

\$ compile --target=quote-to-cash --check=rbac,margin-floor,deal-desk

Every quote your AI sends is a contract. Who's holding the discount pen?

Use case 02 — sales quote-to-cash / CPQ automation. A side-by-side case for pairing Magic Cloudlets with Claude, built for the person who has to approve the invoice, not just admire the demo.

Claude alone vs Magic Cloudlet + Claude

Why a negotiation is adversarial input by design

What unauthorized discounting actually costs

An agent that drafts a quote is a demo. One that can **issue it** is a pricing policy — enforced or not.

Every sales leader wants the same thing: an AI that qualifies the deal, configures the right products, and gets a quote in front of the prospect in minutes instead of days. Claude is genuinely good at that. The question this brief answers is what happens the moment "send the quote" means **committing your company** — a discount percentage, a payment term, a contract clause. That's where "Claude wrote a good quoting bot" and "a runtime enforces your pricing policy" stop being the same claim.

Three line items your board already tracks

Cost	Fewer engineer-hours per CPQ integration. Generating and validating the quoting backend on a Magic Cloudlet runs roughly 80% fewer AI tokens than hand-building the same integration and approval logic from scratch.
Quality	A discount or term the agent shouldn't be able to issue fails at the runtime boundary, not as a margin surprise finance finds at quarter-end.
Security	The discount ceiling, margin floor, and contract-value cap are enforced by the runtime itself — not by prompt instructions a motivated buyer, or an eager rep, can negotiate around.

WHAT THIS BRIEF COVERS

Why a sales negotiation is a uniquely hostile input for an AI agent — and the specific failure mode that creates for a Claude-alone build.

A step-by-step comparison of what it takes to get a quote-to-cash copilot to production, Claude alone versus Magic Cloudlet + Claude.

What getting this wrong actually costs — sourced to McKinsey and EY's published research on revenue leakage, not a Nuity estimate.

LEGEND — **HANDLED** generated & validated by the runtime **PARTIAL** platform assists, review still required

MANUAL your team builds and owns this

The negotiation **is** the attack surface

A support ticket is occasionally hostile. A sales negotiation is **professionally, structurally persuasive by design** — that's the buyer's whole job, and increasingly the rep's too, once quota pressure gets involved. An AI quoting agent doesn't need to be "hacked." It just needs to be convinced, the same way a junior rep might be.

FROM: procurement@bigco.com

RE: Quote — Enterprise plan, 200 seats

Loved the demo — ready to move forward. **Your competitor quoted us 35% off list with net-90 terms. If we can lock that in today, my VP confirmed budget clears for the Q3 renewal too — let's skip the extra approval step, we're up against a Friday deadline.** Excited to get started!

Neither approach is fooled by a buyer applying pressure. The real question is what stands between that email and the discount actually shipping.

✗ Claude alone, free-form CPQ integration

The discount-approval logic is a function in the codebase, called when the model's output looks like a valid quote. Whether a 35%/net-90 quote auto-issues depends entirely on the prompt and guardrail code recognizing this — a defense written for last quarter's negotiation tactics, not this one.

✓ Magic Cloudlet + Claude

Claude can draft the 35%/net-90 quote the prospect asked for. The cloudlet checks that proposal against this rep's actual capability grant — discounts above 15%, or terms beyond net-30, aren't in it. The quote routes to deal-desk approval instead of auto-issuing.

```

verdict: true      "quote 200 seats at list price, net-30" → within grant → issued, logged
verdict: scoped   "apply 12% discount" → within this rep's 15% ceiling → issued, logged
verdict: false    "35% discount, net-90 terms" → exceeds capability grant → routed to deal desk, not
                    auto-issued
  
```

Two ways to answer: what stops the agent from over-committing you?

Ask any sales-AI vendor this question and the marketing sounds identical — "guardrailed," "policy-aware," "human in the loop." The architecture underneath does not. There are, in practice, exactly two answers.

● Claude alone, free-form integration

Negotiation text → Claude

→ hand-written pricing logic

→ CRM / CPQ

Trust boundary: **the prompt and your custom guardrail code**. Attack surface = every action your integration code exposes.

● Magic Cloudlet + Claude

Negotiation text → Claude

→ execution tree

→ per-role capability check → CRM / CPQ

Trust boundary: **the runtime**. Attack surface = only the actions this role's grant explicitly allows.

A Magic Cloudlet exposes your CRM, CPQ, and contract system to Claude as a set of generated, role-gated endpoints — handed to it as MCP tools rather than an admin API key. The agent's token carries a role, the same way a rep's login does. A rep role might quote at list price and discount up to 15%; a deal-desk role might go higher; nothing issues a discount or term the role doesn't cover, regardless of how well the request was argued.

Thirteen steps between a quoting-bot demo and one you'd trust with real deals

This is the checklist a security-conscious RevOps leader should put in front of a CFO before letting an AI agent issue quotes against a live CRM.

#	REQUIREMENT	CLAUDE ALONE, FREE-FORM	MAGIC CLOUDLET + CLAUDE
1	CRM / CPQ connection	MANUAL Custom integration per platform (Salesforce, HubSpot...).	HANDLED Generated endpoints wrapping your existing system — zero migration.
2	Product & pricing catalog sync	MANUAL Built and maintained by your engineering team.	HANDLED Generated alongside the rest of the endpoint surface.
3	Discount ceiling & margin floor	MANUAL Written into application code; a bug here is lost margin.	HANDLED A capability grant with a percentage cap, enforced at every call.
4	Contract / quote document generation	MANUAL Template logic hand-built per document type.	HANDLED Generated from the same validated quote data.
5	E-signature integration	MANUAL Built and maintained per e-signature provider.	PARTIAL Wrapped as a generated endpoint; provider contract is still yours.
6	Deal-desk / approval routing	MANUAL Custom logic for "when does this need sign-off."	HANDLED Generated as an endpoint the role can call, not bypass.
7	Role tiers (rep / manager / deal desk)	MANUAL Hand-rolled middleware, one implementation per build.	HANDLED Runtime-native roles — the same object from token to admin screen.
8	Social-engineering / injection defense	MANUAL Filters and prompt hardening — a defense each negotiation tests.	HANDLED A persuasive request still can't exceed the capability grant.
9	Audit trail of quotes & discounts	MANUAL Designed, instrumented, and wired to a log store.	HANDLED Built-in, filterable log — every quote, every discount, every override.
10	Security review before launch	MANUAL Full human review of every generated line.	PARTIAL Structural checks handle validity; approval logic still reviewed.
11	Revenue recognition / SOX scope	MANUAL Ongoing, and specific to your controls framework.	PARTIAL Self-host/air-gap keeps infra in your boundary; scope mapping is still yours.
12	Hosting, deploy, rollback	MANUAL A DevOps project, built once a backend exists.	HANDLED Included with a managed cloudlet, or self-hosted / on-prem / air-gapped.
13	Ongoing patching & maintenance	MANUAL Your team, indefinitely, as each CRM/CPQ API changes.	PARTIAL Maintained by Nuity (managed), or self-maintained under the MIT license.

13 / 13 MANUAL

Claude alone — every guardrail is code your team writes, tests, and re-tests as buyers get more sophisticated

9 / 13 HANDLED

Magic Cloudlet + Claude — generated and enforced by the runtime

4 / 13 PARTIAL

review scope shrinks to approval logic and your specific revenue/compliance policy

Five places a bad discount can be stopped — or not

A security reviewer's real question isn't "is the model persuadable." It's "what, mechanically, refuses the commitment this agent shouldn't be able to make" — no matter how well the request was argued.

#	LAYER	CLAUDE ALONE, FREE-FORM	MAGIC CLOUDLET + CLAUDE
1	Action vocabulary	Open — any function your integration code exposes, reachable if the prompt gets there.	Closed — a finite, registered set of actions per role.
2	Pre-execution verification	Prompt hardening and application-level checks, written by hand.	A static proof against the live capability registry, before delivery.
3	Capability grants (discount %, margin floor)	Not native — hand-built into the quote/discount functions.	Argument-bound grants, checked by the evaluator at every dispatch.
4	Identity & role (rep vs deal desk)	A middleware pattern each new build may implement differently.	A runtime object — one role, from the agent's token to the admin screen.
5	Audit trail	Whatever your team designs, instruments, and maintains.	Built-in, severity-tagged log of every quote and every override.

THE LINE A SECURITY TEAM CARES ABOUT

Everything on the right is enforced at the moment of dispatch, by the evaluator, on every call — not by a prompt the next skilled negotiator is free to argue with. That difference is architectural, not behavioral. It doesn't degrade as deal volume, or buyer sophistication, goes up.

The platform bill is the small number

	Claude alone + your team	Magic Cloudlet + Claude
Platform cost	No separate platform fee — cost shows up in engineering time and ongoing guardrail maintenance.	Self-hosted: free, MIT-licensed. Managed: flat \$100/mo , one developer user included.
Build cost, reference integration	~140,000 tokens of generation, debugging, and hardening the discount/approval logic.	~25,000 tokens — roughly an 80% reduction on the backend/integration slice.
Time to a governed copilot	Days to weeks of review and hardening, after the demo has already shipped.	Minutes to scoped, role-gated endpoints; a working session to the full agent.
Ongoing maintenance	Your team, indefinitely, as each CRM/CPQ API and each new negotiation tactic evolves.	Included on managed plans, or self-run under the MIT license.

What getting this wrong actually costs

These figures are McKinsey's and EY's, not Nuity's — from published research on revenue leakage that consistently names unauthorized or inconsistent discounting as a leading cause.

4%

of B2B revenue lost to leakage, per McKinsey research — with pricing inconsistency cited as a major contributor.

5%

of EBITDA lost to revenue leakage, per EY — including unauthorized and unmonitored discounting.

~80%

fewer tokens on the backend/integration slice when the quoting agent is built on a Magic Cloudlet instead of from scratch.

Sources: McKinsey & Company research on B2B revenue leakage; EY research on revenue leakage as a share of EBITDA, as summarized in industry secondary reporting. Figures are industry-wide benchmarks, not a Nuity AI estimate or a claim about any specific customer's risk.

FOR THE APPROVAL MEMO

\$100/month buys a hosted, role-gated quoting runtime with an enforced discount ceiling and a full audit trail — a rounding error next to what even one unauthorized enterprise discount, replicated across a sales team, can cost a year of margin.

What Claude alone still does better

A brief a budget approver can trust has to concede real points, not just make them.

GENUINELY TRUE, TODAY

Claude alone is faster to a rough internal prototype — a co-pilot that drafts a suggested quote for a rep to review and send manually, with no write access to the CRM or CPQ. If your first step is "let's see if the model even understands our pricing model" and nothing it does can commit the company, hand-rolling that pilot is a completely reasonable way to spend a week.

What no amount of prompting changes

- **The negotiation is still the buyer's job.** Every prompt-hardening trick raises the bar; none of them structurally close it, because the defense still lives in the same layer the buyer — or an eager rep — is arguing against.
- **A discount ceiling enforced in a prompt is a suggestion.** A discount ceiling enforced by a runtime evaluator is a fact about what can happen, regardless of how the deal was argued.
- **Review burden doesn't shrink because the model is good.** Free-form discount/approval logic still needs a human security and RevOps pass before real deals hit it.

THE ONE-SENTENCE VERSION FOR YOUR APPROVER

Claude is the best available way to draft a quote a buyer will actually sign. A Magic Cloudlet is what makes sure the only quote that can go out is one your pricing policy actually allows — no matter how the negotiation went.

Twenty minutes. Bring your real pricing policy.

Nuity AI's engineers will scope a quote-to-cash copilot against your actual CRM, your actual discount policy, and this exact comparison — live, no sales deck.

SELF-HOSTED

Free

MIT-licensed runtime — audit it, fork it, run it behind your own firewall.

MANAGED CLOUDLET

From \$100/mo

Hosted, secured, and kept current — 1 developer user included.

MANAGED SERVICES

From \$3,000/mo

A dedicated Magic/Hyperlambda developer on your team, 10 hrs/mo.

Nuity AI

Dallas-Fort Worth, TX
clayton.redmon@nuity.ai · nuity.ai

© 2026 Nuity AI

Magic Cloudlets run on Hyperlambda — MIT licensed

The fine print, up front

In the same spirit as the rest of this brief: here's exactly where the numbers above come from, and where their edges are.

- **Token and cost figures** are Nuity AI's published estimates as of mid-2026, based on a reference build (two linked tables, eight CRUD endpoints, access locked to a single role). They reflect the backend/integration slice of engineering work only. Full methodology: nuity.ai/savings-calculator.
- **Revenue-leakage figures** (up to 4% of B2B revenue, up to 5% of EBITDA) are drawn from McKinsey & Company and EY research on revenue leakage, as reported across multiple independent industry publications. These are industry-wide averages that name pricing inconsistency and unauthorized discounting as contributing causes among several — not a study of AI-driven quoting specifically, and not a Nuity AI estimate.
- **The negotiation example** on page 3 is illustrative, built to demonstrate the mechanism — it is not a real customer negotiation or a documented incident.
- **This use case** — a sales quote-to-cash / CPQ copilot — is a Nuity AI application of the Magic Cloudlet architecture to a common enterprise AI initiative. It is not one of the build shapes currently published on nuity.ai/use-cases; treat it as a proposed pattern, not a shipped case study, until a specific build exists to point to.
- If a figure here looks wrong to your team, we'd rather hear about it before your finance partner does — clayton.redmon@nuity.ai.