OpenVault Documentation (Combined Pages)

=== OpenVault ===

Intro
OpenVault is designed to mirror how cognition works, turning complex operations into a clear, navigable system.
This Obsidian vault serves as the central nervous system for OpenClaw ? an AI orchestration platform that manages tasks, automations, and knowledge.
Each Obsidian folder maps to a functional region of the human brain, creating an intuitive architecture for how information flows, gets processed, stored, and acted upon. The UI design helps both humans and AI agents navigate it naturally.

Architecture: Brain vs Body
The system is split into two distinct locations. The Obsidian Vault is the brain ? it remembers, tracks, and documents rules and processes. The Projects folder is the body ? it holds the actual work product, source code, and project-specific documentation.
Brain vs BodyThree core parts and the connections between them
Obsidian Vault
~/OpenVault/Brain: memory, planning, and tracking?
OpenClaw
Agent LayerReads vault context and works in projects?
Projects
~/Projects/Body: source code, docs, and delivery
Why Separate?
Mixing orchestration files with source code creates git merge conflicts, bloated vaults, Obsidian indexing thousands of code files, and agents accidentally editing source code. The separation keeps each system clean and purpose-built.

Why Obsidian Vault?
Keeping this system in Obsidian reduces cognitive load by making critical context easy to scan instead of repeatedly searching code repositories. Linked notes, graph relationships, and structured dashboards speed decisions by making dependencies visible at a glance for both humans and agents. By separating planning from implementation noise, the vault supports faster retrieval, steadier execution, and more sustainable day-to-day operations.

The Two Locations

| | Obsidian Vault (Brain) | Projects Folder (Body) |
|---|---|---|
| Location | ~/OpenVault/ | ~/Projects/ |
| Purpose | Memory, orchestration, rules, tracking, knowledge | Source code, project docs, configs, GitHub repos |
| Git | Vault backup repo (private) | Each project is its own GitHub repo |
| OpenClaw | Reads for HOW to work (skills, prompts, routines) | Reads for WHAT to work on (code, docs, issues) |
| Bridge | tasks.md links to GitHub | CLAUDE.md and docs/ provide context |

The Bridge: tasks.md ? The only project file inside the vault. One per active project. Contains task status, priority, assignee, due dates, and GitHub issue links. Just enough for the Dashboard to aggregate, but no project-specific documentation.
The 5-Step Workflow
? Check the brain ? Read 015-Dashboard for active focus, scan 010-Projects/Active/ for tasks, load skills/prompts/routines
? Go to the workbench ? Navigate to ~/Projects/{{name}}/, read docs/overview.md,

docs/workflow.md, and CLAUDE.md
? Do the work ? Operate on source files in src/ using vault’s skills and prompts. All code changes in Projects folder, never vault.
? Update the brain ? Update tasks.md with status changes, log in 110-Logs/Agent-Runs/, Dashboard auto-aggregates
? Push to GitHub ? Commit and push code changes. tasks.md links to relevant issues and PRs.

=== Obsidian ===

Obsidian Vault
This page documents the Obsidian Vault itself: the folder map, each brain-inspired section, and how information moves through the vault from capture to long-term knowledge. Folders are numbered (000?130) to maintain consistent sort order across file browsers, CLI tools, and agent file access.

Folders

| Folder | Brain Region | Core Function |
|---|---|---|
| 000-Inbox | Sensory Input | Raw captures, unsorted notes, clippings before processing |
| 010-Projects | Prefrontal Cortex | Executive planning, active project management |
| 015-Dashboard | Thalamus | Central routing, system status, daily briefing |
| 020-Tasks | Working Memory | Immediate task tracking: today, upcoming, someday |
| 025-Watchlist | Reticular Activating System | Attention filtering, alerts, monitors, feeds |
| 030-Agents | Neural Pathways | Agent configs, logs, templates, and memory |
| 035-Priorities | Amygdala | Urgency tagging and priority classification |
| 040-Skills | Cerebellum | Procedural memory: reusable skills and commands |
| 050-MCP | Synapses | MCP server connections, tools, resources |
| 060-Prompts | Neocortex | Higher reasoning: system prompts, chains, fragments |
| 070-Routines | Autonomic NS | Automatic + scheduled + triggered workflows |
| 080-Bookmarks | Temporal Lobe | Reference storage: links, articles, tools |
| 090-Knowledge | Long-term Memory | Domain knowledge: engineering, trading, devops, health |
| 095-Connections | Corpus Callosum | Cross-linking, maps of content, tag indexes |
| 100-Context | Prefrontal Context | CLAUDE.md files, rules, personas, scratchpads |
| 110-Logs | Hippocampus | Episodic recording: daily, weekly, decisions, agent runs |
| 120-Templates | DNA / Blueprints | Replicable patterns for all vault content types |
| 130-Archive | Deep Storage | Retired content preserved for reference |

000-Inbox
Sensory InputThe default landing zone for anything captured quickly ? ideas, links, screenshots, voice memos, clippings from the web, or quick notes. Nothing lives here permanently; it gets triaged into the appropriate folder during regular processing.
? Process the Inbox at least once daily (or set up an agent to auto-triage)
? Keep items here less than 48 hours
? Use routing-rules.md in 015-Dashboard to define where items go

010-Projects
Prefrontal CortexEach project gets its own subfolder under Active, On-Hold, or Completed. Projects contain everything needed to execute: an overview, task lists, phase breakdowns, logs, and reference materials.

| File / Subfolder | Purpose |
|---|---|
| Active/ | Projects currently being worked on |
| ProjectName/overview.md | Project summary: goals, scope, stakeholders, timeline |
| ProjectName/tasks.md | Project-specific task list with status tracking |

| | |
|---|---|
| ProjectName/phases/ | Phase breakdowns (e.g., phase-1-design.md, phase-2-build.md) |
| ProjectName/logs/ | Project-specific activity logs and meeting notes |
| ProjectName/references/ | Supporting docs, links, specs relevant to this project |
| On-Hold/ | Projects paused but not abandoned |
| Completed/ | Finished projects preserved for reference |

015-Dashboard

ThalamusThe nerve center of the vault. Provides at-a-glance system status, tells OpenClaw what to focus on, generates daily briefings, and defines routing rules for Inbox items.

| File | Purpose |
|---|---|
| system-status.md | Health status of all automations, agents, backups, and cron jobs |
| active-focus.md | What OpenClaw should prioritize RIGHT NOW ? the current directive |
| daily-brief.md | Auto-generated morning summary: tasks due, alerts, calendar, priorities |
| routing-rules.md | Rules defining how Inbox items get auto-sorted into folders |

020-Tasks

Working MemoryIndividual action items that need to be done. The tasks.md file uses the OpenClaw task tracker format with status emojis, timestamps, and agent attribution.

| File | Purpose |
|---|---|
| today.md | Tasks that must be completed today |
| upcoming.md | Tasks scheduled for the near future (this week / next week) |
| someday.md | Ideas and tasks with no firm deadline ? the backlog |

025-Watchlist

Reticular Activating SystemItems that need passive monitoring. Rather than requiring active work, these trigger attention only when specific conditions are met.

| Subfolder | Purpose |
|---|---|
| Alerts/ | Active alert conditions and thresholds (e.g., price drops, HRV below Y) |
| Monitors/ | Things being tracked: stock prices, health metrics, project deadlines |
| Feeds/ | RSS feeds, API endpoints, data streams, and webhooks to watch |

030-Agents

Neural PathwaysDefines WHO does the work. Each agent has a configuration, produces logs, and can draw on persistent memory split between working, episodic, and semantic memory.

| Subfolder | Purpose |
|---|---|
| Configs/ | Agent definitions: system prompts, personas, tool access, capabilities |
| Logs/ | Agent run history: what they did, when, and results |
| Templates/ | Reusable agent templates for spinning up new agents quickly |
| Memory/Working/ | Current session context and scratchpads ? clears regularly |
| Memory/Episodic/ | Records of specific events and interactions over time |
| Memory/Semantic/ | Consolidated facts, learned patterns, permanent knowledge |

035-Priorities

AmygdalaThe vault’s triage system. Agents auto-classify items using priority-rules.md; humans can override at any time.

| File | Purpose |
|---|---|
| critical.md | ? Must act immediately ? blockers, failures, urgent deadlines |
| high.md | ? This week ? important but not emergency |
| normal.md | ? Scheduled and on track ? proceeding as planned |
| low.md | ? Backlog ? nice to have, no deadline pressure |
| priority-rules.md | Rules for how agents should auto-classify urgency levels |

040-Skills

CerebellumDefines WHAT agents can do. Skills are reusable, composable, and can be shared across agents and tools.

| Subfolder | Purpose |
|---|---|
| Claude-Code/Commands/ | Reusable slash commands and code snippets for Claude Code |
| Claude-Code/Hooks/ | Pre/post hooks and event handlers for Claude Code workflows |
| OpenClaw/ | OpenClaw-specific skill definitions and capability files |
| Shared/ | Skills usable across any agent or tool ? platform-agnostic |
| skill-index.md | Registry of all available skills with descriptions and status |

050-MCP
SynapsesThe communication layer connecting agents to external tools and data sources via Model Context Protocol.

| Subfolder | Purpose |
|---|---|
| Servers/ | MCP server configs and connection details (filesystem, GitHub, Obsidian, etc.) |
| Tools/ | Custom tool definitions exposed via MCP to agents |
| Resources/ | MCP resource definitions ? data sources agents can read |
| Prompts/ | MCP prompt templates served to connected clients |
| mcp-registry.md | Master index of all MCP servers, their tools, and capabilities |

060-Prompts
NeocortexInstructions that direct higher-order processing. Separated from agent configs because prompts are composable and reusable across agents.

| Subfolder | Purpose |
|---|---|
| System/ | System prompts and CLAUDE.md files that define agent behavior |
| Task/ | Reusable task-specific prompts (e.g., summarize, analyze, generate) |
| Chains/ | Multi-step prompt chains and workflows (research ? summarize ? file) |
| Fragments/ | Composable prompt snippets and partials for assembly |

070-Routines
Autonomic NSBackground processes split into conscious (scheduled, triggered) and unconscious (autonomic), mirroring the brain’s voluntary and involuntary actions.

| Subfolder | Purpose |
|---|---|
| Autonomic/ | Always-on processes: backups, syncs, heartbeats, health checks |
| Scheduled/ | Cron jobs and weekly tasks ? set and forget |
| Triggered/ | Event-driven rules: when X happens, do Y |
| Pipelines/ | Multi-step orchestrated workflows chaining multiple actions |
| Checklists/ | Manual routines and SOPs: daily standup, weekly review, monthly audit |

080-Bookmarks
Temporal LobeCurated external references organized by type.

| File | Purpose |
|---|---|
| tools.md | Software, SaaS products, CLIs, and utilities |
| articles.md | Saved articles, blog posts, and long-form reads |
| repos.md | GitHub repositories and open-source projects |
| tutorials.md | How-to guides, courses, and learning resources |
| reference-sites.md | Frequently visited documentation sites and reference portals |

090-Knowledge
Long-term MemoryYour own notes, calculations, strategies, and accumulated expertise organized by domain. What agents reference for domain-specific context.

| Subfolder | Purpose |
|---|---|
| Engineering/ | Natural gas calcs, odorizer specs, pipeline notes, station data |
| Trading/ | Strategies, setups, portfolio notes, market analysis |

| | |
|---|---|
| DevOps/ | Server configs, Coolify notes, Tailscale setup, hosting docs |
| Health/ | HRV insights, sleep data analysis, fitness protocols, wellness tracking |

095-Connections
Corpus CallosumThe meta-layer that links knowledge across domains. Prevents folders from becoming isolated islands.

| File | Purpose |
|---|---|
| map-of-content.md | Master index (MOC) linking related notes across the entire vault |
| tag-index.md | Canonical tags used across the vault with definitions |
| dependency-graph.md | What depends on what: projects ? skills ? agents ? MCP servers |
| Relations/ | Explicit link files connecting items across domains |

100-Context
Prefrontal ContextShapes HOW agents behave in different situations through identity, rules, and contextual awareness.

| Subfolder | Purpose |
|---|---|
| CLAUDE-files/ | CLAUDE.md files for different projects and repos |
| Rules/ | Coding standards, .cursorrules, .clinerules, style guides |
| Personas/ | Role definitions: "act as PM," "act as trader," "act as engineer" |
| Scratchpads/ | Working memory for active agent sessions ? temporary, disposable |

110-Logs
HippocampusEverything that happens gets recorded here. Creates an audit trail and enables agents to learn from past actions.

| Subfolder | Purpose |
|---|---|
| Daily/ | Daily journal entries and standup notes |
| Weekly/ | Weekly review summaries and retrospectives |
| Decisions/ | Decision records with context, options considered, and rationale |
| Agent-Runs/ | Timestamped logs of agent executions: what ran, when, and output |

120-Templates
DNA / BlueprintsStandardized patterns ensuring consistency when creating new content.

| File | Purpose |
|---|---|
| project-template.md | Standard structure for new project folders |
| task-template.md | Format for individual task entries with status tracking |
| daily-log-template.md | Daily journal template with sections for wins, blockers, and focus |
| agent-config-template.md | Standard format for defining a new agent |
| skill-template.md | Template for documenting a new skill |
| mcp-server-template.md | Template for registering a new MCP server connection |
| prompt-template.md | Template for creating reusable prompts |

130-Archive
Deep StorageRetired content that’s no longer active but may be useful someday. Move completed projects, outdated configs, and deprecated skills here rather than deleting them.

How Information Flows Through the Vault
The 9-step lifecycle: Input ? Routing ? Prioritization ? Execution ? Monitoring ? Recording ? Consolidation ? Cross-linking ? Retirement? Input (Sensory) ? New information enters through 000-Inbox
? Routing (Thalamus) ? 015-Dashboard/routing-rules.md determines where items go
? Prioritization (Amygdala) ? Items get urgency tags in 035-Priorities
? Execution (Prefrontal + Cerebellum) ? Tasks get worked using Skills, executed by Agents, following Routines
? Monitoring (RAS) ? 025-Watchlist passively monitors and triggers alerts

? Recording (Hippocampus) ? Everything logged in 110-Logs
? Consolidation (Long-term Memory) ? Insights move to 090-Knowledge
? Cross-linking (Corpus Callosum) ? 095-Connections prevents silos
? Retirement (Deep Storage) ? Completed content moves to 130-Archive

Project Folder Layout

| File / Subfolder | Purpose |
|---|---|
| src/ | Source code, application logic, components, scripts |
| docs/ | All project-specific documentation |
| logs/ | Project-specific activity logs and change history |
| CLAUDE.md | Project-specific agent context and coding instructions |
| .env.example | Template for environment variables (no real values) |
| README.md | Standard GitHub readme for the repository |

The docs/ folder

| File | Purpose |
|---|---|
| docs/overview.md | Project summary: goals, scope, stakeholders, timeline |
| docs/workflow.md | Data flow, architecture diagrams, what triggers what |
| docs/tools.md | Tools, APIs, platforms, third-party services used |
| docs/data-map.md | Database schemas, API endpoints, data sources |
| docs/secrets.md | Environment variable names and vault paths ? never credentials |
| docs/phases/ | Phase breakdowns: phase-1-design.md, phase-2-build.md, etc. |

=== Folders ===

Project Folder
This page focuses on project execution standards after folder setup, including phase management, tasks.md structure, and build-vs-ops task handling.

Phases & Task Management
Every project follows a six-phase lifecycle. Phases provide structure for how work progresses from idea to production, and every task is tagged with the phase it belongs to.

The Six Phases

| # | Phase | Code | Gate (advance when?) |
|---|---|---|---|
| 1 | Planning & Research | 1-Plan | Goals defined, feasibility confirmed, resources identified |
| 2 | Design (UI/UX) | 2-Design | Wireframes approved, UI/UX finalized, prototype reviewed |
| 3 | Development | 3-Dev | Core features built, APIs integrated, code reviewed |
| 4 | Testing (QA) | 4-QA | Critical bugs fixed, performance acceptable, security passed |
| 5 | Deployment | 5-Deploy | Live on production, monitoring active, rollback plan ready |
| 6 | Maintenance | 6-Maint | Ongoing ? no exit gate |

Overlap rules: Phases are overlapping, not rigid sequential. Future-phase tasks can be created and queued but must stay in Queued status until their phase activates. A phase activates when the previous phase’s gate criteria are met.

The tasks.md Specification
One tasks.md exists per active project at 010-Projects/Active/{{ProjectName}}/tasks.md inside the vault.

Header Fields

| Field | Description |
|---|---|
| Repo | GitHub repository URL |
| Local | Local filesystem path (~/Projects/{{name}}) |
| Docs | Path to project documentation (~/Projects/{{name}}/docs/) |
| Branch | Active git branch (typically main) |
| Prefix | 2?3 letter project prefix for globally unique task IDs |
| Current Phase | Active phase with emoji indicator (e.g., ? 3 ? Development) |
| Phase Progress | Visual bar showing all six phases (e.g., ??????) |
| Last Synced | Timestamp of last update |

Active Tasks Table Columns

| Column | Description |
|---|---|
| ID | Globally unique: project prefix + sequential number (e.g., ST-001) |
| Task | Short, actionable description |
| Task Type | Workstream classification: Build or Ops |
| Phase | Phase code: 1-Plan, 2-Design, 3-Dev, 4-QA, 5-Deploy, 6-Maint |
| Status | ? Queued  ? In Progress  ? Completed  ? Failed  ?? Paused  ? Blocked |
| Priority | ? Critical  ? High  ? Normal  ? Low |
| Assignee | claude-code, openclaw, manual, or ? |
| Due | Target completion date |
| GitHub | Link to GitHub issue or PR |

Example tasks.md

```
# {{ProjectName}} ? Tasks
Repo: github.com/{{username}}/{{repo-name}}
Local: ~/Projects/{{repo-name}}
Docs: ~/Projects/{{repo-name}}/docs/
Branch: main
Prefix: {{PX}}-
Current Phase: ? 3 ? Development
Phase Progress: ??????
Last Synced: 2026-02-16 09:00
???
## Active ? Build
| ID | Task | Task Type | Phase | Status | Priority | Assignee | Due | GitHub |
| {{PX}}-001 | Build alert API | Build | 3-Dev | ? In Progress | ? High | claude-code | 2/18 | #12 |
| {{PX}}-002 | Rebalance logic | Build | 3-Dev | ? Queued | ? Normal | openclaw | 2/20 | #14 |
| {{PX}}-003 | Write integration tests | Build | 4-QA | ? Queued | ? Low | ? | ? | ? |
| {{PX}}-004 | Deploy to staging | Build | 5-Deploy| ? Queued | ? Normal | ? | ? | ? |

## Active ? Ops
| ID | Task | Task Type | Category | Status | Priority | Assignee | Due | GitHub |
| {{PX}}-050 | Fix null pointer in parser | Ops | ? Bug | ? In Progress | ? High | claude-code | 2/16 | #15 |
| {{PX}}-051 | Update axios to v1.7 | Ops | ? Dep | ? Queued | ? Low | ? | ? | ? |
| {{PX}}-052 | Add rate limiting | Ops | ? Security | ? Queued | ? High | ? | 2/19 | #16 |
???
## Completed
| ID | Task | Type | Completed | Duration | GitHub | Notes |
| {{PX}}-000 | Scaffold project repo | Build | 2026-02-12 | 2h | #1 | Initial setup |
| {{PX}}-049 | Fix CORS headers | Ops | 2026-02-14 | 20m | #11 | Config fix |

## Blocked
```

| ID | Task | Type | Blocked By | Since | Notes |
|---|---|---|---|---|---|
| {{PX}}-003 | Write integ tests | Build | {{PX}}-001 | 2026-02-15 | Needs API schema finalized |

Build Tasks vs Ops Tasks
Not every task follows a six-phase lifecycle. The system recognizes two distinct task types to avoid unnecessary overhead on one-off changes.

Two Task Types
BUILD Follow the six-phase lifecycle. Features, new systems, or major changes that require planning through deployment. They have phases, dependencies, deliverables, and gate criteria.
OPS Single-action items with no phase progression. Bug fixes, config changes, dependency updates, quick improvements. Created, worked, and closed.

How to Tell Them Apart

| | Build Task | Ops Task |
|---|---|---|
| Scope | Multi-step, multi-day or multi-week | Single action, hours to a day |
| Phases | Tagged with 1-Plan through 6-Maint | No phase ? tagged with a category |
| Dependencies | Often blocked by other tasks or phases | Usually standalone |
| Deliverable | Maps to phase objectives and milestones | The fix or change IS the deliverable |
| Example | Build portfolio rebalance engine | Fix null pointer in parser |
| Lifecycle | Queued ? In Progress ? Completed (across phases) | Queued ? In Progress ? Done |

Ops Task Categories

| Emoji | Category | Use For |
|---|---|---|
| ? | Bug | Defects, errors, broken behavior, regressions |
| ? | Dep | Dependency updates, library upgrades, version bumps |
| ? | Security | Vulnerability patches, auth fixes, hardening |
| ? | Perf | Performance improvements, optimization, caching |
| ? | Config | Environment changes, settings, infrastructure tweaks |
| ? | Docs | Documentation updates, README, inline comments |
| ? | Investigate | Research, debugging, root cause analysis |
| ? | Chore | Refactoring, cleanup, tech debt, code hygiene |

Task ID Numbering

| Type | ID Range | Example |
|---|---|---|
| Build | {{PX}}-001 to {{PX}}-049 | ST-001, ST-002, ST-003 |
| Ops | {{PX}}-050 to {{PX}}-999 | ST-050, ST-051, ST-052 |

Dashboard Aggregation
## Project Status ? {{ProjectName}}
Phase: ? 3-Dev | ??????

### Build Tasks

| In Progress | Queued | Blocked | Done |
|---|---|---|---|
| 1 | 2 | 1 | 1 |

### Ops Tasks

| In Progress | Queued | Done (7d) |
|---|---|---|
| 1 | 2 | 1 |

### ?? Attention
? {{PX}}-050: Fix null pointer (High, in progress)

? {{PX}}-052: Add rate limiting (High, due 2/19)

Revision Log

| Rev | Date | Changes