Butter Money — Take-home Assignment (1 day)

Project: Multi-Bank **Policy Comparator**

Context. Banks publish MITC (Most Important Terms & Conditions) for home loans. The wording, structure, and formats vary (text/PDF/tables). Your task is to turn these diverse docs into a simple, comparable view that helps a non-technical teammate understand "what's the same, what's different, and what's missing."

Your mission. Build a small app that: 1) **Ingests** 3–4 bank MITC documents (we'll provide a pack that includes text and at least one PDF), 2) **Extracts & normalizes** key facts into a structure you design, and 3) **Presents a side-by-side comparison** with short, human-readable explanations and the ability to drill into the underlying evidence.

You are free to use any frameworks, libraries, and LLMs. Keep the scope pragmatic and shippable in 1–2 days.

User stories (lightweight)

- **Upload**: As an ops analyst, I upload a policy for Bank X and see a preview of the normalized data with any warnings.
- **Compare**: I pick 3+ banks and see a compact table organized by sections you think matter most. Cells show whether items are **Same / Diff / Missing / Suspect**.
- **Trace**: When I click a cell, I see the normalized value, a brief explanation, and a pointer to the **exact evidence** (e.g., file + line/page snippet).

Inputs

- We'll share 3–4 MITC files from different banks (mixed formats, small inconsistencies). Assume INR amounts and percentages appear in common Indian notations.

Output/UI

- A simple, clear UI (single page or a couple of screens) that demonstrates the flows above. Focus on **readability and traceability** over design polish.
- Avoid hard-coding bank names or positions—assume new docs could arrive later.

Deliverables

- **Git repo** with code and a **detailed README** explaining how to run locally (scripts or Docker), configuration, and any prerequisites.
- **Short walkthrough video** (3–5 min) showing upload → comparison → evidence drill-down. A link in the README is fine.
- A small **test suite** covering a few core behaviors you consider critical.
- (Optional) An export of your comparison (CSV/JSON) for our review.

Timebox & expectations

- Aim for ~6-8 hours across 1–2 days. Ship a minimal but cohesive slice end-to-end.
- Keep keys out of the repo and make it easy to plug in through env files.

Hints (not requirements)

- **Evidence matters.** Keep a reference to the source (filename + line range, or PDF page + snippet) **at field level**.
- **Normalize units & language.** Expect ₹/Rs, L/Lac, Cr, %, and synonyms (e.g., *credit score ↔ CIBIL, pre-closure ↔ foreclosure, tenure ↔ term*).
- **Conflicts happen.** Detect overlapping or contradictory bands (e.g., LTV thresholds) and surface them as **Suspect** rather than guessing.
- **Dates.** If a doc mentions an "updated/effective" date, prefer that over "today." Keep time zone consistent.
- **Scope choices.** Choose **5–7 sections** you believe are most useful to a loan ops team (e.g., Fees & Charges, Prepayment/Foreclosure, LTV bands, Eligibility, Tenure, Interest-reset/communication, Documents required, Grievance). Explain your picks in the README.

What we'll look for

- **Product sense:** sensible sections, clear comparison signals, helpful copy.
- **Engineering judgment:** simple, extensible data model; clean boundaries between ingest/extract/normalize/compare.

- **Traceability & correctness:** evidence pointers, reasonable normalization, graceful handling of ambiguity.
- **Pragmatism with LLMs:** prompts (if any), fallback/stub strategy.
- **DX & quality:** README clarity, reproducible run, small but meaningful tests, commit hygiene.

Submission

- Share the repository link and the README. Include a link to your video walkthrough.
- If you documented trade-offs/assumptions (recommended), add a brief **DECISIONS.md**.

You don't need to boil the ocean—prioritize the comparison experience and make thoughtful calls where the documents are ambiguous. We care about how you think, not just what you ship.