1. What this system is

This is a small, experimental accounting system built around a single ledger file (ledger.json) and a set of small apps:

- Data Entry app – where you enter journals and postings.
- Statistics app – where you see simple summaries of account history.
- Regression / Trend app – where you see basic trends over time.
- Posting Editor (admin) – where you can inspect and manually edit or delete postings.

All apps share the same ledger, so anything you do in Data Entry immediately shows up in the other apps.
Under the hood, the system follows a simple double-entry accounting model (adapted from a relational design) but stored in JSON rather than in a database.

2. The core concepts

Everything lives inside a single JSON structure called AppData. The main pieces are:

- Accounts – named “buckets” for money or value.
- Journals – groups of postings that represent one transaction.
- Postings – individual lines inside a journal that say “this account changed by this amount”.
- Periods – labels for time (e.g. "2025-01" for January 2025).

In the ledger, these appear as:

- ledger.accounts – your chart of accounts.
- ledger.journals – journal headers.
- ledger.postings – all the lines for all journals.
- ledger.assetTypes and ledger.batches – extra structure you can grow into later.

You don’t need to know the JSON layout to use the system, but it helps to know these words because the apps and docs use them.

3. Accounts: your buckets of money

An account is just a named bucket where you track something:

- Assets – what you own

e.g. Cash, Bank, Equipment, Inventory

- Liabilities – what you owe

e.g. Credit Card, Loan

- Equity – owner’s stake

e.g. Owner’s Equity

- Income – money coming in

e.g. Sales, Service Income

- Expenses – money going out to run things

e.g. Rent, Utilities, Software

Where this appears in the UI
In the Data Entry app, you usually work with account codes (short strings like CASH or RENT). These codes correspond to entries in ledger.accounts.

- When you type an account code in the Data Entry form, the system looks up the matching account by code.
- The various reports let you choose an account code to analyze.

You can think of the chart of accounts as “the list of all buckets my money can flow through”.

4. Journals and postings: how transactions are recorded

Whenever something financially meaningful happens (you pay rent, receive income, move money between accounts), you record it as a journal with one or more postings.

4.1 Journals

A journal is one transaction. It usually has:

- a date
- a period (e.g. "2025-01")
- a description (e.g. "Rent for January")
- one or more postings (the lines inside it)

4.2 Postings

A posting is a single line in a journal:

- an account (which bucket changed)
- an amount (how much it changed)
- a period
- optional memo
- links to the journal, asset type, etc.

Depending on your sign convention, a posting might use:

- positive amounts for “increasing this account” and negative for “decreasing it”, or
- positive for some account types and negative for others.

The important thing is to be consistent with whatever convention your Data Entry script uses.

Where this appears in the UI

In the Data Entry app:

- Clicking New Journal creates a new journal header (date, period, description).
- Each time you click Post, the app calls PostEntry twice:
  - once for the first account (e.g. RENT),
  - once for the second account (e.g. CASH),

with opposite signs on the amount.

In the Posting Editor app (admin):

- You see a list of all postings, including:
  - index (internal position),
  - journal id,
  - account code,
  - period,
  - amount.
- Selecting a posting lets you inspect and manually change its period, amount, and memo (or delete it).

5. Double-entry in this system (important intuition)

This system is based on double-entry accounting:
Every journal has postings that add up to zero.
Whatever goes out of one account comes into another.
In practice:

- If you pay 500 rent in cash:
  - Rent Expense goes up by 500.
  - Cash goes down by 500.
- If you receive 1,200 income to your bank:
  - Bank account goes up by 1,200.
  - Income account goes down or up (depending on sign convention) by 1,200, in the opposite direction of the bank posting.

The key invariant is:

- For every journal, the sum of its postings' amounts is zero.
- For the whole ledger, the sum of all posting amounts is also zero (trial balance).

The Data Entry app makes this natural (by posting equal and opposite amounts), but it's good to keep this rule in mind—especially when using the Posting Editor app.

6. Periods: time labels

A period is a text label describing the time slice a posting belongs to, such as:

- "2025-01" – January 2025
- "2025-Q1" – first quarter of 2025
- "2025" – whole year 2025

The system doesn't enforce a particular format, but consistency is key. If you mix "2025-1" and "2025-01", they will be treated as different periods.
Where this appears in the UI

- In Data Entry, there is a Period textbox when you create or post a journal. Whatever you type there becomes the posting's period.
- In the Statistics and Regression apps, you provide a from-period and to-period. The app uses these to select which postings to include in the analysis.

7. A concrete walkthrough: entering rent in the Data Entry app

Let's turn the concepts into actual clicks.
Example: January rent paid in cash
Scenario: you pay 500 rent in January 2025, out of cash.

1. Open the Data Entry app.
2. Create a new journal
   - Click New Journal.
   - Date: type 2025-01-03 (or your actual date).
   - Period: type 2025-01.
   - Description: type Rent for January.
3. Fill in the posting details
   - In Account 1, type the code for your rent expense account (e.g. RENT).
   - In Account 2, type the code for your cash account (e.g. CASH).
   - In the Amount field, type 500.
4. Post the transaction
   - Click Post.

    - The app will:
        - Create (or reuse) a journal id for this transaction.
        - Add one posting for RENT with +500 (expense up).
        - Add one posting for CASH with –500 (cash down).
5. Check what happened
    - The postings list in the Data Entry form (if present) will show two new lines.
    - In the Posting Editor app, you'll see these postings listed with:
        - their journal id,
        - accounts,
        - period 2025-01,
        - amounts 500 and -500.

Because the amounts are equal and opposite, the journal as a whole still sums to zero.

8. Simple reports, statistics and trends

Once you have some postings in the ledger, the other apps let you "look at" the data in different ways.

8.1 Statistics app (five-number summary)

The Account Statistics app:

- Lets you choose:
    - an account code,
    - a period range (from and to).
- For that account and period range, it builds a series of amounts per period and computes:
    - Minimum
    - First quartile (Q1)
    - Median
    - Third quartile (Q3)
    - Maximum

This can answer questions like:

- "What's the typical monthly rent?"
- "How variable is my monthly sales income?"

8.2 Regression / Trend app

The Regression app:

- Takes the same kind of series (period vs amount).
- Fits a simple line (trend) to it: roughly "amount = a + b × time".
- Reports:
    - Slope – is the series trending up or down, and how steeply?
    - Intercept
    - Maybe a measure of fit (depending on your implementation).

This helps with questions like:

- "Is my income generally increasing over time?"
- "Are my monthly expenses creeping up?"

Both apps read from the ledger; they don't modify data.

9. Fixing mistakes: how to correct your data

Mistakes happen. There are two main ways to fix them, depending on how “serious” the data is.

9.1 While you are still entering a transaction
If you just created a journal and notice a mistake immediately:
- Simplest approach:
  - Delete the journal (if you have a UI for that), or
  - Use the Posting Editor to delete the postings you just made.
- Then re-enter the journal correctly.

For small personal use, this is usually fine.

9.2 After the fact: correction journals (safer, more “proper”)
If the data is already part of your history and you care about auditability or strict double-entry:
- Instead of editing or deleting postings, you can:
  1. Post a correction journal that reverses (negates) the wrong postings.
  2. Post another journal with the correct postings.

This keeps a clear trail of what happened: the original mistake, the reversal, and the fix.

9.3 Using the Posting Editor (admin tool)
The Posting Editor app lets you:
- Select a posting from the list.
- View its:
  - index,
  - posting id,
  - journal id,
  - account id/code,
  - period,
  - amount,
  - memo.
- Change its period, amount, or memo, then Save.
- Or Delete the posting entirely.

⚠ Warning: Editing or deleting postings directly can:
- cause a journal to no longer balance to zero,
- break the global trial balance,
- make reports confusing.

Treat the Posting Editor as an admin/maintenance tool, not as the everyday way to “edit transactions”.

10. A minimal workflow for using the system

Putting it all together, a typical small-scale workflow is:
1. Initial setup
   - Decide your period naming convention (e.g. YYYY-MM).
   - Define a handful of accounts (Cash, Bank, Sales, Rent, etc.) in whatever UI you have for editing accounts, or directly in ledger.json if you’re comfortable with that.
2. Day-to-day
   - Use the Data Entry app to:
     - Create journals (date, period, description).
     - Add postings using account codes and amounts.
   - Aim to make each journal’s postings sum to zero.
3. Review

◦ Use the Statistics app to see typical values and spread.
◦ Use the Regression app to see trends over time.
◦ Occasionally use the Posting Editor or a simple “trial balance” script/check to ensure the overall sum of amounts is still zero.

4. Corrections
◦ For small, fresh mistakes, delete and re-enter.
◦ For more serious/historical corrections, consider using correction journals rather than editing old postings.

11. Glossary / quick reference

Account	A named bucket for tracking value, such as CASH, BANK, SALES, or RENT. Stored in ledger.accounts.
Journal	A single transaction, with a date, period, description, and one or more postings. Stored in ledger.journals.
Posting	One line of a journal: an account + amount (+ period, memo, etc.). Stored in ledger.postings.
Period	A text label for time, e.g. "2025-01" for January 2025. Used to group and filter postings.
Ledger file	The ledger.json file on disk. All state from all apps lives here.
Data Entry app	The screen for entering journals and postings.
Statistics app	The screen for computing simple statistical summaries of an account over time.
Regression app	The screen for analyzing trends (simple linear regression) of an account over time.
Posting Editor	An admin tool for inspecting and manually editing or deleting postings. Powerful, but easy to misuse—use with care.

12. What this system is not

This system is designed as a small educational / experimental tool. It is not:
- a full accounting package,
- a tax engine,
- a multi-user, multi-company system,
- or a replacement for professional accounting software.

It is perfect for:
- learning the ideas behind double-entry accounting,
- experimenting with simple ledger structures,
- and prototyping small workflows.

If you later move to a larger system, understanding how journals, postings, accounts, and periods work here will make the “real thing” much easier to grasp.