

I built the *double-entry engine* — not just the screens on top of one.

Axoris is a multi-tenant ERP and point-of-sale platform for retail businesses. One ASP.NET Core 8 codebase, a database per client, 20+ live businesses taking real money daily. I architected it, wrote it, and I run the production server.

290k

HAND-WRITTEN LINES

818

API ENDPOINTS

20+

LIVE CLIENT DATABASES

48

GL TRANSACTION TYPES

1

ENGINEER

01 The problem most retail software gets wrong

A POS that prints a receipt is easy. A POS whose books still tie eighteen months later, across twenty businesses with twenty differently-shaped charts of accounts, is not.

Most retail systems scatter debit/credit code across the features that need it. Sales posts here, returns posts there, a void gets its own path — and the paths drift. By year two the trial balance doesn't tie and nobody can say which feature broke it.

Axoris has **one posting path**. Every business event in the system — a sale, a goods receipt, a commission accrual, a cash-drawer variance, a foreign-exchange difference — funnels through the same method, which reads a declarative rule table. There is no second place where money is written.

02 How the posting engine works

48 transaction types. Each is a rule set, not a block of code.

#	STEP	WHAT HAPPENS
1	Resolve the ledger	The company's operational ledger, carrying both a ledger type and an accounting standard (Local GAAP / IFRS) — so parallel multi-GAAP ledgers are modelled, not bolted on later
2	Fetch the rules	A rule set per transaction type. Each rule is a mapping type, an entry side, an amount resolver and a memo
3	Resolve the account	A three-tier ladder — explicit context override → configured mapping → coarser fallback → Suspense. This is what lets one engine serve twenty tenants whose charts of accounts are seeded to different depths, with no per-tenant code
4	Validate	Effective-date check: posting to an expired account throws, naming the account code and its expiry date
5	Drop empty lines	A pure-products sale emits no services-revenue credit. Opening a register with a zero float posts nothing at all
6	Assert balance	Debits must equal credits within 0.001 <i>before</i> anything persists, or the entry is rejected outright
7	Write the trail	A source row alongside every entry — bidirectional traceability between any GL line and the business document that caused it

Reversal is first class

Rule sets are written as `isReversal ? reversedRules : forwardRules`. Voiding a sale, a receipt, a stock receive, a COGS posting, an inventory adjustment or a rounding entry re-runs the **same rule set, mirrored**. There is no separate void path to drift out of sync with the forward one.

The transaction types that tell you it's real

Every ledger has `Sale` and `Payment`. These are the ones a hobby project never has, and the ones an accountant asks about first:

- **ForexGainLoss · RoundingDifference · CashShortageSurplus** — the entries that absorb reality
- **AccruedExpense · CommissionAccrual / CommissionReversal** — earned-but-unpaid is a liability on the balance sheet, and a returned sale reverses the accrual
- **StockInTransit · StockInTransitWriteOff** — inter-outlet movement that survives a shipment going missing
- **SaleDiscountChannelAdjustment** — discounts are contra-revenue split by channel (manual / coupon / promotional / loyalty), so the business can answer "what did the coupon programme actually cost us?"

03 Proof — one sale, read straight off the running system

13 September 2026. One item rung through the POS on a demo tenant, tendered cash. This is what the ledger did, unedited.

REFERENCE	MEMO	ACCOUNT	DEBIT	CREDIT
SALE-1	Sale receivable	1120 · Accounts Receivable	0.875	—
	Sales revenue – products (gross)	4100 · Sales Revenue – Products	—	0.875
COGS-1	Cost of goods sold	5101 · COGS – Goods Sold	0.750	—
	Inventory reduction	1130 · Inventory	—	0.750
PAY-1-1	Customer receipt	1115 · Cash Register 1	0.875	—
	AR settlement	1120 · Accounts Receivable	—	0.875
Trial balance, period			2.500	2.500

Five things are visible in those six lines, without reading a line of source:

- **Cash sales route through AR and are then settled** — the textbook treatment, not the shortcut of debiting cash straight against revenue. It is why a cash sale and a customer-account sale are the same code path.
- **COGS posts at moving weighted-average cost** — 0.750 against an 0.875 sale, tying exactly to the 0.125 profit the sales-history row reports. Per-invoice margin is a real computation, not a report-time estimate.
- **The products/services revenue split is live** — revenue landed on `4100 Sales Revenue – Products`, and the services credit was correctly suppressed because the basket had no service line.
- **The account-override path is live** — cash landed on `1115 Cash Register 1`, that register's own account, not a generic cash account.
- **Zero-amount rules are dropped** — opening the register with a zero float produced no journal entry at all. Nothing moved, so nothing posted.

BALANCED The trial balance renders its own assertion line. Statements — trial balance, balance sheet, P&L, general ledger, cash flow, aging — all tie, all support comparative periods, and all drill through to the transactions behind any row.

04 The rest of the platform

Nine modules, 95 screens, 16 bounded contexts. The depth marker matters more than the feature list.

MODULE 01 Sales & POS Register sessions with blind cash counts, parked sales, quotations, refunds, split tender, recurring service contracts with an automated billing run. Editing a posted invoice is judged on <i>what the edit does</i> — the incoming document is diffed against the stored one and each change area answers to its own permission.	MODULE 02 Purchasing Purchase order → goods receipt → purchase invoice as three independent documents. A receipt debits inventory and credits Goods Received Not Invoiced; the bill later clears GRNI to AP, so stock and liability are never coupled to each other's timing. True landed cost with the rounding residue parked so the allocation ties to the GL to the fils.
MODULE 03 Inventory Event-sourced. Movement rows are the ledger; quantity-on-hand is a cached projection with an explicit reconcile that recomputes every balance from the movement stream. Moving weighted average, and hitting zero preserves the cost basis rather than wiping it — so a return that empties a bin doesn't destroy it.	MODULE 06 Reporting Statutory statements with comparative periods, plus operational and stock reporting. Excel, branded PDF and print on nearly every report. Report authorisation is declarative and validated at boot — the application refuses to start if any report endpoint is left unguarded.
MODULE 09 Promotions & pricing Coupons and automatic campaigns. Four reward types, 16 condition types with seven operators, priority and stacking rules, three usage-limit dimensions, an approval workflow above a discount threshold, and a full audit log. Discounts post as contra-revenue split by channel.	PLATFORM Offline-first till An Avalonia 11 / .NET 8 Windows client that keeps selling through a total network outage and reconciles when the line returns. Encrypted local store, outbox, journal. Test-first: 15,600 lines of tests against 28,200 lines of code, including a harness that kills a real process mid-workload.

05 The engineering underneath

TENANCY Hybrid, deliberately A database per client, so the blast radius of any tenancy bug is one customer. An Org → Company → Outlet hierarchy inside each, for customers running several companies across many outlets on a shared catalogue with separate books. The trade is that every schema change is a fan-out — so schema rollout is backup-first, with per-tenant apply logs and rollback notes.	SYNC Distributed by necessity A device-sync server on its own authentication scheme. An <i>allowlist</i> of 19 datasets a till may read — anything off it answers 404, not 403, because a 403 confirms the resource exists. Opaque composite cursors issued by the server, never built by the client, because a bare timestamp loses rows sharing a millisecond and a shop PC's clock cannot be trusted. Quarantine over drop; upsert-by-server-id so a lost cursor costs bandwidth once, not a wedged till.
---	---

MONEY

Precision and idempotency

Every monetary value is `decimal(18,3)` — never a float, never a rounded intermediate. Compute at full precision, round once at the edge. Idempotency keys with unique indexes on every retry-prone path. Document numbering is an atomic increment with bounded collision probing, so a checkout can never hang in a retry loop.

ACCESS

Fail-closed by construction

Three composable permission profiles over 63 feature flags, with an ownership map declared in one taxonomy — a flag disabled on one profile can never be revived by another. Anything unclassified resolves to *deny*. Enforced by startup guards, not by discipline: the application will not boot if a report endpoint is missing its permission attribute.

OPERATIONS

I run it, too

I provision and administer the production server myself — Linux, nginx, systemd, MySQL/MariaDB administration, tuning, backup and the tenant migration process. Scripted release automation, six dated migration runbooks with baselines and rollback notes.

PROCESS

Standards that are enforced

Twelve written non-negotiables covering tenancy, layering, decimal precision, transaction boundaries and idempotency — each with its reasoning. Tool-level hooks block destructive operations against a tenant database and refuse to end a session with a lint warning outstanding. Architecture decisions are recorded, including the ones I reversed.

06 The decision I'm proudest of

I was designing offline sign-in for the till: cache credential hashes locally so a cashier can log in with the network down. Clean design, already accepted, written up.

Then I read how the server actually hashes passwords — **one round of HMAC-SHA512, no work factor**. Against a server that is one hardened box, that is a contained problem. Copied onto every shop-floor PC in twenty businesses, it becomes a catastrophic one: an offline attacker with unlimited time against unstretched hashes.

I reversed my own design. The till now escrows a key rather than caching hashes, and fixing the server-side hashing — Argon2id, constant-time comparison, transparent rehash on next login — is the first thing on my list.

I would rather show you a decision I overturned than a feature I shipped. Anyone can build the feature.

07 Stack

LAYER	TECHNOLOGY
Backend	ASP.NET Core 8 · C# with nullable reference types · EF Core 8 · MySQL 8 / MariaDB via Pomelo · FluentValidation · QuestPDF · EPPlus · MailKit · OpenAPI
Frontend	React 18 · MUI · Tailwind · i18next — English and Arabic with full RTL mirroring, not just translated strings · PWA
Desktop	Avalonia 11 on .NET 8 · encrypted local store · xUnit, FluentAssertions, headless UI tests
Security	JWT in an httpOnly cookie · TOTP and email 2FA with separate login / enrolment / step-up challenge purposes · AES-256-GCM secret protection · constant-time comparison on every code path · delete auditing as a persistence interceptor · fail-closed configuration
Operations	Self-provisioned Linux VPS · nginx · systemd · IIS · MySQL/MariaDB administration · GitHub Actions · scripted releases · per-tenant migration runbooks

Honest about what isn't here: no TypeScript, no managed cloud, no containers. I'd rather you hear that from me than find it in week two.

08 What I'm looking for

A team building software that has to get money right — ERP, accounting, POS, retail tech, fintech, billing. I've been the only engineer for eighteen months, which taught me to manufacture the review a team gives you for free: written standards with the reasoning attached, tool-level guardrails, architecture expressed as failing tests. It is not the same as a colleague disagreeing with me, and that is exactly what I want next.

I'm open to relocation and to remote work with a European or Gulf employer, and I can start immediately.

On seeing it

Axoris is a commercial product I co-own with a business partner, so there's no public repository and I don't hand over source. What I do offer is better: **a live guided walkthrough** — I'll ring a sale through the POS on a demo tenant and open the journal entries it posted, on the call, and answer any question about any part of the code while it's on screen.

Moha. Jihat Hasan

Software engineer · ERP, POS and multi-tenant SaaS · 7 years

md.jihat.engr@gmail.com · [linkedin.com/in/jihathasan](https://www.linkedin.com/in/jihathasan) · +880 1614 021833 (WhatsApp)

Every figure on this page was measured or observed on 13 September 2026. Client names and tenant identifiers are withheld.