TECH STYLE LTD

TECH-STYLE.CO

PRODUCT BROCHURE

Rutba Media FileServer — Product Brochure

A media origin that stores masters only and resizes on request, with an optional asset platform and AI enrichment.

WHAT IS INSIDE

Document	Rutba Media FileServer brochure
Generated	2 September 2026
Online	https://www.tech-style.co/docs/media-fileserver-brochure

Store masters only. Resize on request. Then, if you want it, add an asset index, accounts, versions, collections and AI-generated metadata — each layer optional, each one gated on its own configuration, and none of them changing how the origin behaves when they're off.

Most media stacks store far more than they serve

The conventional approach generates every thumbnail, small, medium and large variant at upload time. In the library that prompted this project, 5,428 originals had become roughly 30,000 files and 4.9 GB — most of those variants never requested, all of them on disk forever, and the whole set needing a re-render every time the design changed.

Rutba keeps one thing: the master. Variants are produced when a browser actually asks for them, served, and cached. When the cache hits its ceiling it evicts the least recently used entries down to about 80% and carries on. Change your breakpoints and the old variants simply age out.

It runs as the origin for images.rutba.pk and is reusable for images.trustlist.uk, so it handles both a retail catalogue and a directory of vendor logos in production.

Stored	Master files only
Variants	Generated on request, cached with LRU
Default cache cap	1 GiB, configurable
Drop-in for	Strapi upload provider

Never upscales.
Requests larger than the master return the master's own dimensions, and aspect ratio is always preserved — so a bad query parameter can't produce a blurry hero image.

THREE LAYERS

Off means unchanged

The governing rule of the codebase: a feature whose backend isn't configured must not alter existing behaviour. No database and it is byte-for-byte the original origin. No ffmpeg and videos simply stream. No API key and there is no AI at all.

NO DEPENDENCIES TO CONFIGURE

Origin — always on

Masters, on-request resizing, the LRU variant cache, Range streaming, multi-volume storage, origin pull-through and clustering. No database required, no build step. No dependencies to configure

GATED ON DB_HOST

Platform — add MySQL

A complete asset index, a background job queue, accounts with roles and MFA, versions, folders, collections, custom metadata, sharing, WebDAV, quotas, trash and audit — plus a web console. Schema created on first boot. Gated on DB_HOST

GATED ON ANTHROPIC_API_KEY

AI — add an API key

Machine-generated tags, captions, alt text, detected text and document summaries through the Claude API, with a hard budget ceiling and human review before anything counts as a real tag. Gated on ANTHROPIC_API_KEY

IMAGES & VIDEO

What it does on the way out

Transformations are expressed as query parameters, so a front end can ask for exactly the render it needs.

On-demand resize

Resize through Sharp with w and h in pixels, capped by MAX_DIM (4000 by default). Never upscales; aspect ratio preserved.

Fit modes

inside (default), cover, contain, outside or fill — the same vocabulary Sharp uses, passed straight through.

Format conversion

Request jpeg, png, webp, avif or auto — which honours the browser's Accept header — with quality from 1–100 (default 80).

Strapi variant prefixes

Drop-in for existing URLs: thumbnail (245), xsmall (64), small (500), medium (750), large (1000), xlarge (1920), remappable via VARIANTS.

Extension swap

The requested extension is a hint. A request for small_x.webp resolves to master x.jpg and keeps the master's format — use ?fm= to convert deliberately.

Video poster frames

?poster or ?thumb returns a still frame as an image — with w, h, fm, q and t=<seconds> for the timestamp — cached like any variant.

On-demand transcoding

?transcode=<height> produces H.264/AAC MP4 between 144p and 2160p. Without an ffmpeg binary the original streams instead — the feature is simply off.

HTTP range streaming

Video, audio, SVG and other non-raster files stream straight from the master store with Range support, so viewers can seek without downloading the whole file.

STORAGE, CACHE & CLUSTER

What it does with what it keeps

Variant cache with a ceiling

Capped by CACHE_MAX_BYTES (1 GiB by default). Every cache hit touches the file, so eviction down to ~80% is a true least-recently-used decision rather than an age heuristic.

Multi-volume masters

Spread masters across directories or mounts, any of which can be read-only. Reads search every volume, so a master on any mount is served and resized transparently.

Placement policies

free writes to the volume with most space, fill to the first with room, route follows prefix rules such as archive/ =archive. Existing masters are always replaced in place.

Origin pull-through

A missing master is fetched from an allow-list of base URLs and persisted locally, then served at the requested size — turning a migration into something lazy rather than a big-bang copy.

Clustering

A fresh upload fans out to eligible peers; a node missing a master asks its peers before reaching for an origin. Replicated writes are marked so receivers don't re-fan-out — no loops.

Public / private zones

Each node is public or private. A public-facing node never replicates, accepts or serves a private master — so a LAN node can push its public uploads up while private ones stay inside.

Traversal protection

Path-traversal guards on every read, origin fetches restricted to configured base URLs, and node-to-node traffic authenticated with a secret separate from the upload token.

Concurrency de-duplication

Simultaneous requests for the same uncached variant collapse into a single render instead of stampeding the CPU.

Cache & CORS headers

Immutable Cache-Control on variants, configurable CORS, and HEAD, OPTIONS and /_health for anything sitting in front of it.

THE ASSET PLATFORM

Add a database, get an asset manager

Point the server at MySQL and a managed platform switches on over the same masters — schema created automatically on first boot, and every part of it inert again the moment the database goes away.

Index & background work

A row for every master

Rows are keyed by path_hash — the sha256 of the full path — and record which volume holds the bytes, so a read looks the volume up instead of probing each one.

Scan & reconcile

Files that arrived by rsync, restore or a mounted volume are invisible until scanned. A scan indexes unknown masters, refreshes drifted rows, and marks vanished ones missing — never deleting them, because the row may carry tags, shares and comments.

Resumable, rate-limited

A killed scan continues from its recorded progress rather than starting over, and a files-per-second limit keeps reconciling a large library from starving live requests.

One job queue

Scans, metadata extraction, enrichment and bulk operations all run as jobs. Workers claim under a lease so a crash returns the job; duplicates collapse, so "scan now" clicked five times is one scan.

Accounts & access

Accounts & RBAC

scrypt-hashed passwords, roles and permission predicates, and sessions that die on logout, expiry, password change, role revocation or account disable.

TOTP two-factor

Standard RFC 6238 — any authenticator app works. Nothing switches on until a working code proves the app is really configured, and ten single-use recovery codes are issued at that moment.

Login throttling

Failures counted per account and per IP over a sliding window, locking from the last attempt. A missing MFA code deliberately isn't counted — the password held, and locking someone out for fumbling a 30-second code is self-inflicted denial of service.

API tokens

The credential for scripts, CI and WebDAV. The plaintext is shown once and only its hash is stored.

Read authorisation

public, mixed or private. A caller is admitted by a cluster secret, an upload token, a session or API token — or by a share link, which is itself the explicit grant.

Safe by construction

The last active admin cannot be disabled, demoted or deleted, and deleting a user never deletes files — ownership is orphaned or reassigned, your choice.

Managing the library

Versions

An overwrite moves the outgoing copy aside before the new one lands, stored outside every served volume. Restoring retains the current bytes first, so a restore is itself undoable. Retention is capped by count and optionally age.

Folders & collections

Folders are real rows that can be renamed or moved as a subtree. Collections cut across paths — a campaign, a client, a release. An asset lives in one folder and any number of collections.

Custom metadata

Define your own fields — text, number, date, enum, multi-select, boolean — set them per asset, and filter on them alongside EXIF and tags. Values are stored in the column their type deserves, so a date filter is a date comparison.

Named renditions

?rendition=web-hero instead of a memorised query string. Retune the definition later and every URL downstream teams shipped follows; an explicit parameter still wins.

Bulk operations

Tag, collect, delete, re-extract or enrich everything matching a filter, as one job. The filter is built by the same code that powers the file listing — so the set you previewed is exactly the set that gets touched.

Comments

Threaded per asset and resolvable. Posting notifies the owner and everyone already in the thread, and nobody else — a feed that includes the whole installation is useless within a week.

Share links

Public links with view or forced-download, optional password, expiry and a download cap enforced atomically. Every serve is counted and audited; revoke instantly.

Trash & recovery

Deletes move the master to a trash area outside the served volumes and mark the row — restore, purge, or empty. Without a database, delete unlinks exactly as it always did.

Duplicate detection

Every upload is hashed as it streams, so duplicate groups and the total reclaimable bytes come for free — no extra disk read.

Storage quotas

Per-user caps checked against Content-Length before the body is stored, with a post-write backstop that rolls the file back so it is never served, replicated or indexed.

WebDAV mount

Mount the whole store — across every volume — as a network drive from Windows, macOS, rclone or davfs2. A DAV upload is indistinguishable from an API upload.

Audit trail

Master writes, logins, share access and administrative changes recorded, so you can answer who did what and when.

A web console, with no build step.
Everything above has a UI at /_ui/ — file browser and search, drag-and-drop bulk upload, previews with EXIF and video metadata, tags as chips, duplicates, trash, shares, users and jobs — served as a plain SPA. Every action it takes is a documented call on the JSON control plane, so anything you can click you can script.

AI ENRICHMENT

Metadata a machine wrote, and a person approved

With an API key configured, assets get generated tags, captions, alt text, detected text and document summaries through the Claude API. All of it is metadata — a master is never modified.

Machine tags are not human tags

Suggestions land in their own table, never in the human tag set. They become real tags only when somebody promotes them — an explicit, audited act — and a rejected suggestion stays rejected, so re-running enrichment never resurrects it.

Every row records the model and the prompt version that produced it. A bad prompt is therefore revocable rather than an archaeology project, and a re-run is auditable.

A refusal, a malformed response or an exhausted budget is recorded as a result rather than retried — it shows up for review instead of burning four more attempts and four more dollars against the same asset.

Cost controls in AI enrichment

LEVER	WHY IT MATTERS
Send a resized variant	Images bill by area. The origin already generates variants, so enrichment uses a bounded one (1568px by default) rather than a 4000px master — the single biggest saving.
Cache the instruction prefix	The taxonomy and output rules are identical on every call, so they sit before the cache breakpoint and only the image varies.
Batch the backfill	Enriching an existing library submits a message batch at half price, skipping assets already done — safe to re-run, and it resumes where it stopped.
Hard monthly ceiling	A budget checked before every call, batches included — not a number reported after the fact. Every call is metered and surfaced in the console.

Image enrichment
Tags, caption, alt text and detected text in one structured call per asset, with suggested values for your own custom metadata fields.

Document enrichment

PDFs and office documents reduced to extracted text, a summary, entities and suggested tags — making document bodies searchable, which a filename index cannot do at all.

Review & override

A console surface to accept, edit, reject, promote a suggestion to a real tag, or re-run one asset or a whole collection. Nothing a model produced is indistinguishable from what a person asserted.

ENDPOINTS

The surface area

A deliberately small public surface — serve, upload, delete, health. Everything the platform adds lives under a reserved underscore namespace, so media path keys are never shadowed.

- GET/<path>?w=&h=&fit=&q=&fm=Resize and serve an image variant
- GET/<path>?rendition=<name>Serve a named rendition
- GET/<path>Serve the master as-is, Range-aware
- GET/uploads/<prefix>_<name>.<ext>Strapi-compatible variant resolution
- PUT/<path>Upload a master (token-gated, versioned, replicated)
- DELETE/<path>Move to trash and propagate to peers
- GET/_healthHealth check
- GET/_api/filesSearch the index — text, tags, type, status, custom fields
- POST/_api/bulkApply an operation to everything matching a filter
- POST/_api/jobsEnqueue a scan or other background job
- POST/_api/ai/enrichEnrich an asset or a filtered batch
- GET/_api/ai/usageToken and spend accounting
- POST/_api/sharesMint a share link with password, expiry and download cap
- GET/_api/storagePer-volume free and total space (admin)
- GET/_ui/Web console
- GET/_dav/WebDAV mount
- GET/_s/<token>Public share link

Running in a few minutes

No build step for either the server or the console. Drop in your masters and start it — everything else is opt-in.

```
# install and run the origin git clone https://github.com/eharain/Rutba-Media-FileServer.git cd Rutba-Media-FileServer npm install node server.js # request a variant curl "http://localhost:3000/photo.jpg?w=750&fm=webp&q=82" # turn on the platform layer DB_URL="mysql://user:pass@localhost/media" node server.js # > console at /_ui/, control plane at /_api/
```

Key configuration

Common environment variables

MASTER_DIR	Where masters live (default ./public)
CACHE_MAX_BYTES	Cache ceiling before LRU eviction (default 1 GiB)
MAX_DIM	Largest requestable dimension (default 4000)
ORIGIN_SOURCES	Allow-listed base URLs for pull-through
CLUSTER_PEERS	Sibling nodes as <baseUrl>|<role>
STORAGE_VOLUMES	Extra master volumes, id:path or id:path|ro
DB_URL	Master switch for the platform layer
TRASH_DIR	Retained deletes and versions — set explicitly in containers
MFA_ENABLED	Offer TOTP enrolment (off by default)
SCAN_ON_BOOT	Reconcile the index once at every boot
ANTHROPIC_API_KEY	Master switch for AI enrichment
AI_MONTHLY_BUDGET_USD	Hard ceiling, checked before every call

INTEGRATIONS & TESTING

Fits what you already run

Strapi upload provider

A provider package makes it a drop-in replacement for a heavier upload stack, honouring Strapi's own variant prefixes so existing URLs keep working.

Next.js image loader

An optional custom loader wires <Image> straight to the drop-formats path.

Migration utilities

Database-driven migration for MySQL and PostgreSQL — copy masters only and rewrite the formats column, cutting file count by roughly five-sixths.

Two test suites

An origin suite that needs no database — proving the gated layers really are off — and a platform suite that creates a throwaway database per run and drops it afterwards.

Tested across runtimes

Both suites run in CI on Node 18, 20 and 22, each printing its own check total so no README line can drift from the real number.

Embeddable

The app is exported as well as executable, so it can be embedded or driven from tests without spawning a process.

RELATED

Other products

EMAIL DELIVERY

Rutba MTA

The other half of the infrastructure pair.

OPEN-SOURCE ERP

Rutba ERP

The platform whose catalogue this serves.

PACKAGES

Open source

Our npm packages and Strapi plugins.

CONTACT

Talk to us

Tech Style Ltd is a UK technology company building enterprise software, procurement intelligence and AI platforms. Tell us what you are trying to do and we will tell you honestly whether we are the right people for it.

This document online	www.tech-style.co/product-media-fileserver
Website	www.tech-style.co
Enquiries	hello@tech-style.co
Products	www.tech-style.co/products
Partner programme	www.tech-style.co/partners

This document was generated on request.
It is built from the live content of tech-style.co at the moment you downloaded it, so it can never quote a price, a version or a status the site has since moved on from. Request a fresh copy any time at www.tech-style.co/downloads.