Contents

RADIANT v5.0.2 - System Evolution: Complete Source Export

For Gemini Evaluation Date: January 9, 2026

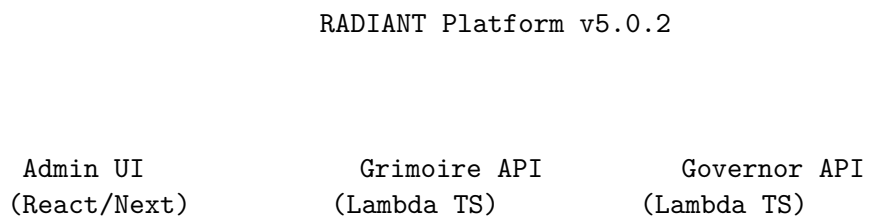
Executive Summary

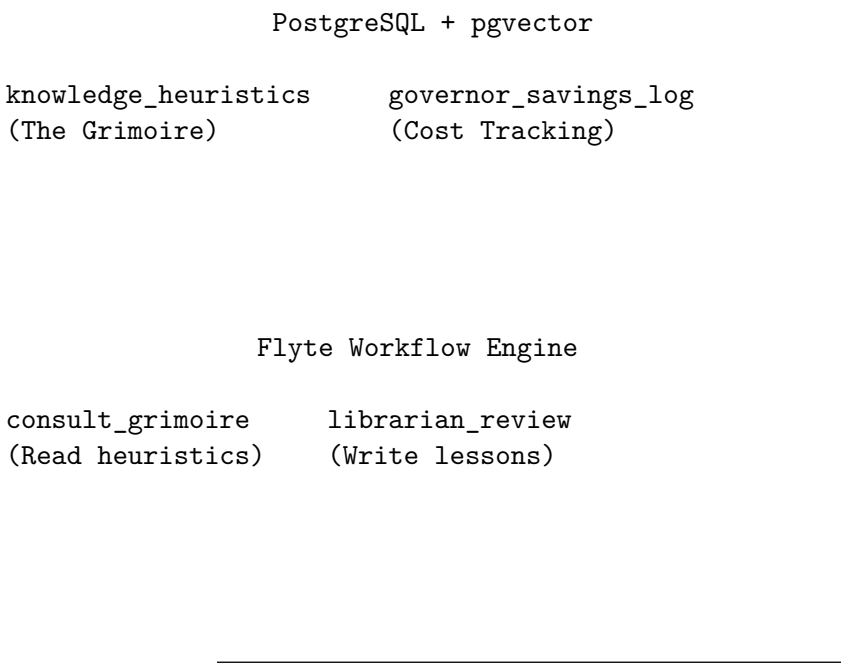
This document contains the complete source code and documentation for RADIANT v5.0.2's "System Evolution" features:

1. **The Grimoire** - Self-optimizing procedural memory system
2. **The Economic Governor** - Cost optimization through intelligent model routing

---

Architecture Overview





File Inventory

Database Schema

File	Purpose
`migrations/V2026_01_09_001__v5_grimoire_governor.sql`	Database tables, indexes, RLS, views

TypeScript Services (AWS Lambda)

File	Purpose
`lambda/shared/services/governor/economic-governor.ts`	Core Governor logic
`lambda/shared/services/governor/index.ts`	Module exports
`lambda/grimoire-api/index.ts`	Grimoire REST API
`lambda/governor-api/index.ts`	Governor REST API

Python Flyte Tasks

File	Purpose
`flyte/workflows/grimoire_tasks.py`	Grimoire Flyte tasks
`flyte/utils/db.py`	RLS-safe database connections
`flyte/utils/embeddings.py`	Vector embedding generation
`flyte/utils/cato_client.py`	Safety validation HTTP client

File	Purpose
`flyte/utils/__init__.py`	Module exports

AWS CDK Infrastructure

File	Purpose
`lib/stacks/grimoire-stack.ts`	Lambda, EventBridge, VPC config
`lambda/grimoire/cleanup.py`	Scheduled cleanup Lambda

Admin Dashboard (React/Next.js)

File	Purpose
`app/(dashboard)/thinktank/grimoire/page.tsx`	Grimoire admin UI
`app/(dashboard)/thinktank/governor/page.tsx`	Governor admin UI

Feature: The Grimoire

Concept

The Grimoire is a **self-optimizing procedural memory system** that enables AI agents to learn from successful executions. It stores "heuristics" - reusable lessons extracted from good outcomes.

Key Operations

1. **consult_grimoire** - Query for relevant heuristics before execution
2. **librarian_review** - Extract lessons after successful execution
3. **cleanup_expired** - Remove stale heuristics (scheduled daily)

Data Model

```
CREATE TABLE knowledge_heuristics (
    id UUID PRIMARY KEY,
    tenant_id UUID NOT NULL,
    domain VARCHAR(50) NOT NULL,
    heuristic_text TEXT NOT NULL,
    context_embedding vector(1536),  -- pgvector for semantic search
    confidence_score FLOAT DEFAULT 0.5,
    source_execution_id VARCHAR(255),
    created_at TIMESTAMPTZ,
    updated_at TIMESTAMPTZ,
    expires_at TIMESTAMPTZ DEFAULT (NOW() + INTERVAL '90 days')
);
```

Security Features

- **RLS** - Tenant isolation via `app.current_tenant_id`
- **Cato Validation** - All heuristics validated before storage/retrieval
- **Fail-Closed for Writes** - Blocks unsafe content from being stored

---

Feature: The Economic Governor

Concept

The Economic Governor uses a **"System 0" classifier** (cheap model) to analyze task complexity and route to the most cost-effective model.

Complexity Scale

Score	Complexity	Routing
1-4	Simple	→ gpt-4o-mini (cheap)
5-8	Medium	→ Original model
9-10	Complex	→ gpt-4o (premium)

Modes

Mode	Behavior
`performance`	No optimization, always use original model
`balanced`	Optimize simple tasks, preserve quality for complex
`cost_saver`	Aggressive optimization (may impact quality)
`off`	Completely disabled

Data Model

```
CREATE TABLE governor_savings_log (
    id UUID PRIMARY KEY,
    tenant_id UUID NOT NULL,
    execution_id VARCHAR(255) NOT NULL,
    original_model VARCHAR(100) NOT NULL,
    selected_model VARCHAR(100) NOT NULL,
    complexity_score INTEGER CHECK (1-10),
    estimated_original_cost DECIMAL(10,6),
    estimated_actual_cost DECIMAL(10,6),
    savings_amount DECIMAL(10,6),
    governor_mode VARCHAR(20) NOT NULL,
    reason TEXT,
    created_at TIMESTAMPTZ
);
```

Complete Source Files

The complete source code is split across 5 companion files for manageability:

Part	File	Contents
1	`GRIMOIRE-GOVERNOR-SOURCE-PART1.md`	Database Schema, Economic Governor Service
2	`GRIMOIRE-GOVERNOR-SOURCE-PART2.md`	Cato API, Governor API (Lambda handlers)
3	`GRIMOIRE-GOVERNOR-SOURCE-PART3.md`	Flyte Tasks (Python)
4	`GRIMOIRE-GOVERNOR-SOURCE-PART4.md`	Python Utilities (DB, Embeddings, Cato Client)
5	`GRIMOIRE-GOVERNOR-SOURCE-PART5.md`	CDK Stack, Cleanup Lambda, Admin UI (React)

Source File Index

#	File	Language	Lines	Part
1	`migrations/V2026_01_09_001__v5_grimoire_governor.sql`	SQL	~170	1
2	`lambda/shared/services/governor/economic-governor.ts`	TypeScript	~260	1
3	`lambda/shared/services/governor/index.ts`	TypeScript	~15	1
4	`lambda/grimoire-api/index.ts`	TypeScript	~150	2
5	`lambda/governor-api/index.ts`	TypeScript	~180	2
6	`flyte/workflows/grimoire_tasks.py`	Python	~400	3
7	`flyte/utils/db.py`	Python	~210	4
8	`flyte/utils/embeddings.py`	Python	~220	4
9	`flyte/utils/cato_client.py`	Python	~250	4
10	`flyte/utils/__init__.py`	Python	~50	4
11	`lib/stacks/grimoire-stack.ts`	TypeScript	~195	5
12	`lambda/grimoire/cleanup.py`	Python	~170	5
13	`app/(dashboard)/brain/TSX/grimoire/page.tsx`	TSX	~300	5
14	`app/(dashboard)/brain/TSX/governor/page.tsx`	TSX	~280	5
15	`flyte/workflows/brain_tank_workflow.py`	Python	~80	5

Total: 15 source files, ~2,930 lines of code