

CaliberVaultII — End-to-End Pipeline, Verification, and Handoff

This document captures the current automation and QA workflow for CaliberVaultII as of November 2025.

Key Directories

- Repo: ``/Users/ghogue/Desktop/CaliberVaultII``
- Overlay: ``/Users/ghogue/Desktop/CaliberVaultII-From-Famous-``
- Public Mirror: ``geraldhogue/CaliberVaultII-public`` (used by Famous)
- Sentinel: ``HARNESS:2025-11-11T20:45Z:fa35a7e7``

Primary Commands

Command	Function
<code>`cvpipe`</code>	overlay → install → test → publish → verify
<code>`cvmemo`</code>	generates send-ready memo with results
<code>`cvrelease`</code>	runs cvpipe + cvmemo + copies memo to clipboard

Pipeline Overview

1. Overlay Sync

Syncs latest Famous drop into repo via rsync (excludes `.git`, `node_modules`, `scripts`, `test-artifacts`).

2. Install

Runs ``npm ci`` or ``npm install`` (deterministic, quiet).

3. TypeCheck

Executes ``tsc -b``; falls back to `commons.json` if needed.

4. Tests

Runs Vitest with verbose reporter; logs to ``test-artifacts/vitest.out.txt``.

5. Publish

Pushes results to public mirror using SSH over port 443; verifies raw files return HTTP 200.

6. Verify

Prints commit SHA (``main_commit``) and file fingerprint (``vitest_sha256``).

Harness Policy

- ``vitest.setup.ts`` is **not frozen** but changes require rationale.
- Famous may update it per drop only with clear documentation.
- Regressions trigger rollback to the previous version.

Known Failure Surfaces and Fixes

1. Supabase Builder: implement ``insert()`` and chainable ``eq()``.
2. Auth Client: ensure ``getSession()`` and ``onAuthStateChange()`` exist on same module.
3. Realtime Client: provide ``client.channel().on().subscribe()`` mock.
4. API Service: ``getAll()->array``, ``create()->object``, ``error->reject``.
5. InventoryOperations: align export/import (named vs default).
6. SmartInstallPrompt: fire ``beforeinstallprompt`` and call ``prompt()``.
7. Barcode Cache: clamp delays; remove 10s waits.
8. ErrorHandler: export constructor + function with correct names.

Verification Protocol

Each run must publish three verifiable artifacts:

1. ``vitest.setup.ts``
2. ``vitest.override.ts``
3. ``vitest.out.txt``

Famous must confirm by pasting:

- ``main_commit`` SHA

- `vitest.out.txt` SHA-256
- harness sentinel at runtime
- first & last 3 lines of `vitest.out.txt`

Diagnostics Tools

- `**pipe-doctor.sh**`: checks overlay sync, remotes, push. Run with `--fix`` to repair.
- `**harness.selftest.spec.ts**`: fails fast when mocks/builders are missing.

Memo Template

`**Subject:**` CaliberVaultII — consume these exact, current artifacts

`**Status:**` Progress / No change / Regression

Artifacts: three raw GitHub links

Verification values: commit + sha256

Summary: test counts, durations

Fix List: explicit remaining issues

Policy: harness expectations

Troubleshooting

- Public stale → update OV path, confirm remote, re-run ``cvpipe``.
- Zsh parse error → never paste TS directly, use heredoc redirection.
- Public 404s → ensure ``gitignore`` has ``!test-artifacts/**``.

Usage Flow

1. ``cvpipe``
2. ``cvmemo`` or ``cvrelease``
3. Send memo to Famous
4. Receive new overlay, repeat.