

Decisions · Platform

ADR-0006: Banis are a registry of pointers over the verbatim corpus; non-SGGS text is a separate labelled layer

🕒 3 min read · Updated 25 September 2026

Why Nitnem banis are a registry of pointers over the verbatim corpus, with non-SGGS text kept as a separate labelled layer.

<https://docs.gurbanisoul.com/adr/0006-bani-registry-over-verbatim-corpus/>

- Status: accepted (2026-09-18)
- Deciders: project owner
- Related: ADR-0002 (comp_id regroup), pipeline/banis/, NOTICE.md (Nitnem bani layer)

Context

The app needs Nitnem / Gutka reading (Japji Sahib, the morning banis, Rehras Sahib, Kirtan Sohila, Sukhmani Sahib, Asa Di Vaar, Ardaas, ...). The corpus has no notion of a bani: section labels only Angs 1–13 and 1353–1430, and comp_type is unreliable. Several banis are not in Sri Guru Granth Sahib Ji at all (Jaap Sahib, Tav-Prasad Savaiye, Benti Chaupai, Ardaas), and Rehras Sahib differs by tradition. The prime directive is that scripture text is never altered and that translations or other text are never blended into it.

Decision

1. **A bani is an ordered list of pointers** (banis, bani_lines). For every Sri Guru Granth Sahib Ji line the pointer is lines.id; the rendered text is always this project's own reconciled corpus and is cited "Sri Guru Granth Sahib Ji · Ang N". Membership and group boundaries are taken from the ShabadOS open database, resolved to our line ids by text match (exact → skeleton → fuzzy, cursor-aware) with explicit, reasoned overrides for the few line-break differences. Inside a group the reading order is our printed order.
2. **Non-SGGS text lives in extra_lines**, a separate table, converted verbatim (nukta, addak and udaat kept), labelled by source (dasam with its Panna, or ardaas), never indexed by FTS, never given an Ang, never given an English translation, and never mixed into lines. The API and the apps carry the source on every line; Save, Share-as-card, Study Trail and Spotlight are available for SGGS lines only.

3. Editorial choices are explicit seed fields, never inherited silently (e.g. the

ਗੁਰਦੇਵ ਮਾਤਾ

salok around Sukhmani Sahib; Asa Di Vaar in its kirtan interleave; SGPC Rehras as default, Taksal as a variant).

4. **Range-defined variants.** A bani that needs no ShabadOS membership at all may instead be declared by `sggs_line_ranges` — our own line-id ranges, in printed order, one group per range (e.g. Asa Di Vaar *as printed*, Ang 462–475, alongside the kirtan interleave). Because a line id is meaningless on its own, every range carries verbatim **text anchors** (the opening line’s prefix, the closing line’s exact text, the markers on the line before it). `build_banis.py` re-checks them on every build and `guard_banis.py` re-checks them in CI, so a future corpus rebuild that shifts ids fails loudly instead of silently pointing the bani at other verses.
5. **Proof and gates.** Scripture stays byte-identical (baseline guard, reconcile attestation). The registry has its own guard (`guard_banis.py`: canonical ranges, dense sequences, pointer integrity, no English column, no SGGS text inside `extra_lines`), gate tests on a throwaway copy, and a golden suite (`contract/golden_banis.ndjson`) pinning every bani’s line sequence and a SHA-256 of its Gurmukhi so any drift fails CI and the Swift parity tests. Non-SGGS text is not covered by the SGGS reconcile proof: it ships to the App Store only after a scholar review is attested in `ios/Resources/NITNEM-REVIEW.md` (TestFlight builds show it labelled “under review”).

Consequences

- Adding or correcting a bani is a change to `bani_seed.json` / `bani_overrides.json` and a rebuild — never a DB edit. Rebuilds are idempotent.
- The “verbatim, proven character-for-character” claim is scoped to Sri Guru Granth Sahib Ji lines everywhere it is made; the non-SGGS layer is described honestly as sourced and reviewed.
- The registry is additive: older DBs without it degrade to `banis_available: false` and the apps hide the feature (capability bit), no code fork.