code>image_manager.unused_check_continue</code>	Continue the saved check for about 8 seconds and save the position and report. Reads only: no file, post or folder changes.
<code>image_manager.unused_collect</code>	Put the files the finished check listed into a new collection named No use found with the check's date (or keep filling that collection) so they can be looked at in the library. Nothing is deleted, moved or edited.

Admin lists / folder bar actions

Action	What it does
<code>image_manager.content_folder_create</code>	Create a private folder in the folder bar of that admin list (Folders for pages and posts must be on for it), at the top or inside a folder. Nothing changes on the website and no address changes.
<code>image_manager.content_folder_rename</code>	Rename the folder. Its items and sub-folders stay in it.
<code>image_manager.content_folder_nest</code>	Move the folder, with its items and sub-folders, under another folder (an empty parent: the top level). A folder never goes into itself or one of its own sub-folders.
<code>image_manager.content_folder_delete</code>	Delete the folder only: its items stay as they are, outside any folder, and its sub-folders move up one level. No post, page or product is changed or deleted.
<code>image_manager.content_folder_move</code>	Put each item (by ID or exact title) into the folder, taking it out of its previous folder; an empty folder takes them out of their folders. Only items you may edit. Nothing changes on the website.

Optimize / Sizes for the page actions

Action	What it does
<code>image_manager.fit_sizes</code>	Start one run over the measured pictures with your compression defaults: up to six extra display sizes per picture where no existing size is within 20 % above a width the measured pages need, never larger than the original and never cropped. Originals stay unchanged.
<code>image_manager.fit_forget</code>	Delete every saved page measurement and measured width. Pictures go back to the sizes their theme gives them; display sizes already made stay.

Measuring pages runs in your browser on the Optimize tab; it is not an AI action.

Settings / Deliver pictures through your CDN actions

Action	What it does
<code>image_manager.cdn_check</code>	Save the CDN address. With deliver on, first write a new small test picture in Image Manager's folder, ask for it through the CDN address (one request to that address) and switch delivery through the CDN on only when it comes back unchanged; then pictures in the uploads folder on public pages point to the CDN and the check runs daily. With deliver off, pictures come from the site.

AffStackDev AI / Native recovery review actions

Action	What it does
<code>image_manager.intent_release</code>	Release this exact uncertain operation's owner reservations after inspecting its native evidence. The prior outcome remains unknown; this never replays, undoes or confirms its business effects. Future changes need a fresh separate approved project. An executing consumer project cannot be released.