

Good work should teach the next attempt.

You correct an AI assistant. The correction matters. Next time, you should not have to teach it again.

Working with an AI assistant can be strangely repetitive. A task begins with a misunderstanding, a missing distinction, or a plausible answer that turns out not to survive contact with the actual work. You correct it. The task narrows. Evidence replaces guesswork. Eventually something works well enough to accept.

Then the conversation ends. A similar task begins somewhere else, and much of the same ground can disappear with it. The obvious temptation is to call this a memory problem: keep more of the conversation, remember more about the person, preserve more context. FCIS starts from a different question. What, exactly, became useful because the work succeeded?

The answer is usually smaller than the experience that produced it. A reusable procedure can say which source owns the truth, which sequence of actions proved reliable, which evidence closes the task, and where the claim must stop. It does not need the private material, failed guesses, incidental names, or every turn of the conversation that happened to reveal that procedure.

That boundary is important. FCIS is not an argument for turning every successful exchange into permanent memory. Most corrections are local. Some are trivial. Some depend so completely on one situation that extracting them would make them misleading. A lesson is worth keeping only when the work has actually established something non-trivial, reusable, and bounded.

The present site supplied a simple example. While trying to explain FCIS, we kept adding things that were individually defensible: a case simulator, an architecture map, status labels, a podcast surface, separate explanatory routes, more headings. The result became harder to understand than the thing it was explaining. We called that over-edited.

The correction was not another visual style. It was a compositional rule: the page is the frame. Readable type repaginates instead of shrinking. Source identity survives reflow. Listening belongs to the source rather than becoming another application inside the page. Complexity can remain rigorous underneath without becoming the reader's burden.

That correction proved useful beyond this one page, so it moved into the canonical Linotype Skill. The interface you are reading is also using that retained procedure. This is FCIS in its most ordinary form: accepted work changes the maintained way the next piece of work is done.

The same distinction applies to distribution. Fountain Coach keeps the canonical procedure in Git. ChatGPT can receive a derived copy through the GitHub-backed marketplace, but that copy does not become the authority. A source revision, a projected marketplace version, and an installed copy are different states, and FCIS keeps those states separate.

So the claim is modest. Good work can leave behind more than a finished result. When a procedure has really been established, it can be kept as maintained source with its evidence and limits still attached. Here and now, that is what has happened: the Linotype correction lives in the Skill, this page uses it, and the source remains inspectable.

[FCIS Skill source](#)

The interface taught the wrong domain.

The discarded drafts are useful evidence. Each one added a plausible interface that changed what the reader was being asked to understand.

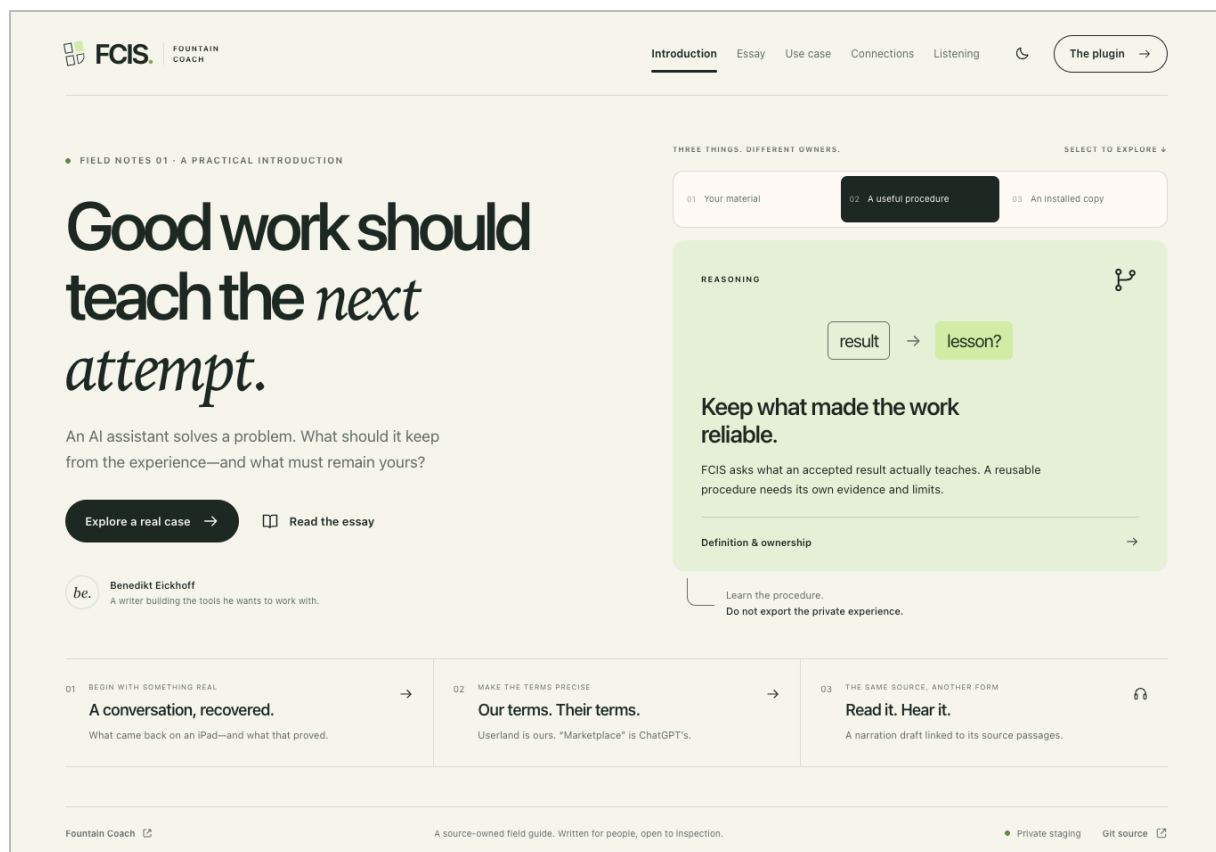
This is an example of domain reasoning. We did not begin with a preferred interface pattern and ask how to fit FCIS into it. We asked what objects actually exist in this domain, who owns them, which relationships have been established, and which representation the reader needs.

The first interface opened with three things and three owners — your material, a useful procedure, an installed copy. Those distinctions were real, but the page made the ownership model into the entrance. It asked the visitor to learn our architecture before the proposition had earned their attention.

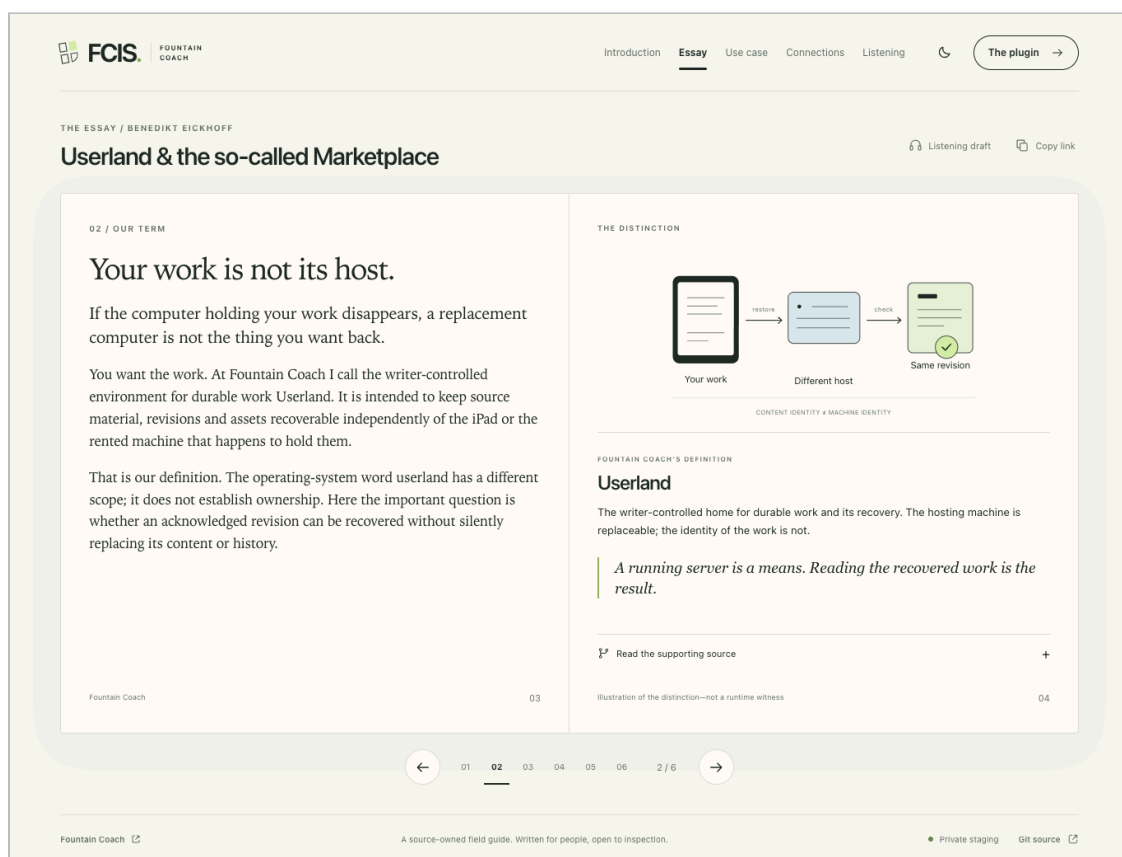
The essay draft then placed a book inside a larger web application. Global navigation, a bordered spread, a second explanatory panel and page controls all competed to explain the same work. “The Page is the Frame” was not a style preference; it removed a representation that the domain did not require.

The listening draft went further. It gave the spoken adaptation its own cover, episode controls, section playback and transcript. That treatment was coherent for a podcast, but wrong for this work. Here there is one essay. A reading is an action on that text, not a second editorial object.

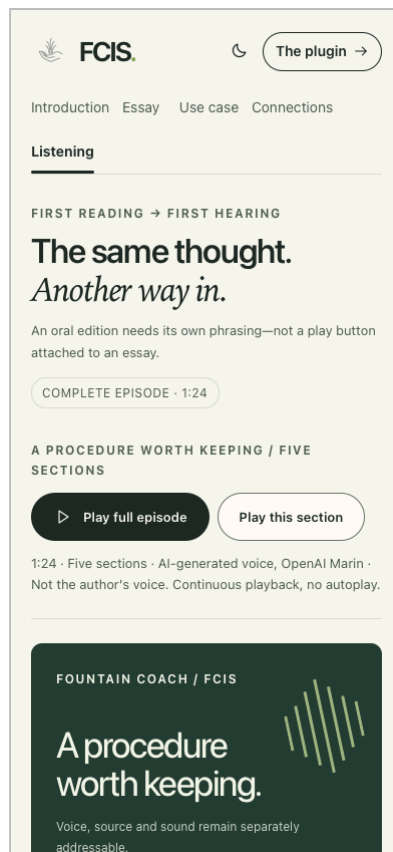
The correction therefore came from the domain, not from taste: one manuscript, one measured page composition, one exact spoken reading, and one PDF projection. The historical interfaces remain because they show the false entities we introduced before the boundary became clear.



Rejected entrance — true distinctions became the product surface. The visitor had to learn “material / procedure / installed copy” before simply reading what FCIS means.



Rejected book-inside-a-site — the essay, the website shell and an explanatory side panel all represented the same argument at once.



Rejected listening product — a coherent podcast interface accidentally promoted the spoken reading into a second editorial object.

FCIS implementation source

Same words, different authorities.

Several terms overlap in ordinary conversation. In this system they name different responsibilities, and confusing them is itself a domain error.

Domain reasoning — Reason from the objects, identities, owners, relationships and evidence that actually exist in the domain. Interface patterns are consequences, not starting assumptions. The rejected FCIS drafts are useful because each introduced a plausible object the domain did not finally need.

FCIS — A Fountain Coach procedure for retaining a non-trivial, reusable and bounded lesson from accepted work. It is not the whole conversation and not a claim that every correction deserves permanent memory.

Skill — OpenAI uses Skill for reusable instructions and supporting resources that help an agent perform a task consistently. In Fountain Coach, the canonical FCIS Skill source is maintained in Git; distributed or installed copies are projections of that source, not its replacement.

Plugin — An OpenAI packaging and integration surface that can include Skills and app capabilities. Plugin is therefore not a synonym for the canonical Skill source.

Tool — A callable capability available to an agent. A tool performs or exposes an operation; a Skill supplies reusable procedure for deciding how to conduct work. The two can cooperate without becoming the same thing.

FountainStore — Fountain Coach's persistence and publication-evidence authority. It stores durable project and publication state under its governed contracts. It is not an OpenAI vector store and not a plugin directory.

FCIS-KIT Instrument Store — Fountain Coach's governed capability plane for admitted instruments. The word "Store" here names a capability registry boundary, distinct from FountainStore persistence.

Vector store / File search — In the OpenAI API, file search retrieves from uploaded files held in vector stores. That is a hosted retrieval mechanism. It does not become Fountain Coach source, publication or capability authority merely because it also

stores information.

MCP — Model Context Protocol is an integration boundary through which tools or external systems can be exposed to a model. Connection does not transfer source ownership or make the connected system authoritative for unrelated state.

Git source / distributed copy / installed copy — Three states, not three competing truths. Git can own the canonical revision; a marketplace or plugin can distribute a derived package; an installed copy can execute locally. FCIS keeps their identities and revisions distinguishable.

Page / audio / PDF — Three projections of one authored work. The Linotype page is the measured reading composition; audio is a timed reading of the authoritative text; PDF is a paginated document projection. None creates a second editorial transcript.

[FCIS Skill — canonical Git source](#)

[FCIS GitHub ↔ ChatGPT distribution record](#)

[Governance — FountainStore](#)

[Governance — FCIS-KIT Instrument Store](#)

[Governance — Skill and maintenance ownership](#)

[OpenAI — Skills in ChatGPT](#)

[OpenAI API — Skills](#)

[OpenAI — Skills in Plugins](#)

[OpenAI API — Tools](#)

[OpenAI API — MCP and connectors](#)

[OpenAI API — File search and vector stores](#)

[OpenAI API — Responses](#)

Source

The procedure, the evidence and the boundaries remain inspectable.

The FCIS Skill is maintained in Fountain-Coach/.github. The Linotype page-frame rule retained from this work lives alongside it as canonical Skill source. The GitHub-backed marketplace carries a derived distribution rather than replacing that ownership.

The Case section uses retained screenshots of earlier FCIS staging interfaces as historical evidence. They show what those drafts actually presented; their captions explain why later domain reasoning removed those representations.

The public FCIS reader is a source-addressed projection of those sources at fcis.fountain.coach. Publication does not turn a distributed or installed copy, an OpenAI vector store, a screenshot, or the public web projection itself into a new source authority.

[FCIS Skill](#)

[Linotype page-frame rule](#)

[GitHub distribution record](#)

[Public source record](#)

[Application repository \(access-controlled\)](#)

[Fountain Coach](#)

[Book](#)

[Fountain Coach Governance](#)

[Instruments](#)

[Status](#)

[Named release manifest](#)

[Publication domain contract](#)

[Accessibility](#)

[Privacy](#)

[Legal notices](#)

[Copyright](#)

[EU compliance](#)