RORI — Project Summary

Research On Regulatory for Industry(s) *Prepared: 2026-02-23* | *Authors: The Frank-cicle & Candi*

What RORI Is

RORI is a curated regulatory knowledge platform with agent-based retrieval. It is designed to ingest large regulatory data corpuses across disparate types and formats, perform deep research and gap analysis between those corpuses, and synthesize findings into actionable plans that can withstand audit and regulatory scrutiny.

RORI is not a chatbot or a general-purpose AI tool. It is a precision research engine where accuracy, consistency, and completeness are non-negotiable.

Dual Purpose

RORI serves two roles simultaneously. First, it is the primary deliverable — a production regulatory research engine. Second, it is the validation case for DevFlywheel, The Frank-cicle's development workflow methodology. Every process decision made during RORI's build feeds back into refining how projects get built going forward.

Target Verticals

Three initial industry verticals have been identified:

1. **Mortgage / First-Time Homebuyers** — Federal and state regulations, GSE selling and servicing guides (Fannie Mae, Freddie Mac), CFPB guidance, fair lending, and down payment assistance programs.
2. **Insurance Regulation** — US state-level regulatory requirements for brokers and agents, coverage mandates, and cross-state compliance.
3. **Medical Regulations for Gig Platforms** — Regulatory landscape impacting clinicians operating through gig economy platforms.

Four Modes of Use

RORI targets four distinct consumer profiles: curated data ingestion across formats and sources, agent-based retrieval with tunable depth and completeness, developer integration as a context manifold for AI systems, and packaged end-user applications for specific verticals.

Where RORI Stands Today

The project has completed its launch phase. Foundational documentation, scaffolding, and planning are in place. No application code has been written yet — by design, the project emphasizes comprehensive specification

before implementation.

What's Been Completed

Project scaffolding is established at both `X:\RORI` and `C:\DATA\RORI` with the DevFlywheel directory structure (`docs/`, `plans/`, `prompts/`, `context/`, `research/`, `scripts/`, `.cursor/`). The Cursor DFW Extension was successfully tested for automatic scaffolding creation.

The project proposal has been written and converted into a polished 12-page PDF with custom infographics and workflow diagrams.

The Macro Development Plan reached v1.1 after the "Nexio bleed" correction — a key early course correction where assumptions from another project (Nexio) were identified and stripped out. The plan is organized into phased development (Phase 0 through Phase 3) plus a parallel Collection & Research swim lane. Every component maps to a feature branch off `main`, scoped for individual contributors.

The Web Scraping Infrastructure Specification (v1.0) is complete. This is the first component spec in the Collection & Research swim lane. It covers tool selection (Firecrawl as primary, Crawlee as fallback), a provider-agnostic adapter architecture, the output envelope schema, the frontend approach (GPT-generated manifests with a slim React monitor), scheduling, error handling, and audit logging.

Journal documentation is current through 2026-02-14, with detailed session tracking across the full launch sprint (seven sessions on day one, plus the spec session on day two).

What's In Progress or Next

Three tracks are queued for continuation:

1. **Research Manifest System specification** — This is the critical dependency. The web scraping infra scrapes only sources on an approved manifest. The manifest system defines how sources are seeded, expanded, reviewed, and approved. Without it, the collection layer has no targets.
2. **Project housekeeping** — Populating the scaffolding files that are still templates: `_ACTIVE_CONTEXT.md`, `_WISHLIST.md`, `_ROADMAP.md` from the proposal and macro plan.
3. **Phase 0 research kickoff** — Deep research into retrieval algorithms, chunking strategies, repository architecture, and agent frameworks for legal/regulatory text. Each produces an Architecture Decision Record (ADR).

Architecture at a Glance

Macro Plan Structure

The development plan runs on two parallel tracks:

Phased Development (phase0–phase3): The core platform — foundation research, the ingestion and indexing engine, the retrieval and agent layer, API and vertical overlays, and scaling infrastructure. Components are sequenced with explicit dependencies.

Collection & Research (collect/*): The data acquisition machinery — web scraping, file system ingestion, AI-assisted research discovery, and source corroboration. This swim lane runs in parallel with Phase 1 and feeds into the ingestion pipeline. It follows a seed → expand → review → approve workflow to keep the repository curated.

The two tracks converge at the ingestion pipeline: the collection swim lane produces staged documents, and the ingestion pipeline processes them into the indexed repository.

Branch Naming Convention

All branches follow: `phase{N}/{component-name}` or `{swim-lane}/{component-name}`

Examples: `phase0/research-retrieval-algorithms`, `phase1/ingestion-pipeline`, `collect/web-scraping-infra`

Key Architectural Decisions Made

Decision	Resolution
Branch model	One branch per component, PR back to main
Swim lane model	Parallel Collection & Research track alongside phased core development
Manifest gating	All collection activity gated by an approved research manifest
Web scraping tooling	Firecrawl (primary), Crawlee (fallback), provider-agnostic adapter layer
Frontend for collection	Option A — GPT-generated manifests with slim React monitoring interface
Spec-before-code	All components specified before implementation begins

---

Key Learnings & Principles

Nexio bleed is real. Working across multiple projects creates gravitational pull toward the most recent architecture in your head. The discipline of grounding each plan in its own defining documents — not borrowed mental models — is critical. This was caught and corrected early in the macro planning phase.

The collection layer is foundational. It's tempting to jump to retrieval algorithms and agent frameworks, but without a disciplined, manifest-driven data acquisition pipeline, there's nothing to retrieve. The seed → expand → review → approve workflow is the backbone of RORI's value proposition: a *curated* repository.

Systematic documentation bridges sessions. Detailed journal entries with session tracking, decision logs, and explicit next-session priorities have proven essential for maintaining continuity across work periods.

Provider-agnostic architecture reduces lock-in risk. The web scraping spec demonstrated this principle — wrapping Firecrawl and Crawlee behind a common adapter interface means the system can switch providers without touching business logic.

Artifacts & Locations

Artifact	Location
Project Scaffolding	`C:\DATA\RORI\` and `X:\RORI\`
Project Proposal (PDF)	`C:\DATA\RORI\docs\`
Macro Development Plan v1.1	`C:\DATA\RORI\plans\RORI_MACRO_DEVELOPMENT_PLAN_v1.1.md`
Web Scraping Spec v1.0	`C:\DATA\RORI\plans\RORI-Web-Scraping-Infra-SPEC-v1.0.md`
Journal Entries	Obsidian vault: `01-IFS-Journal/`
Decision Log	`C:\DATA\RORI\context\_DECISIONS_LOG.md`

This summary reflects project state as of 2026-02-23. Prepared as a review baseline before spec evolution.