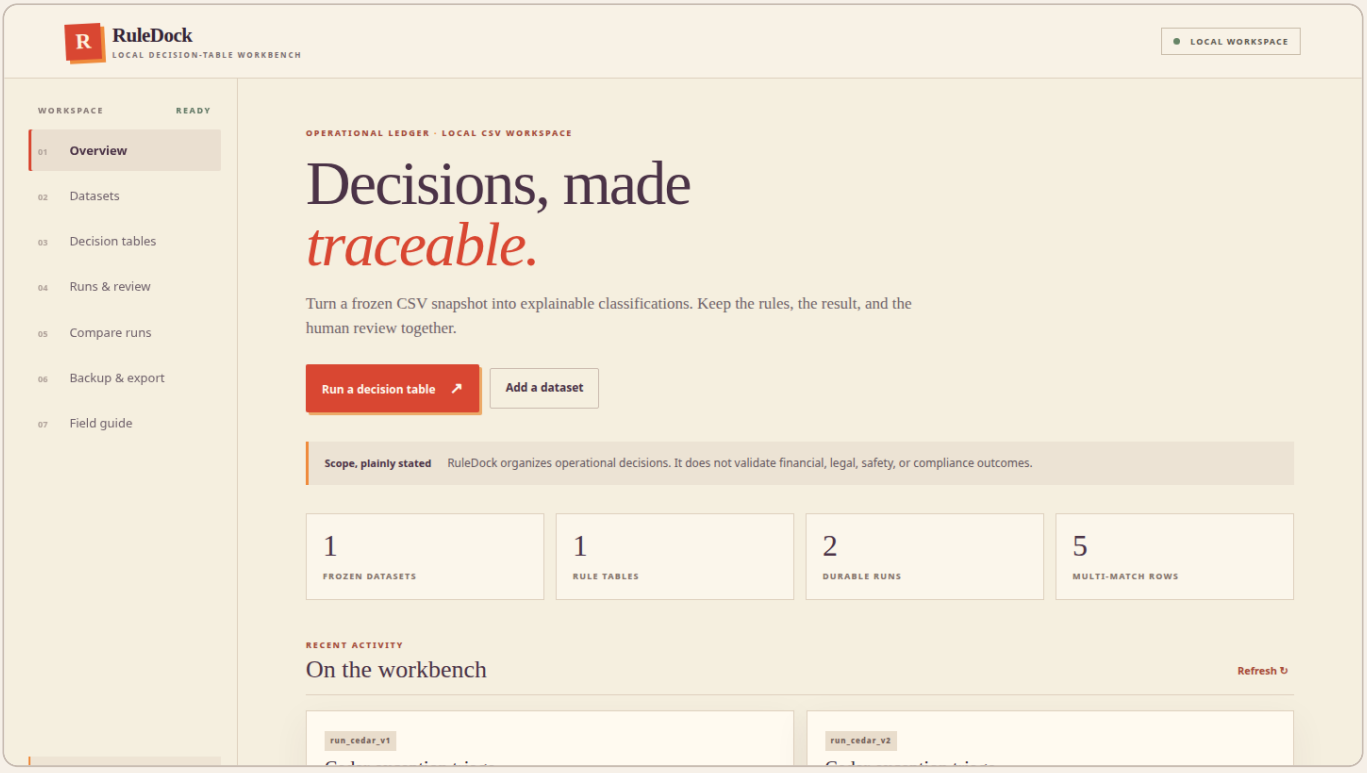
US$89 · ONE-TIME

RuleDock — Local CSV Decision-Table Workbench

Turn a frozen CSV snapshot into explainable classifications. Keep the rules, the result, and human review together on one local computer.

WHAT IT IS

A source-included Python app for one operator to define bounded decision rules, inspect row-level outcomes, review exceptions, and carry results between local workspaces.



Actual app screen captured from the included fictional Cedar Parcel demo. Counts and sample records are illustrative only.

Freeze inputs

Imported CSV becomes an immutable, named dataset snapshot.

Explain outcomes

See matching rules, multiple matches, conflicts, unmatched rows, and numeric diagnostics.

Keep evidence portable

Export a run ZIP with a manifest and portable result files; back up or restore the local workspace.

Product scope: local Python 3.10+ / SQLite application; one operator per workspace. It is not a hosted or team service. RuleDock organizes operational classifications; it does not validate financial, legal, safety, or compliance outcomes. Marketing preview layouts are AI-assisted promotional layouts built around actual fictional app screens; the screenshots in this report are genuine app captures.

From file to reviewed run

A short, explicit workflow—not an automatic decision service.

Start locally

1. Extract the included **RuleDock-v1.0.0** source folder.
2. Install Python 3.10 or newer. No third-party Python package is required; the app uses Python's standard library and SQLite.
3. Open a terminal in the folder and run `python3 app.py`.
4. Open `http://127.0.0.1:8765` in a desktop browser. Stop the local server with `Ctrl+C`.
5. Optional fictional starter workspace: `python3 app.py --demo-data`. The default local data folder is `~/.ruledock`.

Repeatable operator loop

1. Choose comma, tab, or semicolon and import a UTF-8 CSV.
2. Build rules from dataset headers; choose ALL/ANY logic, priority, and first-match or all-match policy.
3. Evaluate, inspect row explanations and diagnostics, then add review statuses and notes.
4. Compare runs only when they share the same immutable dataset; export a run ZIP or workspace backup.



Actual interface capture. Rule authoring is guided through dataset-header, operator, value, ALL/ANY, priority and enabled controls; advanced JSON editing is also present.

Supported input & limits

Runtime: Python 3.10+ and desktop browser; SQLite is included in Python.

Qualification: Linux with Chromium tested. Native Windows and macOS are untested.

Files: UTF-8 CSV, comma/tab/semicolon delimiter; 5 MiB maximum.

Shape: 10,000 data rows; 100 columns; 10,000 characters per cell.

Headers: blank and duplicate headers rejected; rows must match header width.

Rules: up to 200 per table and 50 conditions per rule.

Cedar Parcel: make the rule and the exception visible

A built-in fictional TSV of 8 shipping-exception rows; the demo creates two saved runs to illustrate a rule revision.

Example rules · version 2 · all-match

PRIORITY / LABEL	CONDITIONS
10 · Weather / delay escalation	ANY: `issue eq weather` OR `days_late gte 2`
20 · Heavy parcel review	ALL: `weight_kg gt 10`
30 · Address correction	ALL: `issue eq address`
40 · Priority service follow-up	ALL: `service eq Priority` AND `days_late gte 3`

Text equality is case-sensitive. Contains/starts-with ignore case. Numeric comparisons use exact decimal arithmetic. Rule priority orders evaluation; all-match retains every match.

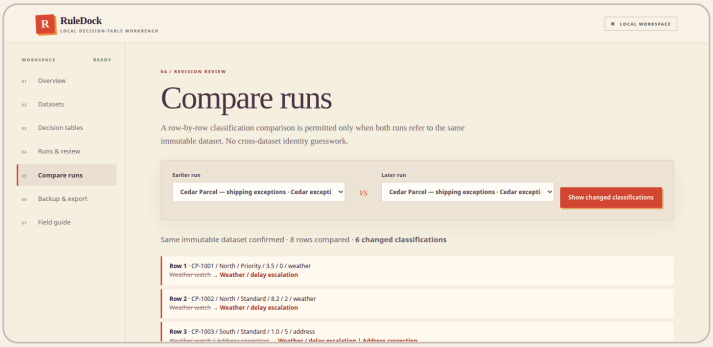
Readable sample rows · run v2

ID	KEY VALUES (SERVICE · KG · LATE · ISSUE)	CLASSIFICATION
CP-1001	Priority · 3.5 · 0 · weather	Weather / delay escalation
CP-1002	Standard · 8.2 · 2 · weather	Weather / delay escalation
CP-1003	Standard · 1.0 · 5 · address	Weather / delay escalation; Address correction
CP-1004	Priority · bad-weight · 4 · weather	Weather / delay escalation; Priority service follow-up
CP-1005	Standard · 12 · 1 · none	Heavy parcel review
CP-1006	Priority · 6.0 · 3 · address	Weather / delay escalation; Address correction; Priority service follow-up
CP-1007	Standard · 2.1 · 0 · weather	Weather / delay escalation
CP-1008	Standard · 5.0 · 1 · none	**Unclassified**

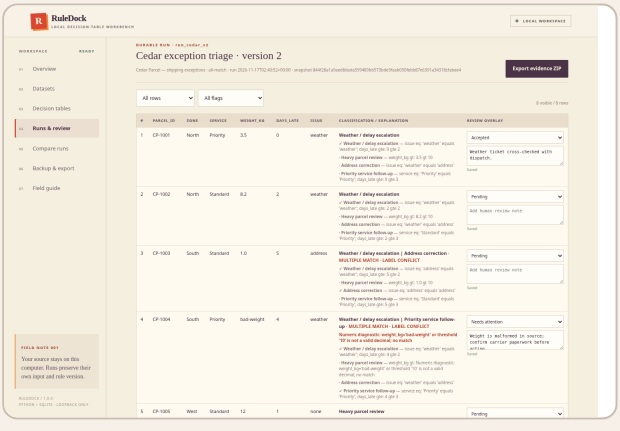
Row 4 trace (CP-1004): the weather rule matches; the weight rule reports `bad-weight` is not a valid decimal and does not match; the priority-service rule also matches. The result is a multi-match, with a numeric diagnostic—not an inferred zero or a silently chosen single label. A human can add a review status and note separately.

What revision comparison means here

The demo's version 1 uses an earlier weather condition and three rules. Version 2 changes its threshold/label and adds the priority-service rule. RuleDock compares row classifications only for runs of the same immutable dataset; this example is illustrative, not customer work.



Actual app view of changed classifications; no inference is made across different datasets.



Actual app capture of the multi-match numeric diagnostic row. Review status/note is an editable overlay, not a rewrite of stored run results.

Know what is—and is not—in the workbench

Included workflow

- ☐ Immutable dataset imports and versioned decision-table rules.
- ☐ ALL/ANY conditions, ordered priority, and first-match/all-match policies. Both inspect enabled-rule overlaps; first-match selects the first label but does not stop evaluation.
- ☐ Row outcomes with match explanations, multi-match/conflict flags, unmatched status and numeric diagnostics.
- ☐ Editable review statuses and notes that remain separate from run outcomes.
- ☐ Same-dataset run comparison of changed classifications.
- ☐ Run ZIP: printable HTML; formula-guarded `source.csv` and `results.csv` (headers and cells); exact original bytes in `source-original.csv.txt` (plain text, not a spreadsheet CSV); dataset and rule snapshots in `results.json`; `rule-snapshot.json`; and a manifest listing filenames and SHA-256 hashes. Also includes local workspace backup and validated restore.
- ☐ Both CSV exports guard formula-triggering values in headers and cells. The separately named original-byte TXT is not a spreadsheet file; review CSV exports before opening them in spreadsheet software.

Not included / not promised

- ☐ No AI, arbitrary scripts, eval, regex, fuzzy matching, or general-purpose data-cleaning suite.
- ☐ No financial, legal, safety, or compliance validation; no decision correctness guarantee.
- ☐ No hosted service, cloud sync, team/LAN access, user accounts, authentication, or API integration.
- ☐ No encryption guarantee or claim that local run evidence/backups are tamper-proof.
- ☐ No specialized date parsing or locale-aware numeric formats; numeric rules require valid finite decimal values.
- ☐ Meaningful changes may be missed; harmless rows may match or fail to match according to the rules supplied. Inspect source data, rules, explanations, and authority before acting.
- ☐ Linux + Chromium tested only. Native Windows/macOS untested; no compatibility promise for them.

Before you start

- ☐ Can you use Python 3.10+ and a desktop browser on the intended Linux computer?
- ☐ Can you keep the CSV, local database and exported backups on a computer you control?
- ☐ Will one designated operator use each local workspace? Do not expose the loopback server to a LAN or public network.
- ☐ Have you chosen a small, testable classification task and reviewed sample cases, conflicts and unmatched rows?

License & purchase

US$89 one-time asking price. Source-included local software, not a subscription. Intended for one purchasing business and one operator per local workspace at a time. Internal business use, source modification, and reasonable internal source/workspace backups are allowed; resale or redistribution of the application/source is not. Generated exports may be shared. Provided as-is; no ongoing support, maintenance, or future-version commitment is promised.

The included buyer **LICENSE.txt** controls. Keep the local computer and backups under your own protection.

Included at purchase: RuleDock v1.0.0 Python source and static interface files. The application uses only Python's standard library and SQLite; no hosted account is part of the product. Setup instructions above are for the included source. This report is a buyer orientation, not a guarantee of fitness for a particular decision.

> **Examples:** Cedar Parcel is fictional. The promotional preview compositions are AI-assisted layouts built around actual fictional app screens; screenshots in this document are captured from the running v1.0.0 app with the built-in fictional demo.