

Data

The editorial ledger

🕒 4 min read · Updated 27 September 2026 · Verified against felff7d on 2026-09-25

What may change in the scripture text and what never does: the decoding rules, the four reviewed editorial rules and their eleven applications, the enforcing gate, and how a concern travels.

<https://docs.gurbanisoul.com/data/editorial-ledger/>

The Gurmukhi in the corpus is verbatim from the source Bir. The pipeline transforms the PDF's *glyph stream* into *logical Unicode* — and that is all it does. Every transform it applies is an entry in one register, `audit/editorial-ledger.jsonl` in `sggs-data`, and a CI gate there fails any scripture change or any new rule that has no entry. This page explains the register; the register itself is the canonical file¹.

Explanation, not scripture. This page describes rules by their codepoint classes and never reproduces a verse. When you need the text, read it verbatim with its Ang from the Ang explorer or the website.

Two kinds of transform, one register

Class	What it is	Entries	Where in code
Decoding (D0–D6)	Undoing what the PDF font did to the byte order: print-job timestamps stripped; a consonant and its halant swapped back into logical order; font glyphs restored to the signs they printed (kanna with bindi, subjoined haha and vava, bindi, hora); ੴ restored from its two-glyph form; a vowel sign re-attached where a stray space split it (the archaic ਊ spellings, 12 cases); the sihari moved after its consonant cluster; stray private-use and Latin characters stripped and the text NFC-normalised	7 rules	<code>fix_text</code> steps 0, 1, 2, 2b, 3, 5
Editorial (R1–R4)	Repairing an <i>impossible</i> Unicode sequence the font emitted — a doubled sihari, a halant before the consonant it belongs to, a detached sihari with a dotted circle, Sahaskriti clusters with a leading sihari — restored to the canonical reading	4 rules, 11 applications	<code>fix_text</code> step 4
Flags (F1–F2)	Things the pipeline <i>notices but does not change</i> : an orphan sihari with no following consonant (11 occurrences), and the rare Sahaskriti ligature glyph decoded to kanna (4, logged for review)	2 flags	logged, never transformed

The eleven applications, one per affected line:

Entry	Rule	Ang	Review
E001 · E002	R1	573 · 586	documented from the start
E003	R3	727	documented from the start
E004	R2	1354	approved 2026-09-24 (scholar review)
E005 · E006	R4	1358 · 1387	approved 2026-09-24 (scholar review)
E007 – E011	R1	1398 · 1402 · 1406 · 1408 · 1409	approved 2026-09-24 (scholar review)

The three earliest applications were itemised when the pipeline was written; the other eight were applied by the same rules and itemised on 2026-09-23, then reviewed and approved by a scholar on 2026-09-24. **If a page or a comment still says “3 logged corrections”, it is out of date:** the register is 4 rules and 11 applications.

The gate

ledger_check.py runs on every pull request in sggs-data, and fails it unless all three hold:

sggs-data/pipeline/ledger_check.py · lines 2–19

```
"""Enforce the editorial ledger: scripture text changes only through a registered,
reviewed entry.

python3 pipeline/ledger_check.py [--base GIT_REF]

Checks (exit 1 on any failure):
1. audit/editorial-ledger.jsonl is well-formed: one header, unique ids, every
   application names a registered editorial rule, and the evidence PDF hash equals
   validation/reconcile-attestation.json.
2. Code == register: the editorial rules in pipeline/sggs_pipeline.py:fix_text
   (step 4: literal replacements + the Sahaskriti regex) are exactly the ledger's
   editorial rules – adding, changing or removing one in code without the ledger
   (or vice versa) fails.
3. With --base: every line whose T0 fields (ang, gurmukhi, text, markers) differ
   between corpus/sggs.jsonl at GIT_REF and now must be a candidate line of an
   application entry added since GIT_REF; lines may never be added or removed.

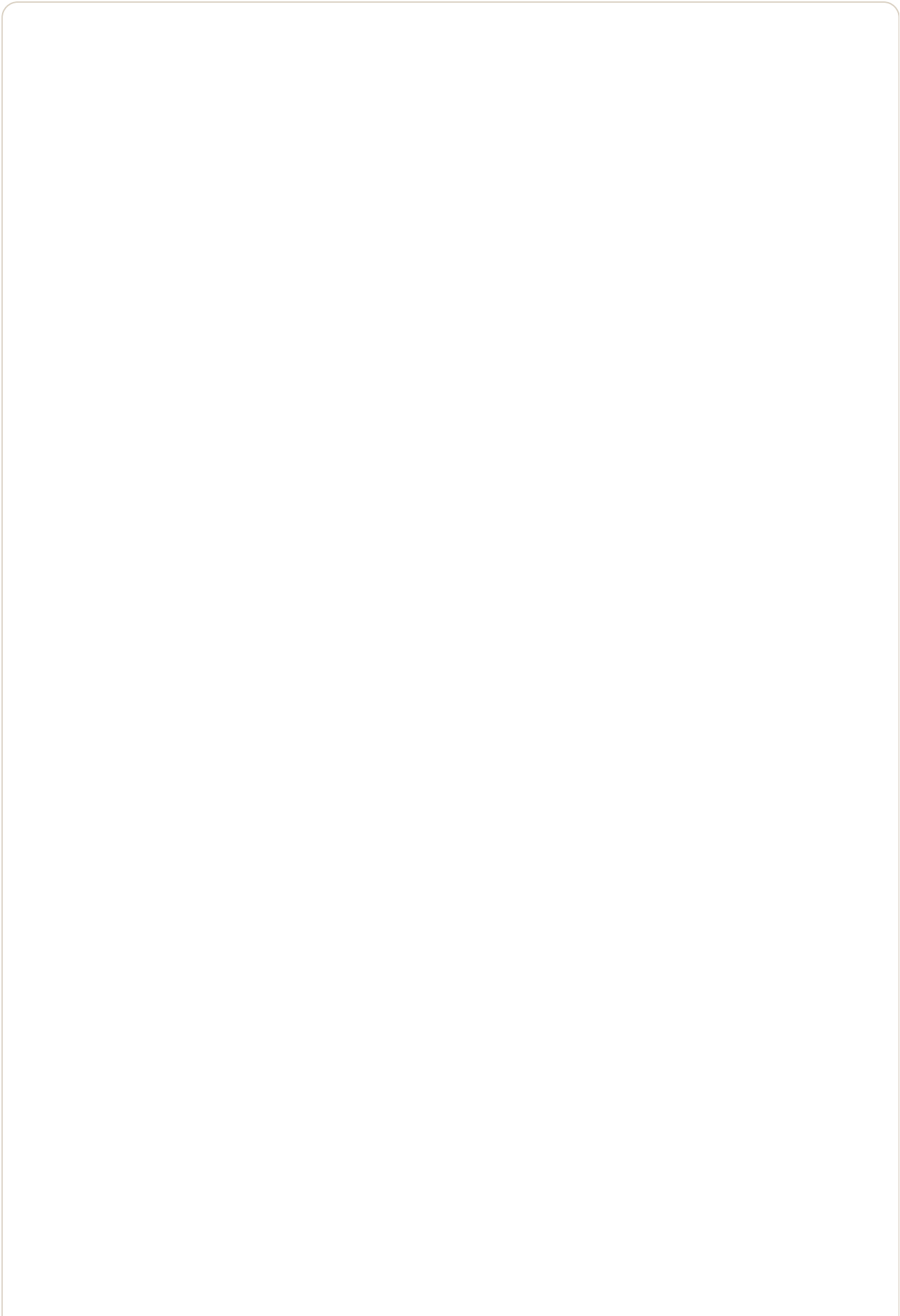
Stdlib only; Python 3.9+. Reads the corpus from git for the base, never writes.
"""
```

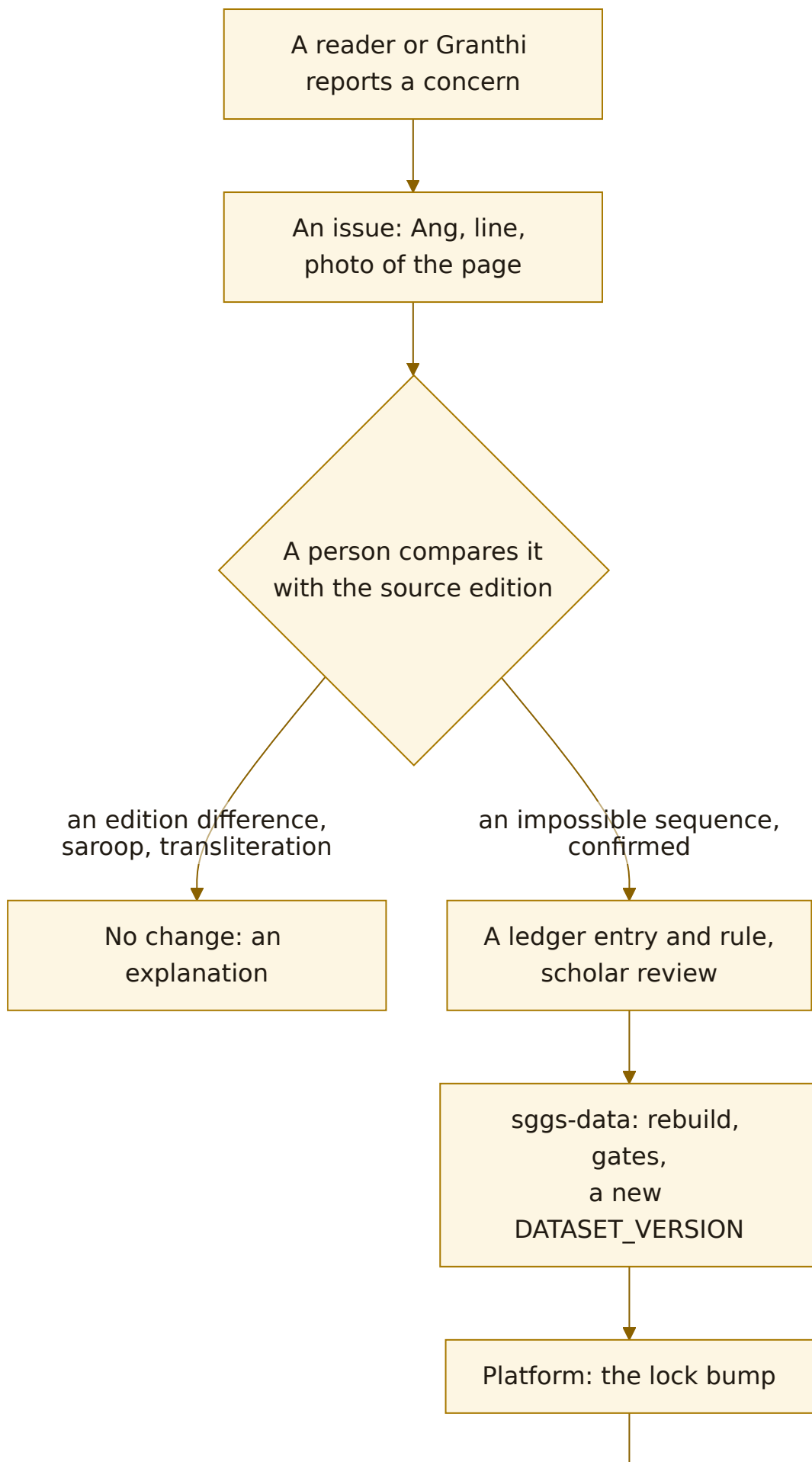
Read from the source at build time — [view these lines on GitHub at c36e317](#) .

Source: [pipeline/ledger_check.py](#)³ in sggs-data.

So a rule cannot be added to the code without the register, a register entry cannot claim a rule the code does not have, and a line's text (ang, gurmukhi, text, markers — the “T0” fields) cannot change between two commits unless a new application entry names that line.

How a fidelity concern travels





App: the same pin, re-archived

A reported text concern goes through review and the ledger before any dataset changes

Most reports are an edition difference, the traditional-saroop rendering (a display choice, the stored text is verbatim) or transliteration (a reading aid, not scripture). Only a confirmed, reviewed repair is ever registered, and it reaches the site and the app as a new dataset — never as an edit here. The support side of this flow is in the **support inbox runbook**; the reverence rules for engineers are in the **reverence checklist**.

Still under review

Known items that are *source-faithful* and deliberately unchanged, tracked row by row so they can never be silently altered (sggs-data's audit/data-quality-baseline.json):

- 21 lines with a vowel sign that has no base consonant (Angs 214, 342, 695, 698, 699, 897, 1203, 1205) — flagged for scholarly review;
- the three closing rubrics that stay one-line compositions (TRAILING_RUBRICS);
- roughly twenty verses the source typography sets like headings;
- one line with an empty translit_norm (id 35328, Ang 829), which a rebuild corrects.

None of these is a text edit; each is a review question, and the answer arrives as a ledger entry or as “leave it as printed”.

Links

1. <https://github.com/Algorythmos-AI/sggs-data/blob/main/audit/editorial-ledger.jsonl>
2. https://github.com/Algorythmos-AI/sggs-data/blob/c36e3172bdf38a61070849bb604b38da89ca5d8b/pipeline/ledger_check.py#L2-L19
3. https://github.com/Algorythmos-AI/sggs-data/blob/main/pipeline/ledger_check.py