Simple AI coding

The **Simple Mode** provides a streamlined AI coding workflow inside the normal left pane. It keeps the app structure familiar while reducing clutter.

Quick start: If you have roughly 5–100 pages of text, you can usually **just run everything** and get decent results. Press **One-click coding** and confirm the short set-up modal, then let it run. You can then go back and adjust the coding (edit links, tweak prompts, re-run specific steps) if you want. For longer texts or high-stakes coding, work incrementally: use the **source limit** in Auto-code (1, 5, 20%, 50%, 100%) and the **Links limit** in Recode to process a sample first, check quality, then scale up.
You can activate it with the **"Simple AI coding"** switch just below the Sources bar. When you sign up and choose to have AI options switched on and active, Simple AI is turned on by default. Other users can switch it on later if they wish.

AI usage consumes **credits** (see Responses Panel); credits renew monthly and do not roll over. Costs depend on model and workflow, but very roughly you might autocode around 30 pages for about 1 credit.

Users with dedicated AI plans receive a larger batch of AI credits each month; other users receive 10 free AI credits per month (the free credits do not stack with paid plans).

The Simple Workflow

When Simple Mode is active:

- the **Sources bar** is hidden (to keep focus),
- the right-hand output tabs stay available,
- the Create/Filter sub-tabs are replaced by one combined simple workflow panel.

The workflow is broken down into six straightforward sections:

1. **One-click coding**: Pipeline runner with a **set-up** modal first. Press **One-click coding** to choose **level of effort** (Flash vs Pro for all four model slots), **Code all sources**, **Skip coded**, **Filter on finish**, and (if the project has links) whether to delete **every** link in the project or **only** links on the sources in scope — then confirm **Run**.
2. Pre-steps: clears Filter Links and turns the filter pipeline **on** before the modal.

3. **Scope**: With **Code all sources** off, only the current Sources bar selection counts (empty bar = all sources). The modal explains how many sources Auto-code will run on, including when **Skip coded** removes already-coded sources (and **Run** is disabled if nothing is left).
4. **Single source** in scope: only **Auto-code** runs (Revise codebook and Recode are skipped so labels are not merged across several AI passes). **Summarise** stays in the intended single-source flow; **Filter on finish** defaults off in that path but you can turn it on.
5. **Several sources** in scope: **Auto-code**, then **Revise codebook**, then **Recode**; **Summarise** is turned off for that run from the modal so the pipeline matches multi-source coding.
6. **Recode target suffix**: Choose blank (simpler — synthesised labels go straight into cause/effect) or e.g. _recoded (keeps raw labels, writes synthesised to temp columns so you can compare).
7. Per-step **Run** buttons still work on their own; the modal suppresses the extra confirms for the sequenced run after you confirm **Run**.
8. **Background**: Give the AI project context before coding. A status tick indicates whether enough background text is set.
9. **Auto-code**: This is where the AI reads your documents and extracts causal links.
10. You can choose to process a small sample first (e.g., `1` or `5` sources) to test your prompt, or process `100%` of them.
11. The "Skip coded" switch ensures you don't waste time and money re-processing documents that already have links.
12. Default model in Simple AI is **Qwen Flash**.
13. **Revise codebook**: Once you have some causal links, the AI can review them and suggest a cleaner, more consistent list of factor labels (a "codebook"). The header tick shows whether the Recode codebook area currently contains suggestions.
14. Includes a **Target clusters** slider; see Target clusters.
15. Optional **Use automatic pre-clustering** switch (default OFF).
16. When pre-clustering is OFF, the AI tries to find the clusters directly from the factor list using the standard Revise codebook prompt. This prompt supports macro replacement: use `[number]` (or `[cluster_count]`) and the **effective** target cluster count is injected at run time (same as the slider logic below).
17. When pre-clustering is ON, the app first groups factor labels semantically using embeddings, then sends those clustered groups to the AI with a separate labelling prompt plus a **Representatives per cluster** slider (`8` to `20`, default `8`).
18. Pre-clustering is more systematic than asking the AI to find all clusters "in its head" from a long raw list. It reduces the black-box / WEIRD-data risk a bit, and may make it easier to preserve more unusual or divergent concepts instead of collapsing them into whatever the model finds most typical.
19. Default model in Simple AI is **Gemini 3 Flash Preview**.

20. **Recode**: Apply the AI's suggested, cleaned-up labels back to your existing causal links. Paste the codebook (from Revise codebook or your own), add a recode instruction, and run.
21. The AI returns index mappings (row → codebook item) rather than full label text, reducing tokens and improving reliability.
22. Default instruction: *"For each raw label give me the NUMBER of the best-matching codebook item by meaning. Use 0 when no codebook item fits. Return only codebook label numbers, never words. Never invent labels."*
23. **Skip recoded**: When on, only processes links that have at least one unrecoded label (cause or effect). Use this when recoding again to focus on remaining work.
24. **Links limit** (1, 5, 20%, 50%, 100%): When not 100%, a random sample of links is recoded. Non-sampled links keep their existing recoded values (or stay blank on first run).
25. The header progress bar is segmented: grey = empty recoded fields, orange = recoded equals original cause/effect, green = recoded non-empty and different.
26. Default model in Simple AI is **Qwen Flash**.
27. **Filter links**: The normal Filter Links panel appears as the final section of the same accordion, so filtering is part of one continuous simple flow.
28. When **Filter on finish** is **on** in the One-click set-up, completing the run applies these analysis filters to the pipeline: **Factor Frequency** (top `12`, counted by **citations**) → **Link Frequency** (top `30`, counted by **citations**). The global Label set controls which `cause`/`effect` columns Recode writes to (no separate “recode suffix” in this panel).

One-click coding (Simple AI)

- Sequencer for the Simple AI pipeline, with one **set-up** modal (see step 1 under The Simple Workflow).
- **Run** starts **Auto-code**; with **more than one** source in scope it continues with **Revise codebook** then **Recode**, stopping on the first non-successful stage. With **exactly one** source in scope, it **stops after Auto-code**.
- Optional link deletion is configured **in the modal** (project-wide vs scoped to sources in scope), not only a single “delete everything” confirm.
- **Recode target**: Use the global Label set below the Sources bar. Create a new suffix there first if you want Recode to fill `cause_suffix` / `effect_suffix` instead of only the default columns.

Background (Simple AI)

- Sets shared project context used by AI coding prompts.
- The status tick indicates whether enough background text is present.

Auto-code (Simple AI)

- Runs AI coding across selected/all sources using your prompt and model.
- **Layout (top → bottom): Summarise** (collapsible block, pale gold panel) with its own prompt and controls; then the main **Model**, **Skip coded**, **Add source prompt**, and **source limit** row; then the main **Prompt** textarea; then **Advanced** (main Auto-code chunk size, concurrency, temperature, thinking, etc.).
- Use source limit + skip coded options to test quickly and avoid rework.
- **Add source prompt** (switch): when ON, each source's optional *Source Prompt* (edit above the source text when viewing a source) is prepended to your main Auto-code prompt for that source. Saved per project in the browser. Use when sources need different context; skip when one background prompt in **Background** is enough.
- **Status line** under the settings shows progress, per-chunk detail, and stop — same behaviour as the former "Code with AI" card.
- **Summarise** (subsection inside Auto-code): optional pre-step with its own prompt history and controls (**model** and **chunk size** for the summarise phase, plus concurrency/retries/temperature/thinking). When enabled, each chunk is summarised first, then Auto-code runs on that summary text. Main **Auto-code** chunking lives under **Advanced** below the main prompt.
- **Iterative prompts**: put `====` alone on a line in your prompt to split into separate iterations; later steps see prior user/assistant turns. Only the last iteration's result is written to links; all iterations appear in Responses.
- **Confirm** before a run shows model, chunking, word count, and cost estimate. **Stop** cancels after current chunk tasks finish.
- Timeouts scale by model and iteration count (cap ~540s total). **Concurrency** (1–5) is in Advanced; raise for speed, lower if you see 429/timeouts.
- Tips on using the prompt history (same chrome as other prompt fields).
- Default model is **Qwen Flash**.

Summarise (Auto-code)

- **What it does**: runs two steps per chunk — first a **Summarise** call (default prompt asks for a **mermaid** diagram of the causal network with verbatim quote edges, loaded from shared `prompts.json`), then **Auto-code** using that structured output. You can replace the default with any summarise instructions you like.
- **When to use it**: helpful when source text is long/noisy and you want an intermediate structured pass before coding; for **multi-source One-click coding** the app turns Summarise off automatically so the pipeline stays a standard table pass.
- **Checkbox in Summarise header**:
- **ON** = run Summarise before coding.

- **OFF** = skip Summarise and use standard one-step Auto-code (main prompt falls back to the default Auto-code prompt from `prompts.json`).
- **Summarise prompt**: separate prompt history from the main Auto-code prompt; defaults are maintained in **`webapp/prompts/prompts.json`** (`ai_simple_summarise_prompt` and `ai_coding_prompt`).
- **Model / Chunk size / Concurrency / Retries / Temperature / Thinking**: apply to the **Summarise** step only (separate from **Advanced** chunk size for Auto-code).
- **Chunking**: the summarise phase uses **Summarise → Chunk size**; the coding phase uses **Advanced → Chunk size** under the main prompt. Both can be set independently.

Revise codebook (Simple AI)

- Suggests a cleaner consolidated codebook from existing links.
- Use this after you have enough coded links for a representative sample.
- Header tick indicates whether the Recode codebook area currently has content.
- **Target clusters**: see Target clusters.
- Optional **Use automatic pre-clustering** switch (default OFF).
- With pre-clustering OFF, the AI clusters the factor list directly from the Revise codebook prompt. That prompt supports `[number]` / `[cluster_count]`.
- With pre-clustering ON, embeddings are used first to group labels semantically, then the AI only has to label those grouped clusters. This is a bit more systematic, less dependent on the AI doing all clustering internally as a black box, and may help preserve unusual or divergent concepts.
- Pre-clustering also adds a **Representatives per cluster** slider (8-20, default 8) and uses a separate labelling prompt.
- Default model is **Gemini 3 Flash Preview**.

Target clusters (Revise codebook)

- The **Target clusters** control is a slider with **50 positions**. The **far left** is **Default**; moving right sets an explicit target of **2** through **50** clusters (one step per cluster count).
- **Default** (far left): the app derives a target count K from the number of **unique factor labels** n in the **current filtered pipeline** (same scope as Revise codebook): $K = \min(\lfloor n/3 \rfloor, 25)$ — at most **25**, or roughly **one label in three** as clusters, whichever is smaller.
- **Explicit** positions (not Default): the requested K is the number shown by the slider (**2–50**). If K is **greater than** n, the run uses n instead (you cannot have more clusters than distinct labels); the app may show a short notice when that cap applies.
- Pre-clustering, embedding clustering, and `[number]` / `[cluster_count]` in prompts all use this **effective** K.

Recode (Simple AI)

- Applies your codebook back onto existing links, turning raw factor labels into cleaner synthesised ones.
- **Recoding** (radio buttons): **AI** — the model maps each raw label to a codebook line by index; **Magnetic** — embedding similarity to codebook lines (same magnet machinery as the pipeline's soft-recode path; similarity threshold). Both write hard updates to links; the names refer to the mapping method, not "soft recoding" in the filter sense.
- **Recode target**: the global Label set (`default` = standard `cause`/`effect`; a suffix = read/write `cause_suffix` / `effect_suffix` in `metadata.custom_columns`, with top-level `cause`/`effect` holding the default-set pair).
- Supports sampled recoding and skip-recoded behavior (skip-recoded only applies when using a non-default label set).
- Header bar shows recode coverage mix across all cause/effect recoded fields.
- Default model is **Qwen Flash**.

Filter links (Simple AI)

- This is the same Filter Links workflow, embedded as the final simple-ai accordion section.
- Use it to refine/select links before reviewing outputs on the right.
- When **One-click coding** finishes with **Filter on finish** enabled, the app applies top-12 factor frequency (citations), then top-30 link frequency (citations) (no longer injects the deprecated Temporary Factor Labels filter).

Advanced Settings

Each section header is clickable and opens/collapses its settings panel. Section headers also include contextual **Help** buttons. The advanced sections are inline (not flyouts), and only one section is expanded at a time.

Inside advanced panels you can:

- Edit the exact **Prompt** the AI uses.
- View your prompt history and load previous prompts.
- Change the **AI Model** (e.g., switch to a "Pro" model for complex reasoning, or a "Flash" model for speed).
- Tweak technical settings like chunk size, concurrency, and temperature.