AI DESKTOP & OS OPS • FILE CLEANUP FIELD GUIDE

AI File System Cleanup & Digital Declutter

Reclaim chaotic `Downloads/` and Desktop folders without ever losing an irreplaceable file—combining dry-run manifests, SHA-256 duplicate sweeps, VLM/OCR receipt renaming, 30-day quarantine staging, and native Mac + Windows automation.

Dry-Run Manifests | SHA-256 Dedup Sweeps | VLM / OCR PDF Naming | 30-Day Quarantine Gate | Mac Hazel vs. Windows | Cowork & Gemini CLI



Plan Before Moving a Byte

Generate readable dry-run manifests (`manifest.csv` or `-WhatIf`) with strict `node_modules` and `.DS_Store` exclusions before touching live folders.

Verify Exact & Visual Clones

Pair `rmlint`, `fdupes`, and Python `hashlib` SHA-256 checksums with VLM/OCR metadata to name scans cleanly by vendor and date.

30-Day Quarantine Shield

Stage redundant installers and clutter in `_quarantine_30d/` with an undo ledger across macOS `Hazel`, Windows `PowerRename`, Cowork, and Gemini CLI.

Why Manifest-First Cleanup Beats Blind Auto-Delete

Pointing an over-eager script or AI assistant straight at a cluttered `Downloads/` or `Desktop/` folder is the fastest way to break local Git repositories, corrupt half-finished browser transfers, or erase tax receipts. The safest approach treats cleanup like a disciplined engineering deployment: scan read-only, exclude active code trees, verify SHA-256 digests, review a dry-run manifest, and stage removals inside a 30-day quarantine folder.

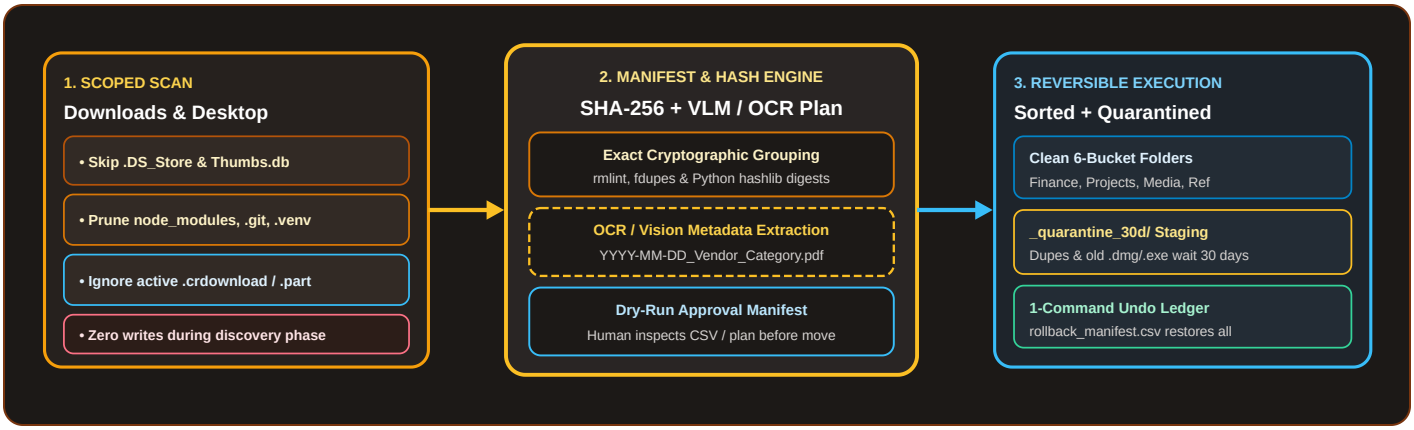
Quick Navigation

1. Mental Model: Dry-Run Pipeline p. 2
2. Exclusions, Setup & Tool Matrix p. 3
3. Annotated Cleanup Cockpit Tour p. 4
4. SHA-256 Sweeps & Dedup Engines p. 5
5. 5-Part Safe Cleanup Formula p. 6
6. VLM & OCR Receipt Renaming p. 7
7. Mac vs. Windows OS Automation p. 8
8. 5 Copy-Paste Cleanup Recipes p. 9
9. Quarantine, Privacy & Rollback p. 10
10. Cross-Tool Declutter Pipeline p. 11
11. Troubleshooting & Pre-Flight p. 12
Golden Rule: Manifest First, Purge Last All

The Golden Rule: Never Rename or Delete in One Hop

Humans create clutter incrementally, so your cleanup workflow should unwind it in reversible layers. Separate **discovery** (hashing bytes and reading PDFs) from **decision** (reviewing your proposed move table) and **execution** (routing keepers to clean buckets while parking discards inside `_quarantine_30d/`).

Architecture: Read-Only Discovery to Reversible Staging



Exclude Dev & OS Trees Before Measuring Anything

Even a modest developer laptop hides 120,000+ identical tiny files inside `node_modules/`, `.venv/`, and `.DS_Store` caches. Always apply a hard ignore list on step zero so your cleanup manifest focuses strictly on real documents, installers, screenshots, and archives.

Safety Exclusions, Hash Tools & Cleanup Matrix

Before invoking a single rename rule or AI planner, lock down what must never be touched and choose the right blend of fast deterministic CLI engines and vision-capable AI helpers for your operating system.

1 Mandatory Exclusion Rules (Never Touch)

- **OS Metadata Files:** Ignore `.DS_Store`, `._*` AppleDouble forks, `Thumbs.db`, and `desktop.ini` so duplicate sweeps never choke on folder-view clutter.
- **Code & Build Trees:** Prune `node_modules/`, `.git/`, `.venv/`, `__pycache__/`, `target/`, `dist/`, and `build/`—identical boilerplate files inside dependencies are structural, not waste.
- **Active Transfers & Stubs:** Skip `*.crdownload`, `*.part`, `*.download`, and unhydrated `.icloud` placeholders unless age exceeds 24 hours.

2 60-Second Cross-Platform Workbench

- **macOS CLI Stack:** Install `brew install rmlint fdupes ocrmypdf`; pair with built-in Spotlight `mdfind`, Finder Folder Actions, and optional **Hazel 6**.
- **Windows 11 Stack:** Install `winget install Microsoft.PowerToys` for **PowerRename** + built-in **PowerShell 7** (`Get-FileHash`, `-WhatIf`).
- **AI Planners:** Run **Claude Cowork** scoped strictly to `Downloads/`, or **Gemini CLI** with `--approval-mode plan --sandbox` for read-only manifests.

Universal Exclude List (Paste Into Every CLI Flag, Script & Agent Prompt)

Exclude paths matching: `.git/**` `node_modules/**` `.venv/**` `dist/**` `target/**` `*.DS_Store` `Thumbs.db` `*.crdownload` `*.part` `_quarantine_30d/**`

Deduplication, Renaming & AI Cleanup Tool Matrix

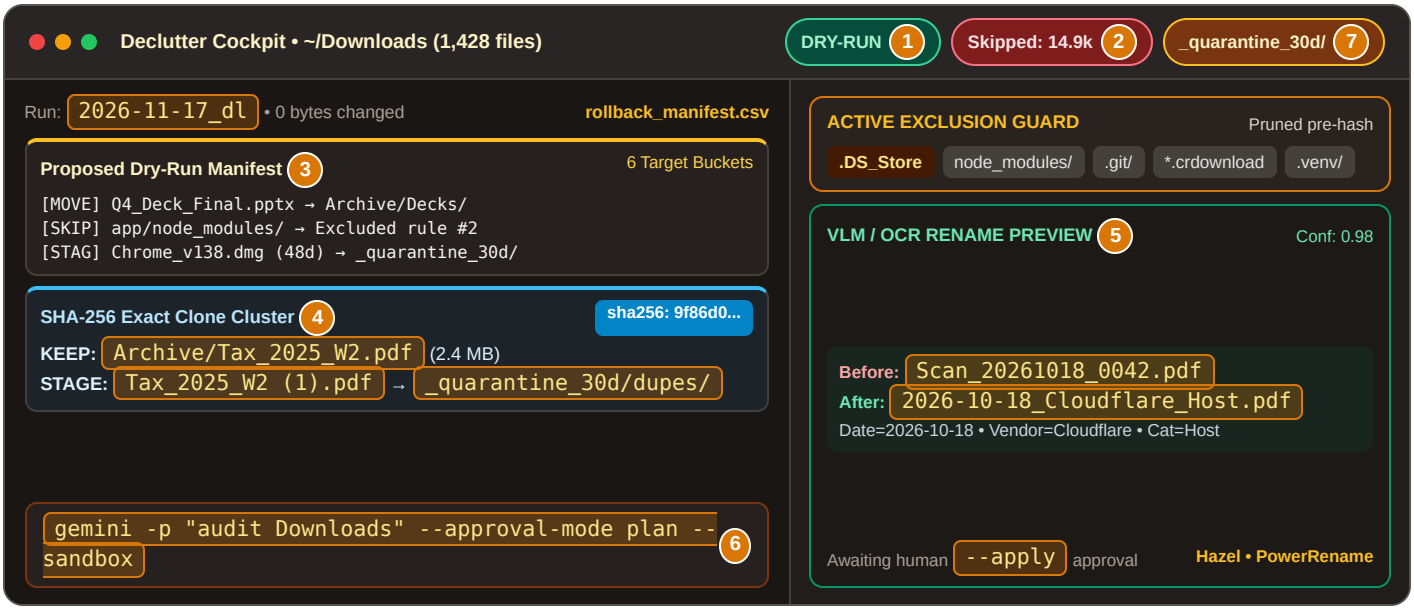
Tool / Engine	OS Support	Verification & Dry-Run Mode	Primary Superpower	Best-Fit Cleanup Task
rmlint *C CLI Linter*	macOS, Linux, Win (WSL2)	Generates `rmlint.sh`; test safely via `./rmlint.sh -n`.	`--algorithm=sha256` + `-p` byte check; ultra fast.	Sweeping large media, backup & archive trees safely.
fdupes / jdupes *Classic CLI*	macOS, Linux, Win ports	**Dry-run by default:** prints dupes; deletes only with `-d`.	Size + MD5 + byte-for-byte check; `-A` skips hidden files.	Quick scriptable text reports & pipe-to-quarantine lists.
Python `hashlib` *Custom Script*	macOS, Windows, Linux	Writes `manifest.csv` first; requires explicit `--apply`.	`file_digest(..., "sha256")` + custom `_quarantine_30d/`.	Cross-OS reproducible sweeps with rollback CSV logs.
Hazel 6 / PowerRename *OS Native GUI*	Hazel (Mac) / PowerToys (Win)	`Option+Cmd+P` Preview + Finder Revert / Live preview grid.	Hazel on-the-fly OCR / PowerRename Regex + EXIF/XMP.	Continuous background inbox rules & visual bulk renames.
Claude Cowork & Gemini CLI	macOS & Windows Desktop / CLI	Approval-gated dry run + Scheduled Tasks / `--approval-mode plan --sandbox`.	Semantic PDF/screenshot reading & smart categorization (Cowork runs on Anthropic servers).	Taming messy `Downloads/` & naming non-PII docs.

Scope Boundary Rule: Never Grant Cleanup Access to Your Home Root

Point Cowork, Gemini CLI, or custom scripts exclusively at leaf folders like `~/Downloads`, `~/Desktop`, or `C:\Users\You\Downloads`. Never aim automated sorters at `~/`, `~/Library`, `AppData`, or active project repositories.

Annotated Tour: Inside a Safe Declutter Cockpit

Whether you inspect a manifest in Claude Cowork, preview rules in Hazel 6, or review a Python/PowerShell dry-run table in your terminal, every trustworthy cleanup workflow exposes the exact same seven visual checkpoints before moving a single file.



1 **Read-Only Dry-Run Mode:** Confirms zero destructive actions occur during scanning—matching `rmlint.sh -n`, PowerShell `-WhatIf`, and Gemini `plan` mode.

2 **Pre-Hash Exclusion Counter:** Shows dev builds, `.DS_Store`, and active browser downloads safely bypassed before hashing begins.

3 **Dry-Run Move Manifest:** Displays every source path, proposed bucket, and target name so you catch surprises before committing changes.

4 **SHA-256 Clone Cluster:** Groups byte-identical files regardless of filename, keeping one canonical original and staging duplicates safely.

5 **VLM / OCR Rename Preview:** Extracts date, vendor, and category from image-only scans into clean `YYYY-MM-DD_Vendor_Category.pdf` names.

6 **Approval-Gated Agent Bar:** Runs Claude Cowork or Gemini CLI inside strict folder sandboxes with explicit human sign-off required to execute moves.

7 **30-Day Quarantine Staging & Ledger:** Routes every duplicate and stale installer into `_quarantine_30d/YYYY-MM-DD/` alongside `rollback_manifest.csv` so you can restore any file in seconds during a full month cooling-off window.

SHA-256 Duplicate Sweeps: Zero-Guesswork Cleanup

Filename heuristics fail in both directions: two different monthly statements might both arrive as `statement.pdf`, while an identical 200 MB installer lurks under `Docker.dmg` and `Docker (1).dmg`. Size pre-filtering followed by SHA-256 hashing finds true clones with mathematical certainty.



Size Bucket → SHA-256 Digest → Quarantine: Two files only reach full cryptographic hashing if their exact byte sizes match first, keeping scans blazingly fast across tens of gigabytes.

Three Proven Duplicate Sweep Engines

1 `rmlint` Script-First Sweep

Rather than deleting live files, `rmlint` writes a clean executable script (`rmlint.sh`). Run `./rmlint.sh -n` first for a dry-run preview; pass `-p` if you want extra byte-by-byte verification alongside SHA-256.

2 `fdupes` Default Dry Run

Classic `fdupes` (or faster C fork `jdupes`) prints duplicate sets without touching disk unless you pass `-d`. Add `-r -n -A -S` to recurse while skipping empty 0-byte files and hidden `.DS_Store` metadata.

3 Python `hashlib.file_digest`

Available natively on both Mac and Windows in Python 3.11+, `hashlib.file_digest(f, "sha256")` streams huge video or ISO files at C speed without loading gigabytes into RAM.

4 Keep Canonical, Stage Clones

Always sort each duplicate cluster to keep the shortest clean path outside `Downloads/` (or oldest original without `(1)`), routing remaining clones straight to `_quarantine_30d/dupes/`.

```
# 1) rmlint SHA-256 sweep (macOS / Linux / WSL2) -- generates rmlint.sh without touching files, then dry-run:
rmlint --algorithm=sha256 --types="duplicates" --output=sh -p ~/Downloads ~/Desktop && ./rmlint.sh -n

# 2) fdupes safe read-only report (skips 0-byte files -n and hidden dotfiles like .DS_Store -A):
fdupes -r -n -A -S ~/Downloads > ~/Desktop/dupes_preview.txt

# 3) Cross-OS Python 3.11+ SHA-256 hash helper (streams chunks cleanly at C speed):
python3 -c 'import hashlib, pathlib; print(hashlib.file_digest(open("invoice.pdf","rb"), "sha256").hexdigest())'
```

Smart Original Selection Rule: Never Let Random Order Pick the Survivor

When deduplicating across `~/Documents` and `~/Downloads`, list your organized archive folder *before* `~/Downloads` on the CLI (or mark it as a tagged keeper with `rmlint ... // ~/Documents`) so copies in Downloads are staged instead of filed originals.

The 5-Part Safe Cleanup & Manifest Formula

Whether you write a Python cleanup script, configure a PowerShell audit, or prompt Claude Cowork and Gemini CLI, reliable file automation separates deterministic rules from AI judgment using five non-negotiable building blocks.

Deterministic Layer (Scripts & OS Tools)

- **Exact Extensions & Paths:** Moving `.dmg` / `.exe` installers, filtering `node_modules/`, and checking file age (>30 days).
- **Cryptographic Checksums:** Computing SHA-256 hashes and verifying post-move byte integrity.
- **Quarantine & Undo:** Writing deterministic CSV rollback ledgers before moving any file.

Semantic AI Layer (Cowork, Gemini, VLMs)

- **Fuzzy Document Triage:** Distinguishing a signed NDA from a draft slide deck or utility receipt.
- **OCR & Screenshot Naming:** Reading vendor logos and dates inside flattened PDFs and PNG screenshots.
- **Folder Proposal:** Clustering 400 random downloads into 5–6 clean, domain-specific folders.

The 5-Part Guarded Declutter Formula

[1. Strict Scope] Target only `~/Downloads` & `~/Desktop`, **[2. Exclusion Guard]** skip `.DS_Store`, `node_modules`, `.git` & `*.crdownload`, **[3. Hash & Schema]** compute SHA-256 + `YYYY-MM-DD_Vendor_Category` names, **[4. Dry-Run Manifest]** write `manifest.csv` & pause for approval, **[5. Quarantine Stage]** and route discards to `_quarantine_30d/` instead of deleting.

Formula Component	What It Enforces	What Fails When Skipped
Strict Scope	Confines AI and scripts to messy drop zones (Downloads/Desktop).	Agents wander into git repos, app libraries, or synced cloud roots.
Exclusion Guard	Prunes dev artifacts, OS dotfiles, and active download partials upfront.	Scripts choke on 100k dependency files or corrupt live browser downloads.
Hash & Schema	Proves identical files via SHA-256 and enforces ISO dates on renames.	Files get mislabeled duplicates or collide on generic vendor names.
Dry-Run Manifest	Outputs an inspectable CSV/Markdown move plan with zero disk writes.	A single bad regex or hallucinated category scrambles 500 files silently.
Quarantine Stage	Holds duplicates and old installers for 30 days with an undo log.	Realizing a week later that an archived installer or tax PDF vanished.

```
# Dry-run manifest generator pattern (safe by default; requires --apply to move files):
SKIP_DIRS = {".git", "node_modules", ".venv", "__pycache__", "dist", "build", "target", "_quarantine_30d"}
SKIP_EXTS = {".crdownload", ".part", ".download", ".tmp"}
# Manifest columns: action | sha256_prefix | size_mb | source_path | proposed_target | reason
# Example row: STAGE_DUPE | 9f86d081 | 2.41 | Downloads/W2 (1).pdf | _quarantine_30d/2026-11-17/W2 (1).pdf | Exact SHA-256 clone
```

VLM & OCR Receipt Renaming: End `Scan_042.pdf` Chaos

Generic filenames like `document(3).pdf`, `IMG_4821.HEIC`, or `Screenshot 2026-11-04.png` bury tax deductions and receipts under unsearchable timestamps. Combining lightweight OCR with Vision-Language Models (VLMs) extracts clean metadata into universal `YYYY-MM-DD_Vendor_Category.pdf` sorting.

1 Text Layer vs. VLM

Digital invoices already contain selectable text. Scanned receipts lack text layers: run `ocrmypdf --deskew --skip-text` first, or inspect page 1 directly with a VLM (Claude, Gemini, or local `qwen-vl`).

2 Strict Filename Schema

Lock output to `YYYY-MM-DD_Vendor_Category.pdf`. Strip spaces, commas, slashes, and corporate suffixes (`Inc`, `LLC`) so filenames sort chronologically across Finder, Explorer, and terminal shells.

3 Low-Confidence Fallback

If a blurry receipt returns missing dates or illegible totals, never guess: move it to `_needs_review/` or stamp file mtime with `_REVIEW` so your tax archive stays 100% trustworthy.

Real-World Before & After Renaming Transformations

Raw Incoming File	Extraction Pipeline	Clean Standardized Output
`Scan_0042.pdf` *Flat scanner image PDF*	`ocrmypdf --deskew --skip-text` + JSON prompt	`2026-10-18_Cloudflare_Hosting.pdf`
`receipt_99182.pdf` *Airline e-ticket email PDF*	Direct `pdfplumber` text extract + category enum	`2026-09-30_Delta_Travel.pdf`
`Screenshot 2026-11-04.png` *Software SaaS checkout snap*	VLM image read → PDF convert or PNG rename	`2026-11-04_Figma_Software.pdf`
`statement(3).pdf` *Monthly utility bill*	Hazel 6 on-the-fly OCR or local Ollama JSON parser	`2026-10-01_ConEd_Utility.pdf`
`tax_locked_2025.pdf` *Encrypted brokerage PDF*	Password prompt skipped → routed safely for review	`_needs_ocr/tax_locked_2025.pdf`

```
# 1) Add searchable OCR text layer to scanned PDFs (skips PDFs that already have text, straightens crooked scans):
ocrmypdf --deskew --skip-text raw_scan.pdf searchable_scan.pdf

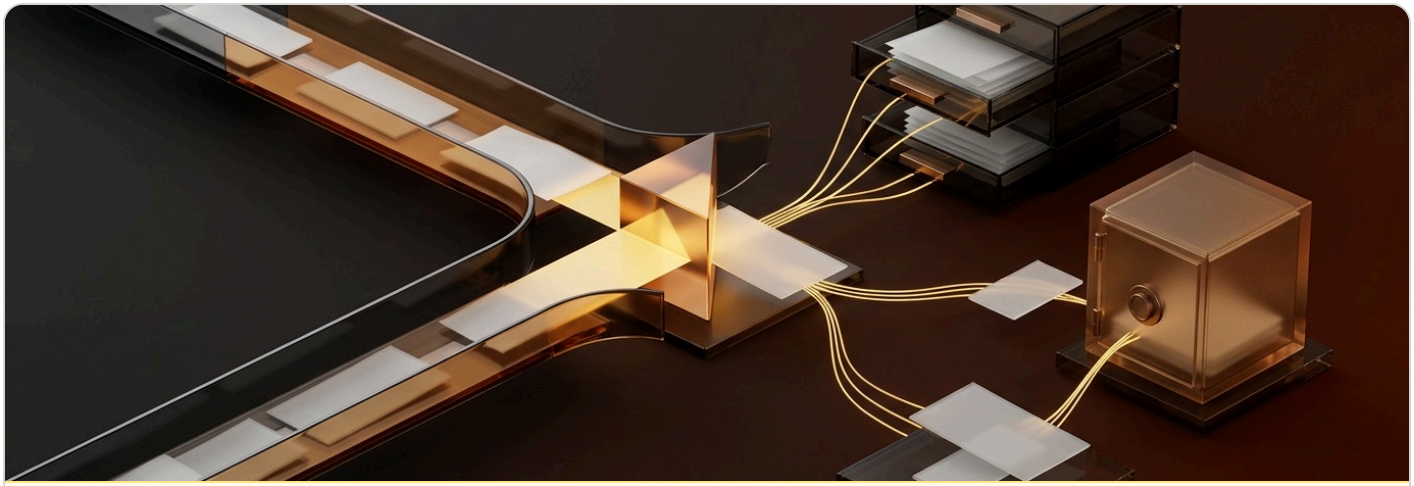
# 2) Structured VLM / LLM JSON prompt for deterministic YYYY-MM-DD_Vendor_Category.pdf dry-run renaming:
"Inspect page 1 of this receipt. Return ONLY JSON:
 {"date": "YYYY-MM-DD", "vendor": "PascalCase", "category": "Hosting|Software|Travel|Utility|Tax", "confidence":
0.95}
 If date or vendor is unreadable, set confidence below 0.70 and return null for missing fields."
```

Collision-Proof Renaming Rule: Append `_02`, Never Overwrite

Two legitimate receipts from the same vendor on the same day (like two Uber rides or separate AWS accounts) will collide on `2026-11-10_Uber_Travel.pdf`. Always check `if dest.exists()` and append `_02`, `_03` before renaming—never silently overwrite!

Mac vs. Windows Automation: Side-by-Side Playbook

Once a deep dry-run cleanup clears your backlog, wire native OS rules so incoming downloads sort themselves quietly every day. macOS pairs **Hazel 6**, **Folder Actions**, and Spotlight `mdfind`; Windows 11 pairs **PowerToys PowerRename**, File Explorer, and **PowerShell**.



Cross-OS Parity: Whether running macOS rules in Hazel and `mdfind` or Windows regex batches in PowerToys PowerRename and PowerShell `-WhatIf`, both ecosystems converge on identical dry-run safety.

Side-by-Side Mac vs. Windows Automation Architecture

Automation Need	macOS Stack (Finder, Hazel 6, Folder Actions, mdfind)	Windows 11 Stack (Explorer, PowerRename, PowerShell)
Continuous Inbox Rules	**Hazel 6:** Watches `~/Downloads` & `Filing Inbox`; supports on-the-fly OCR, encrypted Keychain PDFs, and cloud-stub guards.	**Task Scheduler + PowerShell:** Runs scheduled sorting script cleanly on `$env:USERPROFILE\Downloads` using age & extension filters.
Instant Drop Triggers	**Folder Actions / Shortcuts:** Right-click folder → *Services* → *Folder Actions Setup* to run shell/AppleScript on new drops.	**Power Automate / FileSystemWatcher:** Triggers script or Power Automate Desktop flow whenever PDFs land in `Receipts\Inbox`.
Bulk Pattern & Photo Rename	**Finder Rename + Hazel Tokens:** Select items → right-click *Rename*; Hazel extracts custom text/date tokens automatically.	**PowerToys PowerRename:** Explorer right-click GUI (no native CLI); supports `$1` Regex capture groups, EXIF/XMP photo tags & Copilot hints.
Instant Metadata Search	**Spotlight `mdfind`:** Queries system index instantaneously (`mdfind "kind:pdf invoice"` or stale `.dmg` sweeps).	**PowerShell + Explorer Search:** `Get-ChildItem -File -Recurse` with `Where-Object` filters and File Explorer `datemodified:` chips.
Dry-Run & Undo Safety	**Hazel Preview (`Opt+Cmd+P`):** Tests rules before enabling; Finder right-click **Revert** undoes misfiled Hazel moves.	**`-WhatIf` & PowerRename Grid:** PowerRename shows live Before/After preview; PowerShell `Move-Item -WhatIf` logs moves safely.

```
# macOS Spotlight mdfind audit (lists .dmg/.pkg installers in Downloads untouched for 30+ days, zero changes):
mdfind -onlyin ~/Downloads '(kMDItemFSName == "*.dmg" || kMDItemFSName == "*.pkg") && kMDItemLastUsedDate <
$time.today(-30)'

# Windows PowerShell dry-run staging + SHA-256 manifest (-WhatIf simulates moves without touching disk):
$q = "$env:USERPROFILE\Downloads\_quarantine_30d\$(Get-Date -Format yyyy-MM-dd)"; New-Item $q -ItemType Directory -
Force | Out-Null
Get-ChildItem "$env:USERPROFILE\Downloads" -File | Where-Object { $_.Extension -in ".exe",".msi",".zip" -and
$_.LastWriteTime -lt (Get-Date).AddDays(-30) } |
  ForEach-Object { Move-Item -LiteralPath $_.FullName -Destination $q -WhatIf }
```

5 Real-World Copy-Paste Cleanup & Dry-Run Recipes

Paste these field-tested instructions straight into **Claude Cowork**, **Gemini CLI**, or your terminal scripts. Every single recipe enforces strict read-only dry runs and exclusion lists before a single byte moves.

Recipe & Tool	Copy-Paste Prompt / Command Instruction	Output & Safety Gate
1. Downloads/ Dry Run *Claude Cowork*	`Scope ONLY to ~/Downloads. Do NOT move or delete any files yet. Ignore .DS_Store, node_modules, .git, and *.crdownload. Analyze types, sizes, and duplicates; propose up to 6 clean buckets; write _cleanup_manifest.md and wait for my explicit approval.`	Produces `_cleanup_manifest.md`; zero writes until Turn 2 approval.
2. Plan-Only Clutter Audit *Gemini CLI*	`gemini -p "Inspect ./Downloads and ./Desktop read-only. Skip dev build folders and hidden OS files. Group stale installers (>30d), unfiled PDFs, and suspected clones into a JSON move table estimating reclaimed GB." --approval-mode plan --sandbox`	Hard read-only `plan` mode inside sandbox container.
3. VLM Receipt Renamer *OCR / Cowork / Local AI*	`Read page 1 of every PDF/PNG in ./Inbox_Receipts. Extract ISO date, VendorSlug, and Category (Hosting|Travel|Utility|Tax|Software). Output a dry-run table mapping current name -> YYYY-MM-DD_Vendor_Category.pdf. Route unreadable scans to _needs_ocr/.`	Dry-run rename preview; skips overwrite via `_02` suffix.
4. SHA-256 Safe Dedup *rmlint / Python*	`Sweep ~/Downloads and ~/Desktop using SHA-256 digests excluding .git, node_modules, .venv, and .DS_Store. Keep the oldest clean path outside Downloads; stage all exact clones into _quarantine_30d/dupes/ and write rollback_manifest.csv.`	Stages clones to `_quarantine_30d/` with full SHA-256 undo CSV.
5. Dev Cache & Installer Trim *Mac & Win CLI*	`Find stale node_modules/, target/, .venv/, and .dmg/.exe installers untouched for 60+ days. Verify no uncommitted Git changes exist in sibling repos, print total reclaimable GB, and wait for confirmation before staging to _quarantine_30d/.`	Preserves dirty Git worktrees; frees 10–40 GB safely.

```
# Two-Turn Claude Cowork Approval Pattern (Turn 1 = Read-Only Dry Run; Turn 2 = Safe Quarantine Execution):
# Turn 1: Run Recipe #1 above inside Cowork scoped to Downloads/ only (disable auto-delete in settings).
# Turn 2 (after reviewing _cleanup_manifest.md):
"Approved with edits: leave Tax_2025_Drafts/ untouched. Move all remaining approved files into the 6 buckets,
stage duplicates and installers >30 days old into _quarantine_30d/2026-11-17/, and append every move to
rollback_manifest.csv."
```

Two-Turn Rule: Never Combine Plan & Move in One Prompt

Always make the agent stop after writing the manifest. Reviewing a 30-line dry-run summary takes 20 seconds and catches 99% of wrong-folder assumptions before anything changes on disk.

Cap Top-Level Buckets at Five or Six Folders

Tell Cowork or Gemini CLI never to create a subfolder with fewer than five files. Broad anchors like `01_Finance`, `02_Projects`, `03_Media`, `04_Installers`, and `_quarantine_30d` stay navigable for years.

Quarantine Staging, Privacy & Instant Rollback Safety

Real peace of mind comes from knowing every automated cleanup decision is completely reversible. By parking flagged items in a dated `_quarantine_30d/` holding bay with an undo ledger—and keeping sensitive tax or health PDFs local—you eliminate catastrophic mistakes.

1 The `_quarantine_30d/` Cooling-Off Protocol

- **Dated Staging Batches:** Move old installers, empty folders, and SHA-256 duplicates into `_quarantine_30d/YYYY-MM-DD/` instead of running `rm` or `Remove-Item`.
- **Machine-Readable Undo Ledger:** Log every moved item in `rollback_manifest.csv` with `timestamp`, `sha256`, `original_path`, and `quarantine_path`.
- **30-Day Purge Gate:** Only empty batches older than 30 days after verifying you still have a recent Time Machine, Windows File History, or cloud backup snapshot.

2 Sensitive PDF & Cloud Processing Privacy

- **Cowork Cloud Reality (Oct 2026+):** Cowork tasks now execute on Anthropic's servers (reaching local folders via Desktop) and untrusted PDFs can carry prompt injection—never feed unredacted tax, bank, or medical files to cloud consumer chats.
- **Local OCR + Local Vision:** Use offline `ocrmypdf`, Hazel 6, or a local Ollama model for sensitive receipts, reserving cloud prompts for non-PII or enterprise ZDR plans.
- **Strip GPS Before Sharing:** Before archiving or sharing phone photos, strip EXIF location coordinates using macOS Preview/Shortcuts or PowerShell/ExifTool.

Five-Stage Safety Gate & Recovery Matrix

Safety Gate	Required Guard / Condition	Audit Artifact Created	Recovery Time
0. Pre-Sweep Snapshot	Verify Time Machine / VSS / Cloud sync	Point-in-time OS restore point	Minutes (OS backup)
1. Read-Only Dry Run	`./rmlint.sh -n` / `-WhatIf` / `plan`	`_cleanup_manifest.md` or `.csv`	0 sec (nothing moved)
2. Quarantine Staging	Explicit user `--apply` confirmation	`_quarantine_30d/YYYY-MM-DD/` + undo CSV	<5 sec (single script)
3. Post-Move Verify	Compare SHA-256 before deleting copies	Verified checksum match log	Instant integrity proof
4. 30-Day Final Purge	Folder age >30 days + zero missing refs	Moves batch to OS Trash / Recycle Bin	Still in OS Trash briefly

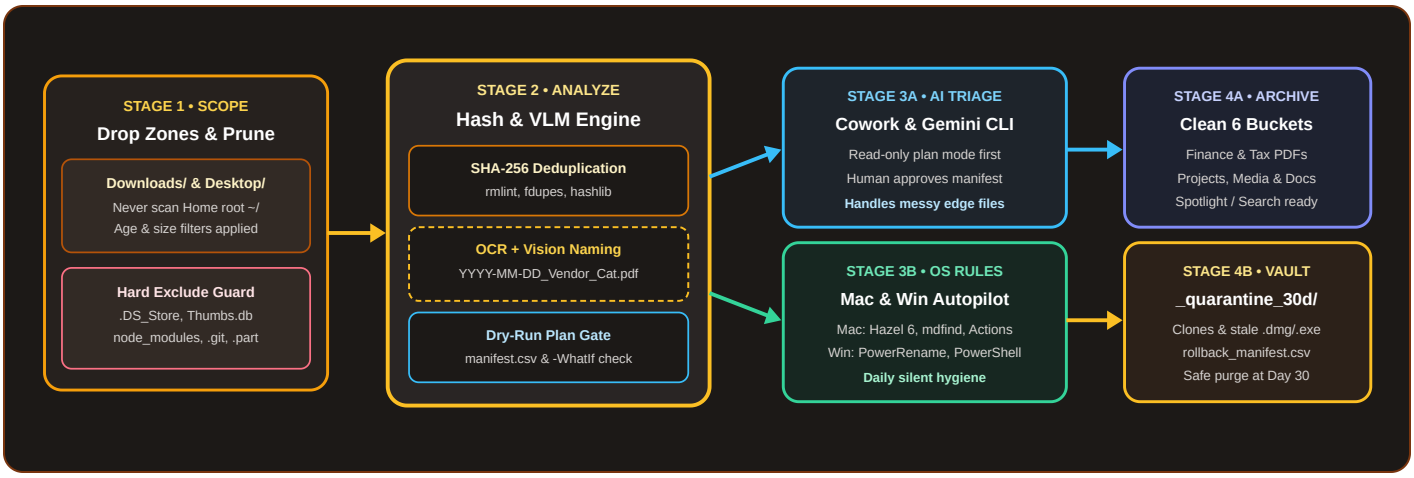
```
# One-command Python Rollback (reads rollback_manifest.csv, verifies SHA-256, restores every file):
import csv, hashlib, shutil, pathlib
for row in reversed(list(csv.DictReader(open("_quarantine_30d/rollback_manifest.csv")))):
    q, orig = pathlib.Path(row["quarantine_path"]), pathlib.Path(row["original_path"])
    if q.exists() and hashlib.file_digest(q.open("rb"), "sha256").hexdigest() == row["sha256"]:
        orig.parent.mkdir(parents=True, exist_ok=True); shutil.move(str(q), str(orig))
```

Cloud-Sync Trap: Watch Out for iCloud Drive & OneDrive Stubs

Deleting or moving a synced folder inside iCloud Drive, Dropbox, or OneDrive deletes it across every linked laptop and phone instantly. Always pause or check cloud sync status before purging `_quarantine_30d/`, and exclude cloud-only placeholder files from blind hash sweeps.

Cross-Tool Synergy: From Messy Drop Zone to Calm Archive

The calmest digital workspace pairs deterministic CLI and OS tools (`rmlint`, `ocrmypdf`, `Hazel 6`, `PowerRename`, PowerShell) for repeatable speed with AI planners (**Claude Cowork** and **Gemini CLI**) for messy, ambiguous documents—all flowing into one unified safety loop.



Your 10-Minute Weekly Declutter Cadence

1 Let OS Rules Catch Routine Noise

Keep **Hazel 6** (Mac) or scheduled **PowerShell** tasks (Windows) clearing 14-day-old installers, zip archives, and screenshots automatically so clutter never piles past a week.

2 Run or Schedule Friday Dry-Run AI Triage

Set a recurring Friday **Scheduled Task** in Claude Cowork (or run **Gemini CLI** with `--approval-mode plan`) on `Downloads/`, approve the manifest, and rename receipts cleanly.

3 Sweep Clones Monthly via SHA-256

Run `rmlint --algorithm=sha256` or Python `hashlib` once a month across media and docs, routing exact clones to `_quarantine_30d/` while preserving canonical archive paths.

4 Empty Day-30 Quarantine Calmly

Glance at month-old folders inside `_quarantine_30d/` after confirming your backup ran; move expired batches to Trash with zero anxiety that you missed something vital.

Simplicity Scales: Stick to Shallow Folders & ISO Filenames

Deep 7-level folder trees collapse back into Desktop dumps because nobody remembers where files live. Pairing 5–6 broad root folders with descriptive `YYYY-MM-DD_Vendor_Category.pdf` names lets Spotlight `mdfind` and Windows Search find anything in 1 second.

Troubleshooting Clinic & 6-Point Pre-Flight Checklist

Reach for this clinic whenever `.DS_Store` floods a duplicate report, scanned PDFs refuse to rename, active browser downloads get moved prematurely, or Hazel skips cloud placeholders—followed by our 6-point pre-flight checklist.

Symptom / Issue	Root Cause	10-Second Fix
1. Scanner choked on 180,000 tiny code files	Cleanup script traversed `node_modules/`, `.git/`, `.venv/`, or `target/`.	Add an early directory prune guard skipping `SKIP_DIRS` before entering subdirectories or hashing any files.
2. Duplicate list full of `.DS_Store` & `Thumbs.db`	Hidden OS folder-layout metadata files share identical default byte headers.	Pass `-A` (`--nohidden`) to `fdupes`, exclude dotfiles in `rmlint`, and filter `.DS_Store` / `desktop.ini` upfront.
3. Browser download corrupted mid-transfer	Folder watcher moved a `.crdownload`, `.part`, or `.download` temp file too early.	Exclude active partial-download extensions and add a rule requiring `modified age > 5 minutes` before sorting.
4. Scanned receipt stayed `Scan_042.pdf`	Image-only scan lacked an embedded text layer for regex or text tools to read.	Run `ocrmypdf --deskew --skip-text` first, enable Hazel 6 on-the-fly OCR, or pass page 1 to a Vision model.
5. PowerToys PowerRename skipped subfolders	Launched only on top-level selection, or regex didn't match target scope dropdown.	Expand subfolders inside Explorer search view, check *Use regular expressions*, or use PowerShell `Get-ChildItem -Recurse -WhatIf`.
6. Hazel 6 rule ignored iCloud / Dropbox files	Files were unhydrated cloud-only stubs rather than downloaded local bytes.	Enable Hazel 6's cloud-only placeholder policy on that folder or wait for local download before OCR inspection.
7. AI agent invented 22 inconsistent folders	Allowed live file edits on Turn 1 without locking bucket names in the prompt.	Enforce `--approval-mode plan` (Gemini) or Turn-1 dry-run manifest (Cowork) capped at 6 fixed bucket names.

6-Point Safe File System Cleanup Pre-Flight Checklist

- ☑ **1. Verify Recent Snapshot:** Confirm Time Machine, Windows File History, or cloud backup is up to date before bulk moves.
- ☑ **2. Lock Scope & Exclusions:** Target only `Downloads/` or `Desktop/` and prune `.git`, `node_modules`, `.DS_Store`, and `*.crdownload`.
- ☑ **3. Run Read-Only Dry Run First:** Inspect `manifest.csv`, `./rmlint.sh -n`, `-WhatIf`, or Gemini `plan` output line by line.
- ☑ **4. Verify SHA-256 & Receipt Names:** Confirm duplicate clusters keep canonical originals and PDFs follow `YYYY-MM-DD_Vendor_Category.pdf`.
- ☑ **5. Stage Discards in `_quarantine_30d/`:** Never delete directly inside scripts; park clones and old installers for a 30-day review window.
- ☑ **6. Keep Rollback CSV Intact:** Store `rollback_manifest.csv` alongside staged files until Day-30 final purge clears safely.

Final Takeaway: Calm Automation Never Makes Irreversible Guesses

Digital clutter builds up because deleting feels risky. When dry-run manifests, SHA-256 checksums, VLM/OCR naming, and a 30-day quarantine bay back every sweep, tidying thousands of files across Mac and Windows becomes effortless, safe, and repeatable.