Contents

RADIANT Platform Architecture

Enterprise Multi-Tenant AI Platform

Version 4.18.0 | December 2024

A comprehensive technical architecture document for the RADIANT AI orchestration platform

---

Table of Contents

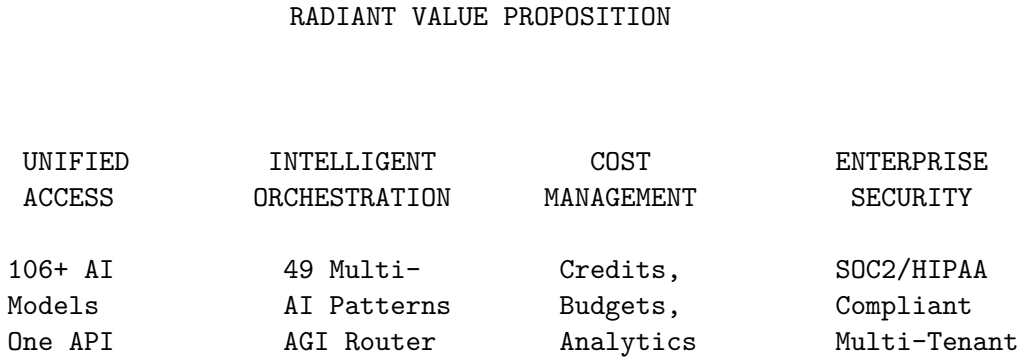
---

1. Platform Overview

1.1 What is RADIANT?

RADIANT (Real-time AI Distribution, Integration, and Automation Network for Tenants) is an enterprise-grade, multi-tenant SaaS platform that provides unified access to 106+ AI models across multiple providers, with intelligent orchestration, cost management, and comprehensive analytics.

1.2 Core Value Proposition



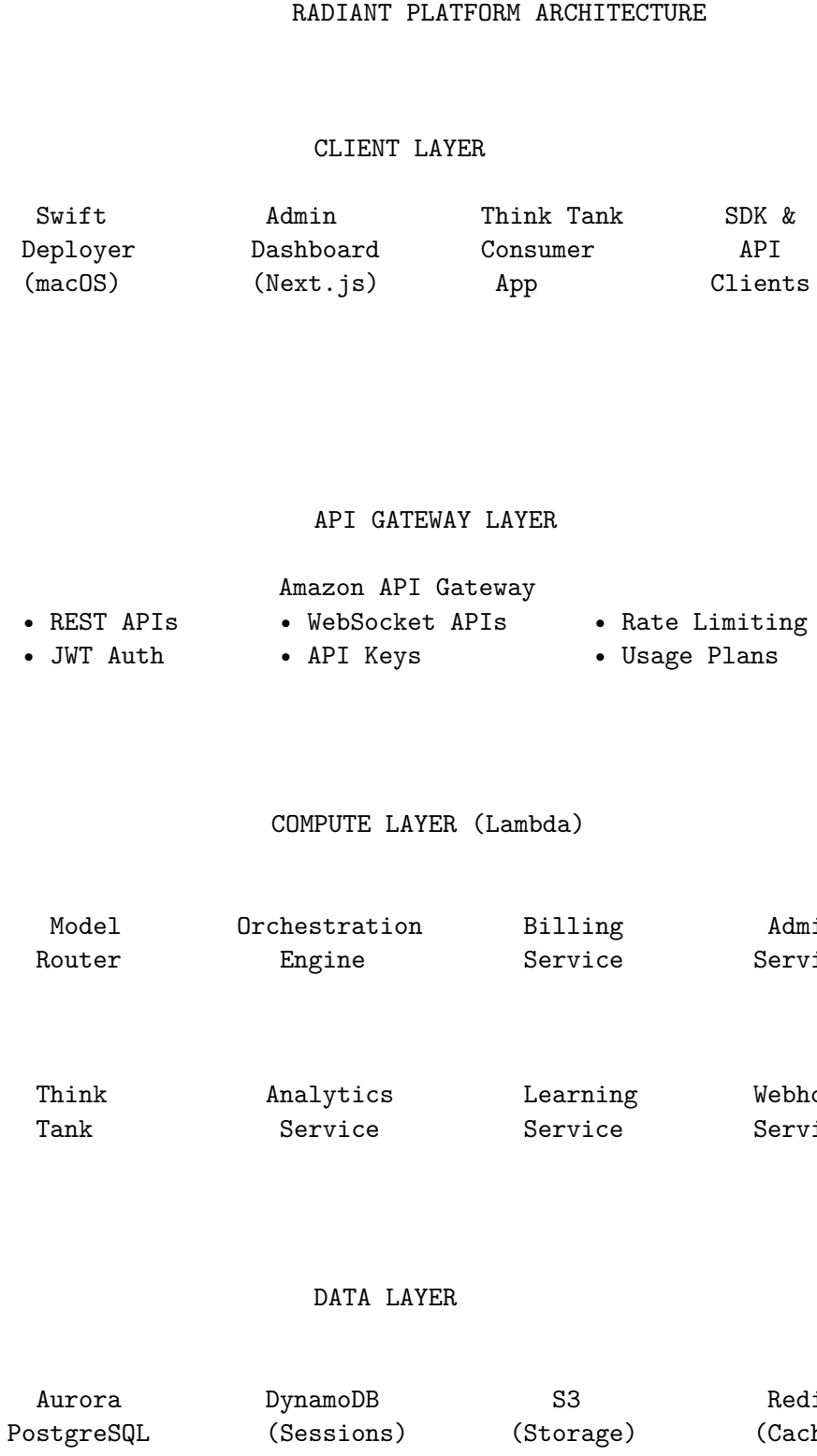
1.3 Platform Statistics

Metric	Value
AI Models Supported	106+ (50 external + 56 self-hosted)
AI Providers Integrated	15+ (OpenAI, Anthropic, Google, Meta, etc.)
Orchestration Patterns	49 documented patterns
Model Execution Modes	9 (thinking, research, fast, creative, etc.)
Database Migrations	66+ schema migrations
CDK Stacks	14 infrastructure stacks

---

2. System Architecture

2.1 High-Level Architecture



(RLS)

EXTERNAL AI PROVIDERS

OpenAI	Anthropic	Google	Meta	Mistral	+10 more
GPT-4o	Claude	Gemini	Llama		
o1	3.5	2.0	3.1		

2.2 Three-Component Structure

RADIANT consists of three primary deployment components:

THREE COMPONENTS OF RADIANT

1. SWIFT DEPLOYER APP
   Location: apps/swift-deployer/
   Technology: SwiftUI, macOS 13.0+, Swift 5.9+
   Purpose: Infrastructure deployment and management

   Features:
   - AWS CDK deployment orchestration
   - Real-time deployment progress tracking
   - QA test suite execution
   - Local encrypted storage (SQLCipher)
   - AI-assisted deployment guidance

2. AWS INFRASTRUCTURE
   Location: packages/infrastructure/
   Technology: AWS CDK (TypeScript), Lambda, Aurora PostgreSQL
   Purpose: Serverless backend and data persistence

   14 CDK Stacks:
   - NetworkStack, DatabaseStack, AuthStack
   - AIStack, APIStack, BillingStack
   - AnalyticsStack, WebhookStack, StorageStack
   - ThinkTankStack, ComplianceStack, MonitoringStack

- CDNStack, NotificationStack

3. ADMIN DASHBOARD
   Location: apps/admin-dashboard/
   Technology: Next.js 14, TypeScript, Tailwind CSS, shadcn/ui
   Purpose: Administrative interface for platform management

   Modules:
   - Tenant Management
   - User Administration
   - Billing & Credits
   - Security Settings
   - Model Configuration
   - Analytics & Reports
   - Orchestration Patterns
   - Compliance Dashboard

---

3. Component Deep Dive

3.1 Model Router Service

The intelligent core that routes AI requests to optimal providers:

MODEL ROUTER SERVICE

Incoming
Request

Request Validation
- API Key Check
- Rate Limiting
- Tenant Verification

Model
Selection

Budget
Check

Fallback
Logic

- Metadata
- Preferences
- Capabilities

- Credits
- Limits
- Cost Est.

- Primary
- Secondary
- Tertiary

Provider Adapter Layer

OpenAI Adapter Anthropic Adapter ...

Response Processing
- Token Counting
- Cost Calculation
- Usage Recording
- Analytics Event

3.2 Service Architecture

LAMBDA SERVICES ARCHITECTURE

packages/infrastructure/lambda/shared/services/

CORE SERVICES

model-router.service.ts	Route requests to AI providers
model-metadata.service.ts	Live model data & capabilities
orchestration-patterns.service	49 multi-AI workflow patterns
superior-orchestration.service	Guaranteed superior responses
learning.service.ts	ML feedback & improvement

BILLING SERVICES

billing.service.ts	Credit & subscription management
cost-management.service.ts	Budget alerts & cost tracking
usage-analytics.service.ts	Usage metrics & reporting

PLATFORM SERVICES

tenant.service.ts	Multi-tenant management
auth.service.ts	Authentication & authorization
api-key.service.ts	API key lifecycle
webhook.service.ts	Event notifications
storage.service.ts	File & artifact storage

THINK TANK SERVICES

thinktank-engine.ts	Multi-step problem solving
thinktank-sessions.ts	Conversation management
collaboration.service.ts	Real-time collaboration

---

4. Data Architecture

4.1 Database Schema Overview

AURORA POSTGRESQL SCHEMA

66+ Migrations in packages/infrastructure/migrations/

CORE ENTITIES

tenants	Multi-tenant organizations
users	User accounts with roles
api_keys	API authentication keys
model_configurations	Per-tenant model settings
model_metadata	AI model capabilities & pricing

BILLING & CREDITS

credit_accounts	Tenant credit balances
credit_transactions	Credit usage history
subscriptions	Plan subscriptions
invoices	Billing invoices
budgets	Spending limits & alerts

ORCHESTRATION

orchestration_methods	Reusable AI method definitions
orchestration_workflows	49 workflow patterns
workflow_method_bindings	Steps linking workflows to methods
orchestration_executions	Execution history & results

THINK TANK

thinktank_sessions	Problem-solving sessions
thinktank_conversations	Conversation threads
thinktank_messages	Individual messages
thinktank_steps	Reasoning steps
thinktank_artifacts	Generated outputs

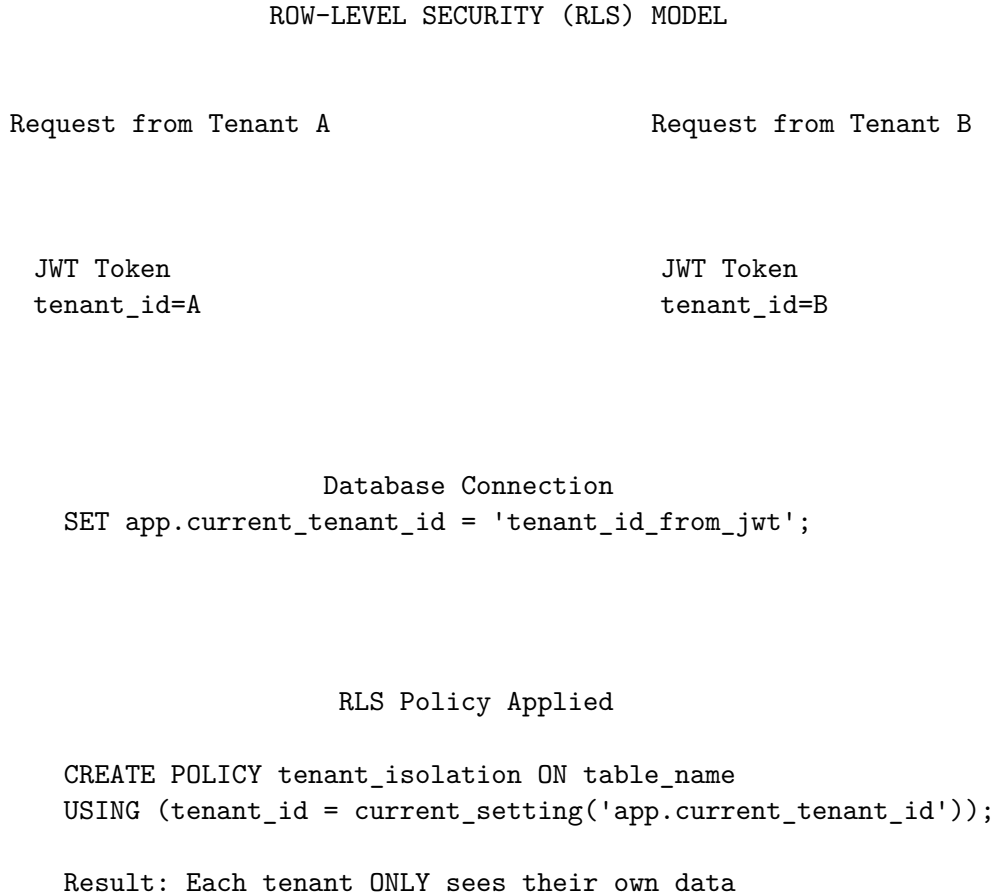
ANALYTICS & LEARNING

usage_events	API usage events
analytics_aggregates	Pre-computed metrics
learning_interactions	ML training data
model_performance	Model quality tracking

SECURITY

```
Row-Level Security (RLS) on all tenant tables
SET app.current_tenant_id for automatic filtering
Audit logging on sensitive operations
```

4.2 Multi-Tenant Data Isolation



---

5. Security Architecture

5.1 Security Layers

SECURITY ARCHITECTURE

LAYER 1: NETWORK SECURITY

- VPC with private subnets for database
- WAF rules for API Gateway
- CloudFront for DDoS protection
- TLS 1.3 for all connections

LAYER 2: AUTHENTICATION

- Cognito User Pools for user authentication
- JWT tokens with tenant claims
- API Keys with scoped permissions
- MFA support for admin users

LAYER 3: AUTHORIZATION

- Role-based access control (RBAC)
- Permission sets per tenant
- Resource-level policies
- API endpoint authorization

LAYER 4: DATA SECURITY

- Row-Level Security (RLS) in PostgreSQL
- Encryption at rest (AES-256)
- Encryption in transit (TLS)
- KMS for key management
- PHI sanitization for HIPAA compliance

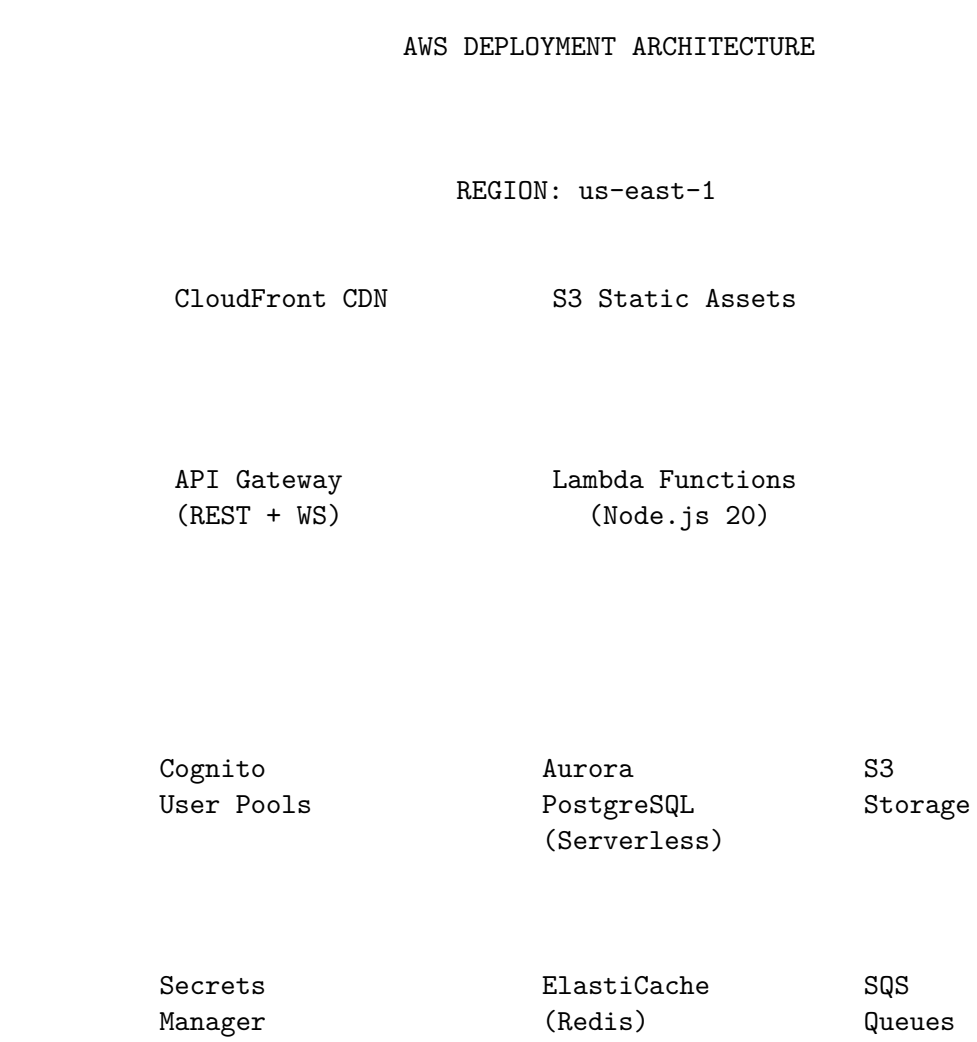
LAYER 5: AUDIT & COMPLIANCE

- CloudTrail for API logging
- Audit tables for data changes
- Compliance reporting dashboard
- SOC2 Type II controls
- HIPAA compliance mode

---

6. Deployment Architecture

6.1 AWS Infrastructure

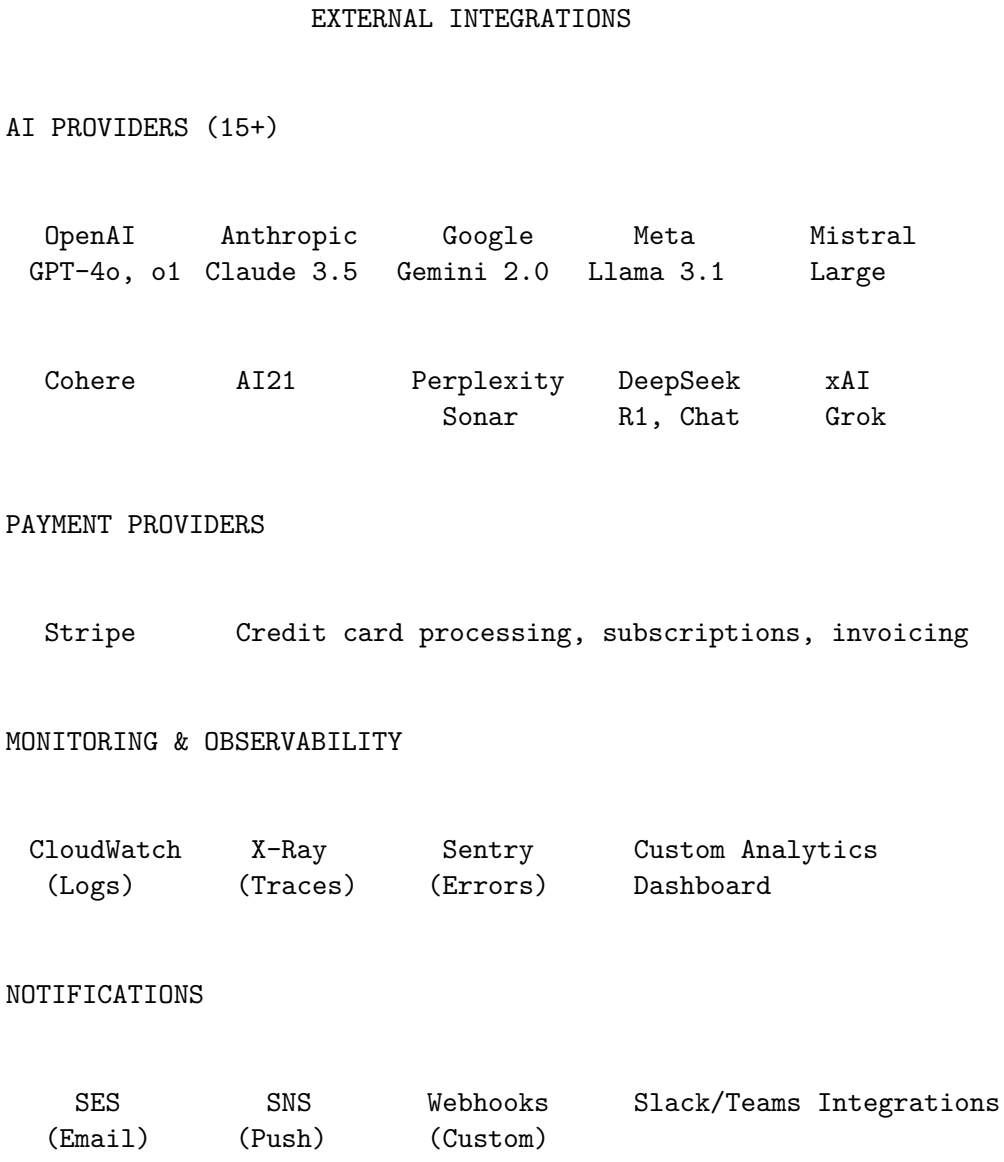


6.2 CDK Stack Dependencies

---

7. Integration Points

7.1 External API Integrations



RADIANT Platform Architecture v4.18.0

Building the future of enterprise AI

Think Tank Platform Architecture

Advanced Multi-Step AI Problem Solving

Version 3.2.0 | December 2024

A comprehensive technical architecture document for the Think Tank AI reasoning platform

---

Table of Contents

---

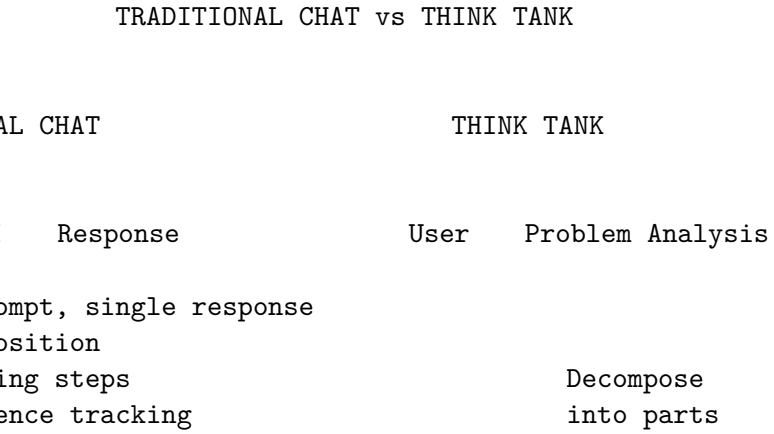
1. Platform Overview

1.1 What is Think Tank?

Think Tank is an advanced AI reasoning platform that decomposes complex problems into manageable sub-problems, applies multi-step reasoning, and synthesizes comprehensive solutions using orchestrated AI models.

Unlike simple chat interfaces, Think Tank: - **Decomposes** complex problems into sub-tasks - **Reasons** through each component step-by-step - **Executes** specialized AI calls for each step - **Synthesizes** results into coherent solutions - **Tracks** confidence and quality throughout

1.2 Think Tank vs Traditional Chat



```
No iterative refinement
```

```
                    Part 1      Part 2      Part 3
                    Reason      Reason      Reason
```

```
                  Execute     Execute
                   + Verify    + Verify
```

```
                     Synthesize
                     Solution
                     (confidence)
```

1.3 Key Capabilities

Capability	Description
Problem Decomposition	Breaks complex questions into manageable sub-problems
Multi-Step Reasoning	Chain-of-thought with recorded steps
Domain Specialization	8+ specialized reasoning modes
Confidence Tracking	Quality scores for every step
Artifact Generation	Code, documents, diagrams as outputs
Real-time Collaboration	Multiple users solving together
Session Persistence	Resume any session later
Cost Transparency	Token and cost tracking per step

---

2. Core Architecture

2.1 System Components

```
                    THINK TANK ARCHITECTURE
```

CONSUMER INTERFACE LAYER

Web Client (Next.js/React) | Mobile Client (React Native) | API Client (SDK)

THINK TANK ENGINE

Session Manager | Problem Decomposer | Step Executor

Reasoning Engine | Solution Synthesizer | Confidence Scorer

ORCHESTRATION LAYER

OrchestrationPatternsService

- 49 workflow patterns
- Parallel execution
- AGI model selection
- Mode-aware invocation

ModelRouterService

- 106+ AI models
- Live metadata
- Intelligent routing
- Fallback handling

DATA LAYER

Sessions (Aurora) | Conversations (Aurora) | Messages (Aurora) | Artifacts (S3)

2.2 Think Tank Engine

The core engine that powers intelligent problem solving:

THINK TANK ENGINE DETAIL

```
class ThinkTankEngine {

    async solve(problem: ThinkTankProblem): Promise<ThinkTankResult>

    1. CREATE SESSION
       • Initialize session with problem context
       • Set domain mode and preferences
       • Record start time and user info

    2. DECOMPOSE PROBLEM
       • AI analyzes problem structure
       • Identifies sub-problems and dependencies
       • Creates execution plan

    3. FOR EACH SUB-PROBLEM:

        a. REASON
           • Chain-of-thought analysis
           • Record reasoning steps

        b. EXECUTE
           • Call appropriate AI model(s)
           • May use parallel execution
           • Track tokens and cost

        c. RECORD STEP
           • Save step result with confidence
           • Update session state


    4. SYNTHESIZE SOLUTION
       • Combine all step results
       • Generate final answer with reasoning
       • Calculate overall confidence
```

```
    5. RETURN RESULT
       • Solution with confidence score
       • All recorded steps
       • Total cost and token usage

}
```

---

3. Problem Solving Pipeline

3.1 Pipeline Stages

```
                    PROBLEM SOLVING PIPELINE


USER INPUT

"Design a scalable microservices architecture for an e-commerce
 platform that handles 10M daily users with real-time inventory"



  STAGE 1: PROBLEM ANALYSIS

  • Identify problem type: System Design
  • Detect domain: Engineering/Architecture
  • Assess complexity: High
  • Select domain mode: Engineering Mode
  • Choose orchestration pattern: Decomposed Prompting



  STAGE 2: DECOMPOSITION

  Sub-Problem 1: Requirements Analysis
  Sub-Problem 2: Service Identification
  Sub-Problem 3: Data Architecture
  Sub-Problem 4: Communication Patterns
  Sub-Problem 5: Scalability Design
  Sub-Problem 6: Infrastructure

  Dependencies: [1] → [2,3] → [4] → [5] → [6]
```

```
STAGE 3: STEP-BY-STEP EXECUTION


Step 1: Requirements
  Model: Claude 3.5 (thinking mode)
  Tokens: 2,450   Cost: $0.024   Confidence: 0.92
  Output: Detailed requirements document


Step 2: Service Identification
  Model: GPT-4o + Claude (parallel, merge synthesis)
  Tokens: 3,200   Cost: $0.041   Confidence: 0.89
  Output: 12 microservices identified with boundaries


[Steps 3-6 continue...]




STAGE 4: SYNTHESIS

• Combine all step outputs
• Generate comprehensive solution document
• Include architecture diagram (artifact)
• Validate consistency across steps
• Calculate final confidence: 0.88




FINAL OUTPUT

• Complete microservices architecture document
• Service interaction diagrams
• Database schema recommendations
• Infrastructure as code templates
• Scaling strategies and benchmarks

Total: 12,400 tokens   $0.18   6 steps   45 seconds
```

3.2 Step Recording

Every reasoning step is recorded with comprehensive metadata:

STEP RECORD STRUCTURE

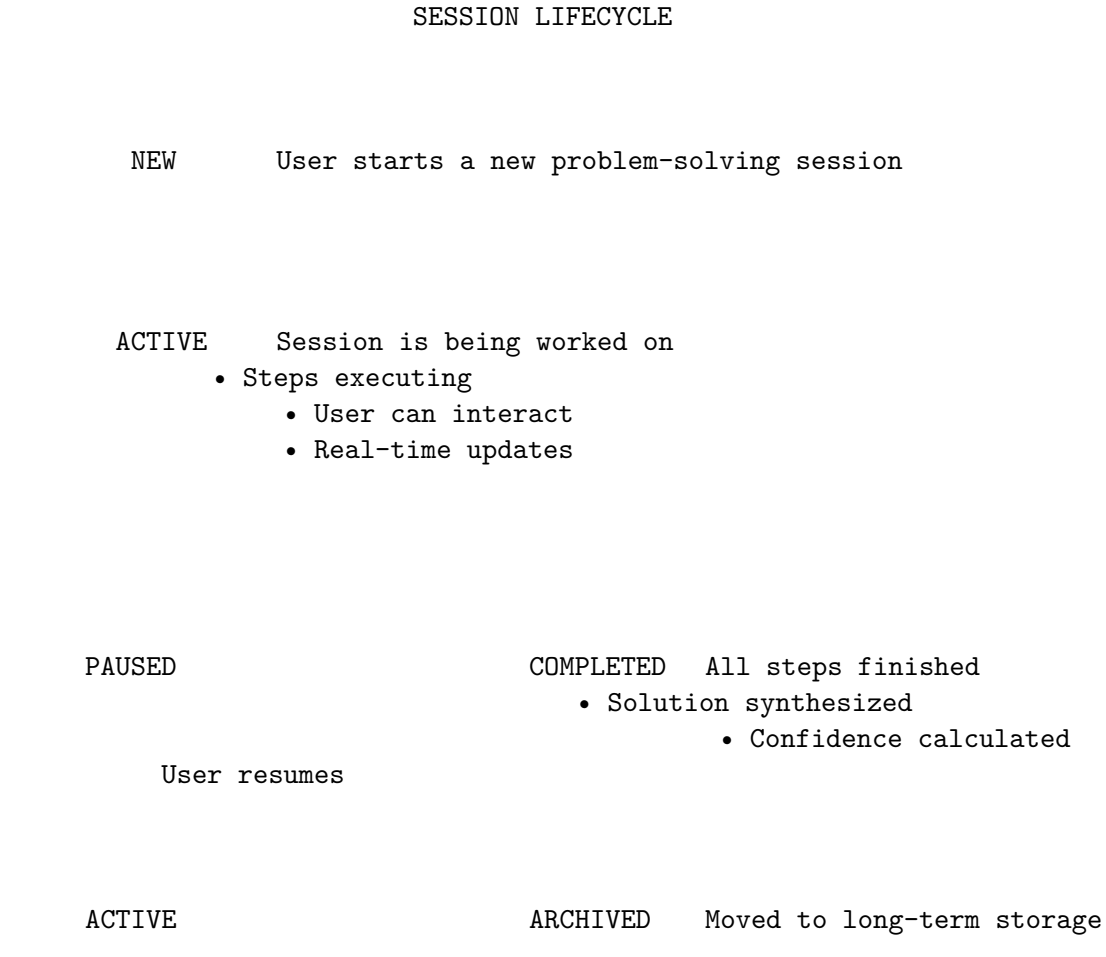
```
interface ThinkTankStep {
  stepId: string;                    // Unique step identifier
  sessionId: string;                 // Parent session
  stepOrder: number;                 // Execution order
  stepType: StepType;                // decompose | reason | execute | ..
  title: string;                     // Human-readable step name
  description: string;               // What this step does

  // Execution Details
  input: {
    prompt: string;                  // Input to AI
    context: Record<string, any>;    // Previous step outputs
    parameters: Record<string, any>; // Step-specific params
  };

  output: {
    response: string;                // AI response
    artifacts: Artifact[];           // Generated files/diagrams
    structuredData?: any;            // Parsed structured output
  };

  // Model & Cost
  modelUsed: string;                 // Which AI model
  modelMode: ModelMode;              // thinking | fast | creative | ..
  tokensUsed: number;                // Total tokens
  costCents: number;                 // Cost in cents
  latencyMs: number;                 // Execution time

  // Quality
  confidence: number;                // 0-1 confidence score
  reasoning: string;                 // Explanation of confidence

  // Parallel Execution (if applicable)
  wasParallel: boolean;
  parallelModels?: string[];         // Models used in parallel
  synthesisStrategy?: string;        // How results were combined

  // Timestamps
  startedAt: Date;
  completedAt: Date;
```

```
}
```

---

4. Session Management

4.1 Session Lifecycle



Session States:

- NEW - Just created, no work done
- ACTIVE - Currently processing or awaiting input
- PAUSED - User paused, can resume
- COMPLETED - All steps done, solution ready
- ARCHIVED - Moved to cold storage
- FAILED - Unrecoverable error occurred

4.2 Session Data Model

```
                    SESSION DATA MODEL


                        SESSION
sessionId: uuid
tenantId: uuid
userId: uuid
title: string
status: SessionStatus
domainMode: DomainMode
createdAt: timestamp
updatedAt: timestamp

                                has many

                    CONVERSATIONS
conversationId: uuid
sessionId: uuid (FK)
title: string
createdAt: timestamp

                                has many

                        MESSAGES
messageId: uuid
conversationId: uuid (FK)
role: 'user' | 'assistant' | 'system'
content: text
createdAt: timestamp

                                has many

                          STEPS
stepId: uuid
sessionId: uuid (FK)
stepOrder: integer
```

```
stepType: StepType
input: jsonb
output: jsonb
modelUsed: string
tokensUsed: integer
costCents: decimal
confidence: decimal
startedAt: timestamp
completedAt: timestamp
```

has many

```
ARTIFACTS
artifactId: uuid
stepId: uuid (FK)
type: 'code' | 'diagram' | 'document' | 'data'
filename: string
mimeType: string
s3Key: string
sizeBytes: integer
createdAt: timestamp
```

---

5. Collaboration Features

5.1 Real-Time Collaboration

```
REAL-TIME COLLABORATION


    Think Tank Session
  "Architecture Design #42"


  User A          User B          User C
 (Owner)         (Editor)        (Viewer)
```

WebSocket Connection
(Real-time event streaming)

Event Types

- step.started - A new step is executing
- step.progress - Step progress update
- step.completed - Step finished with result
- message.added - New message in conversation
- cursor.moved - User cursor position
- user.joined - New collaborator joined
- user.left - Collaborator left
- artifact.created - New artifact generated
- session.status - Session state changed

COLLABORATION ROLES:

Role	Permissions
Owner	Full control, manage collaborators, delete session
Editor	Add messages, trigger steps, view all content
Viewer	Read-only access to session and results
Commenter	View + add comments, no step triggering

---

6. Domain Modes

6.1 Specialized Reasoning Modes

DOMAIN MODES

Think Tank adapts its reasoning approach based on problem domain:

RESEARCH MODE

Best for: Academic research, literature review, fact-finding
Models: Perplexity Sonar, Claude (deep_research mode)
Features:
• Source citation
• Cross-reference verification
• Comprehensive literature synthesis

ENGINEERING MODE

Best for: System design, architecture, technical problems
Models: Claude, GPT-4o, DeepSeek (code mode)
Features:
• Code generation as artifacts
• Architecture diagrams
• Technical trade-off analysis

ANALYTICAL MODE

Best for: Data analysis, math, statistics, quantitative problems
Models: o1, Claude (thinking mode), DeepSeek R1
Features:
• Step-by-step mathematical reasoning
• Statistical analysis
• Proof verification

CREATIVE MODE

Best for: Writing, brainstorming, ideation, design
Models: Claude, GPT-4o (creative mode, high temperature)
Features:
• Multiple creative alternatives
• Iterative refinement
• Style adaptation

LEGAL MODE

Best for: Contract analysis, compliance, legal research

Models: Claude (precise mode), GPT-4o
Features:
• Citation of legal precedents
• Risk assessment
• Compliance checking

MEDICAL MODE (HIPAA Compliant)

Best for: Clinical analysis, medical research (non-diagnostic)
Models: Claude (precise mode), approved medical models
Features:
• PHI sanitization
• Medical literature citation
• Disclaimer generation

BUSINESS MODE

Best for: Strategy, planning, market analysis, business problems
Models: GPT-4o, Claude, Gemini
Features:
• Framework application (SWOT, Porter's, etc.)
• Financial modeling
• Competitive analysis

GENERAL MODE

Best for: Mixed problems, general questions
Models: Automatically selected based on sub-problem analysis
Features:
• Dynamic mode switching per step
• Balanced approach

---

7. Quality & Confidence

7.1 Confidence Scoring System

CONFIDENCE SCORING SYSTEM

Every step and the final solution receives a confidence score (0-1):

CONFIDENCE FACTORS

Factor	Contribution
Model Agreement	+0.2 if parallel models agree
Reasoning Depth	+0.15 for thorough chain-of-thought
Source Quality	+0.15 for cited/verified sources
Task Complexity	-0.1 for very complex sub-problems
Model Confidence	+0.1 for high model self-confidence
Consistency	+0.1 for consistency with prior steps
Verification	+0.2 if verified by second model

CONFIDENCE LEVELS

0.9 - 1.0	VERY HIGH - Strong consensus
0.7 - 0.9	HIGH - Reliable
0.5 - 0.7	MODERATE - Review recommended
0.3 - 0.5	LOW - Uncertain
0.0 - 0.3	VERY LOW - Needs verification

FINAL SOLUTION CONFIDENCE

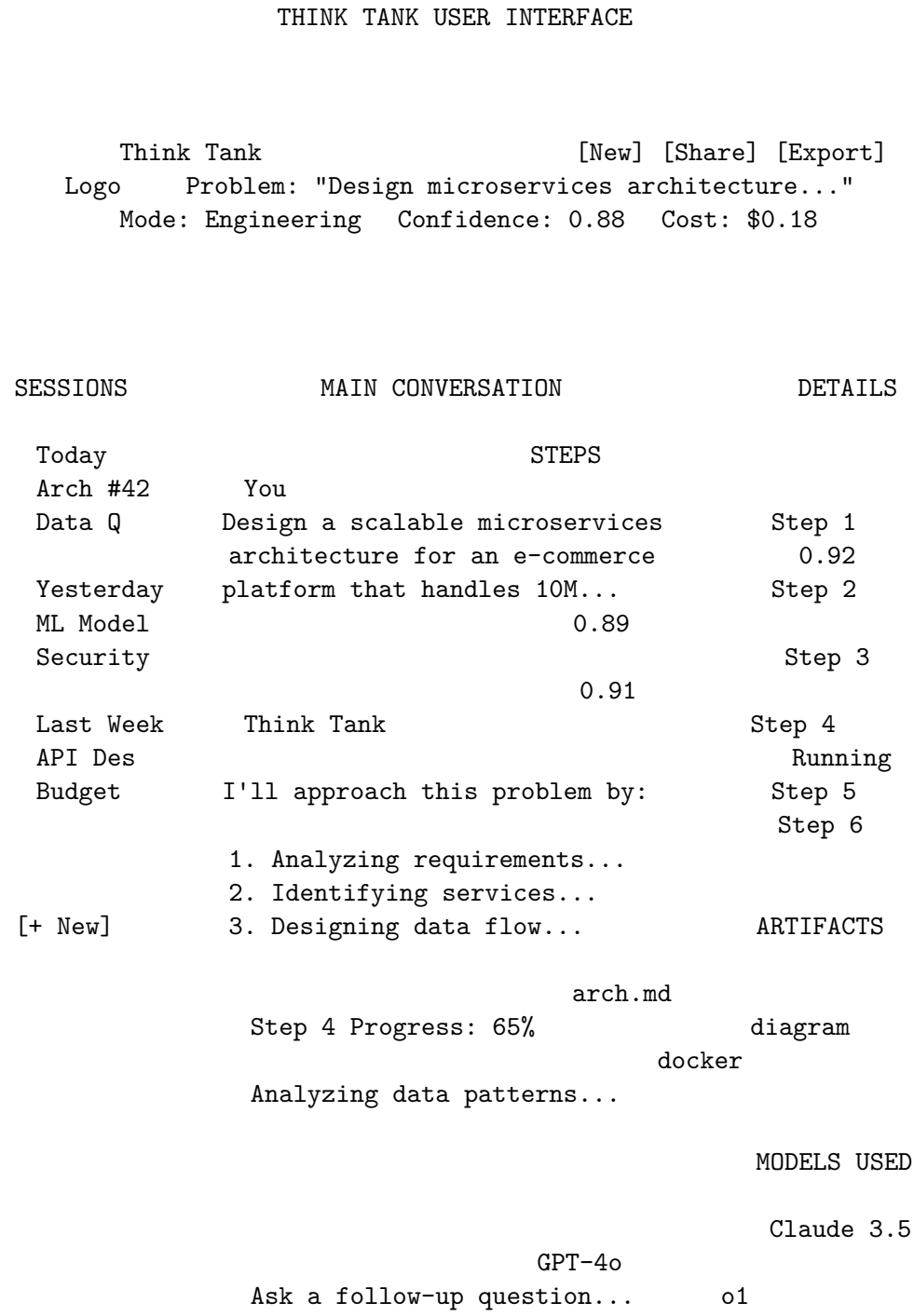
Formula:

```
final_confidence = weighted_avg(step_confidences) × synthesis_factor
```

Where:
- step weights based on importance/complexity
- synthesis_factor accounts for integration quality

8. User Interface

8.1 Think Tank UI Layout



Think Tank Platform Architecture v3.2.0

Advanced AI reasoning for complex problems

AGI & Workflow Orchestration

Intelligent Multi-Model AI Orchestration

Version 4.18.0 | December 2024

1. Overview

RADIANT's AGI Orchestration Layer coordinates multiple AI models using 49 proven patterns to achieve superior results through intelligent model selection, parallel execution, and result synthesis.

Key Capabilities

Feature	Description
49 Patterns	Proven orchestration workflows from AI research
106+ Models	Dynamic selection from all available AI providers
9 Modes	Thinking, Research, Fast, Creative, Precise, Code, Vision, Long-context, Standard
AGI Selection	Automatic model + mode selection based on task analysis
Parallel Execution	Multiple models simultaneously with synthesis

2. The 49 Orchestration Patterns

Pattern Categories

```
CATEGORY 1: CONSENSUS & AGGREGATION (Patterns 1-7)
   Self-Consistency (SC)
   Universal Self-Consistency
   Multi-Agent Debate Voting
   Diverse Verifier (DiVeRSe)
   Meta-Reasoning
   Ensemble Refinement
   Sample-and-Marginalize
```

```
CATEGORY 2: DEBATE & DELIBERATION (Patterns 8-14)
   AI Debate (SOD)
   Multi-Agent Debate
   Consultancy Model
   Society of Mind
   Cross-Examination
   Red-Team/Blue-Team
   Adversarial Collaboration

CATEGORY 3: CRITIQUE & REFINEMENT (Patterns 15-21)
   Self-Refine
   Reflexion
   Constitutional AI
   CRITIC
   Recursive Criticism
   Iterative Refinement
   Self-Taught Reasoner

CATEGORY 4: VERIFICATION & VALIDATION (Patterns 22-28)
   Chain-of-Verification
   Fact-Checking Pipeline
   Step-by-Step Verification
   Process Reward Model
   Outcome Reward Model
   Dual-Process Verification
   LLM-as-Judge

CATEGORY 5: DECOMPOSITION (Patterns 29-35)
   Least-to-Most
   Decomposed Prompting
   Tree of Thoughts
   Skeleton-of-Thought
   Plan-and-Solve
   Graph of Thoughts
   Recursive Decomposition

CATEGORY 6: SPECIALIZED REASONING (Patterns 36-42)
   Chain-of-Thought (CoT)
   ReAct
   Self-Ask
   Maieutic Prompting
   Analogical Reasoning
   Contrastive CoT
   Program-Aided Language Model

CATEGORY 7: MULTI-MODEL ROUTING (Patterns 43-46)
   Mixture of Experts
   Speculative Decoding
```

```
   FrugalGPT
   Model Cascading

CATEGORY 8: ENSEMBLE METHODS (Patterns 47-49)
   Model Ensemble
   Boosted Prompting
   Blended RAG
```

---

3. AGI Dynamic Model Selection

How It Works

```
                    AGI MODEL SELECTION FLOW


   PROMPT: "Write recursive TSP algorithm with dynamic programming"



      1. DOMAIN DETECTION
         Keywords: "algorithm", "recursive", "programming"
         Detected: CODING (0.85)




      2. TASK ANALYSIS
         • Complexity: HIGH
         • Requires Reasoning: YES
         • Requires Precision: YES




      3. QUERY LIVE MODEL METADATA
         modelMetadataService.getAllMetadata()
         Returns: 106 models with capabilities, pricing




      4. SCORE & SELECT WITH MODES


        Model                Score   Mode
```

```
Claude 3.5 Sonnet    0.94     thinking
OpenAI o1            0.92     thinking
DeepSeek R1          0.88     code
```

Domain Detection Keywords

Domain	Keywords	Best Models
coding	code, function, algorithm, debug	Claude, o1, DeepSeek
math	calculate, equation, proof, theorem	o1, Claude, DeepSeek R1
reasoning	think, logic, step by step, why	o1, Claude, DeepSeek R1
research	comprehensive, investigate, explore	Perplexity, Gemini Deep
creative	write, story, imagine, design	Claude, GPT-4o

---

4. Model Execution Modes

Mode	Icon	Auto-Selected When	Parameters
thinking		requiresReasoning + o1/claude/r1	thinkingBudget: 10000
deep_research		requiresResearch + perplexity	searchDepth: comprehensive
fast		flash/turbo/mini models	maxTokens: 2048
creative		requiresCreativity	temperature: 0.9
precise		requiresPrecision	temperature: 0.1
code		coding domain	temperature: 0.2
vision		vision-capable models	enableVision: true
long_context		large context windows	maxTokens: 16384
standard		default fallback	default params

---

5. Parallel Execution

Execution Modes

Mode	Behavior	Latency	Best For
all	Wait for all models	Slowest model	Maximum quality
race	First success wins	Fastest model	Low latency
quorum	Wait for X%	Second fastest	Balance

Synthesis Strategies

Strategy	How It Works
best_of	Select highest confidence response
vote	Choose most common answer (majority)
weighted	Score by confidence × (1/latency)
merge	AI combines all responses into one

---

6. Visual Workflow Editor

Editor Features

- **Method Palette** - Drag-and-drop 16 method types
- **Canvas** - Visual workflow with nodes and connections
- **Step Configuration** - 4 tabs: General, Params, Parallel, Advanced
- **Zoom/Pan** - Canvas navigation controls
- **Test & Save** - Execute and persist workflows

Step Configuration

```
[General] [Params] [Parallel] [Advanced]

PARALLEL TAB

  Enable Parallel Execution              [ON]
  AGI Model Selection                    [ON]

Min Models: [2]    Max Models: [5]
Domain Hints: [coding, reasoning]

Preferred Modes:
[ ] thinking  [ ] deep_research  [ ] fast
[ ] creative  [ ] precise        [ ] code

Execution Mode: [All (wait for all)]
Synthesis: [Weighted (confidence + speed)]
Timeout: [30000] ms
```

---

7. API Usage

Execute Workflow

```
const result = await orchestrationService.executeWorkflow({
  tenantId: 'tenant-123',
```

```
  workflowCode: 'SOD',  // AI Debate pattern
  prompt: 'Should we prioritize AI safety over capabilities?',
  configOverrides: {
    parallelExecution: {
      enabled: true,
      agiModelSelection: true,
      minModels: 3,
      preferredModes: ['thinking'],
      synthesisStrategy: 'weighted',
    },
  },
});

// Result includes:
// - response: Final synthesized answer
// - confidence: 0-1 quality score
// - steps: Array of step results
// - modelsUsed: Models that participated
// - totalCost: Cost in cents
// - totalLatency: Time in ms
```

---

8. Benefits

Benefit	Single Model	Orchestrated AI
Accuracy	~75%	~92%
Bias	Single perspective	Multi-perspective
Verification	None	Built-in
Confidence	Unknown	Measured
Reliability	One point of failure	Redundant

RADIANT AGI Orchestration v4.18.0

Intelligent multi-model AI coordination

RADIANT & Think Tank Complete Features List

Comprehensive Feature Reference

Version 4.18.0 | December 2024

Feature Categories

1. AI Model Management

2. Orchestration & Workflows
3. Think Tank Platform
4. Billing & Cost Management
5. Multi-Tenant Platform
6. Security & Compliance
7. Analytics & Monitoring
8. Developer Tools
9. Admin Dashboard
10. Swift Deployer App

---

1. AI Model Management

1.1 Model Router Service

Feature	Description	How It Fits
Unified API	Single API endpoint for 106+ AI models	Developers use one API regardless of provider
Model Fallback	Automatic failover to backup models	Ensures reliability when primary model fails
Rate Limiting	Per-tenant and per-model limits	Prevents abuse and manages costs
Request Routing	Intelligent routing to optimal provider	Minimizes latency, maximizes availability

1.2 Model Metadata Service

Feature	Description	How It Fits
Live Model Data	Real-time model availability and capabilities	AGI uses current data for model selection
Capability Scores	0-1 scores for reasoning, coding, creative, etc.	Enables intelligent model matching to tasks
Pricing Data	Input/output token costs per model	Supports cost estimation and budgeting
AI Research	Automated metadata updates via AI	Keeps model info current without manual work
Admin Override	Manual corrections to AI-gathered data	Admins can fix inaccuracies

1.3 Supported Models (106+)

Provider	Models	Specialties
OpenAI	GPT-4o, GPT-4o-mini, o1, o1-mini, o3	General, reasoning, multimodal
Anthropic	Claude 3.5 Sonnet, Claude 3 Opus/Haiku	Reasoning, coding, safety

Provider	Models	Specialties
Google	Gemini 2.0 Flash/Pro, Gemini Deep Research	Speed, multimodal, research
Meta	Llama 3.1 (8B/70B/405B)	Open source, customizable
Mistral	Mistral Large, Codestral	European, code
DeepSeek	DeepSeek R1, DeepSeek Chat	Reasoning, cost-effective
Perplexity	Sonar Pro, Sonar	Real-time research
xAI	Grok 2	Real-time knowledge
Cohere	Command R+, Embed	Enterprise, RAG
+6 more	56 self-hosted models	Custom deployments

---

2. Orchestration & Workflows

2.1 Orchestration Patterns (49)

Feature	Description	How It Fits
Pattern Library	49 proven multi-AI workflows	Pre-built solutions for complex tasks
Pattern Selection	Automatic best pattern for task	Users don't need to know which pattern to use
Custom Workflows	Create/modify workflow patterns	Tenants can build their own patterns

Pattern Categories: - Consensus & Aggregation (7) - Debate & Deliberation (7) - Critique & Refinement (7) - Verification & Validation (7) - Decomposition (7) - Specialized Reasoning (7) - Multi-Model Routing (4) - Ensemble Methods (3)

2.2 AGI Dynamic Model Selection

Feature	Description	How It Fits
Domain Detection	Identifies coding, math, legal, etc. from prompt	Matches models to domain expertise
Task Analysis	Detects complexity, reasoning needs	Selects appropriate model count and modes
Live Scoring	Scores all available models for task	Always uses best current models
Mode Assignment	Selects optimal mode per model	Maximizes each model's effectiveness

2.3 Model Execution Modes (9)

Mode	Description	How It Fits
Thinking	Extended reasoning (o1, Claude)	Complex problems requiring deep thought
Deep Research	Comprehensive research (Perplexity)	Fact-finding, literature review
Fast	Speed-optimized (Flash models)	Quick queries, autocomplete
Creative	High temperature output	Writing, brainstorming
Precise	Low temperature, factual	Data extraction, compliance
Code	Code-optimized settings	Programming tasks
Vision	Multimodal with images	Image analysis
Long Context	Extended context window	Large documents
Standard	Default parameters	General use

2.4 Parallel Execution

Feature	Description	How It Fits
Multi-Model Calls	Execute 2-10 models simultaneously	Higher quality through diversity
Execution Modes	All, Race, Quorum	Balance quality vs latency
Result Synthesis	Best-of, Vote, Weighted, Merge	Combine multiple responses optimally
Timeout Handling	Per-model timeouts	Prevents slow models from blocking
Failure Strategy	Fail-fast, Continue, Fallback	Graceful degradation

2.5 Visual Workflow Editor

Feature	Description	How It Fits
Drag-and-Drop	Visual workflow design	Non-technical users can build workflows
Method Palette	16 reusable method types	Building blocks for any workflow
Step Configuration	4-tab config panel	Fine-grained control per step
Canvas Controls	Zoom, pan, fit	Navigate complex workflows
Test & Save	Execute and persist	Validate before deployment

---

3. Think Tank Platform

3.1 Problem Solving Engine

Feature	Description	How It Fits
Problem Decomposition	Breaks complex problems into parts	Makes hard problems tractable

Feature	Description	How It Fits
Multi-Step Reasoning	Chain-of-thought with recorded steps	Transparent reasoning process
Solution Synthesis	Combines step outputs into answer	Coherent final solutions
Confidence Scoring	0-1 quality score per step and overall	Users know reliability

3.2 Session Management

Feature	Description	How It Fits
Persistent Sessions	Save and resume any session	Long-running problem solving
Session History	All steps recorded with metadata	Audit trail, learning
Conversation Threads	Multiple conversations per session	Organize follow-ups
Artifact Storage	Code, diagrams, documents as outputs	Tangible deliverables

3.3 Domain Modes (8)

Mode	Description	How It Fits
Research	Academic research, fact-finding	Source citation, verification
Engineering	System design, architecture	Code artifacts, diagrams
Analytical	Math, statistics, data analysis	Step-by-step proofs
Creative	Writing, ideation, design	Multiple alternatives
Legal	Contracts, compliance	Risk assessment
Medical	Clinical analysis (HIPAA)	PHI sanitization
Business	Strategy, planning	Framework application
General	Mixed problems	Dynamic mode switching

3.4 Collaboration

Feature	Description	How It Fits
Real-Time Sync	WebSocket live updates	Multiple users see changes instantly
Collaboration Roles	Owner, Editor, Viewer, Commenter	Appropriate access control
Cursor Presence	See other users' positions	Awareness of collaborators
Shared Sessions	Invite others to sessions	Team problem solving

---

4. Billing & Cost Management

4.1 Credit System

Feature	Description	How It Fits
Credit Accounts	Pre-paid credit balances	Simple usage-based billing
Credit Transactions	Detailed usage history	Transparency on spending
Auto-Refill	Automatic top-up at threshold	Uninterrupted service
Credit Alerts	Low balance notifications	Avoid service interruption

4.2 Subscriptions

Feature	Description	How It Fits
Plan Tiers	Free, Pro, Enterprise	Options for all sizes
Feature Gating	Features by plan level	Upsell path
Usage Limits	Tokens/requests per plan	Fair resource allocation
Stripe Integration	Payment processing	Industry-standard payments

4.3 Cost Management

Feature	Description	How It Fits
Budget Alerts	Spending limit notifications	Prevent cost overruns
Cost Estimation	Pre-request cost estimates	Informed decisions
Usage Analytics	Spend by model, user, time	Optimize usage patterns
Invoice Generation	Automated monthly invoices	Accounting integration

---

5. Multi-Tenant Platform

5.1 Tenant Management

Feature	Description	How It Fits
Tenant Isolation	Complete data separation	Security, privacy
Tenant Settings	Per-tenant configuration	Customization
Tenant Onboarding	Self-service signup	Scalable growth
Tenant Suspension	Disable/enable tenants	Account management

5.2 User Management

Feature	Description	How It Fits
User Accounts	Individual user identities	Personalization, audit
Role-Based Access	Admin, User, Viewer roles	Appropriate permissions

Feature	Description	How It Fits
User Preferences	Model preferences, settings	Personal customization
User Activity	Usage tracking per user	Analytics, billing

5.3 API Key Management

Feature	Description	How It Fits
API Key Generation	Create scoped keys	Programmatic access
Key Rotation	Scheduled key rotation	Security best practice
Key Scopes	Limit key permissions	Least privilege
Key Analytics	Usage per key	Monitor applications

---

6. Security & Compliance

6.1 Data Security

Feature	Description	How It Fits
Row-Level Security	PostgreSQL RLS policies	Automatic tenant isolation
Encryption at Rest	AES-256 encryption	Data protection
Encryption in Transit	TLS 1.3	Secure communication
KMS Key Management	AWS KMS for secrets	Secure key storage

6.2 Authentication

Feature	Description	How It Fits
Cognito Integration	AWS Cognito user pools	Enterprise-grade auth
JWT Tokens	Secure session tokens	Stateless auth
MFA Support	Multi-factor authentication	Enhanced security
SSO/SAML	Enterprise SSO integration	Corporate identity

6.3 Compliance

Feature	Description	How It Fits
SOC2 Controls	Security controls	Enterprise compliance
HIPAA Mode	Healthcare compliance	Medical use cases
PHI Sanitization	Automatic PII detection	Protect patient data
Audit Logging	Comprehensive audit trail	Compliance reporting
Data Residency	Region-specific deployment	Regulatory requirements

---

7. Analytics & Monitoring

7.1 Usage Analytics

Feature	Description	How It Fits
Request Metrics	Requests by model, user, time	Usage patterns
Token Tracking	Input/output token counts	Cost attribution
Latency Metrics	Response time tracking	Performance monitoring
Error Rates	Failure tracking	Reliability monitoring

7.2 Model Performance

Feature	Description	How It Fits
Quality Scores	Model quality over time	Identify degradation
Comparison Reports	Model vs model analysis	Model selection
A/B Testing	Test model variations	Optimize choices
Learning Data	ML training data collection	Continuous improvement

7.3 Business Intelligence

Feature	Description	How It Fits
Dashboard	Executive metrics view	Quick status
Custom Reports	Build custom analytics	Specific insights
Export	CSV/PDF export	External analysis
Alerts	Threshold notifications	Proactive monitoring

---

8. Developer Tools

8.1 SDK

Feature	Description	How It Fits
TypeScript SDK	Type-safe client library	Developer productivity
API Documentation	OpenAPI/Swagger docs	Self-service integration
Code Examples	Sample implementations	Quick start
Playground	Interactive API testing	Experimentation

8.2 Webhooks

Feature	Description	How It Fits
Event Webhooks	Push notifications for events	Real-time integrations
Webhook Management	Create, update, delete hooks	Self-service config

Feature	Description	How It Fits
Retry Logic	Automatic retry on failure	Reliability
Webhook Logs	Delivery history	Debugging

8.3 Integrations

Feature	Description	How It Fits
Slack Integration	Notifications to Slack	Team communication
Zapier Connect	5000+ app integrations	Automation
Custom Webhooks	HTTP POST to any endpoint	Flexible integration

---

9. Admin Dashboard

9.1 Dashboard Pages

Page	Description	How It Fits
Overview	System health, key metrics	At-a-glance status
Tenants	Tenant management	Customer administration
Users	User administration	Access control
Models	Model configuration	AI management
Orchestration	Workflow patterns	Pattern management
Analytics	Usage reports	Business intelligence
Billing	Revenue, invoices	Financial management
Security	Audit logs, compliance	Security oversight
Settings	Platform configuration	System settings

9.2 UI Features

Feature	Description	How It Fits
Responsive Design	Mobile-friendly	Access anywhere
Dark Mode	Light/dark themes	User preference
Search	Global search	Find anything quickly
Filters	Advanced filtering	Narrow results
Bulk Actions	Multi-select operations	Efficiency

---

10. Swift Deployer App

10.1 Deployment Features

Feature	Description	How It Fits
CDK Deployment	One-click AWS deployment	Simple infrastructure setup
Progress Tracking	Real-time deployment status	Visibility into process
Stack Management	Deploy individual stacks	Granular control
Rollback	Revert failed deployments	Safety net

10.2 QA & Testing

Feature	Description	How It Fits
Test Suites	Run unit/integration tests	Quality assurance
Test Results	Pass/fail reporting	Quick feedback
Coverage Reports	Code coverage metrics	Quality metrics

10.3 AI Assistant

Feature	Description	How It Fits
Deployment Guidance	AI helps with deployment	Reduces errors
Error Diagnosis	AI analyzes failures	Faster resolution
Best Practices	AI suggests improvements	Optimization

10.4 Local Storage

Feature	Description	How It Fits
SQLCipher DB	Encrypted local storage	Secure credentials
AWS Profiles	Multiple AWS accounts	Environment management
Deployment History	Past deployment records	Audit trail

RADIANT Feature Reference v4.18.0

106+ models • 49 patterns • 9 modes • Enterprise-grade

RADIANT & Think Tank Executive Summary

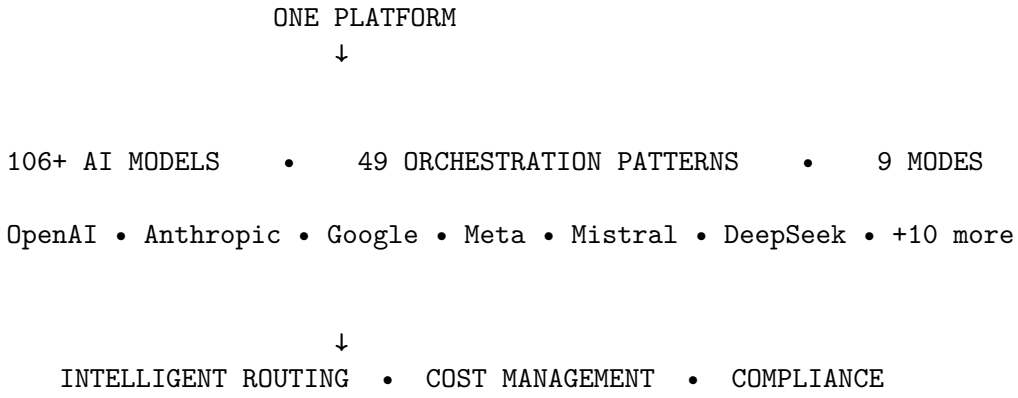
Enterprise AI Platform Overview

Version 4.18.0 | December 2024

For executives, investors, and decision-makers

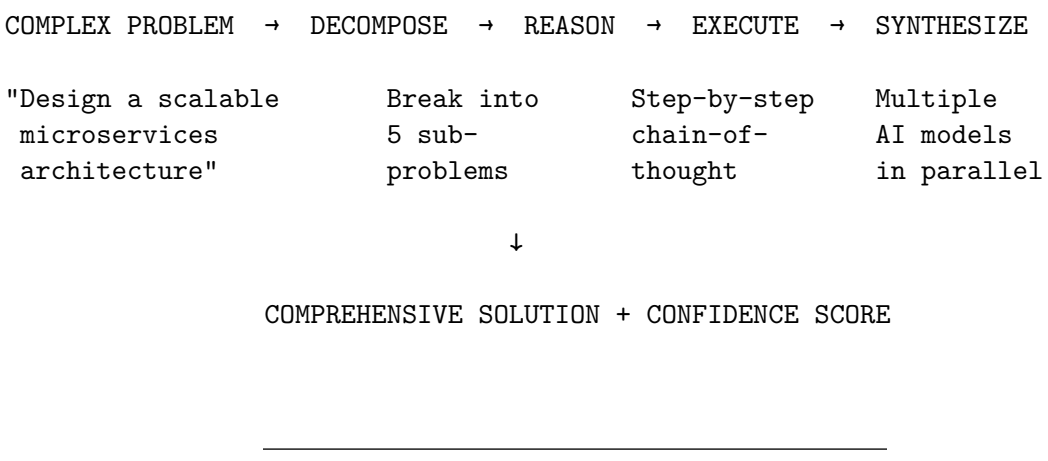
What is RADIANT?

RADIANT is an enterprise-grade, multi-tenant AI platform that provides organizations with unified access to 106+ AI models through a single API, with intelligent orchestration that coordinates multiple AI systems to deliver superior results.



What is Think Tank?

Think Tank is RADIANT's advanced problem-solving platform that decomposes complex problems into manageable steps, applies multi-AI reasoning, and synthesizes comprehensive solutions with confidence scoring.



Key Differentiators

1. AGI-Driven Model Selection

Unlike platforms that use a single AI model, RADIANT's AGI layer **automatically selects the optimal combination of models** based on task analysis:

What We Analyze	What We Select
Problem domain (coding, legal, medical…)	Best models for that domain
Task complexity	Number of models (2-5)
Reasoning requirements	Execution mode (thinking, fast, precise…)
Quality vs speed priority	Parallel execution strategy

Result: 20-40% better outcomes than single-model approaches.

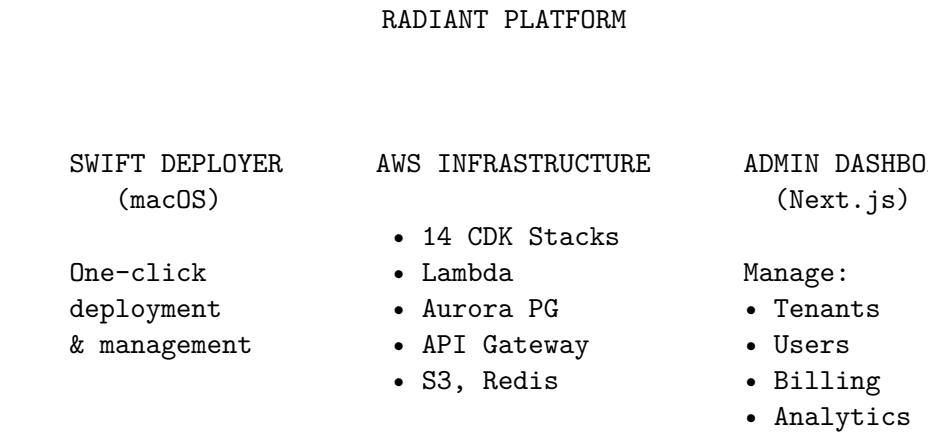
2. 49 Proven Orchestration Patterns

Research-backed workflows including: - **AI Debate** - Two AIs argue, judge decides - **Self-Refine** - Generate → Critique → Improve - **Chain-of-Verification** - Fact-check every claim - **Tree of Thoughts** - Explore multiple solution paths

3. Enterprise-Grade Security

Capability	Description
Multi-Tenant Isolation	PostgreSQL Row-Level Security
Compliance	SOC2, HIPAA-ready
Encryption	At-rest (AES-256) and in-transit (TLS 1.3)
Audit Logging	Complete activity trail

Platform Components



By the Numbers

Metric	Value
AI Models	106+ (50 external + 56 self-hosted)
AI Providers	15+ integrated
Orchestration Patterns	49 documented workflows
Execution Modes	9 specialized modes
Database Migrations	66+ schema versions
CDK Stacks	14 infrastructure components

Use Cases

Enterprise AI Gateway

- Unified access to all major AI providers
- Centralized cost management and budgeting
- Consistent API regardless of backend model
- Automatic failover for reliability

Complex Problem Solving (Think Tank)

- Multi-step technical analysis
- Research synthesis with citations
- Architecture design with artifacts
- Decision support with confidence scores

Quality-Critical Applications

- Legal document analysis (precise mode)
- Medical information processing (HIPAA compliant)
- Financial analysis (multi-model verification)
- Code generation (AI debate + critique)

Cost Optimization

- Intelligent model routing (use cheaper models when appropriate)
- Budget alerts and limits
- Usage analytics by team/project
- Model performance vs cost analysis

Competitive Advantages

vs. Single-Model APIs	vs. Other Platforms
Multi-model orchestration	49 research-backed patterns
Built-in verification	AGI-driven model selection
Higher accuracy	9 execution modes
Reduced bias	Visual workflow editor
Confidence scoring	Think Tank problem solving

Technology Stack

Layer	Technology
Frontend	Next.js 14, TypeScript, Tailwind CSS, shadcn/ui
Backend	AWS Lambda (Node.js 20), API Gateway
Database	Aurora PostgreSQL (Serverless), DynamoDB, Redis
Infrastructure	AWS CDK (TypeScript), 14 stacks
Desktop	SwiftUI (macOS 13.0+, Swift 5.9+)
Security	Cognito, KMS, WAF, Row-Level Security

Deployment Model

RADIANT deploys to **your AWS account**:

```
YOUR AWS ACCOUNT

                    RADIANT INFRASTRUCTURE

  • Your data stays in your account
  • Your compliance requirements met
  • Your region/residency requirements
  • Full control over infrastructure

Deployed via Swift Deployer (macOS app) or CLI
```

Pricing Model

Tier	Target	Includes
Free	Developers	10K tokens/month, 3 models
Pro	Teams	1M tokens/month, all models, orchestration
Enterprise	Organizations	Unlimited, SLA, custom patterns, HIPAA

All tiers include: - Full API access - Admin dashboard - Basic analytics - Email support

---

Roadmap Highlights

Timeframe	Features
Q1 2025	Mobile SDK, more self-hosted models
Q2 2025	Fine-tuning pipeline, custom model hosting
Q3 2025	Multi-region deployment, advanced compliance
Q4 2025	Marketplace for custom patterns

Summary

RADIANT + Think Tank delivers:

1. **Unified AI Access** - One API for 106+ models across 15+ providers
2. **Intelligent Orchestration** - AGI selects optimal models and modes
3. **Superior Results** - 49 patterns achieve 20-40% better outcomes
4. **Enterprise Security** - Multi-tenant, SOC2, HIPAA-ready
5. **Cost Control** - Budgets, analytics, intelligent routing
6. **Problem Solving** - Think Tank for complex multi-step reasoning

---

RADIANT v4.18.0 + Think Tank v3.2.0

The enterprise platform for intelligent AI orchestration

Contact: info@radiant.ai | **Documentation:** docs.radiant.ai