CASE 05 / PERSONAL PROJECT

Vault

An offline vault for passwords, documents, photos and video. No account, no server, nothing on the network to steal.

WHEN	PLATFORMS	MY PART
Aug 2026	Mac	Concept and direction, visual design, how it is used

01

Background and goal

How it was before, what broke, and what the goal was

BEFORE

Every service wants a password. I could not remember them, so I wrote them down. Once written down, they were not safe. I did not trust the built-in password storage, and I did not trust the online password services either.

THE MOMENT IT BROKE

When I had to put things in a folder and then put a password on it as well. This is the opposite of Enigmaa. I simply wanted to build a virtual safe, but not one that relies on the usual password technology.

GOAL

An offline vault, not a password manager. The point is not stronger encryption but having nothing on the network to steal. Passwords, documents, photos and video sit under one lock, because only a place that never goes online is a place for a passport scan.

02

Process

1 **Skeleton**
15 Aug
Four tabs, one data structure. A first version the same day
SwiftUI

2 **Encryption layer**
15 Aug
One master key, wrapped by several keys

3 **Three columns, search**
15 to 16 Aug
Summary, username, password. Search that skips the password itself (v0.3.5)

4 **Public source, notes**
17 Aug
Personal details removed, source made public on GitHub, development notes written
GitHub

5 **Film and landing page**
19 to 20 Aug
A 146-second film. The page rewritten around the offline positioning
HyperFrames
Netlify

6 **v0.3.6, in daily use**
30 Aug
Document attachments can be previewed and downloaded. I have used it myself since

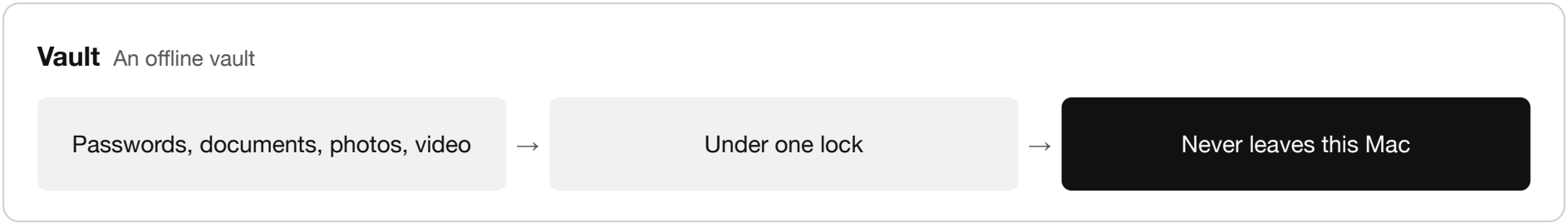
03

The idea

Not harder to steal. Nothing to steal.

A password manager The usual shape

Store passwords → Sync through someone's server → Autofill on every device → Protection rests on trusting that service

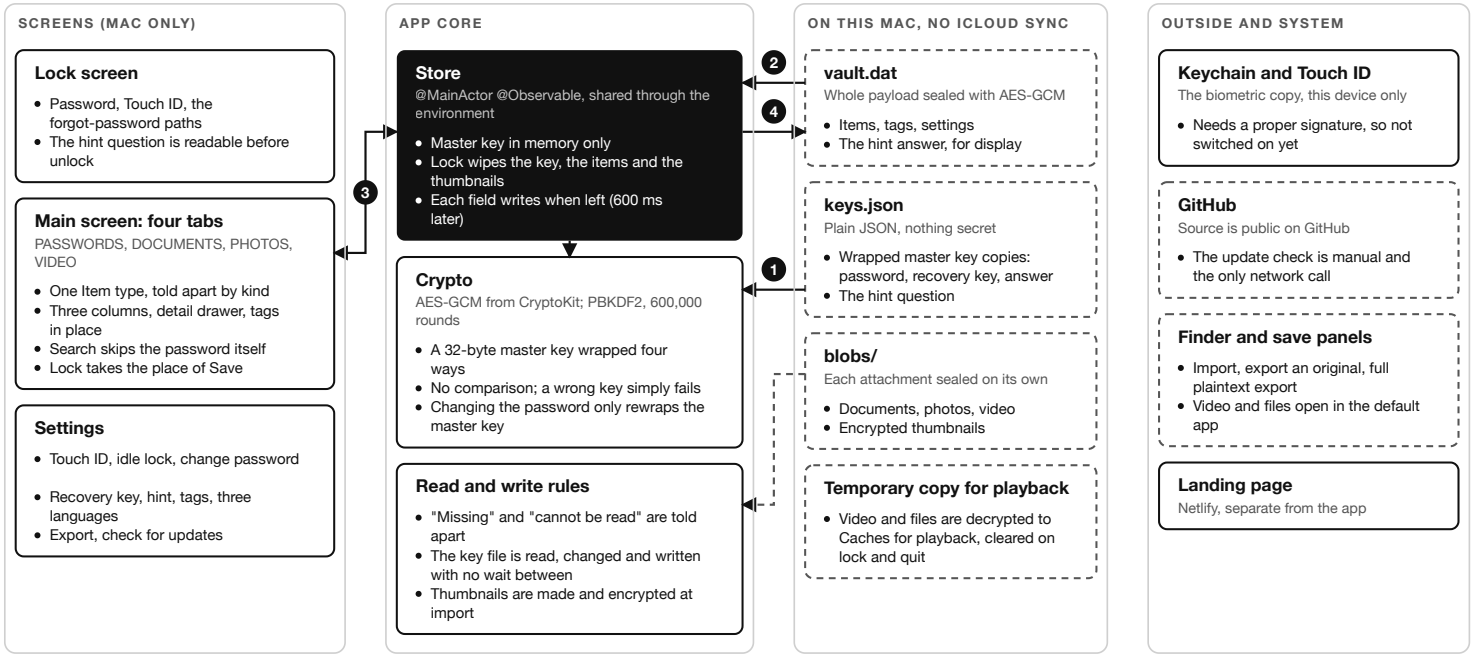


My test "What I want is an offline vault, not just a password keeper." That one sentence decided which features belonged.

04

Architecture

Where the data lives, which layer encrypts, and how the screens get it. Mac only



BUILD AND CHECKS

Mac only, one build
SwiftPM, build.sh, ad-hoc signed, dmg

- macOS 14 and up, v0.3.6, not on the App Store
- Data lives in the Vault folder under Application Support; no UserDefaults

Checks
103 automated checks, real encryption on test data

Interface
Light mode only; ZH, JA, EN

- No iPhone version

1. Any one of the four keys unwraps the master key
2. Decrypt vault.dat; the items go into Store
3. Screens draw from Store; the grid reads only the encrypted thumbnails
4. Edits are written when a field is left, and resealed
5. Lock wipes the key and the contents from memory

Confirmed by me

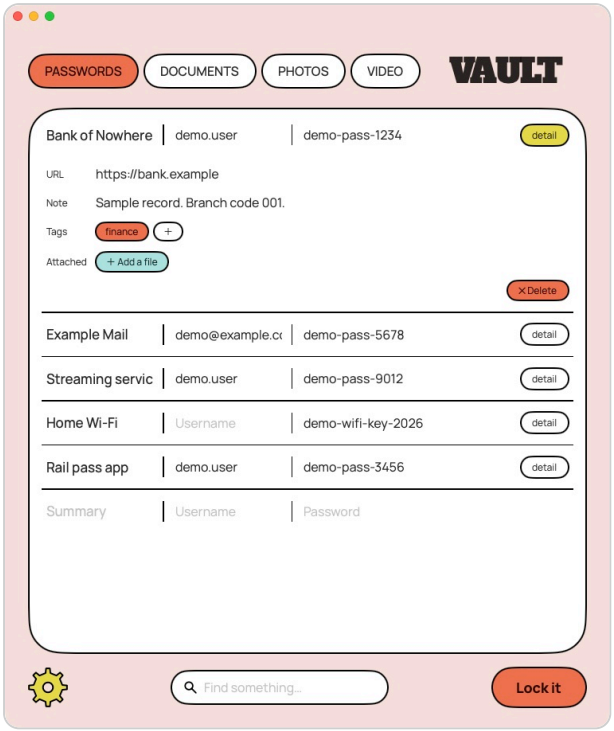
Inferred from the code and my notes, not yet confirmed

The dashed boxes are my inference from the code and my notes, not yet confirmed: what the three files hold (including the hint answer kept for display) [check], the temporary copy for playing video and files [check], the update check being the only network call [check], what goes in and out through Finder and the save panels, including the full plaintext export [check], the build box (not on the App Store, where the data lives) [check]. The key structure itself is in the section The hardest part.

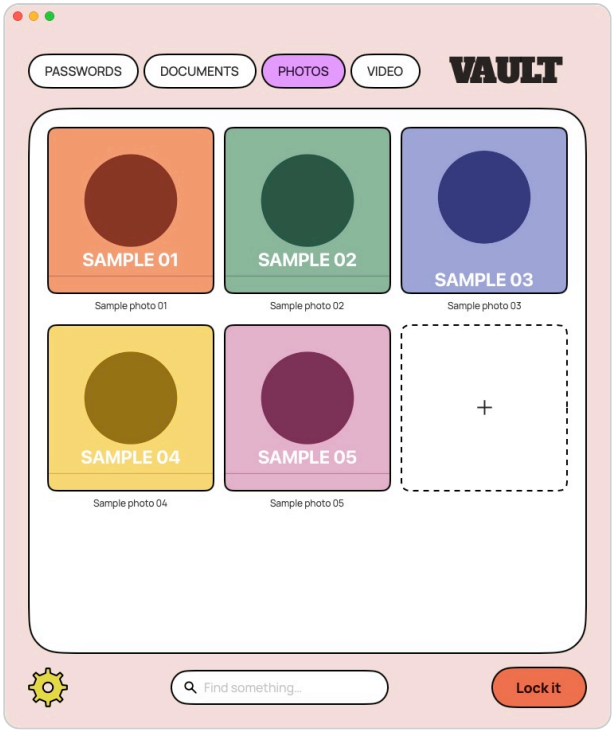
05

Screens

Every screen shows sample data; nothing from my real vault appears.



PASSWORDS, the three columns. Summary, username, password. The open detail shows the URL, a note, tags edited in place and where the delete button sits; all my calls.



PHOTOS, the grid. Thumbnails are stored encrypted and the originals are never decrypted for the grid. The picture-book look was kept even for a security tool.



The lock screen. No password comparison: a wrong password simply does not open the vault. "Forgot your password?" leads to the recovery key and hint answer paths.

06

The hardest part

One master key, four ways to open it



each wraps its own copy of the master key

There is no "compare the password" step. A wrong password simply fails to unwrap the master key. There is no stored password to steal and no yes-or-no check to bypass. Changing the password rewraps the master key; the data itself is not re-encrypted.

Whose call it was This key structure grew out of the conversation with Claude; it is not my design. My calls were on the using side: making the hint answer a real key, where the delete button sits, tags edited in place, the three-column layout.

I did not single out one hardest part of my own; for me the point was the safe itself, one that does not rely on the usual password technology. The key structure above is what Claude picked from the record as the part that took the most technical work.

07

Trade-offs

What was not chosen, and the costs accepted

- ~~A password manager with sync and autofill (did not trust connected services)~~
- ~~A dark, "professional" security look (chose a picture-book style and kept it)~~
- ~~Searching inside passwords (lets someone guess a password a few characters at a time; the exclusion was Claude's suggestion)~~

✓ **Offline, four kinds under one lock, a look of my own**

COSTS (CHOSEN KNOWINGLY)

- No cloud sync; the data lives on one Mac. Settings say so plainly: not yet supported.
- Autofill is weaker than the system's. The feature is simply less capable, and I will not dress that up as a deliberate trade-off.
- If both the password and the recovery key are lost, the data cannot be recovered.
- The hint-answer path is exactly as strong as the answer. I know this and chose it.

08

Problems and fixes

Four found by using it

PROBLEM	WHAT WAS DONE	PRINCIPLE
After regenerating the recovery key, the new key would not open the vault (my report)	A race: read, wait about 135 ms, then write. Another write slipped into the gap. The read now happens after every wait	**A read, a wait, then a write is an open window. Close it**
A file that exists but cannot be read looked like a brand-new vault, and the next save would overwrite the real data with nothing	Separated "does not exist" from "cannot be read"; the second now stops with an error	**Never let "cannot read" become "empty"**
Touch ID could not be enabled on the self-signed build (the system refused to register the key)	Left the code in place, used password unlock, and waited for a proper developer signature. As of Oct 2026, after joining the Apple Developer Program in September, Touch ID has not been switched on yet.	**Some walls are signatures, not code**
A default value sitting in an input field looked like a placeholder, and I typed the real answer into it	Took defaults out of the fields	**Do not put a default where an answer goes**

09

Working split

I set the positioning and how it is used; AI designed the keys and built it

MY ROLE Concept Direction Visual How it is used Release

01

ME
Set the positioning: an offline vault

02

ME
Draw the main screen in Figma, choose the picture-book look

AI
Build the first version

03

AI
Design the key structure and the search rule

ME
Accept; ask for the hint answer to become a real key

04

ME
Use it daily, report what breaks

AI
Find the cause, fix, add checks

05

ME
Correct attribution: say which ideas were Claude's, fix the public page

06

ME
Narration and direction of the film and page

AI
Make the film and the page

10

Results

4	4	103	3	0.3.6
Kinds under one lock passwords, documents, photos, video	Ways to open password, Touch ID, recovery key, hint answer	Automated checks real encryption on test data	Interface languages ZH, JA, EN	In my daily use since Aug 2026 Mac

11

Claude's view

The collaborator's opinion, not written by the author.

Claude's view Collaborator's opinion

WORKED WELL

- Put the positioning in one sentence of his own: an offline vault, not a password keeper. That sentence decided which things belong together.
- Corrected attribution against his own interest: said the search rule was Claude's idea and had the already-public page changed.
- Chose a picture-book look for a security tool and did not drift towards a dark "professional" style.

COULD BE BETTER

- Touch ID did not work on the self-signed build. After joining the developer programme it has still not been switched on, as of Oct 2026.
- The key structure was checked only by us; there has been no third-party cryptography review.
- The only user is the author. There is no evidence yet that others want it for the same reason.
- Cloud sync and autofill fall short of the system's own tools, and the author says so himself.

AUTHOR Names what is not his idea · Judges from the user's seat · Keeps his own visual language

A personal project. No real vault data appears anywhere in this page or in the film. All screenshots use sample data.