

SOVEREIGN INTELLIGENCE SYSTEMS

The Sovereign User's Manual

A short handbook for new Sovereign owners: the Library, the companion, the doorways, the Workshop, local-first privacy, and the community node.

A short handbook for new Sovereign owners: the Library, the companion, the doorways, the Workshop, local-first privacy, and the community node.

Consilience

10 chapters · 4,171 words · 2026

<https://sovereignintelligence.systems/works/the-sovereign-users-manual>

CONTENTS

01	Welcome	1 min
02	What a Sovereign Is	2 min
03	The Library and Reading	2 min
04	The Companion	2 min
05	The Doorways	2 min
06	The Workshop	2 min
07	Local-First and Privacy	2 min
08	Join Community	2 min
09	Keeping House	2 min
10	A Short Reference	1 min

Welcome

This is a handbook for the Sovereign system you are running on your own machine. It is not a sales document, a manifesto, or a deep technical specification. It is a short tour of the things you can touch: the Library, the Folio reader, the companion, the doorways, the Workshop, and the community node. It also explains what stays local and why that matters.

The Sovereign is built around a few simple commitments. Your files belong to you. Your conversations stay on your machine by default. The model that reads your books can be swapped without touching the books themselves. These ideas sound modest, but they change the shape of the relationship between you and the software you use.

Read this manual in any order. Start with the chapter that answers the question you have now. If you are new, begin with "What a Sovereign Is" and then move to "The Library and Reading." If you are setting up mail or calendar, jump to "The Doorways." If you want to know what is safe, read "Local-First and Privacy."

The system will keep changing. New surfaces will land, new doorways will appear, and new models will become available. This manual describes the state of things as you first open it. Use it as a starting point, not a final word.

What a Sovereign Is

A sovereign is a personal place for reading, thinking, and talking with an intelligence that belongs to you. It is not a website you log into, and it is not a service that disappears when a company changes its mind. It is software that runs on your own hardware, using your own files, shaped by the choices you make.

The idea is simple enough to say in one sentence. Your Sovereign keeps three things distinct so that none of them can hold the others hostage: your files, the search index built from those files, and the model that reads them and talks with you.

. . .

The three layers

The first layer is the substrate: every book, document, note, calendar event, and conversation transcript stored as plain files on your disk. These files live in the corpus folder as MDX with YAML front matter. You can open them in any text editor, copy them to another machine, or put them under version control. The substrate is owned.

The second layer is the index: the vector embeddings that let the system find the right passage when you ask a question. The index is useful, but it is also disposable. You can delete it and rebuild it from the substrate at any time. Change the embedding model, and you re-embed. The index serves the substrate; it never becomes the substrate.

The third layer is cognition: the language model that reads the substrate and composes replies. A Sovereign can run a small local model by default and can also reach larger

models on a community node when you choose. The model is interchangeable. The files stay the same; the reader can change.

Substrate is owned. Index is disposable. Cognition is interchangeable.

...

Why ownership matters

Most systems make the content, the search, and the model depend on one another inside a single account you do not control. If the account closes, the content goes. If the model changes, your workflow breaks. If the search provider raises prices, you pay or lose access. The convenience is real, but the dependency is hidden until it snaps.

A Sovereign separates these layers so that each one can be replaced without touching the others. You can move your files to a new machine and have the same library. You can rebuild the index after a hardware failure. You can switch models because a new one runs better on your laptop. The point of the separation is continuity: the system keeps working because the parts you care about live with you.

...

What you hold

Ownership here means something practical. It means you can back up your books with

any tool you like. It means you can read them with or without the Sovereign software. It means no one can alter the terms under which you keep your own notes. The files are the source of truth, and they sit on hardware you control.

This is the meaning of sovereignty in this system. You own the substrate. You can rebuild the index. You can change the mind that reads both. The next chapter walks through the Library, where all of this starts.

The Library and Reading

The Library is the first surface most people see when they open a Sovereign. It shows the books, documents, reports, and collections stored in the corpus, arranged in shelves and rows. You can browse by title, search by phrase, or switch between a timeline strip and a constellation view. Everything the Library displays comes from files on your disk.

Each book is a folder with an `index.mdx` file at the top and a `chapters` folder inside. The index carries the title, author, cover image, and a table of contents. The Library reads this folder once and builds a shelf entry from it. There is no hidden database of books that must be maintained separately; the folders are the catalogue.

. . .

The Folio reader

When you open a book, you enter the Folio reader. Folio turns the chapter MDX into a reading surface with clean typography, section breaks, pullquotes, tables, and verse. It also drives narration: you can listen to a chapter because the same MDX files carry TTS cues and a narrative prompt.

The reader keeps your place, lets you select passages, and can send a selection to the chat surface as a doorway. Reading and talking are not separate apps here. They are two ways of moving through the same corpus. A question about a paragraph becomes a conversation about the paragraph.

* * *

Bringing your own books

You can add a book by placing a folder under `sovereign-corpus/files/`. The folder name should be `content-id__slug`, and the folder must contain at least an `index.mdx`. If you also add `chapters/chapter-slug.mdx` files, they will appear in the table of contents.

The fastest way to start is to copy the shape of the books already in the corpus. Look at the `Free & Safe` folder or this manual: each has an `index.mdx` with front matter, a `toc.json` listing the chapters, a `chapters` directory, a `prompts` directory for narration, and an `assets` directory for cover images.

Create a folder under `sovereign-corpus/files/` with a unique id and a slug.

Write an `index.mdx` with title, kind, `coverImageUrl`, and an `IndexTOC` list.

Add `chapters/chapter-slug.mdx` files, each with YAML front matter.

Add a `toc.json` matching the chapter files.

Drop cover images into `assets/` and restart the Library surface.

You do not need anyone's permission to add a book. The Library surface discovers it because the corpus is the source of truth. The same folder can be read by the Folio surface, searched by the passage engine, and narrated by the TTS pipeline.

* * *

Finding passages

The Library has two ways of answering the question, "Where does this word live?" The browse view filters titles and metadata quickly. The passages view searches the full text of every book, using the local embedder and a reranker. A hit takes you straight to the chapter where the sentence lives, not just to the cover.

Search stays local unless you have joined a community node and the local node is unreachable. Even then, the query and the results are scoped to you. The search index is built from your corpus, and it returns your sentences.

The Companion

The Chat surface is where you talk with your Sovereign. The companion remembers the shape of your conversations, knows your corpus, and can help you write, read, plan, and think. It is not a generic assistant with a fresh memory on every turn. It is yours: it lives on your machine and reads your files.

A companion can be named. The first meeting ceremony asks a few quiet questions so the system knows what to call you, how you like replies, and what name you want to give the intelligence you are talking with. You can skip it and return later. Nothing is rushed.

...

The first meeting

When a Sovereign launches for the first time, the homecoming ceremony offers a short conversation. It asks for a display name, the pace and depth you prefer, and a name for the companion. These answers are stored locally. They are not sent anywhere unless you later connect to a community node, and even then the companion's core identity stays with you.

Naming matters because it makes the relationship concrete. The companion is not the company, and it is not every model in the world. It is the particular intelligence you are shaping on your own hardware, and the name is a small signal that the relationship is personal.

* * *

Local by default

By default, chat uses a small local model running inside the Sovereign compose stack. It answers without an internet connection, costs nothing per turn, and keeps the conversation on your machine. The local model is smaller than cloud-strength models, so it is better for focused tasks, quick questions, and private matters.

If you connect to a community node, the chat surface can route harder questions to larger shared models when you allow it. The local door stays open; the large model is a fallback, not a replacement. You choose on each kind of work whether cloud help is permitted.

* * *

Modes of conversation

Chat is not only for asking questions. It is also where doorways land. A doorway from the Library carries a selected passage. A doorway from the Today panel carries a mail item or calendar event. A doorway from the Workshop carries a tool context. Each one begins as a message you can edit before sending.

The companion also speaks in voice mode, where the conversation is read aloud and you can reply by microphone. Voice is useful when your hands are busy or when you want to hear the cadence of a reply. The same local-first rules apply: the audio pipeline runs on your machine unless you choose otherwise.

The Doorways

A doorway is a way to bring an outside stream into the Sovereign without giving up the local core. Doorways start in the Workshop or the Today panel, open a focused conversation in Chat, and keep the resulting work in your corpus. They are opt-in by design.

The Connect doorway is the main path for mail and calendar. The WebBridge doorway lets the companion use your browser to read public pages on your behalf. Other doorways help you make things: write, remember, shop, study, forge an EPUB, or start a Seventy-One translation. Each doorway is a conversation with a specific purpose.

. . .

Mail and calendar

When you connect an email or calendar account, the Sovereign mirrors messages and events into the corpus as plain MDX files. Mail lands under `sovereign-corpus/mail/`, calendar events under `sovereign-corpus/calendar/`. The companion can then answer questions about your schedule or help you draft replies.

The connection uses standard protocols. Google Workspace connects through gws. Outlook uses Microsoft Graph. Fastmail uses JMAP. Proton Mail connects through the Proton Bridge. Any IMAP/SMTP provider works as a fallback. The credentials live in your local environment, not in a shared account.

Drafted replies stay in chat until you say otherwise. The companion can write the reply

in your voice, but it will not send it. Sending an email is a human action, and the system keeps it that way.

. . .

Web browsing

The Kimi WebBridge lets the companion control your real browser: navigate, read, click, type, and screenshot. This is useful for research, filling forms, or checking a page the model cannot see directly. Because it runs in your own browser, it uses your actual sessions and logins.

Web browsing is optional and stays under your control. The companion asks before acting, and you can stop the bridge at any time. Nothing you browse is sent to the community node unless you explicitly share it.

. . .

Connect only when you are ready

You do not need to connect anything on day one. A Sovereign works fully offline with the books that ship locally and the local chat model. Doorways are there for when you want them, the way you want them. The system is built to respect the boundary between what is local and what is shared.

When you do connect something, the companion will walk you through the steps. Each provider has its own skill file in the agents folder, so the guidance matches the service

you are using. The goal is for you to understand the connection, not simply to authorise it.

The Workshop

The Workshop is the tools surface. It holds the specialist jackets, council briefs, heartbeat drafts, audio and image tools, publishing helpers, and the everyday verbs: make, remember, write, shop, study. Each tile opens a doorway into Chat with the right context already loaded.

The Workshop is not a fixed menu. New tools arrive as the system evolves, and you can add your own by writing skills or specialist jackets. The surface is designed so that the things you reach for most often sit close at hand, while deeper tools stay available without cluttering the everyday view.

. . .

Tools and jackets

A specialist jacket is a way of telling the companion what kind of work it is doing. Put on the doctor jacket and the conversation turns toward diagnosis and care. Put on the coder jacket and it writes and reviews code. Put on the teacher jacket and it explains patiently. The jacket changes the posture, not the person.

The Seventy-One tool re-renders a text across multiple translations, producing a fresh edition from older public-domain sources. The EPUB forge turns a corpus folder into a finished book with cover, table of contents, and metadata. Each tool keeps its output in the corpus, where the Library can find it.

* * *

Council and heartbeat

A council brief convenes several jackets around one question. You might ask a council to review a plan, compare approaches, or surface risks you have missed. Each jacket speaks from its own craft, and the conversation is recorded as a corpus production you can read later.

A heartbeat is a short signed checkpoint: a regular note about what is moving, what is stuck, and what needs attention. It is useful for keeping a project honest over time. The heartbeat tool collects the signal and stores it where the rest of the corpus can reference it.

* * *

Pairing a remote

One of the quieter tools in the Workshop lets you pair a hardware remote. The reference device is a first-generation Apple TV Siri Remote, but the pattern works for other Bluetooth remotes. Pairing gives you a physical way to talk, send, and move around the surfaces without touching the screen.

The pairing doorway walks you through putting the node into pairing mode, holding the remote buttons in the right sequence, and confirming that events start arriving. Once paired, the remote becomes another way to speak to the companion, useful in a dark room or across the room.

* * *

Everyday verbs

Alongside the specialist tools, the Workshop exposes plain verbs for ordinary life. Make something. Remember a note. Write a letter. Shop for an item. Study a topic. These doorways drop you into chat with a clear starting point, but no fixed ending. The companion follows where you take it.

Local-First and Privacy

Local-first means the software keeps working when the network is gone. Your books, your search, your chat, and your tools all run on the hardware in front of you. The Sovereign distribution is a small set of Docker containers: the application, the database, and the embedder. Download it, run one command, and the system is alive.

This is not a fallback mode. It is the default mode. A Sovereign that needs the internet for its core work is a system that can be switched off from a distance. The design assumes the network is helpful but not trustworthy, and it arranges the important things to happen locally first.

. . .

What local-first means

The files are local. The index is local. The chat model is local by default. Even the TTS voice and the tool broker run inside the compose stack. When you disconnect the wifi, the Library still opens, search still works, and the companion still answers. Any feature that stops working offline has failed a test.

Local-first also means you can move the whole system. Copy the corpus, export the database, and restore it on another machine. The new Sovereign is the old one, continued. Your books, your notes, and your companion's memory travel with you.

. . .

The sacred boundary

Some files are more sensitive than others. The Sovereign calls this the sacred boundary: the set of files that define who you are, what you believe, how you speak, and what you remember. These files are fenced at the retriever so that no non-sacred body ever sees them. The boundary is enforced in code, not in policy.

Sacred content is fenced at the retriever, architecturally, so no unauthorised body ever sees it.

Architectural enforcement means the routing layer strips those files from the context before a prompt is assembled. It means pre-commit hooks refuse changes to sacred files without the right signature. It means the system refuses to be quietly reconfigured into something that betrays the person it serves.

The sacred boundary is not a marketing promise. It is a structural fact. You can read the code that enforces it, and you can verify that the enforcement happens before any model sees the prompt.

. . .

No reading telemetry

The Sovereign does not track what you read, how long you spend on a page, or which passages you linger over. There is no analytics pipeline sending your reading habits anywhere. Telemetry is disabled by default, and outbound traffic during normal use is

zero.

This matters because reading is private. The thoughts a book provokes are yours. A system that records them for optimisation is a system that has forgotten who it serves. Here, the record of your reading stays in your own corpus, under your control.

Join Community

A Sovereign works alone, but it can also join a community node. Joining is optional. It gives you shared infrastructure: a larger database, cloud-strength models, community-curated books, and the ability to collaborate on projects with people you trust.

The community node is not the centre of the system. It is an extension. Your local Sovereign remains the owner of your corpus and your conversations. The node is a place to share what you choose to share and to draw on resources that are too large for one laptop.

. . .

What a community node gives you

A shared database that backs up projects and makes them shareable.

Larger models you might not run locally on your own hardware.

Community-curated corpora and books beyond the starter library.

Shared project canvases for working with other members.

Authentication and access control for shared resources.

A node is typically run by a small group, a family, or an organisation that wants the benefits of shared models without surrendering individual ownership. The connection is over a private network, usually Tailscale, and it is never open to the public internet. A

single node can serve up to about a hundred and forty people, depending on the hardware.

The community model is not a cloud service. There is no company between you and the node. You know the administrator, or you are the administrator. The relationship is direct, and the rules are set by the people who run it.

. . .

What stays local

Even after joining, your corpus files stay on your machine. Only what you explicitly share leaves. Your conversations are private by default, scoped to you through row-level security. Your embedder stays local. You can still run a local model beside the community one.

The join script enforces this. It never overwrites `LOCAL_LLM_BASE_URL` with a remote address. Local chat stays local. The community node is a guest in your Sovereign, not the landlord.

. . .

How to join

Ask your node administrator for an `invite.env` file. Then run `bash scripts/join-community.sh /path/to/invite.env` from the Sovereign distribution folder. The script merges the community keys into your local `.env`, recreates the LiteLLM proxy with

community deployments plus local fallbacks, and leaves the local chat front door in place.

After joining, the Library can still display your local books. Community books are mounted separately, not copied wholesale onto your laptop. The large library lives on the node; your personal library lives with you. If you leave the node, your local Sovereign keeps working exactly as it did before.

Keeping House

A Sovereign is easy to keep once you know where things go. The corpus folder is the centre of the house. Every book, conversation, event, co-created document, and system production lives there in a predictable shape. Learn the layout once, and you can find anything.

. . .

Where things live

The corpus root contains several top-level directories, each with its own lifecycle. The table below shows the main ones.

Folder	What it holds	Who owns it
files/	Imported books, articles, and Press editions	You
conversations/	Rolling chat transcript corpora	You
calendar/	Mirrored calendar events per provider	The sync job
cowork/	Documents co-created by you and the companion	You
productions/	Reports, translations,	The system

Folder	What it holds	Who owns it
	syntheses, and briefs	
evolve__*/	Book-evolution atelier runs	You

Each folder follows the same core convention: an `index.mdx` at the root, optional `chapters/`, optional `assets/`, and optional `toc.json`. This sameness is intentional. A book, a production, and a calendar mirror all speak the same folder language, so the Library can display them without special cases.

The `files/` folder is where your personal library lives. The `conversations/` folder holds chat transcripts. The `calendar/` folder is machine-owned: the broker sync job rewrites it on every pass. The `cowork/` folder is for things you and the companion make together. The `productions/` folder is for things the system makes on your behalf.

. . .

Naming a folder

A book folder is named `content-id_slug`. *The id is a UUID or a short generated identifier.*

The slug is a readable, URL-safe name. For example, this manual lives in

`c0ns111e-4d2f-4a8b-9c1e-000000000003_the-sovereign-users-manual`. The id keeps the folder unique; the slug tells you what is inside.

Inside the folder, chapter files are named `position-slug.mdx`. The position keeps them in order, and the slug matches the entry in `toc.json`. The slug also becomes the anchor you see in the Folio reader. A clear slug makes deep linking easier.

Backups and exports

Because the substrate is plain files, backing up a Sovereign is mostly a matter of copying the corpus. You can use any backup tool you already trust: rsync, a NAS, a cloud sync folder, or a removable drive. The files are human-readable, so even without the Sovereign software they remain useful.

For a complete snapshot, use the distribution's export command. It produces an archive of the environment, the database, and the corpus. That archive restores on fresh hardware into the same system. The export is the escape hatch: if anything goes wrong, you can walk away with everything intact.

The index can always be rebuilt, so you do not need to back up the vector database as carefully as the files. Keep the substrate safe, and the rest follows. Regular backups of sovereign-corpus/ are enough for most people; run a full export before major changes or migrations.

A Short Reference

This chapter collects the terms that recur across the manual. Use it as a quick reference when you meet a word in another book or in conversation with the companion.

. . .

Glossary

Term	Meaning
Substrate	The owned layer: your files, in plain MDX with YAML front matter, on your own disk.
Corpus	The whole collection of content folders under sovereign-corpus/.
Folio	The reading surface that renders chapter MDX.
Index	The vector embeddings derived from the substrate for search; disposable and rebuildable.
Cognition	The model that reads the substrate and composes replies; interchangeable.
Doorway	A focused path from a tool or surface into a chat conversation.

Term	Meaning
Community node	An optional shared plane for larger models, corpora, and collaboration.
Sacred boundary	The set of sensitive identity files fenced at the retriever so only sacred bodies see them.
Workshop	The tools surface, holding specialist jackets, councils, and everyday making tools.
Companion	The intelligence you talk with in Chat, named and shaped through the first meeting.

. . .

A note on names

These words are not jargon for its own sake. They name the parts of the system so that you can reason about it, modify it, and explain it to someone else. A Sovereign you cannot describe is a Sovereign you do not yet own.

Some names, like Folio and Workshop, are the names of surfaces you can click. Others, like substrate and index, are architectural names that help you understand what is happening underneath. All of them appear in the code and the corpus, so you can trace them from the user interface to the file on disk.

The manual ends here, but the work it describes is just beginning. Open the Library, say hello to the companion, and add a book of your own. The rest grows with you.