The Knowledge Architecture Audit

A self-diagnostic for anyone whose second brain has stopped serving them.

Eight pages. Twenty minutes. No tools required.

*A free resource from **kanon.**, a horus. project.*
kanonmethod.com

Why this exists

You have a system. Notion, Obsidian, Roam, Logseq, a folder of markdown, all of the above. You've spent dozens of hours on it. You're not getting it back.

This audit will tell you, in plain language, why.

It's not a sales pitch. The diagnostic is the whole document. If you finish it and conclude that your setup is fine, that's a real outcome. We'd rather you know than buy something you don't need.

If you finish it and conclude that the architecture is the problem, you'll know what to fix and roughly in what order.

Twenty minutes. Eight short sections. Each one ends with a yes/no. Score yourself at the end.

Page 1

The Monday Morning Test

The single test every personal knowledge system must pass:

> ***Does it help you decide anything on Monday morning?***

Not "does it store useful things." Not "does it look impressive in graph view." Decide. As in: change what you do this week.

If the honest answer is "I'd need to spend an hour reorganizing first," the system is no longer working for you. You are working for it.

- ■ **Yes my system informs decisions in under five minutes most weeks**
- ■ **No I rarely consult it for decisions, or it takes too long when I do**

Page 2

The Search Test

Pick a topic you wrote about a year ago. Something you remember caring about. Search for it now.

What you should see: every relevant note, in some order, with the strongest one near the top.

What you probably see: dozens of fragments, no obvious priority, half of them stale, no idea which is the canonical version.

Why this matters. A system that cannot tell its present self from its past self is not a knowledge system. It is a heap.

- ■ **Yes my search returns prioritized, current results**
- ■ **No my search returns a heap I have to re-read**

Page 3

The Provenance Test

Open your three most important notes from the last six months. For each, answer:

1. Where did this information come from? (Email? Article? Conversation? My own thought?)
2. When was it true? (Last week? Two years ago? Never verified?)
3. Have I summarized it, or kept the original?

If you summarized at capture, the original is gone. Your future self has no way to verify your past self.

If you can't tell what's a quote, what's a paraphrase, and what's your own conclusion, your AI definitely can't either. That's why it makes things up when you ask it.

- ■ **Yes my notes preserve sources, dates, and original wording**
- ■ **No my notes are a blend of paraphrase and assertion with unclear origin**

Page 4

The AI Test

Hand your notes to Claude or GPT. Ask a real question. Something that should require it to connect three or four pieces of your thinking.

Most people, when they do this, get one of:

- A confident, generic answer that doesn't use their notes at all.
- A confused mash-up that mixes what they wrote with what the model already knew.
- The model giving up and asking for clarification.

Why this happens. Your notes are a folder of plain text. The model has no schema, no typed relations, no notion of what's a source versus a conclusion. It does its best. Its best is bad.

This is not the model's fault. It is the architecture's fault. Karpathy's *LLM Wiki* (April 2026) showed that an AI can build and maintain a knowledge base, if the substrate is structured for it. Most personal setups are not.

■ **Yes my AI gives reliable, grounded answers from my notes**

■ **No my AI is unreliable on my own data**

Page 5

The Schema Test

Open three notes about *different but related* things. For example: a note about a customer, a note about a project for that customer, a note about a deliverable in that project.

Question: are the relationships between these notes *typed*?

Specifically:

• Is there a defined difference between "mentions" and "is about"?
• Between "depends on" and "was caused by"?
• Between "contradicts" and "updates"?

If your only link is "this note links to that note," you have a graph in the visual sense, not in the semantic sense. The agent can see edges. It cannot reason on them.

■ **Yes my links carry meaning beyond "these are related"**

■ **No my links are untyped, every connection is the same color**

Page 6

The Portability Test

Imagine Notion shuts down tomorrow. Or Obsidian. Or Logseq. (Any of them could.)

Could you take your knowledge to a new tool without losing what makes it yours?

You probably have an *export*. A folder of markdown, or a JSON dump, or a CSV.

The export is not your data. Your data is the relationships between the pieces, the schemas you developed, the tags you defined, the cross-links. Most of that lives in the app's internal representation. It does not survive export.

A real personal knowledge system is *substrate-first*. The store outlives the tool.

- ■ **Yes I have a canonical store that survives any single tool**
- ■ **No my data and my tool are the same thing**

Page 7

The Compounding Test

Ask: is my system *more* useful today than it was a year ago, or less?

A working knowledge system compounds. More notes means more connections, more reliable retrieval, better surfacing. Like a city that gets better as it grows.

A failing knowledge system inverts. More notes means more noise, harder search, more time grooming, less time using. Like a basement.

If you've been at this for over a year and the curve is still negative, the architecture is the cause. No amount of new tags will fix it.

- ■ **Yes my system gets more useful as I add to it**
- ■ **No my system gets harder to use the more I add**

Score and Read

Count your **No** answers above.

0 to 1 No. Your architecture is sound. Add scale, and it will keep working. You probably don't need our help.

2 to 3 No. You have one or two specific gaps. Most likely: untyped relations and lost provenance. The starter kit fixes both. The book explains why.

4 to 5 No. You are sitting on a system that is actively losing value as it grows. You need to either rebuild on a real substrate, or stop adding to the existing one. Both the starter kit and the done-for-you build address this. The DFY build is faster but more expensive.

6 to 7 No. Your system is not a knowledge system. It is a heap of files you feel guilty about. The fastest path is the done-for-you build, because rebuilding from inside is harder than starting clean.

What to do with this result

Three options, in increasing order of commitment:

1. **Read the book** when it ships. EUR 25 standalone, included in the starter kit.
2. **Get the starter kit** (EUR 79, or EUR 49 for the first 30 founding members). Schema, scripts, Notion template, ten worked examples. Do it yourself, in a weekend.
3. **Have us build it** (EUR 1,500, four slots per month). Two weeks. We migrate, schema, curate the first 200 nodes, set up email ingest. You own the result.

If none of those feel right, you still have a free audit of your own architecture, which is more than most people have. That alone is worth twenty minutes.

Where to find us

kanon. is a horus. project. We are building this in the open.

- Newsletter: weekly notes on building schema-first knowledge systems.
- Founding-member program: thirty seats, then closed.
- Direct contact: hello@kanonmethod.com

structure inside · reach outside.
kanon. · a horus. project · 2026
by Daniel Bürge & Claude