PRODUCT REQUIREMENTS DOCUMENT

CloudTrim

Productized Cloud Cost Audit Platform

One-week, fixed-fee AWS audits for Los Angeles startups. A Terraform-native engine that scans, remediates, and verifies — cutting cloud spend 30% with zero downtime, and showing its math.

> **$8,704 → $5,891** monthly run rate, measured
> **32.32%** verified reduction · re-audited
> **65 / 65** remediation actions API-verified

CloudTrim
Los Angeles, California · v1.0

SEPTEMBER 2026 · CONFIDENTIAL

Table of Contents

1. Executive Summary

CloudTrim is the delivery engine for a productized cloud cost audit practice: fixed-fee, one-week engagements for Los Angeles startups that cut AWS spend by 30% or more with zero customer-facing downtime. The product compresses what takes a senior consultant roughly 35 hours of manual console spelunking into a 40-minute automated pipeline, then converts every finding into a verified, auditable remediation. The consultant sells the outcome; the engine guarantees the rigor.

$8,704 → $5,891	32.32%	65 / 65	1.8 mo
monthly run rate, measured (CUR billing)	verified reduction, re-audited	remediation actions API-verified	client payback on $5,000 fee

The platform's core claim is not a projection — it is a measured, reproducible result. The end-to-end demo in this repository seeds a realistic 18-month-old startup AWS account via Terraform (110 infrastructure resources), ingests its billing export in the AWS Cost and Usage Report format, audits it with a 22-rule catalog backed by live CloudWatch telemetry, remediates Tier 0/1 findings through real AWS API calls, and re-audits the result: monthly spend falls from $8,703.59 to $5,890.53 — a 32.32% verified reduction worth $33,757 per year — with all 65 remediation actions individually verified by API read-back and a billing-vs-inventory reconciliation variance of just -0.2%.

This document specifies the product behind that result: the problem and market context, the functional requirements and rule catalog, the zero-downtime remediation methodology that makes the guarantee safe, the system architecture (which runs unchanged against the moto AWS API emulator for demos and real AWS for engagements), the business model and unit economics of the fixed-fee tiers, and the go-to-market plan for the LA startup ecosystem. It is written for the founding consultant-operator, early engineering hires, and the first lighthouse clients who will read it as the contract for what this product does and does not do.

2. Problem and Market Context

Cloud waste at startups is chronic, structural, and almost never deliberate. Feature velocity outruns cost hygiene: staging environments multiply, sandbox experiments are never torn down, snapshots accumulate for years, and log groups grow without retention because nobody was ever assigned to care. Industry surveys (Flexera State of the Cloud 2024) consistently report that enterprises believe roughly 27-32% of cloud spend is wasted, and startups are worse — they have no FinOps staff, no tagging discipline, and no dashboards anyone reads. The spend is invisible until the runway conversation, which is exactly the wrong time to discover it.

The ideal customer profile is concrete: seed-through-Series-B startups in the Los Angeles ecosystem (Santa Monica, Playa Vista, Culver City, DTLA) running $5,000-$50,000 per month on AWS, with 5-100 engineers. At the midpoint of that range, a 30% reduction returns $16,000-$18,000 per month — more than

the entire audit fee, every month, indefinitely. The buyer is usually the CTO or VP of Engineering who suspects the waste but cannot spare a sprint to hunt it; the champion is often the finance lead watching the AWS invoice climb quarter over quarter. Almost none of these companies can justify a FinOps hire, and the big consultancies will not take a $5k engagement.

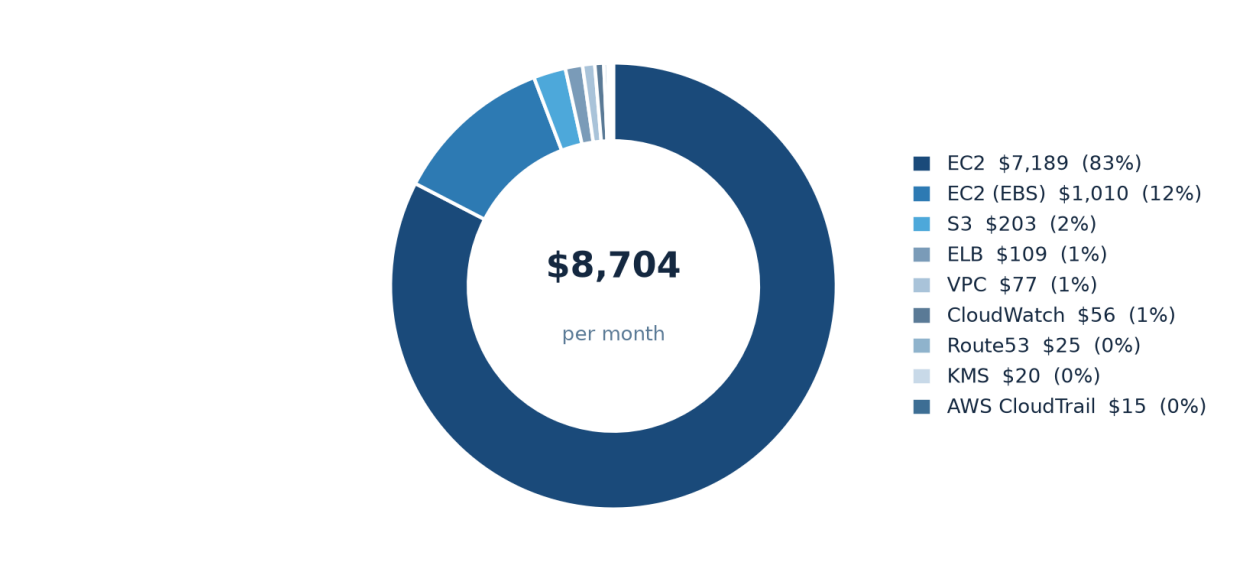

Figure 1 — Audited account spend by service, monthly run rate from the CUR billing export (measured, demo account).

The existing alternatives all fail this customer. Native AWS tooling (Cost Explorer, Compute Optimizer) surfaces hints but requires the exact expertise the client lacks and produces no remediation. FinOps SaaS platforms price for enterprises and demand ongoing operational commitment. Freelance consultants deliver variable quality with no verification. The gap in the market is a productized audit: fixed fee, fixed timeline, defined deliverable, and a quantified, verified outcome. That is the slot CloudTrim occupies.

3. Solution Overview and Positioning

CloudTrim is a consultant-operated audit engine with a four-stage pipeline: Connect, Audit, Remediate, Verify. Connect provisions access (read-only IAM in live engagements; a Terraform-seeded emulator account in demos) and ingests billing data in the AWS Cost and Usage Report format. Audit runs a read-only inventory sweep with boto3 — paginated describe calls across EC2, EBS, ELBv2, S3, CloudWatch Logs, and VPC — cross-referenced against CloudWatch telemetry and CUR billing lines, then evaluates a 22-rule catalog that emits evidence-backed findings: every finding cites the API observation, the metric, and the billing line that justify it, plus the savings math and the remediation action.

Remediate is where productization earns its fee. Findings are classified into four risk tiers; Tier 0 (pure waste) and Tier 1 (live, in-place AWS operations) are executed during the engagement week through the same APIs a human would call, with each action verified by reading state back before it is marked applied. Tier 2/3 changes — rightsizing, NAT consolidation, Graviton migrations, Savings Plans commitments — are never auto-applied; they ship as a costed roadmap with generated Terraform modules the client's team

reviews and applies through their own CI. Verify closes the loop: billing is recomputed from the post-remediation state and a full re-audit measures the realized reduction. The client sees the before, the after, and the audit trail connecting them.

Three design commitments differentiate the product. First, billing is the source of truth: every audit cross-checks the CUR run-rate against a bottom-up cost model built from live inventory and list pricing, and reports the variance — catching CUR lag and pricing drift instead of silently trusting either source. Second, remediation is Terraform-native: the engine generates reviewable modules with import blocks, so clients keep version-controlled infrastructure changes rather than console archaeology. Third, the guarantee is narrow and honest: zero-downtime applies to Tier 0/1 actions only, every claim is verified by API read-back, and anything an environment cannot verify is reported as pending rather than assumed.

> *"We cut your AWS bill 30% in a week, with zero downtime, and we show our math."*

4. Functional Requirements

Requirements are organized MoSCoW-style. Each functional requirement carries an acceptance criterion expressed in terms of the shipped demo or live engagement behavior, because a requirement that cannot be observed in the working system is a wish, not a requirement. The rules catalog itself — the product's core IP — is specified rule-by-rule in Appendix A.

Must have (v1 shipped)

ID	Requirement	Acceptance criterion
FR-01	Endpoint-agnostic AWS client layer (aws.py): every boto3 client routes through one seam controlled by AWS_ENDPOINT_URL; identical code path for emulator and live AWS.	Demo and live mode differ only by environment variable; no code branches on mode.
FR-02	Read-only inventory scanner with pagination across EC2, EBS, ELBv2, S3, CloudWatch (metrics + logs), VPC, and STS identity.	A 110-resource seeded account is fully inventoried in one pass; scanner performs zero mutating calls.
FR-03	CUR 2.0 parser: resource-daily aggregation, per-resource and per-usage-type rollups, billed-days tracking.	Parsed summary reproduces the generator's monthly run-rate to the cent; billed-days matches spec.
FR-04	22-rule audit catalog with evidence, severity, risk tier, savings math, and remediation action per finding.	Seeded demo yields 99 findings across 10 firing rules; each finding lists at least two evidence strings.
FR-05	Reconciliation check: bottom-up inventory cost model vs CUR run-rate, variance reported.	Demo variance within +/-0.5% (measured -0.2% pre, -0.3% post remediation).
FR-06	Tier 0/1 remediation executor with per-action API read-back verification and full audit log (before/after state, API calls, timestamps).	Demo run: 65/65 actions verified, 0 pending; log persisted in SQLite.
FR-07	Terraform remediation generator emitting modules with 1.5+ import blocks (gp3 conversion, S3 lifecycle) plus provider wiring.	Generated module passes terraform init/plan; import blocks adopt live resources (verified: 24 imported).
FR-08	Client-facing PDF audit report with executive stats, findings table, verified remediation log, roadmap, and methodology.	10-page report generated from the audit store; headline numbers match engine state.
FR-09	Dashboard: live summary, findings browser with tier filters and evidence expansion, remediation log, before/after by service, methodology view.	Dashboard renders the live audit state over the engine API; actions trigger pipeline jobs.
FR-10	Portable deployment: docker-compose with pinned moto (AWS API emulator), engine, and web images; Codespace devcontainer bootstrap.	docker compose up + make demo reproduces the full pipeline on a clean Docker host.

Should have

ID	Requirement	Acceptance criterion
FR-11	Savings Plans and Graviton roadmap analysis with modeled (conservative) rates.	Roadmap findings fire on steady-state workloads with modeled payback.
FR-12	Snapshot-before-delete guard policy for unattached volumes >= 100 GB.	Guarded deletions create safety snapshots; grace-tagged stopped instances skip termination.
FR-13	Live-only rule coverage: RDS idle/gp3, ElastiCache, Lambda reserved concurrency, classic ELB.	Rules evaluate quietly where APIs are absent (emulator) and fire on live accounts.

Will not have (v1)

ID	Decision	Rationale
FR-14	Multi-cloud (GCP/Azure) in v1 — deferred; AWS covers the LA ICP concentration.	Out of scope for v1; revisit after 20 engagements.
FR-15	Self-serve SaaS onboarding — the consultant operates the engine; a self-serve tier is a v2 decision.	Productization stays consultant-operated in v1.

5. Zero-Downtime Remediation Methodology

The zero-downtime guarantee is a risk classification discipline, not a slogan. Every finding is assigned to one of four tiers before anything moves. Tier 0 actions remove resources that provably serve no traffic — an unattached volume, a snapshot no AMI references, an unassociated Elastic IP, a load balancer whose target groups have zero targets, uncommitted multipart upload parts. Tier 1 actions are in-place AWS operations that the platform itself performs live: gp2 to gp3 volume conversion via ModifyVolume, S3 lifecycle configuration, CloudWatch log retention, and stopping instances whose CPU has averaged below 5% for 14 days of telemetry. Neither tier touches anything a customer request can reach.

Tier	Class	Examples	Downtime	Execution
T0	Pure waste removal	Unattached volumes, stale snapshots, orphaned EIPs, idle ALB, MPU leaks	None — nothing serves traffic	Engine, during audit week
T1	Live, non-disruptive	gp2→gp3 conversion, S3 lifecycle, log retention, idle-instance stop	None — in-place AWS operations	Engine, during audit week
T2	Scheduled maintenance	Rightsizing, volume shrinks, NAT consolidation	Minutes, client window	Client via generated Terraform
T3	Planned initiative	Graviton, Spot migration, Savings Plans	Zero (rolling) or n/a	Client engineering roadmap

Table 2 — Risk-tier classification governing the zero-downtime guarantee.

Execution is verification-first. The executor captures before-state, performs the action through the real AWS API, then reads the state back and only marks the finding applied if the observed end-state matches the intent — a stopped instance must actually be stopped, a converted volume must actually report gp3, a released address must actually be gone. Volumes of 100 GB or larger get a safety snapshot before deletion, and instances stopped by the engine are tagged with a 30-day grace timestamp so long-cleanup logic never terminates something the client may still want. Anything that cannot be verified is reported as pending, never as done. In the measured demo run all 65 actions verified on the first pass; the one prior run that hit an

emulator consistency delay stayed honestly pending until the retry logic landed.

Tier 2 and Tier 3 changes are where the bigger money lives — rightsizing under-utilized instances, consolidating redundant NAT gateways, Graviton rebuilds, spot capacity, Savings Plans coverage — and they are deliberately not automatic. Each ships as a costed roadmap item with effort and risk labels, and the infrastructure changes arrive as generated Terraform modules using import blocks, so the client's team reviews the diff and applies it through their own CI with rollback they already trust. The audit report explicitly models roadmap tiers independently and states the combined bound, because stacked discounts on the same baseline are not additive and the product does not pretend otherwise.

6. System Architecture

The system is three containers and one seam. moto (pinned at 5.2.3, server mode) provides the AWS-compatible API endpoint for the demo account — full community coverage of every service the engagement touches, including ELBv2 and S3 lifecycle configuration, which community-edition LocalStack gates behind its Pro tier (LocalStack Pro is a drop-in alternative). The engine container carries the Python package plus the Terraform binary: the FastAPI service, the CLI, the scanner, rule engine, CUR pipeline, remediation executor, report generator, and an SQLite audit store that records every scan, finding, and verified action. The web container serves the Next.js dashboard, which reads the engine API and triggers pipeline jobs. Docker Compose wires the three together with healthchecks and bind-mounted output directories so report PDFs and generated Terraform modules land on the host filesystem.

Figure 2 — Runtime topology and engagement pipeline; one seam (aws.py) separates demo mode from live AWS.

The architectural invariant that makes the demo real is that there is exactly one place the environment is known: the client factory in aws.py. With AWS_ENDPOINT_URL set, every boto3 client points at the emulator; unset, the identical code talks to real AWS through the standard credential chain, and the

Terraform modules carry the same variable for provider endpoints. Nothing else in the engine — not the scanner, not the rules, not the remediator, not the report — branches on demo versus live. The emulator fidelity gap is handled honestly at the verification layer: any action the environment cannot land stays pending in the audit log.

For engagements, the same engine runs from the consultant's laptop or a CI runner against the client account: an IAM role restricted to read-only describe/list/get for the audit phase, with remediation either executed through a scoped write policy or handed to the generated Terraform modules applied by the client. The CUR export is delivered from the client's S3 billing bucket into the same parser the demo uses. Data handling stays single-account, single-region (us-west-2) per engagement; no cross-client data ever touches the same database, and the SQLite store is handed over or destroyed at engagement close.

7. Data and Cost Model

Money enters the system through one door: the Cost and Usage Report. The engine parses the standard CUR 2.0 column schema (line_item/UnblendedCost, UsageType, ResourceId, product attributes) from resource-daily aggregation files, rolling usage up per service, per resource, and per usage type, and tracking how many days each resource has been billed — which is how zombie snapshots are caught: a snapshot billed 60 of the last 60 days with no AMI reference is waste regardless of what the console's creation-date column claims. In the demo, the CUR file is generated from live API state so billing always reconciles with inventory; in live engagements, the same parser consumes the client's real CUR export.

Savings math runs on a pricing table of public us-west-2 on-demand list rates (EC2 per-hour, EBS and S3 per GB-month, ALB and NAT per-hour, EIP idle-hours), with modeled conservative rates for roadmap levers (62% spot discount, 27% one-year Compute Savings Plans, list-price deltas for Graviton). Each finding's monthly savings derives from observed quantities — instance hours, provisioned GB, billed days — times these rates, never from guesses. The pricing table is a swap-in module: production deployments refresh it from the AWS Price List API, and drift shows up immediately in the reconciliation check rather than in a client dispute.

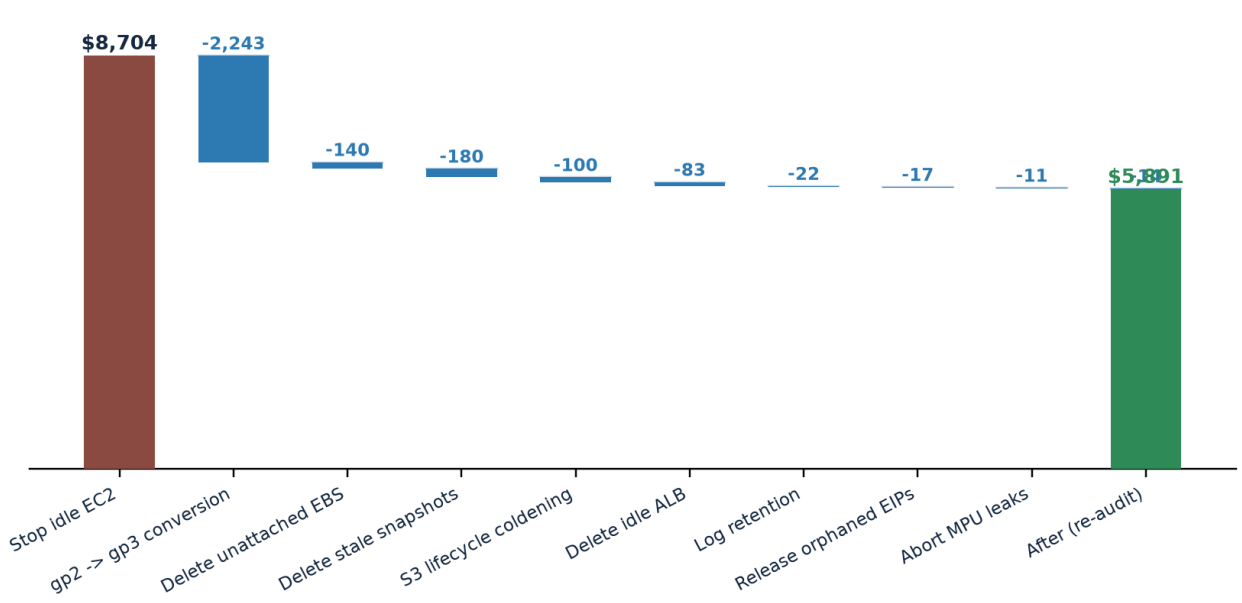

Figure 3 — Measured savings waterfall: $8,703.59 baseline to $5,890.53 post-remediation by rule category (all Tier 0/1, verified).

The reconciliation check is the honesty mechanism: after every audit, the engine rebuilds the monthly run-rate bottom-up from inventory times list pricing and compares it to the CUR-derived rate. The measured demo variance is -0.2% before remediation and -0.3% after — the residual is the trailing-window artifact of two recently-created snapshots, a textbook example of the kind of nuance an auditor should surface, not hide. In live engagements, variance above a few percent flags CUR delivery lag (new resources) or pricing drift (table needs refresh), and the report says which.

−0.2%	−0.3%
billing vs inventory variance, pre-remediation	billing vs inventory variance, post-remediation

8. Business Model and Unit Economics

Pricing is fixed-fee and outcome-anchored, in three tiers. The $3,000 Lite audit covers accounts under $10k/month: full scan, findings report, Tier 0 cleanup, one re-audit. The $5,000 Standard audit — the flagship — covers accounts to $30k/month and adds Tier 1 live remediation, the generated Terraform modules, and the 30-day roadmap review call. The $8,000 Deep-dive covers to $60k/month and adds Savings Plans modeling, a rightsizing workshop with the engineering team, and a quarterly re-audit subscription option at $1,500. All tiers share the same engine; the tiers price complexity and follow-up, not effort, because the engine has already collapsed the effort curve.

Tier	Price	Scope	Includes
Lite	$3,000	Accounts < $10k/mo	Full audit, findings report, Tier 0 cleanup, re-audit
Standard	$5,000	Accounts < $30k/mo	Adds Tier 1 live remediation, Terraform modules, roadmap review
Deep-dive	$8,000	Accounts < $60k/mo	Adds Savings Plans modeling, rightsizing workshop, quarterly option
Re-audit	$1,500/q	Past clients	Quarterly re-scan + drift report (near-pure margin)

Table 3 — Fixed-fee engagement tiers.

The unit economics work because delivery cost is decoupled from list price. A manual audit of this scope takes a senior consultant roughly 35 hours: discovery calls, console and CLI archaeology, spreadsheet modeling, report writing, and a remediation plan the client may never execute. The engine performs the audit pass in about 40 minutes and the full engagement — including client calls, remediation execution, verification, and report review — in roughly 6 consultant hours. At the Standard tier that is an effective rate north of $800/hour on work the market prices at $150, and the margin is quality-verified by construction: every number in the report traces to a billing line and every action to an API read-back.

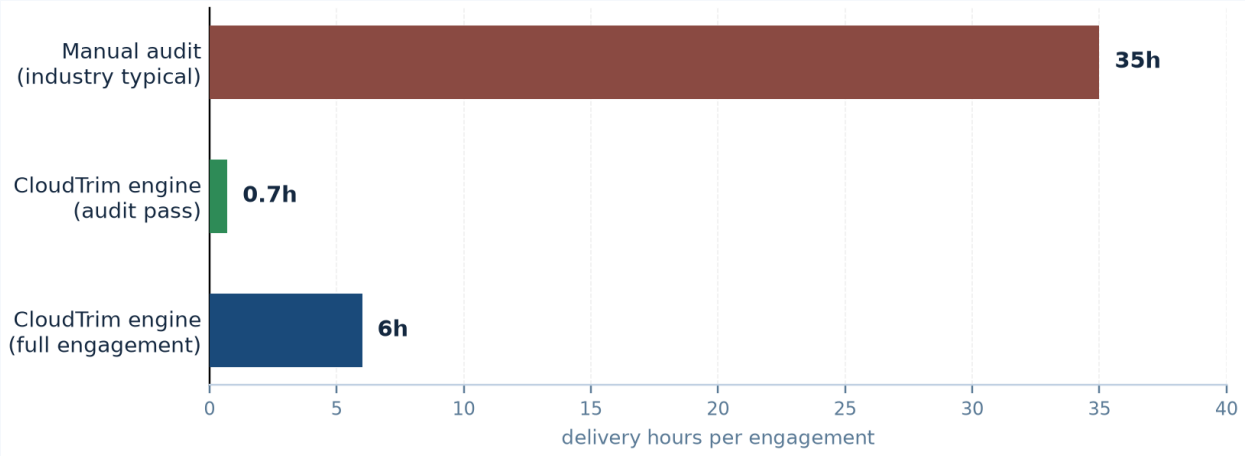

Figure 4 — Delivery hours per engagement: manual audit vs engine-assisted (engine pass ~40 min; full engagement with client touchpoints ~6 h).

The client's ROI arithmetic closes the sale. The measured demo account — $8,704/month, squarely mid-ICP — realizes $2,813/month in verified savings against a $5,000 fee: 1.8-month payback, $33,757 annualized, before touching the $3,632/month roadmap. An engagement that finds less than the fee's worth of savings is still deliverable, but the positioning makes the expected case explicit: the audit pays for itself inside two billing cycles, and the roadmap compounds it. Referral economics follow naturally — startup CTOs talk to each other, and a verified 32% cut is a story that gets retold.

9. Go-to-Market Plan

Go-to-market starts where the ICP physically concentrates: the Santa Monica-Playa Vista-Culver City startup corridor, plus DTLA and the accelerator ecosystems (Science Inc. alumni, AmplifyLA portfolio circles, the AWS and DevOps meetups that fill every month). The motion is founder-led consulting sales with a productized posture: fixed scope, fixed fee, one-week turnaround, and a guarantee stated in numbers. The free lead-magnet is a 30-minute mini-scan — read-only access, a findings preview with the top three savings opportunities, no remediation — which doubles as qualification: if the mini-scan finds under $1k/month, the honest answer is the Lite tier or a raincheck.

The first 90 days target three lighthouse clients at the Standard tier, selected for retellable results: one seed-stage dev-tools company, one Series B consumer startup, one B2B SaaS with visible staging sprawl.

Each engagement produces a case study with the before/after chart, the verified action count, and a client quote — the exact artifacts the demo generates natively. Months four through six systematize: referral loop through CTO circles and the LA DevOps community, a quarterly re-audit subscription for past clients ($1,500/quarter, near-pure margin since the engine re-runs in minutes), and partnership conversations with the LA-focused fractional-CFO shops who already sit in the finance conversation.

The positioning line stays blunt on purpose: 'We cut your AWS bill 30% in a week, with zero downtime, and we show our math.' Every claim in that sentence is backed by the engine's verification layer, which means sales never outruns delivery. The anti-goal is drifting into hourly consulting or enterprise FinOps tooling — both are different businesses with different economics, and both would dilute the one thing that makes this practice sellable: a fixed price for a verified outcome.

Phase	Window	Focus	Target outcome
Lighthouse	Days 1-90	3 LA startups at Standard tier; case studies with before/after charts	3 retellable verified results
Systematize	Days 91-180	Referral loop via CTO circles, LA DevOps community; re-audit attach	40%+ referral rate, 30% attach
Partner	Days 181-270	Fractional-CFO and accelerator partnerships	2 steady referral channels

Table 4 — 30-60-90 day go-to-market sequence.

10. Demo and Validation Results

The demo is the proof-of-work for every claim in this document, and it runs from a clean Docker host in one command: docker compose up, then make demo. Terraform applies the seed module (110 resources: 26 instances across production, worker, staging, sandbox, and dev fleets; 31 volumes; 2 load balancers; 4 S3 buckets; 3 log groups; VPC networking), the seeder layers in 14 days of CloudWatch telemetry, 22 snapshots with billing history, 2 unused AMIs, 3 incomplete multipart uploads, redundant NAT gateways, and a 60-day CUR billing export derived from live API state. The audit then finds what was planted — plus nothing that was not.

The measured results below are from the canonical run of the shipped code (commit-stamped, reproducible on the pinned emulator versions). Findings: 99 across 14 rule categories; applied-tier savings $2,809.48/month identified, with an additional $3,632.44/month on the Tier 2/3 roadmap. Remediation: 65 actions executed, 65 verified by API read-back, zero pending, zero rolled back. Reconciliation: billing vs bottom-up inventory within 0.3% before and after. And the headline: monthly run rate $8,703.59 to $5,890.53 — a 32.32% verified reduction, $33,756.72 annualized, achieved entirely through Tier 0 and Tier 1 actions with no customer-facing endpoint touched.

Metric	Before	After	Delta
Monthly run rate (CUR)	$8,703.59	$5,890.53	−$2,813.06 (−32.32%)
Annualized run rate	$104,443	$70,686	−$33,757
Findings (audit / re-audit)	99	34	Tier 0/1 fully cleared
Remediation actions	n/a	65	65 verified, 0 pending

Metric	Before	After	Delta
Reconciliation variance	−0.2%	−0.3%	within tolerance
Roadmap savings identified	$3,632/mo	n/a	Tier 2/3, costed
Customer-facing downtime	n/a	n/a	zero

Table 5 — Measured E2E results, canonical run of the shipped code (docker compose + make demo reproduces them).

11. KPIs, Roadmap and Risks

Success metrics exist at two levels: the product and the practice. Product-level, the engine is healthy when audit delivery stays under one hour end-to-end, savings-found per engagement stays above 25% of spend for ICP accounts, remediation verification success stays at 100% on supported APIs, and reconciliation variance stays under 2% on live accounts. Practice-level, the first year targets 30-40 engagements at an average fee of $5,000 (75% Standard tier mix), a referral rate above 40%, and a quarterly re-audit attach rate of 30% — the numbers that make this a practice that scales on reputation rather than headcount.

Metric	Target	Level
Audit delivery time (end-to-end)	< 1 hour	product
Savings found per ICP engagement	> 25% of spend	product
Remediation verification success	100% on supported APIs	product
Reconciliation variance (live)	< 2%	product
Engagements, year 1	30-40 @ ~$5,000 avg	practice
Referral rate	> 40%	practice
Quarterly re-audit attach	30%	practice

Table 6 — Success metrics for product and practice.

The v1-to-v2 roadmap sequences depth before breadth. Near-term: live-mode hardening (Cost Explorer API integration for accounts without CUR set up, Pricing API refresh pipeline, Organizations/multi-account fan-out for the clients whose 'account' is actually twelve), then the Savings Plans optimizer (commitment sizing against trailing usage with reservation-aware modeling), and EBS/right-sizing recommendations driven by CloudWatch percentiles rather than averages. Mid-term: GCP support for the mixed-cloud startups that show up in the ICP, and a self-serve mini-scan portal that turns the lead magnet into a funnel with no consultant time in the first touch. Explicitly not on the roadmap: becoming a monitoring platform. Audits are the product.

The risk register is short because the scope is narrow. Pricing drift is a maintenance risk — list prices move, and stale rates understate savings — mitigated by the reconciliation check surfacing drift mechanically and a Pricing API refresh in the v2 plan. Emulator fidelity is a demo risk, not a product risk: moto and LocalStack occasionally lag real AWS behavior, which the verification layer converts into honest pending statuses rather than false claims. Trust is the adoption risk — startups are rightly nervous about write access — mitigated by the read-only audit phase, scoped remediation policies, and the option to deliver everything as Terraform the client applies themselves. Consultant bandwidth is the scale risk, addressed by the quarterly re-audit product whose delivery cost approaches zero. The residual risks are ordinary business risks: sales cycle length and

ICP concentration, both watched monthly.

Appendix A: Rules Catalog Specification

The catalog below is the normative specification for the engine's 22 rules: detection logic, savings formula, risk tier, remediation path, and where each rule is visible (the pinned demo emulators or live AWS only). Savings formulas reference the pricing module's rates; thresholds are configuration constants documented per rule.

Rule	Svc	Detection logic	Savings formula	Tier	Remediation	Visible
EC2-IDLE-STOP	EC2	Running instance, 14-day CloudWatch avg CPU < 5% (>= 10 datapoints)	instance rate x 730 h	1	stop_instances + 30-day grace tag	demo + live
EC2-RIGHTSIZE	EC2	Running instance, avg CPU 5-20%; stateful workload	(current - one-size-down rate) x 730 h	2	stop -> modify type -> start, via TF/ASG	demo + live
EC2-STOPPED-ZOMBIE	EC2	Stopped >= 30 days (no compute hours in CUR, no CW data)	attached EBS GB x rate	2	snapshot, terminate, delete volumes	demo + live
EC2-GRAVITON	EC2	Steady x86 workload (avg CPU > 20%) on m5/c5 family	(x86 - arm64 rate) x 730 h	3	arm64 AMI rebuild, rolling replace	demo + live
EC2-SPOT	EC2	Interruption-tolerant workload (tag: stateless/batch/worker/ci)	rate x 62% spot discount x 730 h	3	ASG mixed-instances policy	demo + live
SAVINGS-PLANS	EC2	On-demand compute baseline > $300/mo, stable	baseline x 27% (1-yr Compute SP)	3	commitment purchase sized to floor	demo + live
EBS-UNATTACHED	EBS	Volume state 'available' (no attachment)	GB x volume-type rate	0	snapshot (if >= 100 GB) + delete	demo + live
EBS-GP2-GP3	EBS	gp2 volume, in-use, >= 20 GB	GB x (gp2 - gp3 rate) = 20% off	1	modify_volume, live in-place	demo + live
EBS-OVERSIZED	EBS	In-use volume, read+write ops < 1.0/s over 14 days	(GB - target GB) x gp3 rate	2	snapshot -> smaller volume -> swap	demo + live
SNAP-STALE	EBS	Snapshot billed >= 45 of last 60 days, no AMI reference	stored GB x $0.05/GB-mo	0	delete snapshot	demo + live
AMI-STALE	EC2	AMI unused by instances, backing snapshots billed >= 45 days	sum of snapshot GB x rate	0	deregister + delete snapshots	live (moto echoes 1 BDM)
EIP-ORPHAN	VPC	EIP with no association	$0.005/hr x 730 h	0	release address	demo + live
ELB-IDLE	ELB	ALB with 0 registered targets, CUR shows hours+LCU	ALB hours + LCU from CUR	0	delete LB + target groups	demo + live
NAT-CONSOLIDATE	VPC	>= 2 NAT gateways in one VPC	(count - 1) x $0.045/hr x 730 h	2	route-table consolidation	demo + live
S3-LIFECYCLE	S3	No lifecycle config, storage > $10/mo (CUR)	storage x 45% blended (60% cold)	1	lifecycle: IA 30d, Glacier IR 90d, MPU abort	demo + live

Rule	Svc	Detection logic	Savings formula	Tier	Remediation	Visible
S3-MPU-LEAK	S3	Incomplete multipart uploads listed	parts GB x standard rate	0	abort uploads + lifecycle guard	demo + live
CW-LOGS-RETENTION	CloudWatch	Log group retention = Never Expire	storage x 60% (90-day steady state)	1	put_retention_policy 90d	demo + live
RDS-IDLE	RDS	DB instance 14-day avg CPU < 5%	instance + storage rates	1	snapshot + stop	live only
RDS-GP3	RDS	DB storage type gp2	GB x (gp2 - gp3 RDS rate)	1	modify_db_instance, live	live only
ELASTICACHE-IDLE	ElastiCache	Cluster utilization review vs engine metrics	node rate x 50%	2	rightsize node type / replica count	live only
LAMBDA-CONCURRENCY	Lambda	Reserved concurrency with low invocations	headroom value (advisory)	2	drop reservation after metric review	live only
ELB-CLASSIC	ELB	Classic ELB present	advisory (deprecated generation)	2	ALB migration, DNS cutover	live only

Table 7 — Normative rule specification (22 rules, 4 risk tiers). Rates reference the pricing module; thresholds are configuration constants.