Package 'pensar'

June 25, 2026

Type Package

Title LLM Wiki Engine

Version 0.6.4

Date 2026-06-24

Description Personal wiki engine with a large language model (LLM) as research assistant. Supports guided sessions through a 'Claude Code' <https://github.com/anthropics/claude-code> skill bundle and autonomous research runs from R via autoresearch(). Results land in a structured vault of markdown pages with 'YAML' frontmatter and wikilinks, ready for hand-editing in your favourite editor alongside the LLM. Vaults are seeded with 'CLAUDE.md' and 'AGENTS.md' so 'Claude Code', 'Codex' <https://github.com/openai/codex>, and other agents share the same operating instructions. Can adopt an existing 'Obsidian' <https://obsidian.md/> vault in place via init_vault(adopt = TRUE).

License Apache License (>= 2)

URL https://github.com/cornball-ai/pensar

BugReports https://github.com/cornball-ai/pensar/issues

SystemRequirements pandoc (for vault_export()), git (for vault_commit())

Imports curl, digest, stringdist, yaml

Suggests jsonlite, llm.api, saber, simplermarkdown, tinytest

VignetteBuilder simplermarkdown

Encoding UTF-8

NeedsCompilation no

Author Troy Hernandez [aut, cre] (ORCID: <https://orcid.org/0009-0005-4248-604X>), cornball.ai [cph]

Maintainer Troy Hernandez <troy@cornball.ai>

Repository CRAN

Date/Publication 2026-06-24 22:50:02 UTC

Contents

autoresearch . . . 2
backlinks . . . 4
dedup . . . 5
default_vault . . . 6
ingest . . . 6
ingest_agent_context . . . 7
ingest_briefing . . . 8
ingest_repo . . . 9
ingest_url . . . 10
init_vault . . . 11
lint . . . 13
log_entry . . . 13
manifest_path . . . 14
migrate_briefings_to_repos . . . 15
outlinks . . . 16
page_context . . . 16
pensar_skill_path . . . 17
print.pensar_research . . . 18
read_manifest . . . 18
recent_activity . . . 19
related_pages . . . 19
search_pages . . . 20
show_page . . . 21
status . . . 22
tags . . . 22
update_index . . . 23
update_manifest . . . 24
use_vault . . . 25
vault_commit . . . 26
vault_export . . . 26
vault_graph . . . 27
vault_registry . . . 28

Index **30**

`autoresearch` *Autonomous research loop into a pensar vault*

Description

Runs a bounded, package-owned research workflow. R controls the loop, source ingestion, wiki writes, index refresh, and logging; model calls are limited to structured decisions such as query planning, source selection, evidence extraction, and page drafting.

The default search backend uses Tavily via `TAVILY_API_KEY`. The default model backend uses `llm.api` when credentials are available and otherwise falls back to deterministic heuristics. Tests and integrations can pass fake `search_backend`, `fetch_backend`, and `model_backend` functions.

Usage

```
autoresearch(topic, vault = default_vault(), search_backend = NULL,
             fetch_backend = NULL, model_backend = NULL, program = NULL,
             force = FALSE, overwrite = TRUE, update = TRUE, slug = NULL,
             provider = "auto", model = NULL, verbose = TRUE)
```

Arguments

topic	Character. Research topic, free-form string.
vault	Character. Vault path.
search_backend	Function with signature function(query, n) returning a data.frame with at least title, url, and snippet.
fetch_backend	Function with signature function(url) returning a list with url, status_code, content_type, body, and fetched_at.
model_backend	Function with signature function(task, input, program) returning structured lists for the internal autoresearch tasks. When supplied, overrides provider and model.
program	Optional list or YAML path overriding the default autoresearch program. Vault-level _research/program.yml overrides package defaults when program is NULL.
force	Logical. Allow writes into adopted vaults.
overwrite	Logical. Allow planned wiki pages to overwrite existing wiki files. Set FALSE to make existing-page updates fail instead of replacing content.
update	Logical. When TRUE (default), planned pages whose slug matches an existing wiki page run through a revise_page model task that reads the existing body and produces an edit-aware revision, so hand-written prose survives a re-run. When FALSE, the planner's new draft body replaces the existing body wholesale.
slug	Optional character. When supplied, force the synthesis row's slug to this value, bypassing the title-overlap collision guard. Useful for explicitly amending an existing wiki page (slug = "my-existing-page" updates wiki/my-existing-page.md in place) or naming a fresh synthesis page. The synthesis row is the first type == "analysis" entry in the planner output, else row 1. Other planned rows (concepts, entities) keep their planner-assigned slugs and still go through normal collision and dedup checks.
provider	Provider for the default llm.api model backend: "auto" (default; picks whichever of ANTHROPIC_API_KEY, OPENAI_API_KEY, or MOONSHOT_API_KEY is set), "anthropic", "openai", "moonshot", "ollama", or "heuristic" (force the deterministic fallback).
model	Optional model name for the default llm.api backend.
verbose	Logical. Print phase progress.

Value

A list of class pensar_research with topic, program, queries, search results, filed sources, extracted claims, written pages, synthesis metadata, and model usage.

Examples

```
## Not run:
vault <- file.path(tempdir(), "ar-example")
init_vault(vault, rproj = FALSE, agent_instructions = FALSE)
use_vault(vault)
Sys.setenv(TAVILY_API_KEY = "tvly-...")
res <- autoresearch("transformer scaling laws")
print(res)
show_page(res$synthesis$slug)

## End(Not run)
```

`backlinks` *Find backlinks to a page*

Description

Scans all markdown files in the vault for `[[wikilinks]]` that reference the target page.

Usage

```
backlinks(page, vault = default_vault())
```

Arguments

`page`	Page name (without `.md` extension).
`vault`	Path to the vault directory.

Value

A data.frame with columns `source` (page name) and `file` (path relative to the vault).

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("See [[seed]] for context.", type = "articles",
       source = "demo", vault = v)
backlinks("seed", vault = v)
unlink(v, recursive = TRUE)
```

dedup *Find candidate duplicate pages*

Description

Compares every non-system page pair by Jaro-Winkler title similarity and tag-set Jaccard overlap. Pairs whose combined score exceeds `threshold` are written to `_proposals/dedup.md` for human review.

Usage

```
dedup(vault = default_vault(), threshold = 0.7)
```

Arguments

`vault`	Vault path.
`threshold`	Minimum combined score, in [0, 1]. Default 0.7.

Details

The combined score weights title similarity at 0.6 and tag overlap at 0.4 (`0.6 * jw_sim + 0.4 * tag_jaccard`). Title similarity is computed on lowercased trimmed titles; pages with no title fall back to `node_id`.

Pensar never auto-merges. The proposals file is for human review.

Value

A data.frame of proposed pairs (invisibly): `page_a`, `page_b`, `title_similarity`, `tag_overlap`, `combined_score`. Sorted by combined score descending.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("A.", type = "articles", source = "alpha-foo", vault = v)
ingest("B.", type = "articles", source = "alpha-foos", vault = v)
dedup(v)
unlink(v, recursive = TRUE)
```

default_vault *Resolve the active vault path*

Description

Returns the path of the vault pensar will act on, resolved from the same opt-in sources as every other entry point: the PENSAR_VAULT environment variable, a walk-up for schema.md, then options("pensar.vault") (set by use_vault). This is the public getter paired with the use_vault() setter; call it to confirm which vault a bare ingest() or status() will use before running them.

Usage

```
default_vault()
```

Details

Per CRAN policy there is no implicit home-filespace fallback. If no source is configured this errors with a setup hint rather than guessing a path. For the resolution source as well as the path (useful when debugging "why is it filing to the wrong vault?"), call status(), which reports both.

Value

Character string: the normalized vault path. Errors if no vault is configured.

See Also

use_vault to set the vault for a session.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
use_vault(v)
default_vault()
options(pensar.vault = NULL)
unlink(v, recursive = TRUE)
```

ingest *Ingest content into the vault*

Description

Writes content to raw/{type}/, generates a filename from source and date, adds YAML frontmatter, updates index.md, and appends to log.md.

Usage

```
ingest(content, type = c("articles", "chats", "briefings", "matrix"), source,
       title = NULL, tags = NULL, vault = default_vault(), force = FALSE)
```

Arguments

content	Character string or character vector (lines) of content.
type	Content type: "articles", "chats", or "matrix". The legacy type "briefings" is still accepted but deprecated; for repo artifacts use ingest_repo().
source	Short identifier for the content source (e.g., URL, session ID, project name).
title	Optional title. If NULL, derived from source.
tags	Optional character vector of tags.
vault	Path to the vault directory.
force	In adopted vaults (init_vault(adopt = TRUE)), ingest() refuses to write by default. Pass TRUE to write into the adopted tree anyway. Native vaults ignore this parameter.

Value

The path to the written file, invisibly.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("Hello, world.", type = "articles", source = "demo",
       vault = v)
status(v)
unlink(v, recursive = TRUE)
```

---

ingest_agent_context *Snapshot saber's assembled agent context into the vault*

Description

Calls saber::agent_context() to assemble the current memory / project-instructions / global-instructions / identity string for an agent, then writes it into the vault through the existing ingest() pipeline (so the manifest, index, log, and auto-commit all kick in).

Usage

```
ingest_agent_context(agent = c("claude", "codex", "corteza"),
                     vault = default_vault(), project_dir = getwd(),
                     workspace_dir = NULL, ...)
```

Arguments

agent	One of "claude", "codex", "corteza". Passed to saber::agent_context().
vault	Vault path.
project_dir	Project directory passed to saber::agent_context(). Defaults to getwd().
workspace_dir	Optional workspace dir (e.g., ~/.corteza/workspace) passed through to saber for SOUL.md / USER.md resolution.
...	Forwarded to saber::agent_context(); use this for the include_* overrides documented there.

Details

Saber stays in pensar's Suggests: the wrapper guards with requireNamespace("saber") and errors with an install hint if the package isn't available. It also gates on the agent_context() export specifically so the wrapper degrades cleanly against older saber versions (pre-0.4) that don't ship the function, instead of failing R CMD check's static analysis or crashing at runtime. Users who don't want this wrapper pay no mandatory dependency cost.

Returns silently with a message (and no write) when saber returns an empty context, so the vault doesn't accumulate empty snapshots.

Value

The relative path of the written page, invisibly. Returns NULL invisibly when saber returns an empty context.

Examples

```
## Not run:
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest_agent_context("claude", vault = v)
unlink(v, recursive = TRUE)

## End(Not run)
```

---

ingest_briefing *Generate and ingest a saber briefing*

Description

Deprecated. Use ingest_repo() with the default enrich = "auto" (which writes a briefing.md for R packages under raw/repos/<name>/) instead. This function now delegates to ingest_repo() after warning.

Usage

```
ingest_briefing(project = NULL, scan_dir = path.expand("~"),
                vault = default_vault())
```

Arguments

project	Project name. If NULL, inferred from the git root of the current working directory.
scan_dir	Directory to search for the project. Defaults to path.expand("~").
vault	Path to the vault directory.

Value

Invisibly, the path(s) written by ingest_repo().

Examples

```
## Not run:
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest_briefing(project = "pensar", vault = v)

## End(Not run)
```

---

ingest_repo *Ingest a repository snapshot into the vault*

Description

Captures repo state pinned to a commit SHA so wiki claims can cite a specific point in time. Writes one or more artifacts under raw/repos/<name>/<artifact>.md:

briefing.md type: repo-briefing – saber digest of HEAD (regenerable). Requires the saber package.

ast.md type: repo-ast – exported and internal symbols from saber::symbols() (regenerable). Requires saber.

snapshot.md type: repo-snapshot – commit-pinned metadata: SHA, origin URL, branch, tracked file listing, recent commits.

Usage

```
ingest_repo(path, name = NULL, ref = "HEAD",
            enrich = c("auto", "package", "none"),
            artifacts = c("briefing", "ast", "snapshot"), files = NULL,
            tags = NULL, vault = default_vault())
```

Arguments

path	Path to a local git repo. Tilde-expanded.
name	Repo identifier used as the directory name under raw/repos/. Defaults to basename(path).
ref	Git ref to snapshot. Default "HEAD".

enrich	One of "auto" (detect R package and enrich), "package" (force package digest; errors if no DESCRIPTION), or "none" (snapshot only). Default "auto".
artifacts	Subset of c("briefing", "ast", "snapshot") to write. Default writes all that apply for the chosen enrich mode.
files	Optional file globs (relative to the repo root) whose tracked paths are listed in snapshot.md. Contents are not stored. Default "R/*.R" for R packages, NULL otherwise.
tags	Optional character vector of tags applied to every written artifact.
vault	Path to the vault directory.

Details

Re-running overwrites the artifact files in place; git tracks history. This supersedes ingest_briefing(), which is now deprecated.

Value

Character vector of paths written, invisibly.

Examples

```
## Not run:
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest_repo("~/corteza", vault = v)

## End(Not run)
```

---

ingest_url *Ingest content from a URL*

Description

Fetches url (10s timeout), refuses non-2xx responses or content types outside text/JSON/XML, and writes the body into the vault through ingest(). If the manifest already records this URL as a source, returns the existing page path without re-fetching.

Usage

```
ingest_url(url, vault = default_vault(), type = "articles", title = NULL,
           tags = NULL)
```

Arguments

url	URL to fetch.
vault	Vault path.
type	Ingest type. Default "articles".
title	Optional page title. If NULL, derived from HTML <title>, or falls back to the URL.
tags	Optional character vector of tags.

Details

For HTML responses the page's <title> is extracted and used as the page title when title is not supplied.

Value

The relative path of the written (or existing) page, invisibly.

Examples

```
## Not run:
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest_url("https://example.com", vault = v)
unlink(v, recursive = TRUE)

## End(Not run)
```

---

init_vault *Initialize a pensar vault*

Description

Creates the vault directory structure and seeds the control files: schema.md, index.md, log.md, and (by default) agent instruction files for Claude Code and Codex.

Usage

```
init_vault(path = default_vault(), rproj = TRUE, agent_instructions = TRUE,
           adopt = FALSE, commit = NULL, force = FALSE)
```

Arguments

path	Path to the vault directory. No implicit default: pass an explicit path, or configure one via PENSAR_VAULT, use_vault(), or a walk-up schema.md marker (either in the current directory or in a vault/ subdir). Per CRAN policy pensar will not silently write to the user's home filespace.
rproj	If TRUE (default), also write an RStudio project file ({basename(path)}.Rproj). The project file makes a vault stored under a hidden directory (e.g., one configured via PENSAR_VAULT pointing at ~/.local/share/...) easy to open as an RStudio project, since RStudio's GUI normally refuses to create projects inside hidden folders. Code indexing is disabled in the project file since the vault contents are markdown, not R source. The file is a harmless ~14-line INI stub; delete it anytime if you prefer not to use RStudio. Pass rproj = FALSE to skip it entirely.
agent_instructions	If TRUE (default), write CLAUDE.md and AGENTS.md with identical content orienting an AI agent to work in this vault (CLI reminders, editing rules, ingest workflow). If you don't plan to start an AI agent session in the vault, pass FALSE.
adopt	Opt-in read-only adopt mode. When TRUE the function writes only a minimal adopted schema.md (carrying adopted: true frontmatter), plus log.md and index.md if absent. No raw/ or wiki/ scaffolding is created and no auto-commit runs. Use this when pointing pensar at an existing Obsidian vault whose layout you don't want to change. After adoption, update_index() and status() switch to registry-driven enumeration; reads work normally and ingest() refuses writes unless force = TRUE.
commit	Auto-commit gate. NULL (default) commits the initial scaffold only when the target directory is pensar-owned (empty, or already shaped like a pensar vault). TRUE commits unconditionally (after force = TRUE writes); FALSE skips the commit even for pensar-owned directories. Forcing pensar into foreign content does not by itself grant permission to commit to that content's history.
force	Write gate. FALSE (default) refuses to scaffold when the target directory already contains files or a git history that aren't pensar's. TRUE scaffolds anyway. Use sparingly.

Value

The vault path, invisibly. Returns NULL invisibly when the safety gate refused to scaffold.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
list.files(v, recursive = TRUE)
unlink(v, recursive = TRUE)
```

lint *Vault health check*

Description

Scans the vault for orphan pages (no incoming wikilinks), broken wikilinks (pointing to nonexistent pages), and tag clusters with no wiki synthesis.

Usage

```
lint(vault = default_vault(), min_cluster_size = 3L)
```

Arguments

vault	Path to the vault directory.
min_cluster_size	Minimum number of raw pages sharing a tag to suggest a wiki page. Default 3.

Value

A list with class pensar_lint.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("Refers to [[absent]].", type = "articles", source = "demo",
       vault = v)
lint(v)
unlink(v, recursive = TRUE)
```

log_entry *Append a log entry*

Description

Appends a structured entry to log.md with timestamp, operation type, and message.

Usage

```
log_entry(message, operation = "note", vault = default_vault())
```

Arguments

message	Description of what happened.
operation	Operation type (e.g., "init", "ingest", "lint").
vault	Path to the vault directory.

Value

Invisible NULL.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
log_entry("Reviewed wiki/seed.md", operation = "review", vault = v)
unlink(v, recursive = TRUE)
```

---

manifest_path *Canonical manifest path inside a vault*

Description

Returns <vault>/.pensar/manifest.yml. Pensar uses this file for per-source ingest provenance and an opt-in path -> page_uid address map.

Usage

```
manifest_path(vault)
```

Arguments

vault	Vault path.

Details

The manifest is written by ingest() and ingest_repo() after a successful page write. Read-only operations (vault_registry(), update_index(), and status()) never touch it. The directory is created lazily by update_manifest(), so simply asking for the path does not materialize .pensar/.

Value

Absolute path to the manifest file.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
manifest_path(v)
unlink(v, recursive = TRUE)
```

migrate_briefings_to_repos *Migrate legacy briefings to the repo provenance layout*

Description

Walks raw/briefings/, classifies each file as a briefing or AST artifact, maps the slug to a repo name, keeps the newest file per (repo, artifact) pair, and moves it to raw/repos/<repo>/<artifact>.md. Older duplicates are dropped if drop_old = TRUE (git history retains them).

Usage

```
migrate_briefings_to_repos(vault = default_vault(), dry_run = TRUE,
                           drop_old = TRUE,
                         rename_map = c(llamaR = "corteza", llamar = "corteza"))
```

Arguments

vault	Path to the vault.
dry_run	If TRUE (default), prints planned moves and link rewrites without touching files. Set FALSE to apply.
drop_old	If TRUE (default), removes superseded dated briefings after the chosen file is moved. If FALSE, leaves them in place.
rename_map	Named character vector mapping legacy slug stems to current repo names. Defaults handle the llamaR -> corteza rename. Pass an extended map if your vault carries other renames.

Details

Wikilinks are rewritten in wiki/*.md only – raw/ is left untouched, per the immutability rule. Aliases (e.g. [[2026-04-30-corteza|corteza]]) are preserved.

Run with dry_run = TRUE (the default) first and review the planned operations before committing.

Value

Invisibly, a data.frame with columns file, repo, artifact, action, destination.

outlinks *Find outlinks from a page*

Description

Scans a single page for [[wikilinks]] and returns the targets. Mirror of backlinks() in the forward direction.

Usage

```
outlinks(page, vault = default_vault())
```

Arguments

page	Page name (without .md extension).
vault	Path to the vault directory.

Value

A data.frame with columns target (page name) and exists (logical: whether the target page exists in the vault).

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
fp <- ingest("Cites [[seed]] and [[missing]].", type = "articles",
             source = "demo", vault = v)
outlinks(tools::file_path_sans_ext(basename(fp)), vault = v)
unlink(v, recursive = TRUE)
```

page_context *Structured context for a single page*

Description

Returns the frontmatter, a short body head, the page's outlinks, and its backlinks in one struct so callers don't have to call four functions. Resolves name through the registry, so a query like "Notes/Foo" works alongside the basename style.

Usage

```
page_context(name, vault = default_vault(), body_chars = 300L)
```

Arguments

name	Page name, path, alias, or page_uid.
vault	Vault path.
body_chars	Maximum chars of body to return. Default 300.

Value

A list with components path (relative path), node_id, frontmatter (named list), body_head (string), outlinks (data.frame), backlinks (data.frame), class = "pensar_page_context".

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
fp <- ingest("Body text here.", type = "articles", source = "demo",
             vault = v)
ctx <- page_context(tools::file_path_sans_ext(basename(fp)),
                    vault = v)
names(ctx)
unlink(v, recursive = TRUE)
```

---

pensar_skill_path *Locate pensar's bundled skill directory*

Description

Pensar ships markdown skill bundles under inst/skills/pensar/. Returns the absolute path to the bundle root, or to a specific skill when skill is given. Useful for symlinking pensar skills into an agent's skill directory, e.g. ln -s $(Rscript -e 'cat(pensar::pensar_skill_path())') \ ~/.claude/skills/pensar.

Usage

```
pensar_skill_path(skill = NULL)
```

Arguments

skill	Optional skill name (e.g., "autoresearch"). NULL returns the bundle root.

Value

Absolute path. Returns an empty string when the skill is not installed (matching system.file() behavior).

Examples

```
pensar_skill_path()
pensar_skill_path("autoresearch")
```

print.pensar_research *Print a pensar research session result*

Description

Print a pensar research session result

Usage

```
## S3 method for class 'pensar_research'
print(x, ...)
```

Arguments

x	A pensar_research object.
...	Unused.

Value

x, invisibly.

read_manifest *Read the pensar manifest for a vault*

Description

Returns a normalized list with version, created, sources, and address_map fields. When the manifest file is missing, returns a fresh empty struct with the current date as created; the manifest file is not written.

Usage

```
read_manifest(vault)
```

Arguments

vault	Vault path.

Details

Malformed YAML logs a warning and returns the empty struct so callers can keep going.

Value

A list with components version (integer), created (YYYY-MM-DD string), sources (named list of per-source records), and address_map (named list mapping relative path to page_uid).

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
read_manifest(v)$sources
unlink(v, recursive = TRUE)
```

---

recent_activity *Recent vault activity from log.md*

Description

Parses entries from `log.md` (the format written by `log_entry()`). Returns entries from the last `days` days, newest first.

Usage

```
recent_activity(vault = default_vault(), days = 7L)
```

Arguments

`vault`	Vault path.
`days`	Window in days. Default 7.

Value

A data.frame with columns `timestamp` (POSIXct), `operation`, `message`, sorted newest first.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
recent_activity(v, days = 30)
unlink(v, recursive = TRUE)
```

---

related_pages *Pages related to a target by shared tags + co-citation*

Description

Heuristic scoring: `score` = `#shared tags` + `#shared outlinks`. Both are unweighted set intersections. The target page itself is excluded from the result. Ties are broken by alphabetical `path`.

Usage

```
related_pages(name, vault = default_vault(), k = 10L)
```

Arguments

name	Page to find related pages for. Same resolution as find_page().
vault	Vault path.
k	Number of related pages to return. Default 10.

Value

A data.frame with columns path, node_id, title, score, sorted by score descending then path.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
fp <- ingest("[[other]]", type = "articles", source = "a",
             tags = c("x", "y"), vault = v)
ingest("[[other]]", type = "articles", source = "b",
       tags = c("x"), vault = v)
related_pages(tools::file_path_sans_ext(basename(fp)),
              vault = v, k = 5)
unlink(v, recursive = TRUE)
```

---

search_pages *Search pages by title, tags, aliases, or (optionally) body*

Description

Substring match (case-insensitive). Default scope is registry-only fields: title, tags, and front-matter aliases. With in_body = TRUE the body of each page is scanned too; that reads every file and is slower.

Usage

```
search_pages(query, vault = default_vault(), type = NULL, in_body = FALSE)
```

Arguments

query	Substring to search for.
vault	Vault path.
type	Optional type filter. When supplied, only pages whose registry type field equals type are considered.
in_body	If TRUE, also search the body of each page.

Value

A data.frame with columns path, node_id, title, type, and matched_in (character identifying where the substring was found: "title", "tag:<tag>", "alias:<alias>", or "body").

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("Body cites [[other]].", type = "articles",
       source = "demo-article", vault = v)
search_pages("demo", vault = v)
unlink(v, recursive = TRUE)
```

---

show_page *Show a page with its connections*

Description

Returns the page content alongside its outgoing and incoming wikilinks. Use this when you need to review or edit a page: the outlinks show what raw sources the page cites; the backlinks show what depends on it.

Usage

```
show_page(page, vault = default_vault())
```

Arguments

page	Page name (without .md extension).
vault	Path to the vault directory.

Value

A list with class pensar_page.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
fp <- ingest("Body text.", type = "articles", source = "demo",
             vault = v)
show_page(tools::file_path_sans_ext(basename(fp)), vault = v)
unlink(v, recursive = TRUE)
```

status *Vault status summary*

Description

Returns page counts by category, total pages, and wikilink count. When `vault` is `NULL` (default), the vault is resolved via `PENSAR_VAULT`, walk-up from `getwd()`, or `options("pensar.vault")`, and the source of the match is recorded for display.

Usage

```
status(vault = NULL)
```

Arguments

`vault`	Path to the vault directory. `NULL` (default) triggers automatic resolution.

Value

A list with class `pensar_status`, including the resolved `vault` path and a `source` label (`"env"`, `"walkup"`, `"walkup-subdir"`, `"option"`, or `"explicit"`).

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
status(v)
unlink(v, recursive = TRUE)
```

tags *Audit tag usage against a controlled vocabulary*

Description

Reads every tag from the registry, optionally compares against a taxonomy file (`_meta/taxonomy.md`, a markdown bullet list of allowed tags), and writes a proposals report to `_proposals/tags.md`. Unknown tags get near-miss suggestions via Jaro-Winkler distance against the taxonomy.

Usage

```
tags(vault = default_vault(), taxonomy = NULL, near_miss_threshold = 0.15)
```

Arguments

vault	Vault path.
taxonomy	Optional path to a taxonomy file. Defaults to <vault>/_meta/taxonomy.md when present, otherwise no taxonomy is loaded and the report lists all used tags by frequency.
near_miss_threshold	Maximum Jaro-Winkler distance for an unknown tag to be suggested as a typo of a taxonomy entry. Default 0.15.

Details

Pensar never auto-renames. The proposals file is for human review.

Value

A list with components used (data.frame of tag / count), unknown (data.frame of unknown tags with optional suggestions), and unused_taxonomy (character vector of taxonomy entries with zero usage). Invisible.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("hi", type = "articles", source = "demo",
       tags = c("foo", "bar"), vault = v)
tags(v)
unlink(v, recursive = TRUE)
```

---

update_index *Update the vault index*

Description

Scans all markdown files in the vault and regenerates index.md as a categorized catalog with wikilinks and titles.

Usage

```
update_index(vault = default_vault())
```

Arguments

vault	Path to the vault directory.

Value

The path to index.md, invisibly.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("Body.", type = "articles", source = "demo", vault = v)
update_index(v)
unlink(v, recursive = TRUE)
```

`update_manifest` *Update the pensar manifest for a vault*

Description

Patches a manifest record for one page. Writes *only* `.pensar/manifest.yml`; never edits other tools' manifest formats (`.manifest.json`, `.raw/.manifest.json`).

Usage

```
update_manifest(vault, source = NULL, path = NULL, page_uid = NULL,
                address = NULL, hash = NULL, ingested_at = NULL)
```

Arguments

`vault`	Vault path.
`source`	Source identifier (URL, session id, etc.). Optional.
`path`	Relative path inside the vault that this update is about. Required if any of the other fields are set.
`page_uid`	Stable page identity from frontmatter `id` / `address`. Goes into both the `sources` record (when other source fields are set) and the `address_map`.
`address`	Alias for `page_uid` that only writes to `address_map`. Useful when the caller wants to record an address without touching the source record.
`hash`	Content hash (typically `paste0("sha1:", ...)` from `digest::digest()`).
`ingested_at`	Timestamp string. Defaults to current time when any source-shaped field is set.

Details

`source`, `hash`, or `ingested_at` together write or refresh a `sources[path]` record. `page_uid` (or `address`, which is treated as an alias) writes an `address_map[path]` record. A call with only `path` and `page_uid` updates the address map without touching the `sources` record.

Value

The manifest path, invisibly.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
update_manifest(v, source = "demo",
                path = "raw/articles/demo.md",
                hash = "sha1:abc")
read_manifest(v)$sources[["raw/articles/demo.md"]]
unlink(v, recursive = TRUE)
```

---

use_vault *Remember a vault path for this R session*

Description

Sets `options("pensar.vault")` so subsequent pensar calls resolve to `path` without repeating the argument. Persist by adding `pensar::use_vault("~/wiki")` to `~/.Rprofile` as a global default. Both `PENSAR_VAULT` and a project-local `schema.md` found via walk-up (in the current directory or a `vault/` subdir) will override this option (see `default_vault` resolution order).

Usage

```
use_vault(path)
```

Arguments

`path`	Path to your pensar vault directory.

Value

The resolved path, invisibly.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
use_vault(v)
status()
options(pensar.vault = NULL)
unlink(v, recursive = TRUE)
```

vault_commit *Commit vault changes to git*

Description

No-op if the vault is not a git repo or if there are no changes. Stages all changes (respecting `.gitignore`), commits with the given message, and optionally pushes to remotes.

Usage

```
vault_commit(message, vault = default_vault(), push = NULL)
```

Arguments

message	Commit message.
vault	Path to the vault directory.
push	If NULL (default), honors PENSAR_AUTO_PUSH. Pass TRUE or FALSE to override.

Details

Honors the PENSAR_AUTO_PUSH environment variable: if set to "0" or "false" (case-insensitive), skips the push step. Otherwise, pushes to every configured remote.

Value

TRUE if a commit was made, FALSE otherwise (invisibly).

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
# Returns FALSE invisibly: no .git in this temp vault.
vault_commit("noop", vault = v, push = FALSE)
unlink(v, recursive = TRUE)
```

vault_export *Export the vault to static HTML*

Description

Renders every markdown page in the vault to HTML, resolving [[wikilinks]] to relative anchor tags. Output is a standalone site that can be served from any static file server or opened via file://.

Usage

```
vault_export(vault = default_vault(), out_dir = default_site_dir())
```

Arguments

vault	Path to the vault directory.
out_dir	Destination directory. No default: pass an explicit path or set PENSAR_SITE_DIR (per CRAN policy, pensar will not silently render into a home-filespace location).

Details

No default out_dir: pass an explicit path or set the PENSAR_SITE_DIR environment variable. Per CRAN policy pensar will not silently render into a home-filespace location (e.g., tools::R_user_dir()). PENSAR_SITE_DIR is the recommended escape hatch – point it at a Syncthing folder so edits propagate to other devices on export.

Requires the pandoc command-line tool to be available.

Export is incremental. The first export to a given out_dir is a full build; subsequent exports re-render only pages whose source changed, plus any page whose wikilinks point at a name that was added, removed, or moved (so broken/resolved links stay correct). State is kept in .pensar-export-cache.yml inside out_dir; delete it to force a full rebuild. The stylesheet and site index are always regenerated (they don't invoke pandoc).

Value

The output directory path, invisibly.

Examples

```
if (nzchar(Sys.which("pandoc"))) {
    v <- tempfile("vault-")
    init_vault(v, rproj = FALSE, agent_instructions = FALSE)
    ingest("Body.", type = "articles", source = "demo", vault = v)
    vault_export(v, out_dir = tempfile("site-"))
    unlink(v, recursive = TRUE)
}
```

---

vault_graph *Render a vault's wikilink graph as SVG*

Description

Scans every markdown page in the vault (excluding control files), extracts [[wikilinks]] as edges, and renders the result via saber::graph_svg(). Node tooltips carry the page type, tags, and date from YAML frontmatter; broken wikilinks (targets with no matching page) appear as external nodes with a distinct tooltip.

Usage

```
vault_graph(vault = default_vault(), width = 1600L, height = 1200L, ...)
```

Arguments

vault	Path to the vault directory.
...	Passed through to saber::graph_svg() (e.g., iterations, seed).
width, height	Viewport in pixels. Defaults (1600 x 1200) are larger than saber::graph_svg()'s defaults since vaults tend toward many nodes.

Value

Character vector of SVG lines. Write with writeLines().

Examples

```
## Not run:
# Requires a version of 'saber' that exports graph_svg().
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("Cites [[other]].", type = "articles", source = "demo",
       vault = v)
svg <- vault_graph(v)
writeLines(svg, tempfile(fileext = ".svg"))

## End(Not run)
```

---

vault_registry *Build a structured registry of a vault's pages*

Description

Scans every .md file in vault and returns a data.frame with one row per page. Used internally to resolve wikilinks and query frontmatter without re-scanning the filesystem on every call.

Usage

```
vault_registry(vault = default_vault(), cache = c("session", "user", "none"),
               refresh = FALSE)
```

Arguments

vault	Vault path.
cache	Cache policy: "session" (default), "user", or "none".
refresh	If TRUE, rebuild and overwrite the cache.

Details

Identity columns:

- node_id is the current link-resolution identity (path-aware basename via name_from_path()). This is what [[page]] matches against.
- page_uid is a stable identity sourced from frontmatter id: or address:. NA if the page declares neither. Stable identity is opt-in via frontmatter; pensar never fabricates one from a path hash, because path hashes change on rename.

Cache levels:

- "session" (default): memoized in a package-level environment, keyed by vault path. No disk write. Invalidates when any .md file's mtime changes.
- "user": persisted to tools::R_user_dir("pensar", "cache"). CRAN-safe location; never writes inside the vault itself (.pensar/ is reserved for vault-owned state).
- "none": rebuild on every call.

Value

A data.frame with columns: path, node_id, page_uid, title, aliases, type, category, tags, sources, links_out, system_file. Aliases / tags / links_out are list-columns. The type and category fields come from frontmatter verbatim; callers compute an effective type as type when present, falling back to category otherwise.

Examples

```
v <- tempfile("vault-")
init_vault(v, rproj = FALSE, agent_instructions = FALSE)
ingest("Body cites [[other]].", type = "articles", source = "demo",
       vault = v)
reg <- vault_registry(v)
nrow(reg)
unlink(v, recursive = TRUE)
```

Index

autoresearch, 2

backlinks, 4

dedup, 5
default_vault, 6

ingest, *6*, 6
ingest_agent_context, 7
ingest_briefing, 8
ingest_repo, *7*, *8*, 9
ingest_url, 10
init_vault, 11

lint, 13
log_entry, 13

manifest_path, 14
migrate_briefings_to_repos, 15

outlinks, 16

page_context, 16
pensar_skill_path, 17
print.pensar_research, 18

read_manifest, 18
recent_activity, 19
related_pages, 19

search_pages, 20
show_page, 21
status, *6*, 22

tags, 22

update_index, 23
update_manifest, 24
use_vault, *6*, 25

vault_commit, 26
vault_export, 26
vault_graph, 27
vault_registry, 28