URL Ledger

90-Day MVP Build Plan & Engineering Backlog

From audit wedge to governed URL asset system of record

Build thesis: The MVP does not need to become a full AI platform. It needs to become the canonical ledger where humans, analytics systems, and future agents read URL asset truth, request actions, and reconcile outcomes.

Product category	Website Asset Intelligence
MVP wedge	45-Day URL Portfolio Repricing Audit
Core system	URL asset ledger + ratings + evidence + action queue
Strategic end state	System of record and transaction layer for URL-level asset value

Confidential working blueprint - prepared for product, engineering, delivery, and pilot planning

1. Executive build summary

The first build target is not a full-featured platform. The first build target is a credible, usable, audit-backed system of record that can ingest a domain, normalize URL assets, score structural variables, generate a recovery backlog, and produce evidence-backed reports.

- Use the 45-Day URL Portfolio Repricing Audit as the wedge and data collection motion.
- Turn each audit into a canonical URL asset ledger with persistent URL IDs, lineage, snapshots, ratings, evidence, and recommended actions.
- Ship enough automation to make the audit repeatable, not enough automation to hide the analyst judgment that creates trust.
- Build the rating and evidence layer before the agent action layer. Agents can query the ledger only after the ledger has reliable state.
- Design every feature so it can eventually serve three consumers: human operators, executive buyers, and external/internal AI agents.

MVP question	Answer
What must be true by Day 90?	A pilot customer can connect or export data, see a normalized URL ledger, receive ratings, review evidence, approve actions, and download audit/board outputs.
What should be avoided?	Do not overbuild generic content recommendations, AI writing, broad workflow automation, or a full BI suite before the ledger and scoring spine are proven.
What is the proof point?	The product can convert messy URL inventory into a ranked, dollarized, governed recovery backlog with evidence attached to each high-priority action.
What is the strategic moat?	Historical URL state, normalized benchmarks, explainable ratings, governance policies, evidence packs, and action-outcome reconciliation.

2. Product principles

Principle	Engineering implication
URL is the unit of account	Every record, score, event, action, and evidence artifact ties back to a stable URL asset ID or cluster ID.
Truth before automation	The MVP prioritizes reconciliation, lineage, scoring confidence, and evidence over autonomous action.
Channel-agnostic by design	Organic, AI citation, paid, email, social, referral, direct, CRM, and agent observations share a common observation model.
Audit wedge, platform path	Audit outputs must be generated from the same objects that become persistent platform objects.
Human + agent governance	Permissions, action requests, approvals, and audit trails must be built as primitives, not retrofitted later.
Explainability is a feature	Every score must show why it exists, which signals created it, and how confidence was calculated.
No black-box recommendations	Actions are packaged with evidence, expected impact, effort, dependencies, and measurement plan.
System of record, not another dashboard	The product reconciles fragmented tools into canonical URL asset truth, rather than merely visualizing another metric feed.

3. MVP product scope

Included in MVP	Deferred from MVP
Domain/project setup, URL import, sitemap/crawl/GSC/GA4 export ingestion	Full real-time connector marketplace
Canonical URL registry, dedupe, redirects, lineage, cluster assignment	Perfect automated canonicalization across every edge case
Baseline 13-variable scoring and rating explainability	Machine-learned benchmark ratings across all industries
Value-at-risk and recovery backlog model	Fully automated multi-touch revenue attribution engine
Evidence pack generation and analyst notes	Legal-grade assurance certification
Action queue with owner, approval status, and measurement plan	Autonomous writeback to CMS/search console without approval
Executive report, one-pager, CSV export, API-ready data objects	Full self-serve enterprise BI builder
Agent-ingress read endpoints and policy concepts	Full external agent marketplace

MVP slogan: Ingest the portfolio. Normalize the assets. Score the risk. Rank the recovery. Attach the evidence. Govern the actions.

4. Target user journeys

User	Journey	Success moment
Founder / analyst	Creates a project, imports URL inventory, validates signals, scores assets, assembles the report.	The audit report can be generated from structured ledger data, not manually assembled from scratch.
CMO / VP Growth	Reviews value at risk, top decaying clusters, action backlog, and 90-day roadmap.	They can defend what to fund, what to freeze, what to refresh, and what to protect.
SEO / content lead	Reviews individual URL ratings, root causes, evidence, and implementation instructions.	They know exactly which URLs to refresh, merge, redirect, protect, or expand.
RevOps / analytics lead	Checks attribution mappings, conversion definitions, and confidence levels.	They trust the dollarization because assumptions and data lineage are visible.
Agent / automation layer	Queries approved URL asset state, requests actions, and reads policy restrictions.	The agent does not improvise against a website. It transacts against governed state.

5. 90-day build roadmap

Phase	Build focus	Exit criteria
Days 1-15: Foundation	Project setup, URL asset schema, ingestion stubs, URL registry, canonical ID logic, basic admin.	Can create a project, ingest URL rows, assign URL_IDs, dedupe obvious variants, and view a URL registry.
Days 16-30: Signal joins	Sitemap/crawl import, GSC/GA4 export import, CRM/revenue placeholder mapping, snapshot tables.	Can join performance observations to URL assets and show baseline metrics per URL and cluster.
Days 31-45: Ratings v1	13-variable scoring, rating explanations, confidence tiers, analyst overrides, evidence notes.	Can produce health/risk/decay/fit ratings with reason codes and confidence.
Days 46-60: Recovery backlog	Action recommendations, value-at-risk calculation, priority queue, owner fields, dependency flags.	Can produce a top-25 recovery backlog ranked by impact, effort, confidence, and urgency.
Days 61-75: Governance + evidence	Policy states, protected assets, approval workflow, evidence artifacts, change logs.	Can distinguish safe actions, blocked actions, and protected assets with evidence attached.
Days 76-90: Outputs + pilot hardening	Report generator, exports, dashboard polish, QA, pilot onboarding, API/agent read model.	Can run one pilot audit end-to-end and deliver the report, ledger export, and action backlog from the system.

6. Engineering epic map

Epic	Build scope	Acceptance test
E1. Project workspace	Customer/domain records, project status, users, roles, pilot metadata.	A project can contain one or more domains, data sources, snapshots, users, and reports.
E2. URL asset registry	URL_ID creation, canonical URL, variants, status, template type, cluster, owner.	Every imported URL resolves to a stable asset record with lineage and current state.
E3. Ingestion pipeline	CSV upload, sitemap import, crawl export import, GSC/GA4 export import.	Analysts can upload messy source files and map columns without engineering help.
E4. Normalization engine	Canonicalization rules, trailing slash rules, parameter stripping, duplicate detection, redirects.	The same URL is not counted five different ways across tools.
E5. Signal warehouse	Snapshot storage, channel observations, query/page metrics, events, revenue proxies.	Each URL has historical observations and import lineage.
E6. Scoring engine	13-variable scores, rollups, ratings, confidence, reason codes, manual overrides.	Every score can be explained, challenged, and revised without breaking audit history.
E7. Value-at-risk model	Yield, durability, risk, decay velocity, recoverable value, confidence bands.	Leaders see dollarized exposure and recovery potential by URL and cluster.
E8. Action queue	Refresh, merge, retire, redirect, protect, expand, monitor actions with owners and dependencies.	The audit creates an executable backlog, not just observations.
E9. Evidence layer	Screenshots, exports, analyst notes, source citations, reason-code evidence, report appendices.	Every recommendation has an evidence trail.
E10. Governance layer	Protected assets, policy gates, approval states, permissions, blocked actions, no-touch rules.	High-risk pages cannot be changed without explicit approval.
E11. Reporting outputs	Executive report, one-pager, ledger export, backlog export, quarterly report shell.	Client-facing deliverables can be generated from structured data.
E12. API/agent ingress	Read endpoints, policy response, action request schema, audit log for agent queries.	Agents can consume state safely without becoming the source of truth.

7. Sprint-level backlog

Sprint	Primary tickets	Definition of done
Sprint 1	Create app shell; projects/domains/users tables; URL import UI; URL_ID generator; basic registry view.	A user can create a project, import URLs, and see a clean URL table with stable IDs.
Sprint 2	Add sitemap/crawl/GSC/GA4 import types; column mapping; import validation; source_file table.	A user can import multiple sources and see validation warnings before commit.
Sprint 3	Build canonicalization rules; duplicate grouping; variant mapping; redirect/lineage fields.	Obvious duplicates and variants reconcile to one asset or lineage group.
Sprint 4	Create snapshot/observation tables; page metrics view; cluster rollup; channel coverage flags.	A URL can show performance history and imported signal coverage.
Sprint 5	Implement 13-variable scoring v1; reason codes; confidence tier; analyst override flow.	Each URL has a variable scorecard with explanations and override history.
Sprint 6	Build ratings rollup; A-D rating; risk class; decay class; portfolio health summary.	The system produces portfolio-level rating distributions and top-risk lists.
Sprint 7	Create value-at-risk model; yield/durability/risk math; recoverable	A leader can see dollarized exposure by URL, cluster, and failure mode.

Sprint	Primary tickets	Definition of done
	value assumptions; confidence bands.	
Sprint 8	Build action queue; action types; impact/effort/dependency fields; owner/approval state.	The system generates and manages a ranked recovery backlog.
Sprint 9	Add evidence artifacts; note templates; screenshot/upload support; report evidence packs.	Each high-priority action has attached evidence and analyst rationale.
Sprint 10	Add protected assets, policy rules, approval thresholds, no-touch status, action risk gates.	The product can block or escalate risky actions based on policy.
Sprint 11	Generate report DOCX/PDF shell; CSV exports; executive summary; top-25 backlog appendix.	An audit deliverable can be exported from product data.
Sprint 12	Pilot hardening; QA; seed demo data; API read endpoint; agent query log; admin cleanup.	One pilot can run end-to-end with repeatable outputs and minimal manual patching.

8. Feature requirements by module

Module	Must-have requirements	Nice-to-have later
Workspace	Projects, domains, users, roles, statuses, pilot metadata, data source inventory.	Multi-tenant SSO, billing, client portal branding.
URL registry	Stable URL_ID, canonical URL, variants, status, cluster, template, lifecycle, owner.	Automated similarity clustering, visual URL graph explorer.
Data ingestion	CSV upload, source type selection, column mapping, validation, commit/rollback.	Direct OAuth connectors, scheduled sync, warehouse reverse ETL.
Scoring	13-variable score fields, rating formulas, reason codes, confidence levels, overrides.	Industry benchmarks, ML weighting, auto-calibrated confidence.
Backlog	Action type, expected impact, effort, dependency, owner, approval state, measurement plan.	Jira/Asana/ClickUp sync, workflow automations.
Evidence	Attach notes, source files, screenshots, exports, before/after metrics, rationale.	Evidence notarization, certification pack, external auditor mode.
Governance	Protected asset flags, policy thresholds, approval rules, no-touch pages, audit trail.	Role-based action simulations, automated rollback playbooks.
Reports	Executive PDF/DOCX, one-pager, ledger CSV, backlog CSV, board-ready summary.	Interactive client portal, scheduled quarterly 10-K generation.
Agent ingress	Read-only API, policy response, action request schema, query logs.	Writeback APIs, external agent marketplace, protocol integrations.

9. 13-variable scoring implementation

Variable family	MVP signals	Output
1. Content health	Freshness, completeness, current status, basic quality notes.	Health score + refresh need.
2. Content decay	Clicks/impressions/sessions trend, peak date, decline velocity.	Decay class + half-life estimate.
3. Content dilution	Low-value inventory share, weak pages inside cluster, thin scale.	Dilution risk by cluster.
4. Cannibalization	Overlapping queries/pages, duplicate intent, cluster crowding.	Merge/redirect candidates.
5. Content waste	Low traffic, low fit, low conversion, stale no-owner pages.	Retire/contain list.
6. Content investment	Age, update history, asset type, strategic cost proxy.	Protect vs fix vs abandon logic.
7. Content ROI	Conversion mapping, revenue proxy, pipeline influence, RPM/LPV.	Value contribution estimate.
8. Performance	Sessions, clicks, impressions, CTR, conversions, engagement proxy.	Performance grade.
9. Intent alignment	TOFU/MOFU/BOFU, branded, commercial, transactional, support.	Intent fit + journey imbalance.
10. Psychographic fit	Decision reassurance, trust proof, objection coverage, message mismatch.	Message risk notes.
11. Authority/entity strength	Internal links, backlinks proxy, author/brand proof, topical role.	Authority contribution score.
12. Technical/indexation	Indexability, canonicals, redirects, depth, schema, CWV, render risk.	Structural risk score.
13. AI/agent readiness	Extractability, citation readiness, policy visibility, machine-readable evidence.	Agent-readiness class.

10. Value-at-risk calculation v1

The MVP model should be explainable before it is precise. Use transparent assumptions and confidence bands. The objective is better capital allocation, not false accounting certainty.

Concept	MVP formula / logic
URL yield	Recent sessions or clicks x revenue/conversion proxy, adjusted by landing-page value or CRM mapping where available.
Decay exposure	Observed decline from baseline or peak, adjusted by seasonality notes and confidence.
Structural risk	Weighted combination of technical/indexation, cannibalization, dilution, and governance signals.
Recoverability	Combination of strategic fit, fix effort, historical authority, and clarity of root cause.
Value at risk	Yield x decay exposure x structural risk confidence.
Recoverable value	Value at risk x recoverability factor x implementation confidence.
Priority score	Recoverable value x confidence / effort, with protected-asset gating.
Confidence band	High where source data, signal agreement, and root cause are strong; low where assumptions dominate.

11. Data model priorities for engineering

Object	Purpose	Critical fields
Project	Customer/domain workspace.	project_id, customer_name, domain, status, start_date, owner.
URL Asset	Canonical unit of account.	url_id, canonical_url, status, template_type, cluster_id, owner, protected_flag.
URL Variant	Maps duplicates, parameters, redirects, prior URLs.	variant_url, url_id, relationship_type, detected_source.
Import Source	Tracks files/connectors used.	source_id, type, file_name, imported_by, imported_at, validation_status.
Snapshot	Point-in-time state.	snapshot_id, url_id, captured_at, source_id, channel, metric_set.
Observation	Channel event or metric.	observation_id, url_id, metric_name, value, period, source_id.
Score	Variable-level scoring.	score_id, url_id, variable_id, score_value, confidence, reason_codes.
Rating	Rollup grade.	rating_id, url_id, rating, health, risk, decay, fit, confidence.
Action	Recommended or approved action.	action_id, url_id, action_type, priority, impact, effort, status, owner.
Evidence	Proof for score/action.	evidence_id, url_id, action_id, artifact_type, note, source_ref, created_by.
Policy	Governance rules.	policy_id, scope, rule_type, threshold, approval_required, blocked_actions.
Agent Query	Agent read/request audit trail.	query_id, agent_id, url_id, permission_scope, response_status, timestamp.

12. API and agent ingress MVP

Endpoint / interface	Purpose	MVP behavior
GET /projects/{id}/urls	Return URL asset registry.	Read-only, filterable by cluster, rating, status, protected flag.
GET /urls/{url_id}	Return canonical URL asset state.	Includes identity, rating, current risk, recommended action, policy state.
GET /urls/{url_id}/evidence	Return evidence artifacts and reason codes.	Read-only with redaction of sensitive notes if required.
GET /clusters/{cluster_id}	Return cluster rollup.	Ratings, value at risk, top issues, top actions.
POST /action_requests	Allow an agent or user to request an action.	Creates pending request; does not execute changes.
GET /policies/evaluate	Check whether an action is allowed.	Returns allow, escalate, or block with reason.
GET /reports/{id}/export	Retrieve generated report/export.	Authenticated download of approved output.
Agent query log	Audit all access.	Logs who/what queried which URL state and what response was given.

13. Technical architecture v1

Layer	Recommended MVP approach
Frontend	React/Next.js or equivalent; focus on tables, filters, scorecards, action queue, and report previews.

Layer	Recommended MVP approach
Backend API	Node/TypeScript, Python/FastAPI, or Rails/Django; choose team familiarity over novelty.
Database	Postgres for canonical ledger; JSONB for flexible source payloads; later warehouse sync optional.
File storage	Object storage for imports, evidence artifacts, exports, screenshots, and report packs.
Workers	Background jobs for imports, normalization, scoring recalculation, report generation.
Auth	Email/password or magic link for MVP; role-based permissions at project level.
Reporting	Generate DOCX/PDF from templates; CSV exports for ledger and backlog.
Analytics	Track product usage, report generation, action queue changes, and pilot outcomes.
Security	Read-only data bias, import audit trail, least-privilege roles, protected asset gating.
Future warehouse	BigQuery/Snowflake/ClickHouse sync once pilots require enterprise scale.

14. QA, validation, and data quality gates

Gate	Validation rule
Import validation	Required columns mapped, row counts shown, URL format checked, duplicate rows flagged.
URL normalization QA	Sample 100 URL assets; confirm variants, canonicals, redirects, and parameters map correctly.
Metric join QA	Imported page metrics reconcile to source totals within acceptable tolerance.
Score QA	Top/bottom 25 URLs manually reviewed against evidence to confirm score direction.
Value-at-risk QA	Assumptions visible; confidence bands applied; no dollar figures shown without a source/proxy label.
Action QA	Every top-25 action has action type, expected impact, effort, owner, dependency, and evidence.
Report QA	Executive summary, tables, and appendix reconcile to ledger data.
Governance QA	Protected assets cannot be moved into approved action state without escalation.
Export QA	CSV and PDF/DOCX exports match current project state and include timestamp/source notes.
Pilot QA	A non-builder analyst can repeat the workflow from import to report using the playbook.

15. Team and operating cadence

Role	Responsibility in first 90 days
Product owner	Owns scope, pilot outcomes, user stories, acceptance criteria, and prioritization.
Full-stack lead	Owns application architecture, database model, API, frontend, and engineering standards.
Data engineer	Owns ingestion, normalization, source validation, scoring jobs, and exports.
Analyst / domain expert	Owns scoring rubric, reason codes, evidence standards, QA, and report quality.
Designer / UX	Owns registry, scorecard, action queue, dashboard, and report preview workflows.
Delivery lead	Owns pilot intake, access, client communication, and audit timeline.
Security/advisor	Reviews data access, customer permissions, and agent-ingress policy assumptions.

Weekly rhythm	Output
Monday build review	Confirm sprint scope, blockers, and acceptance criteria.
Wednesday data/model review	Review imports, normalization quality, scores, and edge cases.
Friday demo	Demo working product against a pilot scenario, not abstract feature completion.
Weekly risk review	Update scope cuts, pilot dependencies, security risks, and credibility risks.
End-of-sprint readout	Ship notes, demo link, QA status, and next sprint backlog.

16. Pilot readiness checklist

Area	Ready when
Data intake	The customer can provide sitemap/crawl, GSC, GA4, and revenue proxy exports using a documented workbook.
Ingestion	Analyst can import and validate source files without developer intervention.
Ledger	All URLs have stable IDs, status, cluster, current metrics, and source lineage.
Scoring	At least 80% of important URLs have variable scores and confidence levels.
Backlog	Top 25 actions have evidence, impact, effort, owner, and measurement plans.
Report	Executive report and appendices generate from structured data with minimal manual editing.
Governance	Protected assets, approval gates, and no-touch rules are visible in the action queue.
Security	Data access is read-only or import-based, with clear customer approval.
Support	Delivery team has SOP, troubleshooting steps, and escalation path.
Outcome	The customer can understand what to fix, what not to touch, and what to fund next.

17. Risks and scope controls

Risk	Control
Trying to build full SEO suite	Keep scope anchored to ledger, rating, evidence, action queue, and reports.
Trying to automate perfect recommendations	Use analyst-reviewed action generation for MVP; automate structure, not judgment.
Dollarization credibility risk	Label revenue source and confidence; show proxy assumptions; avoid false precision.
Connector rabbit hole	Start with exports and imports; add direct connectors after pilot pain is proven.
Scoring complexity	Use transparent formulas and reason codes; benchmark later.
Agent hype drift	Agent ingress is read/request/reconcile, not autonomous publishing.
Report quality gap	Treat reports as generated products with QA, not incidental exports.
Messy URL normalization	Document edge cases, expose mapping, and allow analyst overrides.
Buyer confusion	Position as website asset intelligence and system of record, not AI search tracker.
Overcustomization per pilot	Separate customer-specific assumptions from reusable product objects.

18. Definition of done for the first commercial MVP

The first commercial MVP is done when URL Ledger can run a real customer through intake, import, scoring, backlog generation, governance review, report export, and audit-to-platform expansion using the same canonical URL asset records.

- A customer domain can be represented as a normalized URL asset portfolio.
- The portfolio can be scored across the 13 structural variables with visible confidence levels.
- The system can show value at risk, recoverable value, and priority actions by URL and cluster.
- Every top recommendation has an evidence trail and measurement plan.
- Protected assets and approval rules are enforced in the backlog.
- The executive report, one-pager, and CSV exports can be generated from product data.
- A pilot user can see the difference between audit output and the ongoing ledger subscription path.
- The product creates a reason for renewal: recurring monitoring, quarterly reporting, benchmarks, governance, and agent ingress.

19. Next build decisions

Decision	Default recommendation
Build vs no-code prototype	Use a real app shell and database if the goal is investor/customer credibility; use no-code only for clickable demo support.
Connector priority	Start with CSV/export ingestion for crawl, GSC, GA4, sitemap, CRM/payment proxy. Build OAuth later.
Scoring formula ownership	Version formulas in code and store score explanations in database for auditability.
First pilot vertical	Choose a portfolio with 500+ URLs, obvious revenue mapping, and visible decay/waste pain.
First monetization	Sell 45-day audit and credit part of the audit fee toward the first-year ledger subscription.
First benchmark asset	Use pilot baselines to create anonymous distributions only after normalization standards are stable.
First agent feature	Read-only URL state endpoint plus action request endpoint with policy evaluation.
First renewal hook	Quarterly Content 10-K report plus continuous decay/risk monitoring.

20. Appendix: sample epics as tickets

Ticket ID	Ticket	Epic
T-001	Create project and domain model	Workspace
T-002	Build CSV upload and column-mapping flow	Ingestion
T-003	Generate stable URL_ID and canonical URL field	Registry
T-004	Create URL variant and duplicate mapping table	Normalization
T-005	Import sitemap/crawl/GSC/GA4 exports	Ingestion
T-006	Create metric snapshot and observation tables	Data
T-007	Create cluster assignment and rollup view	Registry
T-008	Implement 13-variable score model v1	Scoring
T-009	Add reason codes and confidence tiers	Scoring
T-010	Add analyst override and override history	Governance
T-011	Build value-at-risk calculator	Economics
T-012	Create ranked action queue	Actions
T-013	Add action types and approval states	Actions
T-014	Add evidence artifact uploads and notes	Evidence
T-015	Create protected asset and policy rule model	Governance
T-016	Generate executive report from ledger data	Reporting
T-017	Generate top-25 backlog export	Reporting
T-018	Create portfolio overview dashboard	Dashboard
T-019	Create URL asset detail page	Dashboard
T-020	Build read-only API for URL asset state	Agent ingress
T-021	Build action request endpoint	Agent ingress
T-022	Log agent/user queries and action requests	Audit trail
T-023	Seed demo project and sample data	Demo
T-024	Run pilot QA checklist end-to-end	Pilot
T-025	Prepare pilot handoff and subscription upsell view	Commercial

21. Closing build thesis

URL Ledger wins by becoming the durable record underneath website asset decisions. The MVP should not chase every surface where content performs. It should build the spine that allows every surface - search, AI answers, paid, social, email, CRM, BI, and agents - to be reconciled against one URL asset truth.

The practical path is simple: sell the audit, use the audit to populate the ledger, use the ledger to create ratings, use ratings to generate governed actions, use actions to prove outcomes, and use outcomes to compound benchmarks and trust. That is the product loop.