Contents

SECTION 7: EXTERNAL PROVIDERS & DATABASE SCHEMA (v2.3.0 - CANONICAL)

Dependencies: Sections 0-6 **Creates:** 21 provider integrations, Complete PostgreSQL schema, all migrations

âš ï¸ IMPORTANT: CANONICAL DATABASE SCHEMA

This section contains the ONLY database migration files. Ignore any migration snippets in other sections - use ONLY the migrations defined here.

Canonical Table Names (enforced throughout):

Table	Purpose
`tenants`	Multi-tenant isolation
`users`	End users (not admins)
`administrators`	Platform admins
`invitations`	Admin invitations
`approval_requests`	Two-person approval workflow
`providers`	AI provider configurations
`models`	AI model definitions
`usage_events`	Request-level usage tracking
`invoices`	Billing invoices
`audit_logs`	Compliance audit trail

Table	Purpose
revenue_entries	Individual revenue events (subscriptions, credits, AI markup)
cost_entries	Infrastructure costs (AWS, external AI, platform fees)
revenue_daily_aggregates	Pre-computed daily revenue/cost summaries
model_revenue_tracking	Per-model revenue breakdown with markup analysis
accounting_periods	Month-end close tracking
reconciliation_entries	Accounting adjustments
revenue_export_log	Audit trail for accounting exports
preprompt_templates	Reusable pre-prompt patterns with weights
preprompt_instances	Actual pre-prompts used in AGI plans
preprompt_feedback	User feedback with attribution analysis
preprompt_attribution_scores	Learning correlations per factor
preprompt_learning_config	Admin-configurable learning parameters
preprompt_selection_log	Selection reasoning audit trail
user_memory_rules	User personal AI interaction rules
preset_user_rules	Pre-seeded rule templates users can add
user_rule_application_log	Tracks when user rules are applied
memory_categories	Hierarchical categorization of memory types
ai_ethics_standards	Industry AI ethics frameworks (NIST, ISO, EU AI Act)
ai_ethics_principle_standards	Mapping principles to standard sections
provider_rejections	Track provider/model rejections with fallback chain
rejection_patterns	Learn rejection patterns for smarter fallback
user_rejection_notifications	Notify users of rejected requests
model_rejection_stats	Per-model rejection statistics
rejection_analytics	Daily aggregated rejection stats by model/provider/mode
rejection_keyword_stats	Track violation keywords for policy review
tenant_users	End-user accounts with RLS isolation (v5.1.1)
platform_admins	Platform administrator accounts (v5.1.1)
service_api_keys	Service API key metadata and rate limits (v5.1.1)
service_api_key_audit	Partitioned API key usage audit log (v5.1.1)
tenant_sso_connections	SAML/OIDC SSO configuration per tenant (v5.1.1)
litellm_gateway_config	LiteLLM Gateway scaling parameters (v5.1.1)
system_component_health	Component health tracking for dashboard (v5.1.1)
system_alerts	Active system alerts (v5.1.1)
cortex_config	Per-tenant Graph-RAG configuration (v5.52.25)
cortex_entities	Knowledge entities with vector embeddings (v5.52.25)
cortex_relationships	Entity relationships with temporal validity (v5.52.25)
cortex_chunks	Text chunks with embeddings for RAG (v5.52.25)
cortex_activity_log	Cortex activity tracking (v5.52.25)
cortex_query_log	Cortex query analytics (v5.52.25)
oauth_clients	Registered third-party OAuth applications (v5.52.26)
oauth_authorization_codes	Short-lived OAuth auth codes (v5.52.26)
oauth_access_tokens	OAuth access token hashes (v5.52.26)
oauth_refresh_tokens	OAuth refresh tokens with rotation (v5.52.26)

Table	Purpose
oauth_user_authorizations	User consent records for OAuth apps (v5.52.26)
oauth_scope_definitions	Admin-configurable OAuth scopes (v5.52.26)
oauth_audit_log	Partitioned OAuth audit log (v5.52.26)
tenant_oauth_settings	Per-tenant OAuth configuration (v5.52.26)
oauth_signing_keys	RSA keys for JWT signing (v5.52.26)
mfa_backup_codes	One-time MFA recovery codes (v5.52.28)
mfa_trusted_devices	30-day device trust tokens (v5.52.28)
mfa_audit_log	Partitioned MFA event audit log (v5.52.28)
uds_conversations.detected_language	Auto-detected content language for search (v5.52.29)
uds_conversations.search_vector_simple	Simple tsvector for fallback search (v5.52.29)
uds_conversations.search_vector_english	Language-specific tsvector (v5.52.29)
rejected_prompt_archive	Archive rejected prompts with full content for analysis
routing_decision_cache	Semantic vector cache for brain router decisions
storage_tier_config	Adaptive storage configuration per deployment tier
ethics_config_presets	Externalized ethics frameworks (secular, religious, etc.)
tenant_ethics_config	Per-tenant ethics preset selection
preprompt_shadow_tests	Shadow A/B tests for pre-prompt optimization
preprompt_shadow_samples	Individual samples from shadow tests
preprompt_shadow_settings	Global shadow testing configuration
uncertainty_events	Logprob-based uncertainty detection events
user_gold_interactions	Highly-rated interactions for few-shot learning (pgvector)
synthesis_sessions	MoA parallel generation sessions
synthesis_drafts	Individual model drafts in synthesis
synthesis_results	Final synthesized responses
verification_sessions	Cross-provider adversarial verification sessions
verification_issues	Issues found by adversarial models
code_execution_sessions	Code sandbox execution sessions
code_execution_runs	Individual execution attempts with patches
intelligence_aggregator_config	Per-tenant Intelligence Aggregator configuration
prompt_patterns	Common prompt patterns for typeahead matching
user_prompt_history	User prompt history with embeddings for suggestions
suggestion_log	Typeahead suggestion usage tracking
result_ideas	Ideas shown with AI responses
proactive_suggestions	Push notification suggestions
trending_prompts	Popular prompts by domain
agi_ideas_config	Per-tenant AGI Ideas configuration
agi_learned_prompts	Persisted prompts with success rates and embeddings
agi_learned_ideas	Learned idea patterns with click rates
prompt_idea_associations	Links between prompts and effective ideas
agi_learning_events	Raw learning signals for analysis
agi_learning_aggregates	Pre-computed learning statistics
agi_model_selection_outcomes	Model selection outcomes with ratings
agi_routing_outcomes	Routing path effectiveness tracking

Table	Purpose
agi_domain_detection_feedback	Domain detection accuracy feedback
agi_orchestration_mode_outcomes	Mode effectiveness per context
agi_response_quality_metrics	Response quality dimensions
agi_preprompt_effectiveness	Preprompt effectiveness tracking
agi_user_learning_profile	User preferences learned over time
agi_unified_learning_log	Single source of truth for all learning
response_feedback	Enhanced feedback with 5-star ratings and comments
feedback_summaries	Pre-aggregated feedback statistics by scope
feedback_config	Per-tenant feedback configuration
reasoning_trees	Tree of Thoughts reasoning sessions
knowledge_entities	GraphRAG knowledge entities
knowledge_relationships	GraphRAG entity relationships
research_jobs	Deep Research job tracking
job_queue	Async job processing queue
lora_adapters	Dynamic LoRA adapter registry
generated_ui	Generative UI component tracking
cognitive_architecture_config	Per-tenant cognitive feature config
consciousness_test_results	Consciousness test results
genesis_personas	Cato mood definitions (Balanced, Scout, Sage, Spark, Guide)
user_persona_selections	User's selected Cato mood
cato_governor_state	Precision Governor decision history
cato_cbf_definitions	Control Barrier Function definitions
cato_cbf_violations	CBF violation log
cato_veto_log	Sensory veto events (hard stops)
cato_fracture_detections	Intent-action misalignment detection
cato_epistemic_recovery	Epistemic recovery event tracking
cato_human_escalations	Human escalation queue
cato_audit_trail	Merkle-verified append-only audit log
cato_audit_tiles	Audit batching for S3 anchoring
cato_audit_anchors	S3 Object Lock anchor references
cato_tenant_config	Per-tenant Cato safety configuration
cato_cloudwatch_alarm_mappings	CloudWatch alarm to veto signal mappings
cato_entropy_jobs	Async entropy check job tracking
cato_cloudwatch_sync_log	CloudWatch sync audit log
cato_api_persona_overrides	API-level session persona overrides
consciousness_profiles	Aggregated consciousness profiles
emergence_events	Consciousness emergence events
deep_thinking_sessions	Extended reasoning sessions
consciousness_parameters	Adjustable consciousness parameters
global_workspace	Global Workspace Theory state
recurrent_processing	Recurrent Processing state
integrated_information	IIT/Phi state
persistent_memory	Unified experience state
world_model	World-model grounding state
self_model	Self-awareness state

Table	Purpose
introspective_thoughts	Self-reflective thoughts
curiosity_topics	Curiosity tracking
creative_ideas	Creative synthesis storage
imagination_scenarios	Mental simulations
attention_focus	Attention/salience tracking
affective_state	Emotion-like signals
affective_events	Affective event log
autonomous_goals	Self-directed goals
exploration_sessions	Topic exploration sessions
semantic_memories	Long-term semantic knowledge
generated_apps	Think Tank generated interactive apps
app_interactions	User interactions with generated apps
user_app_preferences	User preferences for app factory
app_templates	Pre-built app templates
consent_records	GDPR consent tracking
gdpr_requests	GDPR data subject requests
data_retention_policies	Configurable data retention
phi_access_log	HIPAA PHI access audit trail
cato_genesis_state	Genesis boot sequence state tracking
cato_development_counters	Atomic counters for developmental gates
cato_developmental_stage	Current developmental stage (Piaget-inspired)
cato_circuit_breakers	Circuit breaker states and configuration
cato_circuit_breaker_events	Circuit breaker event history
cato_neurochemistry	Emotional/cognitive state (anxiety, curiosity, etc.)
cato_tick_costs	Per-tick cost tracking
cato_pricing_cache	AWS pricing table cache
cato_pymdp_state	PyMDP active inference state
cato_pymdp_matrices	Active inference matrices (A, B, C, D)
cato_consciousness_settings	Loop configuration settings
cato_loop_state	Consciousness loop execution state
data_processing_agreements	Sub-processor DPAs
data_breach_incidents	Security breach tracking
hipaa_config	Per-tenant HIPAA configuration
generative_ui_feedback	User feedback on generated UI
ui_improvement_requests	Real-time improvement requests
ui_improvement_sessions	Live AGI improvement sessions
ui_improvement_iterations	Session iteration history
ui_feedback_learnings	AGI learnings from feedback
ui_feedback_config	Per-tenant feedback configuration
ui_feedback_aggregates	Pre-computed feedback analytics
generated_multipage_apps	Multi-page web applications
app_pages	Individual pages within multi-page apps
app_versions	Version history for multi-page apps
app_deployments	Deployment tracking for apps
multipage_app_templates	Pre-built app templates
app_analytics	Usage analytics for generated apps

Table	Purpose
multipage_app_config	Per-tenant multi-page app config
self_hosted_model_metadata	Comprehensive metadata for 56 self-hosted AI models
model_orchestration_preferences	Tenant-specific model selection preferences
self_hosted_model_usage	Usage analytics for self-hosted models per tenant
model_selection_history	History of model selections by the orchestration engine
orchestration_methods	Shared reusable AI orchestration methods (70+)
orchestration_workflows	System (49) and user-defined orchestration workflows
workflow_method_bindings	Links methods to workflows with parameter overrides
workflow_customizations	Tenant-level customizations of system workflows
orchestration_executions	Workflow execution tracking with audit
orchestration_step_executions	Individual step execution records
user_workflow_templates	User-saved workflow templates with sharing
orchestration_audit_log	Audit trail for all workflow changes
thinktank_media_capabilities	Media input/output capabilities for Think Tank integration
model_proficiency_rankings	Ranked proficiency scores for models across domains and modes
model_discovery_log	Log of model discoveries with proficiency generation status
result_derivations	Comprehensive history of how Think Tank results are derived
derivation_steps	Individual steps in the execution plan
derivation_model_usage	Models used during result generation with token and cost tracking
derivation_timeline_events	Timeline events for visualization of execution flow
domain_ethics_config	Per-tenant domain ethics configuration
domain_ethics_custom_frameworks	Custom and built-in professional ethics frameworks
domain_ethics_audit_log	Audit log of ethics checks performed
domain_ethics_framework_overrides	Tenant-specific framework overrides
domain_ethics_violation_patterns	Custom violation patterns for frameworks
model_registry	Central registry of all AI models with capabilities
model_endpoints	Communication endpoints for models with auth and health
model_sync_config	Configuration for timed sync service
model_sync_jobs	History of sync job executions
new_model_detections	Newly detected models pending processing
model_routing_rules	Rules for routing requests to models
ethics_pipeline_log	Log of all ethics checks at prompt/synthesis levels
ethics_rerun_history	History of workflow reruns triggered by violations
ethics_pipeline_config	Per-tenant ethics pipeline configuration
inference_components_config	Per-tenant SageMaker Inference Components configuration
shared_inference_endpoints	Shared SageMaker endpoints hosting multiple models
inference_components	Model components on shared endpoints
tier_assignments	Current and recommended hosting tiers per model

Table	Purpose
tier_transitions	History of tier changes for models
component_load_events	Model load/unload event history
inference_component_events	Audit log of inference component events
consciousness_heartbeat_log	Consciousness heartbeat execution log
ethics_frameworks	Externalized ethics frameworks (secular, religious, etc.)
tenant_ethics_selection	Per-tenant ethics framework selection
user_persistent_context	User-level persistent context entries with embeddings
user_context_extraction_log	Context extraction audit trail
user_context_preferences	Per-user context learning preferences
consciousness_predictions	Active Inference predictions with outcomes
learning_candidates	High-value interactions flagged for LoRA training
lora_evolution_jobs	Weekly LoRA training job tracking
prediction_accuracy_aggregates	Aggregated prediction accuracy by context
consciousness_evolution_state	Track consciousness evolution across generations
local_ego_config	Per-tenant Local Ego configuration (shared model approach)
shared_ego_endpoints	Shared Ego SageMaker endpoint status
ego_processing_log	Local Ego processing decisions and recruitment
ego_config	Per-tenant Zero-Cost Ego configuration
ego_identity	Persistent identity (name, narrative, traits)
ego_affect	Real-time emotional state
ego_working_memory	Short-term memory with 24h expiry
ego_goals	Active and historical goals
ego_injection_log	Ego context injection audit trail
library_registry_config	Per-tenant library assist configuration
open_source_libraries	Global registry of open-source tools
tenant_library_overrides	Per-tenant library customization
library_usage_events	Library invocation audit trail
library_usage_aggregates	Pre-computed library usage statistics
library_update_jobs	Library registry update job tracking
library_version_history	Library version change history
library_registry_metadata	Global library registry metadata
library_execution_config	Per-tenant execution configuration
library_executions	Execution records with metrics and billing
library_execution_queue	Priority queue for pending executions
library_execution_logs	Execution debug logs
library_executor_pool	Executor pool status for auto-scaling
library_execution_aggregates	Pre-computed execution statistics
conscious_orchestrator_decisions	Logs decisions made by conscious orchestrator (plan/clarify/defer/refuse)
ecd_metrics	ECD verification results per request (score, entities, refinements)
ecd_audit_log	Full audit trail with original and final responses
ecd_entity_stats	Aggregated entity extraction statistics by type
distillation_training_data	Teacher model reasoning traces for student training

Table	Purpose
inference_student_versions	Student model versions with metrics
distillation_jobs	Distillation training job tracking
semantic_cache	Vector similarity cache for responses
semantic_cache_metrics	Cache performance metrics
metacognition_assessments_v2	Enhanced confidence assessments
reward_training_data	Preference comparison training data
reward_model_versions	Reward model version tracking
counterfactual_candidates	Routing decision records for counterfactual analysis
counterfactual_simulations	Alternative path simulation results
knowledge_gaps	Detected knowledge gaps from uncertainty
curiosity_goals	Autonomous exploration goals
causal_links	Turn-to-turn causal relationships
conversation_turns	Conversation history for causal tracking
reasoning_traces	Full request traces (partitioned by month)
reasoning_outcomes	User feedback on reasoning traces
hitl_question_batches	HITL question batch records
hitl_rate_limits	HITL rate limit configuration per scope
hitl_question_cache	HITL deduplication cache with TTL and pgvector embeddings
hitl_voi_aspects	HITL VOI aspect tracking
hitl_voi_decisions	HITL VOI decision records
hitl_abstention_config	HITL abstention detection settings
hitl_abstention_events	HITL abstention event log
hitl_escalation_chains	HITL escalation chain configuration
gateway_instances	Gateway instance registry with heartbeat tracking
gateway_statistics	Gateway time-series metrics (5-minute buckets)
gateway_configuration	Gateway per-tenant and global settings
gateway_alerts	Gateway alert and incident tracking
gateway_sessions	Gateway active connection tracking
gateway_audit_log	Gateway admin action audit trail
code_quality_snapshots	Periodic code coverage/quality metrics
test_file_registry	Source files and test status tracking
json_parse_locations	JSON.parse migration tracking
technical_debt_items	Technical debt items by priority/status
code_quality_alerts	Code quality regression alerts
agents	Sovereign Mesh agent registry with AI config
agent_executions	Agent OODA execution state
agent_iteration_logs	Detailed agent iteration tracking
apps	Sovereign Mesh app registry (3,000+ apps)
app_connections	OAuth/API credentials for apps
app_learned_inferences	AI learning loop for apps
ai_helper_calls	AI Helper usage tracking
ai_helper_cache	AI Helper response caching
ai_helper_config	AI Helper system/tenant configuration
workflow_blueprints	Pre-flight workflow provisioning
capability_checks	Capability verification results

Table	Purpose
cato_decision_events	Cato decision transparency events
cato_war_room_deliberations	Cato War Room deliberation capture
hitl_queue_configs	HITL approval queue configuration
hitl_approval_requests	HITL approval request records
execution_snapshots	Time-travel debugging snapshots
replay_sessions	Replay/what-if analysis sessions
thinktank_user_consents	GDPR consent records
thinktank_gdpr_requests	GDPR data subject requests
thinktank_security_config	Per-tenant security settings
thinktank_rejections	User rejection notifications
thinktank_user_preferences	User model and UI preferences
thinktank_ui_feedback	UI feedback collection
thinktank_ui_improvement_sessions	AI UI improvement sessions
thinktank_multipage_apps	User-generated multipage apps
thinktank_voice_sessions	Voice transcription records
thinktank_file_attachments	File attachment storage
brand_kits	AI Report Writer brand customization (logo, colors, fonts)
report_templates	Reusable AI report structures
generated_reports	AI-generated reports with content
report_smart_insights	Extracted insights from reports (denormalized)
report_exports	Report export records with S3 references
report_chat_history	Interactive report modification history
report_schedules	Scheduled automatic report generation

Migration 154: Cato Advanced Configuration (v6.1.1)

Extends the Genesis Cato Safety Architecture with configurable parameters and new supporting tables.

New Columns on cato_tenant_config:

Column	Type	Default	Description
enable_redis	BOOLEAN	TRUE	Enable Redis state persistence
redis_rejection_ttl_seconds	INTEGER	60	TTL for rejection history
redis_persona_override_ttl_seconds	INTEGER	300	TTL for persona overrides
redis_recovery_state_ttl_seconds	INTEGER	600	TTL for recovery state
enable_cloudwatch_vetoes	BOOLEAN	TRUE	Enable CloudWatch alarm integration
cloudwatch_sync_interval_seconds	INTEGER	60	CloudWatch sync polling interval
cloudwatch_alarm_mappings	JSONB	'{}'	Custom alarm mappings
enable_async_entropy	BOOLEAN	TRUE	Enable async entropy checks
entropy_async_threshold	NUMERIC(5,4)	0.6	Threshold for async processing
entropy_job_ttl_hours	INTEGER	24	Entropy job result retention
entropy_max_concurrent_jobs	INTEGER	10	Max concurrent entropy jobs

Column	Type	Default	Description
fracture_word_overlap_weight	NUMERIC(5,4)	0.20	Fracture detection weight
fracture_intent_keyword_weight	NUMERIC(5,4)	0.25	Fracture detection weight
fracture_sentiment_weight	NUMERIC(5,4)	0.15	Fracture detection weight
fracture_topic_coherence_weight	NUMERIC(5,4)	0.20	Fracture detection weight
fracture_completeness_weight	NUMERIC(5,4)	0.20	Fracture detection weight
fracture_alignment_threshold	NUMERIC(5,4)	0.40	Fracture alignment threshold
fracture_evasion_threshold	NUMERIC(5,4)	0.60	Fracture evasion threshold
cbf_authorization_check_enabled	BOOLEAN	TRUE	Enable model authorization checks
cbf_baa_verification_enabled	BOOLEAN	TRUE	Enable BAA verification
cbf_cost_alternative_enabled	BOOLEAN	TRUE	Enable cost alternative suggestions
cbf_max_cost_reduction_percent	NUMERIC(5,2)	50.00	Target cost reduction

New Tables:

Table	Purpose
cato_cloudwatch_alarm_mappings	CloudWatch alarm to veto signal mappings
cato_entropy_jobs	Async entropy check job tracking
cato_cloudwatch_sync_log	CloudWatch sync audit log

cato_cloudwatch_alarm_mappings Schema:

Column	Type	Description
id	UUID	Primary key
tenant_id	UUID	Tenant reference
alarm_name	VARCHAR(255)	CloudWatch alarm name
alarm_name_pattern	VARCHAR(255)	Regex pattern for alarm matching
veto_signal	VARCHAR(50)	Veto signal to activate
veto_severity	VARCHAR(20)	warning, critical, emergency
is_enabled	BOOLEAN	Enable/disable mapping
auto_clear_on_ok	BOOLEAN	Auto-clear when alarm resolves
description	TEXT	Mapping description

cato_entropy_jobs Schema:

Column	Type	Description
id	UUID	Primary key
tenant_id	UUID	Tenant reference
session_id	UUID	Session reference
job_id	VARCHAR(100)	Unique job identifier
status	VARCHAR(20)	pending, processing, completed, failed

Column	Type	Description
prompt	TEXT	Original prompt
response	TEXT	AI response to check
model	VARCHAR(100)	Model used
check_mode	VARCHAR(20)	Check mode (sync, async, deep)
entropy_score	NUMERIC(5,4)	Calculated entropy score
consistency	NUMERIC(5,4)	Consistency score
is_potential_deception	BOOLEAN	Deception flag
deception_indicators	JSONB	Detailed indicators
expires_at	TIMESTAMPTZ	Job expiration time

cato_cloudwatch_sync_log Schema:

Column	Type	Description
id	UUID	Primary key
tenant_id	UUID	Tenant reference (NULL for global)
sync_type	VARCHAR(20)	scheduled, manual, alarm_event
alarms_checked	INTEGER	Number of alarms checked
alarms_in_alarm	INTEGER	Alarms in ALARM state
vetos_activated	INTEGER	Vetos activated this sync
vetos_cleared	INTEGER	Vetos cleared this sync
success	BOOLEAN	Sync success flag
error_message	TEXT	Error details if failed
duration_ms	INTEGER	Sync duration

Default Alarm Mappings Seeded:

Alarm	Signal	Severity
radiant-system-cpu-critical	SYSTEM_OVERLOAD	emergency
radiant-system-memory-critical	SYSTEM_OVERLOAD	emergency
radiant-security-breach	DATA_BREACH_DETECTED	emergency
radiant-compliance-alert	COMPLIANCE_VIOLATION	critical
radiant-anomaly-detection	ANOMALY_DETECTED	warning
radiant-model-health	MODEL_UNAVAILABLE	warning

Migration 155: Cato Spec Alignment (v6.1.1 Patch 3)

Fixes mood attributes to match Genesis Cato v2.3.1 specification and adds tenant default mood support.

Mood Attribute Fixes:

Mood	Attribute	Was	Now
Sage	discovery	0.8	0.6

Mood	Attribute	Was	Now
Spark	achievement	0.7	0.5
Spark	reflection	0.6	0.4
Guide	discovery	0.7	0.5
Guide	reflection	0.5	0.7

New Column on `cato_tenant_config`:

Column	Type	Default	Description
`default_mood`	VARCHAR(50)	'balanced'	Tenant-specific default mood override

New Table: `cato_api_persona_overrides`

Stores API-level persona overrides for sessions (temporary, session-based).

Column	Type	Description
`id`	UUID	Primary key
`tenant_id`	UUID	Tenant reference
`session_id`	UUID	Session reference
`user_id`	UUID	Optional user reference
`persona_name`	VARCHAR(50)	Mood name (balanced, scout, sage, spark, guide)
`expires_at`	TIMESTAMPTZ	When override expires
`created_at`	TIMESTAMPTZ	Creation timestamp
`created_by`	UUID	Admin who set the override
`reason`	TEXT	Optional reason for override

Migration 152: Advanced Cognition (v6.1.0)

Table	Purpose
`distillation_training_data`	Teacher reasoning traces with quality scores
`inference_student_versions`	Student model versions with SageMaker endpoints
`distillation_jobs`	Training job lifecycle tracking
`semantic_cache`	Vector similarity cache with pgvector embeddings
`semantic_cache_metrics`	Hourly cache performance metrics
`metacognition_assessments_v2`	Confidence, entropy, escalation decisions
`reward_training_data`	Winning/losing response pairs with signals
`reward_model_versions`	Reward model artifacts and accuracy
`counterfactual_candidates`	Original routing decisions to analyze
`counterfactual_simulations`	Alternative model comparison results
`knowledge_gaps`	Topics with low user satisfaction
`curiosity_goals`	Budget-controlled autonomous exploration

Table	Purpose
causal_links	Reference, elaboration, correction relationships
conversation_turns	Turn content for causal analysis
reasoning_traces	Partitioned trace storage (monthly partitions)
reasoning_outcomes	Regeneration, edit, dwell time signals

Migration 133: ECD Tables - Truth Engine™ (v6.0.4-S1)

Table	Purpose
ecd_metrics	Per-request ECD verification results with scores and entity counts
ecd_audit_log	Full response audit trail for divergence analysis
ecd_entity_stats	Daily aggregated stats by entity type (grounded/divergent)

ECD Configuration (added to system_config): - ECD_ENABLED - Enable verification loop - ECD_THRESHOLD - Max acceptable divergence (0.1) - ECD_MAX_REFINEMENTS - Auto-correction attempts (2) - ECD_BLOCK_ON_FAILURE - Block failed responses - ECD_HEALTHCARE_THRESHOLD - Healthcare threshold (0.05) - ECD_FINANCIAL_THRESHOLD - Financial threshold (0.05) - ECD_LEGAL_THRESHOLD - Legal threshold (0.05) - ECD_ANCHORING_ENABLED - Critical fact anchoring - ECD_ANCHORING_OVERSIGHT - Send to oversight queue

Helper Functions: - get_ecd_stats(tenant_id, days) - ECD statistics - get_ecd_trend(tenant_id, days) - Score trend over time - get_ecd_entity_breakdown(tenant_id, days) - Entity type stats - log_ecd_metrics(...) - Log verification results - update_ecd_entity_stats(...) - Update entity aggregates

Migration 105: Consciousness Enhancements (v4.18.19)

Column/Table	Change
open_source_libraries.description_embedding	vector(1536) for Vector RAG semantic search
ego_working_memory.consolidated	Boolean flag for memory consolidation tracking
introspective_thoughts.thought_type	VARCHAR(50) for idle thought categorization
semantic_memories.source	VARCHAR(50) for consolidation source tracking
conscious_orchestrator_decisions	New table for architecture inversion decision logging

Type Imports

```
import {
  AIProvider,
```

```
  AIModel,
  EXTERNAL_PROVIDERS,
  PROVIDER_ENDPOINTS,
  Administrator,
  Invitation,
  ApprovalRequest,
  UsageEvent,
  Invoice,
  AuditLog,
} from '@radiant/shared';
```

RADIANT v2.2.0 - Prompt 7: External Providers & Database Schema/Migrations

> **Prompt 7 of 9** | Target Size: ~85KB | Version: 3.7.0 | December 2024

OVERVIEW

This prompt creates the external AI provider integrations and complete database schema:

1. **External Providers** - 21 AI provider configurations with LiteLLM routing
2. **Provider Categories** - Text, Image, Video, Audio, 3D, Embeddings, Search
3. **LiteLLM Configuration** - Complete config.yaml for all external providers
4. **PostgreSQL Schema** - Complete schema with Row-Level Security (RLS)
5. **DynamoDB Tables** - Session, cache, and real-time data tables
6. **Database Migrations** - Versioned migration scripts
7. **Pricing Configuration** - Dynamic pricing with 40% external / 75% self-hosted markup

---

ARCHITECTURE

```
Ã¢â€ Å’Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â
Ã¢â€ â€š                              EXTERNAL PROVIDERS LAYER                          Ã¢â€ â€š
Ã¢â€ Å"Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â
Ã¢â€ â€š                                                                              Ã¢â€ â€š
Ã¢â€ â€š   Client Request                                                             Ã¢â€ â€š
Ã¢â€ â€š        Ã¢â€ â€š                                                                         Ã¢â
Ã¢â€ â€š        Ã¢â€"Â¼                                                                         Ã¢â€
Ã¢â€ â€š   Ã¢â€ Å’Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢
Ã¢â€ â€š   Ã¢â€ â€š  Lambda Router  Ã¢â€ â€šÃ¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€"Â¶Ã¢â€ â€š   Lite
Ã¢â€ â€š   Ã¢â€ â€š  (Prompt 4)     Ã¢â€ â€š     Ã¢â€ â€š   Gateway        Ã¢â€ â€š     Ã¢â€ â€š S
Ã¢â€ â€š   Ã¢â€ â€ Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ Â¬Ã¢â€ â‚¬Ã¢â€
Ã¢â€ â€š           Ã¢â€ â€š                              Ã¢â€ â€š
Ã¢â€ â€š           Ã¢â€ â€š                              Ã¢â€"Â¼
Ã¢â€ â€š   Ã¢â€ Å’Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢â€"Â¼Ã¢â€ â‚¬Ã¢â€ â‚¬Ã¢
```


EXTERNAL PROVIDER ROUTING
TEXT GEN
IMAGE
VIDEO
OpenAI
DALL-E 3
Anthropic
Midjourney
Google
Stability
xAI
Replicate
DeepSeek
Ideogram
Mistral
Flux
EMBEDDINGS
SEARCH
3D
OpenAI
Perplexity
Cohere
Exa
Voyage
Tavily
Google
You.com
METERING & BILLING
Token counting
Cost calculation
Usage aggregation
40% markup (external)
75% markup (self-hosted)
DATABASE LAYER
AURORA POSTGRESQL
DYNAMODB
Tenants
Users
Admins
Providers
Models
Usage
Billing
Audit Logs
RLS Policies
Sessions
Chat History
Model States
Cache
WebSocket Conne
Approval Reques
TTL Cleanup

DIRECTORY STRUCTURE

```
packages/infrastructure/
Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ lib/
Ã¢â€ â€š   Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ config/
Ã¢â€ â€š   Ã¢â€ â€š   Ã¢â€ â€ Ã¢â€ â¬Ã¢â€ â¬ providers/
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ index.ts                  # All provider export
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ text.providers.ts         # Text generation pro
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ image.providers.ts        # Image generation pr
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ video.providers.ts        # Video generation pr
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ audio.providers.ts        # Audio/speech provid
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ embedding.providers.ts    # Embedding providers
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ search.providers.ts       # Search providers
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ 3d.providers.ts           # 3D generation provi
Ã¢â€ â€š   Ã¢â€ â€š       Ã¢â€ â€ Ã¢â€ â¬Ã¢â€ â¬ pricing.config.ts         # Pricing configurat
Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ litellm/
Ã¢â€ â€š   Ã¢â€ â€ Ã¢â€ â¬Ã¢â€ â¬ config/
Ã¢â€ â€š       Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ external.yaml                # External provider config
Ã¢â€ â€š       Ã¢â€ â€ Ã¢â€ â¬Ã¢â€ â¬ complete.yaml                # Combined config (external
Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ migrations/
Ã¢â€ â€š   Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ 001_initial_schema.sql          # Base tables
Ã¢â€ â€š   Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ 002_tenant_isolation.sql       # RLS policies
Ã¢â€ â€š   Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ 003_ai_models.sql              # Model registry
Ã¢â€ â€š   Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ 004_usage_billing.sql          # Metering tables
Ã¢â€ â€š   Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ 005_admin_approval.sql        # Admin workflow tables
Ã¢â€ â€š   Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ 006_self_hosted_models.sql    # Self-hosted additions (Pro
Ã¢â€ â€š   Ã¢â€ Å"Ã¢â€ â¬Ã¢â€ â¬ 007_external_providers.sql    # External provider tables
Ã¢â€ â€š   Ã¢â€ â€ Ã¢â€ â¬Ã¢â€ â¬ migrations.ts                 # Migration runner
Ã¢â€ â€ Ã¢â€ â¬Ã¢â€ â¬ dynamodb/
    Ã¢â€ â€ Ã¢â€ â¬Ã¢â€ â¬ tables.ts                      # DynamoDB table definitions
```

PART 1: EXTERNAL PROVIDER DEFINITIONS

Provider Pricing Structure

All external providers use a **40% markup** on base costs: - Base cost: Provider's published rate - Billed cost: Base Ãfâ€" 1.40

Self-hosted models (Prompt 6) use a **75% markup** on infrastructure costs.

packages/infrastructure/lib/config/providers/index.ts

```
/**
 * External Provider Registry
 * All external AI providers with pricing and capabilities
 * Pricing includes 40% markup over provider costs
```

```
 *
 * RADIANT v2.2.0 - December 2024
 */

export * from './text.providers';
export * from './image.providers';
export * from './video.providers';
export * from './audio.providers';
export * from './embedding.providers';
export * from './search.providers';
export * from './3d.providers';
export * from './pricing.config';

import { TEXT_PROVIDERS } from './text.providers';
import { IMAGE_PROVIDERS } from './image.providers';
import { VIDEO_PROVIDERS } from './video.providers';
import { AUDIO_PROVIDERS } from './audio.providers';
import { EMBEDDING_PROVIDERS } from './embedding.providers';
import { SEARCH_PROVIDERS } from './search.providers';
import { THREE_D_PROVIDERS } from './3d.providers';

// ============================================================================
// PROVIDER TYPES
// ============================================================================

export type ProviderCategory =
  | 'text_generation'
  | 'image_generation'
  | 'video_generation'
  | 'audio_generation'
  | 'speech_to_text'
  | 'text_to_speech'
  | 'embedding'
  | 'search'
  | '3d_generation'
  | 'reasoning';

export interface ExternalProvider {
  id: string;
  name: string;
  displayName: string;
  category: ProviderCategory;
  description: string;
  website: string;
  apiBaseUrl: string;
  authType: 'api_key' | 'oauth' | 'bearer';
  secretName: string;
  enabled: boolean;
```

```
  regions: string[];
  models: ExternalModel[];
  rateLimit?: {
    requestsPerMinute: number;
    tokensPerMinute?: number;
  };
  features: string[];
  compliance?: string[];
}

export interface ExternalModel {
  id: string;
  modelId: string;
  litellmId: string;
  name: string;
  displayName: string;
  description: string;
  category: ProviderCategory;
  capabilities: string[];
  contextWindow?: number;
  maxOutput?: number;
  inputModalities: ('text' | 'image' | 'audio' | 'video')[];
  outputModalities: ('text' | 'image' | 'audio' | 'video' | '3d')[];
  pricing: ExternalProviderPricing;
  deprecated?: boolean;
  successorModel?: string;
}

export interface ExternalProviderPricing {
  type: 'per_token' | 'per_request' | 'per_second' | 'per_image' | 'per_minute';
  inputCostPer1k?: number;      // $/1K tokens (input)
  outputCostPer1k?: number;     // $/1K tokens (output)
  baseCostPerRequest?: number;  // Base $/request
  costPerSecond?: number;       // $/second (video/audio)
  costPerImage?: number;        // $/image
  costPerMinute?: number;       // $/minute (audio)
  markup: number;               // Multiplier (1.40 for 40%)
  billedInputPer1k?: number;    // Final billed rate
  billedOutputPer1k?: number;   // Final billed rate
}

// ============================================================================
// AGGREGATED REGISTRY
// ============================================================================

export const ALL_EXTERNAL_PROVIDERS: ExternalProvider[] = [
  ...TEXT_PROVIDERS,
  ...IMAGE_PROVIDERS,
```

```
  ...VIDEO_PROVIDERS,
  ...AUDIO_PROVIDERS,
  ...EMBEDDING_PROVIDERS,
  ...SEARCH_PROVIDERS,
  ...THREE_D_PROVIDERS,
];

export const PROVIDER_BY_ID = new Map<string, ExternalProvider>(
  ALL_EXTERNAL_PROVIDERS.map(p => [p.id, p])
);

export const ALL_EXTERNAL_MODELS: ExternalModel[] =
  ALL_EXTERNAL_PROVIDERS.flatMap(p => p.models);

export const MODEL_BY_ID = new Map<string, ExternalModel>(
  ALL_EXTERNAL_MODELS.map(m => [m.id, m])
);

export const MODEL_BY_LITELLM_ID = new Map<string, ExternalModel>(
  ALL_EXTERNAL_MODELS.map(m => [m.litellmId, m])
);

// ============================================================================
// HELPER FUNCTIONS
// ============================================================================

export function calculateCost(
  model: ExternalModel,
  inputTokens: number,
  outputTokens: number,
  additionalUnits?: { images?: number; seconds?: number; minutes?: number }
): { baseCost: number; billedCost: number } {
  const pricing = model.pricing;
  let baseCost = 0;

  if (pricing.type === 'per_token') {
    baseCost =
      (inputTokens / 1000) * (pricing.inputCostPer1k || 0) +
      (outputTokens / 1000) * (pricing.outputCostPer1k || 0);
  } else if (pricing.type === 'per_request') {
    baseCost = pricing.baseCostPerRequest || 0;
  } else if (pricing.type === 'per_image') {
    baseCost = (additionalUnits?.images || 1) * (pricing.costPerImage || 0);
  } else if (pricing.type === 'per_second') {
    baseCost = (additionalUnits?.seconds || 0) * (pricing.costPerSecond || 0);
  } else if (pricing.type === 'per_minute') {
    baseCost = (additionalUnits?.minutes || 0) * (pricing.costPerMinute || 0);
  }
```

```
  return {
    baseCost,
    billedCost: baseCost * pricing.markup,
  };
}

export function getProviderSecrets(): Map<string, string> {
  const secrets = new Map<string, string>();
  for (const provider of ALL_EXTERNAL_PROVIDERS) {
    secrets.set(provider.id, provider.secretName);
  }
  return secrets;
}
```

packages/infrastructure/lib/config/providers/text.providers.ts

```
/**
 * Text Generation Providers - RADIANT v4.12
 * OpenAI (GPT-5, GPT-4.1, O-Series), Anthropic (Claude 4.5), Google (Gemini 3),
 * xAI (Grok 4), DeepSeek (V3), Mistral, Cohere, Perplexity
 *
 * Updated: December 2024
 */

import { ExternalProvider, ExternalModel } from './index';

const MARKUP = 1.40; // 40% markup

// =============================================================================
// OPENAI - GPT-5, GPT-4.1, and O-Series Models
// =============================================================================

const OPENAI_MODELS: ExternalModel[] = [
  // GPT-5 Series (Flagship - December 2024)
  {
    id: 'openai-gpt-5-2',
    modelId: 'gpt-5.2',
    litellmId: 'gpt-5.2',
    name: 'gpt-5.2',
    displayName: 'GPT-5.2',
    description: 'Most capable flagship model with configurable reasoning and 400K context',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'reasoning', 'function_calling', 'json_mode', 'streaming'
    contextWindow: 400000,
    maxOutput: 128000,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
```

```
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.00175,
      outputCostPer1k: 0.014,
      cachedInputCostPer1k: 0.000175,
      markup: MARKUP,
      billedInputPer1k: 0.00175 * MARKUP,
      billedOutputPer1k: 0.014 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-12-01',
      family: 'gpt-5',
      version: '5.2',
      isLatest: true,
      supportsBatch: true,
      batchDiscount: 0.5,
    },
  },
  {
    id: 'openai-gpt-5-1',
    modelId: 'gpt-5.1',
    litellmId: 'gpt-5.1',
    name: 'gpt-5.1',
    displayName: 'GPT-5.1',
    description: 'Previous flagship with full capabilities and 400K context',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'reasoning', 'function_calling', 'json_mode', 'streaming'
    contextWindow: 400000,
    maxOutput: 128000,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.00125,
      outputCostPer1k: 0.01,
      markup: MARKUP,
      billedInputPer1k: 0.00125 * MARKUP,
      billedOutputPer1k: 0.01 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-10-01',
      family: 'gpt-5',
      version: '5.1',
    },
  },
  {
    id: 'openai-gpt-5-mini',
    modelId: 'gpt-5-mini',
```

```
    litellmId: 'gpt-5-mini',
    name: 'gpt-5-mini',
    displayName: 'GPT-5 Mini',
    description: 'Fast, affordable GPT-5 variant for high-volume tasks',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'function_calling', 'json_mode', 'streaming'],
    contextWindow: 400000,
    maxOutput: 128000,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.00025,
      outputCostPer1k: 0.002,
      markup: MARKUP,
      billedInputPer1k: 0.00025 * MARKUP,
      billedOutputPer1k: 0.002 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-11-01',
      family: 'gpt-5',
      version: 'mini',
    },
  },
  {
    id: 'openai-gpt-5-nano',
    modelId: 'gpt-5-nano',
    litellmId: 'gpt-5-nano',
    name: 'gpt-5-nano',
    displayName: 'GPT-5 Nano',
    description: 'Ultra-fast, most cost-efficient GPT-5 for classification and simple tasks',
    category: 'text_generation',
    capabilities: ['chat', 'function_calling', 'json_mode', 'streaming'],
    contextWindow: 400000,
    maxOutput: 128000,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.00005,
      outputCostPer1k: 0.0004,
      markup: MARKUP,
      billedInputPer1k: 0.00005 * MARKUP,
      billedOutputPer1k: 0.0004 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-11-01',
      family: 'gpt-5',
```

```
    version: 'nano',
  },
},
// GPT-4.1 Series (1M Context - April 2024)
{
  id: 'openai-gpt-4-1',
  modelId: 'gpt-4.1-2025-04-14',
  litellmId: 'gpt-4.1-2025-04-14',
  name: 'gpt-4.1',
  displayName: 'GPT-4.1',
  description: 'Best coding model with industry-leading 1M context window',
  category: 'text_generation',
  capabilities: ['chat', 'vision', 'function_calling', 'json_mode', 'streaming', 'agents', '
  contextWindow: 1000000,
  maxOutput: 32768,
  inputModalities: ['text', 'image', 'video'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.002,
    outputCostPer1k: 0.008,
    markup: MARKUP,
    billedInputPer1k: 0.002 * MARKUP,
    billedOutputPer1k: 0.008 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-04-14',
    family: 'gpt-4.1',
    version: '4.1',
    isLatest: true,
    supportsFineTuning: true,
  },
},
{
  id: 'openai-gpt-4-1-mini',
  modelId: 'gpt-4.1-mini-2025-04-14',
  litellmId: 'gpt-4.1-mini-2025-04-14',
  name: 'gpt-4.1-mini',
  displayName: 'GPT-4.1 Mini',
  description: 'Fast 1M context model beating GPT-4o benchmarks at lower cost',
  category: 'text_generation',
  capabilities: ['chat', 'vision', 'function_calling', 'json_mode', 'streaming'],
  contextWindow: 1000000,
  maxOutput: 32768,
  inputModalities: ['text', 'image'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
```

```
    inputCostPer1k: 0.0004,
    outputCostPer1k: 0.0016,
    markup: MARKUP,
    billedInputPer1k: 0.0004 * MARKUP,
    billedOutputPer1k: 0.0016 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-04-14',
    family: 'gpt-4.1',
    version: 'mini',
  },
},
{
  id: 'openai-gpt-4-1-nano',
  modelId: 'gpt-4.1-nano-2025-04-14',
  litellmId: 'gpt-4.1-nano-2025-04-14',
  name: 'gpt-4.1-nano',
  displayName: 'GPT-4.1 Nano',
  description: 'Fastest 1M context model for classification and autocomplete',
  category: 'text_generation',
  capabilities: ['chat', 'function_calling', 'json_mode', 'streaming'],
  contextWindow: 1000000,
  maxOutput: 16384,
  inputModalities: ['text'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.0001,
    outputCostPer1k: 0.0004,
    markup: MARKUP,
    billedInputPer1k: 0.0001 * MARKUP,
    billedOutputPer1k: 0.0004 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-04-14',
    family: 'gpt-4.1',
    version: 'nano',
  },
},
// O-Series Reasoning Models
{
  id: 'openai-o3',
  modelId: 'o3-2025-04-16',
  litellmId: 'o3-2025-04-16',
  name: 'o3',
  displayName: 'O3',
  description: 'Most powerful reasoning model for math, science, and complex coding',
  category: 'reasoning',
```

```
    capabilities: ['chat', 'reasoning', 'vision', 'function_calling', 'web_browsing', 'code_exe
    contextWindow: 200000,
    maxOutput: 100000,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.002,
      outputCostPer1k: 0.008,
      markup: MARKUP,
      billedInputPer1k: 0.002 * MARKUP,
      billedOutputPer1k: 0.008 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-04-16',
      family: 'o-series',
      version: 'o3',
      isLatest: true,
      reasoningModel: true,
    },
  },
  {
    id: 'openai-o3-pro',
    modelId: 'o3-pro-2025-06-10',
    litellmId: 'o3-pro-2025-06-10',
    name: 'o3-pro',
    displayName: 'O3 Pro',
    description: 'Extended compute O3 for maximum reasoning accuracy on hardest problems',
    category: 'reasoning',
    capabilities: ['chat', 'reasoning', 'vision', 'function_calling', 'streaming'],
    contextWindow: 200000,
    maxOutput: 100000,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.02,
      outputCostPer1k: 0.08,
      markup: MARKUP,
      billedInputPer1k: 0.02 * MARKUP,
      billedOutputPer1k: 0.08 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-06-10',
      family: 'o-series',
      version: 'o3-pro',
      reasoningModel: true,
    },
```

```
},
{
  id: 'openai-o4-mini',
  modelId: 'o4-mini-2025-04-16',
  litellmId: 'o4-mini-2025-04-16',
  name: 'o4-mini',
  displayName: 'O4 Mini',
  description: 'Fast reasoning model, best on AIME math benchmarks',
  category: 'reasoning',
  capabilities: ['chat', 'reasoning', 'vision', 'function_calling', 'streaming'],
  contextWindow: 200000,
  maxOutput: 100000,
  inputModalities: ['text', 'image'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.0011,
    outputCostPer1k: 0.0044,
    markup: MARKUP,
    billedInputPer1k: 0.0011 * MARKUP,
    billedOutputPer1k: 0.0044 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-04-16',
    family: 'o-series',
    version: 'o4-mini',
    isLatest: true,
    reasoningModel: true,
  },
},
{
  id: 'openai-o3-mini',
  modelId: 'o3-mini-2025-01-31',
  litellmId: 'o3-mini-2025-01-31',
  name: 'o3-mini',
  displayName: 'O3 Mini',
  description: 'Small reasoning model optimized for programming and math with configurable ef
  category: 'reasoning',
  capabilities: ['chat', 'reasoning', 'streaming'],
  contextWindow: 200000,
  maxOutput: 100000,
  inputModalities: ['text'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.0011,
    outputCostPer1k: 0.0044,
    markup: MARKUP,
```

```
      billedInputPer1k: 0.0011 * MARKUP,
      billedOutputPer1k: 0.0044 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-01-31',
      family: 'o-series',
      version: 'o3-mini',
      reasoningModel: true,
      reasoningEffort: ['low', 'medium', 'high'],
    },
  },
  // Legacy GPT-4o (still available for audio)
  {
    id: 'openai-gpt-4o',
    modelId: 'gpt-4o-2024-08-06',
    litellmId: 'gpt-4o-2024-08-06',
    name: 'gpt-4o',
    displayName: 'GPT-4o',
    description: 'Multimodal model with native audio input/output capabilities',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'audio', 'function_calling', 'json_mode', 'streaming'],
    contextWindow: 128000,
    maxOutput: 16384,
    inputModalities: ['text', 'image', 'audio'],
    outputModalities: ['text', 'audio'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.0025,
      outputCostPer1k: 0.01,
      markup: MARKUP,
      billedInputPer1k: 0.0025 * MARKUP,
      billedOutputPer1k: 0.01 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-08-06',
      family: 'gpt-4o',
      version: '4o',
      legacy: true,
      supportsAudio: true,
    },
  },
];

export const OPENAI_PROVIDER: ExternalProvider = {
  id: 'openai',
  name: 'openai',
  displayName: 'OpenAI',
  category: 'text_generation',
```

```
  description: 'Leading AI lab providing GPT-5, GPT-4.1, and O-series reasoning models',
  website: 'https://openai.com',
  apiBaseUrl: 'https://api.openai.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/openai',
  enabled: true,
  regions: ['us-east-1', 'eu-west-1'],
  models: OPENAI_MODELS,
  rateLimit: { requestsPerMinute: 10000, tokensPerMinute: 10000000 },
  features: ['streaming', 'function_calling', 'vision', 'audio', 'json_mode', 'batch_api', 'ag
  compliance: ['SOC2', 'GDPR', 'HIPAA'],
  metadata: {
    foundedYear: 2015,
    headquarters: 'San Francisco, CA',
    totalModels: OPENAI_MODELS.length,
    lastUpdated: '2024-12-01',
  },
};

// ============================================================================
// ANTHROPIC - Claude 4.5, 4, and 3.5 Series
// ============================================================================

const ANTHROPIC_MODELS: ExternalModel[] = [
  // Claude 4.5 Series (Latest - November 2024)
  {
    id: 'anthropic-claude-opus-4-5',
    modelId: 'claude-opus-4-5-20251101',
    litellmId: 'anthropic/claude-opus-4-5-20251101',
    name: 'claude-opus-4.5',
    displayName: 'Claude Opus 4.5',
    description: 'Most intelligent model for coding, agents, and computer use',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'tool_use', 'computer_use', 'extended_thinking', 'streami
    contextWindow: 200000,
    maxOutput: 64000,
    inputModalities: ['text', 'image', 'pdf'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.005,
      outputCostPer1k: 0.025,
      cachedInputCostPer1k: 0.0005,
      markup: MARKUP,
      billedInputPer1k: 0.005 * MARKUP,
      billedOutputPer1k: 0.025 * MARKUP,
    },
    metadata: {
```

```
    releaseDate: '2024-11-01',
    family: 'claude-4.5',
    version: 'opus',
    isLatest: true,
    supportsBatch: true,
    batchDiscount: 0.5,
    bedrockId: 'anthropic.claude-opus-4-5-20251101-v1:0',
    vertexId: 'claude-opus-4-5@20251101',
  },
},
{
  id: 'anthropic-claude-sonnet-4-5',
  modelId: 'claude-sonnet-4-5-20250929',
  litellmId: 'anthropic/claude-sonnet-4-5-20250929',
  name: 'claude-sonnet-4.5',
  displayName: 'Claude Sonnet 4.5',
  description: 'Best balance of intelligence, speed, and cost - recommended for most tasks',
  category: 'text_generation',
  capabilities: ['chat', 'vision', 'tool_use', 'computer_use', 'extended_thinking', 'streamin
  contextWindow: 200000,
  maxOutput: 64000,
  inputModalities: ['text', 'image', 'pdf'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.003,
    outputCostPer1k: 0.015,
    cachedInputCostPer1k: 0.0003,
    longContextInputPer1k: 0.006,
    longContextOutputPer1k: 0.0225,
    markup: MARKUP,
    billedInputPer1k: 0.003 * MARKUP,
    billedOutputPer1k: 0.015 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-09-29',
    family: 'claude-4.5',
    version: 'sonnet',
    isLatest: true,
    isRecommended: true,
    supportsBatch: true,
    batchDiscount: 0.5,
    betaContextWindow: 1000000,
    bedrockId: 'anthropic.claude-sonnet-4-5-20250929-v1:0',
    vertexId: 'claude-sonnet-4-5@20250929',
  },
},
{
```

```
    id: 'anthropic-claude-haiku-4-5',
    modelId: 'claude-haiku-4-5-20251001',
    litellmId: 'anthropic/claude-haiku-4-5-20251001',
    name: 'claude-haiku-4.5',
    displayName: 'Claude Haiku 4.5',
    description: 'Fastest Claude with near-frontier performance for high-volume tasks',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'tool_use', 'streaming', 'batch'],
    contextWindow: 200000,
    maxOutput: 64000,
    inputModalities: ['text', 'image', 'pdf'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.001,
      outputCostPer1k: 0.005,
      cachedInputCostPer1k: 0.0001,
      markup: MARKUP,
      billedInputPer1k: 0.001 * MARKUP,
      billedOutputPer1k: 0.005 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-10-01',
      family: 'claude-4.5',
      version: 'haiku',
      isLatest: true,
      supportsBatch: true,
      batchDiscount: 0.5,
      bedrockId: 'anthropic.claude-haiku-4-5-20251001-v1:0',
      vertexId: 'claude-haiku-4-5@20251001',
    },
  },
  // Claude 4 Series
  {
    id: 'anthropic-claude-opus-4',
    modelId: 'claude-opus-4-20250514',
    litellmId: 'anthropic/claude-opus-4-20250514',
    name: 'claude-opus-4',
    displayName: 'Claude Opus 4',
    description: 'Previous flagship for complex analysis and extended tasks',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'tool_use', 'computer_use', 'extended_thinking', 'streamin
    contextWindow: 200000,
    maxOutput: 64000,
    inputModalities: ['text', 'image', 'pdf'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
```

```
    inputCostPer1k: 0.015,
    outputCostPer1k: 0.075,
    markup: MARKUP,
    billedInputPer1k: 0.015 * MARKUP,
    billedOutputPer1k: 0.075 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-05-14',
    family: 'claude-4',
    version: 'opus',
    bedrockId: 'anthropic.claude-opus-4-20250514-v1:0',
    vertexId: 'claude-opus-4@20250514',
  },
},
{
  id: 'anthropic-claude-sonnet-4',
  modelId: 'claude-sonnet-4-20250514',
  litellmId: 'anthropic/claude-sonnet-4-20250514',
  name: 'claude-sonnet-4',
  displayName: 'Claude Sonnet 4',
  description: 'Balanced performance and speed for general use',
  category: 'text_generation',
  capabilities: ['chat', 'vision', 'tool_use', 'computer_use', 'extended_thinking', 'streamin
  contextWindow: 200000,
  maxOutput: 64000,
  inputModalities: ['text', 'image', 'pdf'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.003,
    outputCostPer1k: 0.015,
    markup: MARKUP,
    billedInputPer1k: 0.003 * MARKUP,
    billedOutputPer1k: 0.015 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-05-14',
    family: 'claude-4',
    version: 'sonnet',
    bedrockId: 'anthropic.claude-sonnet-4-20250514-v1:0',
    vertexId: 'claude-sonnet-4@20250514',
  },
},
// Claude 3.5 Series (Legacy)
{
  id: 'anthropic-claude-haiku-35',
  modelId: 'claude-3-5-haiku-20241022',
  litellmId: 'anthropic/claude-3-5-haiku-20241022',
```

```
    name: 'claude-3.5-haiku',
    displayName: 'Claude 3.5 Haiku',
    description: 'Fast, affordable legacy model for high-volume tasks',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'tool_use', 'streaming'],
    contextWindow: 200000,
    maxOutput: 8192,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.0008,
      outputCostPer1k: 0.004,
      markup: MARKUP,
      billedInputPer1k: 0.0008 * MARKUP,
      billedOutputPer1k: 0.004 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-10-22',
      family: 'claude-3.5',
      version: 'haiku',
      legacy: true,
    },
  },
];

export const ANTHROPIC_PROVIDER: ExternalProvider = {
  id: 'anthropic',
  name: 'anthropic',
  displayName: 'Anthropic',
  category: 'text_generation',
  description: 'AI safety company providing Claude 4.5 series - the most intelligent AI assista
  website: 'https://anthropic.com',
  apiBaseUrl: 'https://api.anthropic.com/v1',
  authType: 'api_key',
  secretName: 'radiant/providers/anthropic',
  enabled: true,
  regions: ['us-east-1', 'eu-west-1'],
  models: ANTHROPIC_MODELS,
  rateLimit: { requestsPerMinute: 4000, tokensPerMinute: 400000 },
  features: ['streaming', 'tool_use', 'vision', 'extended_thinking', 'batch_api', 'computer_use
  compliance: ['SOC2', 'GDPR', 'HIPAA'],
  metadata: {
    foundedYear: 2021,
    headquarters: 'San Francisco, CA',
    totalModels: ANTHROPIC_MODELS.length,
    lastUpdated: '2024-11-01',
    cloudPartners: ['AWS Bedrock', 'Google Vertex AI'],
```

```
  },
};

// ============================================================================
// GOOGLE - Gemini 3, 2.5, and 2.0 Series
// ============================================================================

const GOOGLE_MODELS: ExternalModel[] = [
  // Gemini 3 Series (Preview - December 2024)
  {
    id: 'google-gemini-3-pro',
    modelId: 'gemini-3-pro-preview',
    litellmId: 'gemini/gemini-3-pro-preview',
    name: 'gemini-3-pro',
    displayName: 'Gemini 3 Pro',
    description: 'Latest flagship with advanced reasoning and native 1M context',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'audio', 'video', 'function_calling', 'thinking', 'groundi
    contextWindow: 1000000,
    maxOutput: 65536,
    inputModalities: ['text', 'image', 'audio', 'video', 'pdf'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.002,
      outputCostPer1k: 0.012,
      longContextInputPer1k: 0.004,
      longContextOutputPer1k: 0.018,
      markup: MARKUP,
      billedInputPer1k: 0.002 * MARKUP,
      billedOutputPer1k: 0.012 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-12-01',
      family: 'gemini-3',
      version: 'pro',
      isPreview: true,
      isLatest: true,
    },
  },
  {
    id: 'google-gemini-3-flash',
    modelId: 'gemini-3-flash-preview',
    litellmId: 'gemini/gemini-3-flash-preview',
    name: 'gemini-3-flash',
    displayName: 'Gemini 3 Flash',
    description: 'Fast Gemini 3 variant for high-volume multimodal tasks',
    category: 'text_generation',
```

```
    capabilities: ['chat', 'vision', 'audio', 'video', 'function_calling', 'thinking', 'streami
    contextWindow: 1000000,
    maxOutput: 65536,
    inputModalities: ['text', 'image', 'audio', 'video'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.0005,
      outputCostPer1k: 0.003,
      markup: MARKUP,
      billedInputPer1k: 0.0005 * MARKUP,
      billedOutputPer1k: 0.003 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-12-01',
      family: 'gemini-3',
      version: 'flash',
      isPreview: true,
    },
  },
  // Gemini 2.5 Series (Stable)
  {
    id: 'google-gemini-25-pro',
    modelId: 'gemini-2.5-pro',
    litellmId: 'gemini/gemini-2.5-pro',
    name: 'gemini-2.5-pro',
    displayName: 'Gemini 2.5 Pro',
    description: 'Stable flagship with thinking mode and 1M context',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'audio', 'video', 'function_calling', 'thinking', 'groundi
    contextWindow: 1000000,
    maxOutput: 65536,
    inputModalities: ['text', 'image', 'audio', 'video', 'pdf'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.00125,
      outputCostPer1k: 0.01,
      longContextInputPer1k: 0.0025,
      longContextOutputPer1k: 0.015,
      markup: MARKUP,
      billedInputPer1k: 0.00125 * MARKUP,
      billedOutputPer1k: 0.01 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-09-01',
      family: 'gemini-2.5',
      version: 'pro',
```

```
    isLatest: true,
    isRecommended: true,
  },
},
{
  id: 'google-gemini-25-flash',
  modelId: 'gemini-2.5-flash',
  litellmId: 'gemini/gemini-2.5-flash',
  name: 'gemini-2.5-flash',
  displayName: 'Gemini 2.5 Flash',
  description: 'Fast multimodal with thinking capabilities',
  category: 'text_generation',
  capabilities: ['chat', 'vision', 'audio', 'video', 'function_calling', 'thinking', 'stream
  contextWindow: 1000000,
  maxOutput: 65536,
  inputModalities: ['text', 'image', 'audio', 'video'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.0003,
    outputCostPer1k: 0.0025,
    markup: MARKUP,
    billedInputPer1k: 0.0003 * MARKUP,
    billedOutputPer1k: 0.0025 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-09-01',
    family: 'gemini-2.5',
    version: 'flash',
  },
},
{
  id: 'google-gemini-25-flash-lite',
  modelId: 'gemini-2.5-flash-lite',
  litellmId: 'gemini/gemini-2.5-flash-lite',
  name: 'gemini-2.5-flash-lite',
  displayName: 'Gemini 2.5 Flash-Lite',
  description: 'Ultra cost-efficient for high-volume processing',
  category: 'text_generation',
  capabilities: ['chat', 'vision', 'function_calling', 'streaming'],
  contextWindow: 1000000,
  maxOutput: 65536,
  inputModalities: ['text', 'image'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.0001,
    outputCostPer1k: 0.0004,
```

```
      markup: MARKUP,
      billedInputPer1k: 0.0001 * MARKUP,
      billedOutputPer1k: 0.0004 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-09-01',
      family: 'gemini-2.5',
      version: 'flash-lite',
    },
  },
  // Gemini 2.0 Series
  {
    id: 'google-gemini-2-flash',
    modelId: 'gemini-2.0-flash-001',
    litellmId: 'gemini/gemini-2.0-flash-001',
    name: 'gemini-2.0-flash',
    displayName: 'Gemini 2.0 Flash',
    description: 'Fast multimodal with real-time capabilities',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'audio', 'video', 'function_calling', 'streaming'],
    contextWindow: 1000000,
    maxOutput: 8192,
    inputModalities: ['text', 'image', 'audio', 'video'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.0001,
      outputCostPer1k: 0.0004,
      markup: MARKUP,
      billedInputPer1k: 0.0001 * MARKUP,
      billedOutputPer1k: 0.0004 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-06-01',
      family: 'gemini-2.0',
      version: 'flash',
      hasFreeVersion: true,
    },
  },
];

export const GOOGLE_PROVIDER: ExternalProvider = {
  id: 'google',
  name: 'google',
  displayName: 'Google AI',
  category: 'text_generation',
  description: 'Google AI providing Gemini 3, 2.5, and 2.0 models with native 1M context',
  website: 'https://ai.google.dev',
```

```
  apiBaseUrl: 'https://generativelanguage.googleapis.com/v1beta',
  authType: 'api_key',
  secretName: 'radiant/providers/google',
  enabled: true,
  regions: ['us-east-1', 'eu-west-1', 'ap-northeast-1'],
  models: GOOGLE_MODELS,
  rateLimit: { requestsPerMinute: 1000, tokensPerMinute: 4000000 },
  features: ['streaming', 'function_calling', 'vision', 'audio', 'video', 'thinking', 'groundi
  compliance: ['SOC2', 'GDPR', 'HIPAA'],
  metadata: {
    foundedYear: 1998,
    headquarters: 'Mountain View, CA',
    totalModels: GOOGLE_MODELS.length,
    lastUpdated: '2024-12-01',
    cloudPlatform: 'Google Vertex AI',
  },
};

// ============================================================================
// XAI (GROK) - Grok 4 and 3 Series
// ============================================================================

const XAI_MODELS: ExternalModel[] = [
  // Grok 4 Series (Latest)
  {
    id: 'xai-grok-4',
    modelId: 'grok-4-0709',
    litellmId: 'xai/grok-4-0709',
    name: 'grok-4',
    displayName: 'Grok 4',
    description: 'xAI flagship with advanced reasoning capabilities',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'reasoning', 'function_calling', 'streaming'],
    contextWindow: 256000,
    maxOutput: 256000,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.003,
      outputCostPer1k: 0.015,
      markup: MARKUP,
      billedInputPer1k: 0.003 * MARKUP,
      billedOutputPer1k: 0.015 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-07-09',
      family: 'grok-4',
```

```
    version: '4',
    isLatest: true,
  },
},
{
  id: 'xai-grok-4-1-fast',
  modelId: 'grok-4-1-fast-reasoning',
  litellmId: 'xai/grok-4-1-fast-reasoning',
  name: 'grok-4.1-fast',
  displayName: 'Grok 4.1 Fast',
  description: 'Best value: near-flagship performance at 1/15th price with 2M context',
  category: 'text_generation',
  capabilities: ['chat', 'vision', 'reasoning', 'function_calling', 'streaming', 'tool_calli
  contextWindow: 2000000,
  maxOutput: 2000000,
  inputModalities: ['text', 'image'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.0002,
    outputCostPer1k: 0.0005,
    markup: MARKUP,
    billedInputPer1k: 0.0002 * MARKUP,
    billedOutputPer1k: 0.0005 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-11-01',
    family: 'grok-4.1',
    version: 'fast-reasoning',
    isLatest: true,
    isRecommended: true,
    bestValue: true,
  },
},
{
  id: 'xai-grok-4-fast',
  modelId: 'grok-4-fast-reasoning',
  litellmId: 'xai/grok-4-fast-reasoning',
  name: 'grok-4-fast',
  displayName: 'Grok 4 Fast Reasoning',
  description: 'Fast reasoning with massive 2M context window',
  category: 'reasoning',
  capabilities: ['chat', 'vision', 'reasoning', 'function_calling', 'streaming'],
  contextWindow: 2000000,
  maxOutput: 2000000,
  inputModalities: ['text', 'image'],
  outputModalities: ['text'],
  pricing: {
```

```
    type: 'per_token',
    inputCostPer1k: 0.0002,
    outputCostPer1k: 0.0005,
    markup: MARKUP,
    billedInputPer1k: 0.0002 * MARKUP,
    billedOutputPer1k: 0.0005 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-09-01',
    family: 'grok-4',
    version: 'fast-reasoning',
  },
},
// Grok 3 Series
{
  id: 'xai-grok-3',
  modelId: 'grok-3',
  litellmId: 'xai/grok-3',
  name: 'grok-3',
  displayName: 'Grok 3',
  description: 'Previous flagship with real-time X/Twitter knowledge',
  category: 'text_generation',
  capabilities: ['chat', 'vision', 'function_calling', 'streaming'],
  contextWindow: 131072,
  maxOutput: 131072,
  inputModalities: ['text', 'image'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.003,
    outputCostPer1k: 0.015,
    markup: MARKUP,
    billedInputPer1k: 0.003 * MARKUP,
    billedOutputPer1k: 0.015 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-03-01',
    family: 'grok-3',
    version: '3',
  },
},
{
  id: 'xai-grok-3-mini',
  modelId: 'grok-3-mini',
  litellmId: 'xai/grok-3-mini',
  name: 'grok-3-mini',
  displayName: 'Grok 3 Mini',
  description: 'Smaller, faster Grok for everyday tasks',
```

```
    category: 'text_generation',
    capabilities: ['chat', 'function_calling', 'streaming'],
    contextWindow: 131072,
    maxOutput: 16384,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.0003,
      outputCostPer1k: 0.0005,
      markup: MARKUP,
      billedInputPer1k: 0.0003 * MARKUP,
      billedOutputPer1k: 0.0005 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-03-01',
      family: 'grok-3',
      version: 'mini',
    },
  },
  // Grok 2 Series
  {
    id: 'xai-grok-2',
    modelId: 'grok-2-1212',
    litellmId: 'xai/grok-2-1212',
    name: 'grok-2',
    displayName: 'Grok 2',
    description: 'Stable previous generation model',
    category: 'text_generation',
    capabilities: ['chat', 'vision', 'function_calling', 'streaming'],
    contextWindow: 131072,
    maxOutput: 131072,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.002,
      outputCostPer1k: 0.01,
      markup: MARKUP,
      billedInputPer1k: 0.002 * MARKUP,
      billedOutputPer1k: 0.01 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-12-12',
      family: 'grok-2',
      version: '2',
      legacy: true,
    },
```

```
  },
];

export const XAI_PROVIDER: ExternalProvider = {
  id: 'xai',
  name: 'xai',
  displayName: 'xAI (Grok)',
  category: 'text_generation',
  description: 'xAI providing Grok 4 series with industry-leading 2M context and competitive pr
  website: 'https://x.ai',
  apiBaseUrl: 'https://api.x.ai/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/xai',
  enabled: true,
  regions: ['us-east-1'],
  models: XAI_MODELS,
  rateLimit: { requestsPerMinute: 1000, tokensPerMinute: 1000000 },
  features: ['streaming', 'function_calling', 'vision', 'reasoning', 'web_search', 'x_search',
  compliance: ['SOC2'],
  metadata: {
    foundedYear: 2023,
    headquarters: 'San Francisco, CA',
    totalModels: XAI_MODELS.length,
    lastUpdated: '2024-11-01',
    serverTools: ['web_search', 'x_search', 'code_execution', 'document_search'],
    toolCallCost: 0.005,
  },
};

// ============================================================================
// DEEPSEEK - V3 Models (Best Value in Market)
// ============================================================================

const DEEPSEEK_MODELS: ExternalModel[] = [
  {
    id: 'deepseek-chat',
    modelId: 'deepseek-chat',
    litellmId: 'deepseek/deepseek-chat',
    name: 'deepseek-chat',
    displayName: 'DeepSeek Chat (V3.2)',
    description: 'Best value in market: frontier performance at 90% lower cost',
    category: 'text_generation',
    capabilities: ['chat', 'function_calling', 'json_mode', 'streaming', 'prefix_completion'],
    contextWindow: 128000,
    maxOutput: 8192,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
```

```
    type: 'per_token',
    inputCostPer1k: 0.00028,
    outputCostPer1k: 0.00042,
    cachedInputCostPer1k: 0.000028,
    markup: MARKUP,
    billedInputPer1k: 0.00028 * MARKUP,
    billedOutputPer1k: 0.00042 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-12-01',
    family: 'deepseek-v3',
    version: '3.2',
    isLatest: true,
    isRecommended: true,
    bestValue: true,
    openAICompatible: true,
  },
},
{
  id: 'deepseek-reasoner',
  modelId: 'deepseek-reasoner',
  litellmId: 'deepseek/deepseek-reasoner',
  name: 'deepseek-reasoner',
  displayName: 'DeepSeek R1',
  description: 'Advanced reasoning model rivaling O1 at fraction of cost',
  category: 'reasoning',
  capabilities: ['chat', 'reasoning', 'function_calling', 'streaming', 'thinking_in_tool_use
  contextWindow: 128000,
  maxOutput: 64000,
  inputModalities: ['text'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.00028,
    outputCostPer1k: 0.00042,
    cachedInputCostPer1k: 0.000028,
    markup: MARKUP,
    billedInputPer1k: 0.00028 * MARKUP,
    billedOutputPer1k: 0.00042 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-12-01',
    family: 'deepseek-r1',
    version: 'r1',
    isLatest: true,
    reasoningModel: true,
    openAICompatible: true,
  },
```

```
  },
];

export const DEEPSEEK_PROVIDER: ExternalProvider = {
  id: 'deepseek',
  name: 'deepseek',
  displayName: 'DeepSeek',
  category: 'text_generation',
  description: 'Most cost-effective frontier AI: 90% cheaper than competitors with comparable c
  website: 'https://deepseek.com',
  apiBaseUrl: 'https://api.deepseek.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/deepseek',
  enabled: true,
  regions: ['global'],
  models: DEEPSEEK_MODELS,
  rateLimit: { requestsPerMinute: 1000 },
  features: ['streaming', 'function_calling', 'reasoning', 'json_mode', 'context_caching', 'pre
  compliance: [],
  metadata: {
    foundedYear: 2023,
    headquarters: 'Hangzhou, China',
    totalModels: DEEPSEEK_MODELS.length,
    lastUpdated: '2024-12-01',
    openAISDKCompatible: true,
    bestValueProvider: true,
  },
};

// ============================================================================
// MISTRAL - European Frontier Models
// ============================================================================

const MISTRAL_MODELS: ExternalModel[] = [
  {
    id: 'mistral-large',
    modelId: 'mistral-large-2411',
    litellmId: 'mistral/mistral-large-2411',
    name: 'mistral-large',
    displayName: 'Mistral Large 2.1',
    description: 'European flagship for complex reasoning and enterprise tasks',
    category: 'text_generation',
    capabilities: ['chat', 'function_calling', 'json_mode', 'streaming'],
    contextWindow: 128000,
    maxOutput: 4096,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
```

```
    type: 'per_token',
    inputCostPer1k: 0.002,
    outputCostPer1k: 0.006,
    markup: MARKUP,
    billedInputPer1k: 0.002 * MARKUP,
    billedOutputPer1k: 0.006 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-11-01',
    family: 'mistral-large',
    version: '2.1',
    isLatest: true,
  },
},
{
  id: 'mistral-medium',
  modelId: 'mistral-medium-2505',
  litellmId: 'mistral/mistral-medium-2505',
  name: 'mistral-medium',
  displayName: 'Mistral Medium 3',
  description: 'Balanced European model for general enterprise tasks',
  category: 'text_generation',
  capabilities: ['chat', 'function_calling', 'json_mode', 'streaming'],
  contextWindow: 128000,
  maxOutput: 4096,
  inputModalities: ['text'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.0004,
    outputCostPer1k: 0.002,
    markup: MARKUP,
    billedInputPer1k: 0.0004 * MARKUP,
    billedOutputPer1k: 0.002 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-05-01',
    family: 'mistral-medium',
    version: '3',
  },
},
{
  id: 'mistral-small',
  modelId: 'mistral-small-2409',
  litellmId: 'mistral/mistral-small-2409',
  name: 'mistral-small',
  displayName: 'Mistral Small 2',
  description: 'Cost-efficient for everyday tasks',
```

```
    category: 'text_generation',
    capabilities: ['chat', 'function_calling', 'json_mode', 'streaming'],
    contextWindow: 128000,
    maxOutput: 4096,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.0001,
      outputCostPer1k: 0.0003,
      markup: MARKUP,
      billedInputPer1k: 0.0001 * MARKUP,
      billedOutputPer1k: 0.0003 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-09-01',
      family: 'mistral-small',
      version: '2',
    },
  },
  {
    id: 'mistral-pixtral-large',
    modelId: 'pixtral-large-2411',
    litellmId: 'mistral/pixtral-large-2411',
    name: 'pixtral-large',
    displayName: 'Pixtral Large',
    description: 'Vision model for OCR, document analysis, and image understanding',
    category: 'vision',
    capabilities: ['chat', 'vision', 'function_calling', 'streaming'],
    contextWindow: 128000,
    maxOutput: 4096,
    inputModalities: ['text', 'image'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.002,
      outputCostPer1k: 0.006,
      markup: MARKUP,
      billedInputPer1k: 0.002 * MARKUP,
      billedOutputPer1k: 0.006 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-11-01',
      family: 'pixtral',
      version: 'large',
    },
  },
  {
```

```
  id: 'mistral-codestral',
  modelId: 'codestral-2501',
  litellmId: 'mistral/codestral-2501',
  name: 'codestral',
  displayName: 'Codestral',
  description: 'Specialized code model supporting 80+ languages with fill-in-the-middle',
  category: 'code',
  capabilities: ['chat', 'code_completion', 'fim', 'streaming'],
  contextWindow: 256000,
  maxOutput: 8192,
  inputModalities: ['text'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.0002,
    outputCostPer1k: 0.0006,
    markup: MARKUP,
    billedInputPer1k: 0.0002 * MARKUP,
    billedOutputPer1k: 0.0006 * MARKUP,
  },
  metadata: {
    releaseDate: '2024-01-01',
    family: 'codestral',
    version: '2501',
    supportedLanguages: 80,
  },
},
{
  id: 'mistral-magistral-medium',
  modelId: 'magistral-medium-2509',
  litellmId: 'mistral/magistral-medium-2509',
  name: 'magistral-medium',
  displayName: 'Magistral Medium',
  description: 'Reasoning model with configurable thinking effort levels',
  category: 'reasoning',
  capabilities: ['chat', 'reasoning', 'function_calling', 'streaming'],
  contextWindow: 128000,
  maxOutput: 8192,
  inputModalities: ['text'],
  outputModalities: ['text'],
  pricing: {
    type: 'per_token',
    inputCostPer1k: 0.001,
    outputCostPer1k: 0.003,
    markup: MARKUP,
    billedInputPer1k: 0.001 * MARKUP,
    billedOutputPer1k: 0.003 * MARKUP,
  },
```

```
    metadata: {
      releaseDate: '2024-09-01',
      family: 'magistral',
      version: 'medium',
      reasoningModel: true,
      reasoningEffort: ['low', 'medium', 'high'],
    },
  },
];

export const MISTRAL_PROVIDER: ExternalProvider = {
  id: 'mistral',
  name: 'mistral',
  displayName: 'Mistral AI',
  category: 'text_generation',
  description: 'European frontier models with GDPR compliance and specialized coding/vision va
  website: 'https://mistral.ai',
  apiBaseUrl: 'https://api.mistral.ai/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/mistral',
  enabled: true,
  regions: ['eu-west-1', 'us-east-1'],
  models: MISTRAL_MODELS,
  rateLimit: { requestsPerMinute: 1000 },
  features: ['streaming', 'function_calling', 'vision', 'code_completion', 'fim', 'reasoning',
  compliance: ['GDPR', 'SOC2'],
  metadata: {
    foundedYear: 2023,
    headquarters: 'Paris, France',
    totalModels: MISTRAL_MODELS.length,
    lastUpdated: '2024-11-01',
    europeanProvider: true,
  },
};

// ============================================================================
// COHERE - Enterprise RAG Models
// ============================================================================

const COHERE_MODELS: ExternalModel[] = [
  {
    id: 'cohere-command-r-plus',
    modelId: 'command-r-plus-08-2024',
    litellmId: 'cohere/command-r-plus-08-2024',
    name: 'command-r-plus',
    displayName: 'Command R+',
    description: 'Enterprise model optimized for RAG and complex tool use',
    category: 'text_generation',
```

```
    capabilities: ['chat', 'rag', 'function_calling', 'streaming'],
    contextWindow: 128000,
    maxOutput: 4096,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.0025,
      outputCostPer1k: 0.01,
      markup: MARKUP,
      billedInputPer1k: 0.0025 * MARKUP,
      billedOutputPer1k: 0.01 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-08-01',
      family: 'command-r',
      version: 'plus',
      isLatest: true,
    },
  },
  {
    id: 'cohere-command-r',
    modelId: 'command-r-08-2024',
    litellmId: 'cohere/command-r-08-2024',
    name: 'command-r',
    displayName: 'Command R',
    description: 'Balanced model for general enterprise tasks',
    category: 'text_generation',
    capabilities: ['chat', 'rag', 'function_calling', 'streaming'],
    contextWindow: 128000,
    maxOutput: 4096,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.00015,
      outputCostPer1k: 0.0006,
      markup: MARKUP,
      billedInputPer1k: 0.00015 * MARKUP,
      billedOutputPer1k: 0.0006 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-08-01',
      family: 'command-r',
      version: 'base',
    },
  },
  {
```

```
    id: 'cohere-embed-v4',
    modelId: 'embed-v4.0',
    litellmId: 'cohere/embed-v4.0',
    name: 'embed-v4',
    displayName: 'Embed V4',
    description: 'State-of-the-art embeddings for semantic search and RAG',
    category: 'embeddings',
    capabilities: ['embeddings'],
    contextWindow: 128000,
    maxOutput: 0,
    inputModalities: ['text'],
    outputModalities: ['embeddings'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.0001,
      outputCostPer1k: 0,
      markup: MARKUP,
      billedInputPer1k: 0.0001 * MARKUP,
      billedOutputPer1k: 0,
    },
    metadata: {
      releaseDate: '2024-10-01',
      family: 'embed',
      version: 'v4',
      dimensions: 1024,
    },
  },
];

export const COHERE_PROVIDER: ExternalProvider = {
  id: 'cohere',
  name: 'cohere',
  displayName: 'Cohere',
  category: 'text_generation',
  description: 'Enterprise AI specialized in RAG and semantic search',
  website: 'https://cohere.com',
  apiBaseUrl: 'https://api.cohere.ai/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/cohere',
  enabled: true,
  regions: ['us-east-1', 'eu-west-1'],
  models: COHERE_MODELS,
  rateLimit: { requestsPerMinute: 1000 },
  features: ['streaming', 'function_calling', 'rag', 'embeddings', 'rerank'],
  compliance: ['SOC2', 'GDPR'],
  metadata: {
    foundedYear: 2019,
    headquarters: 'Toronto, Canada',
```

```
    totalModels: COHERE_MODELS.length,
    lastUpdated: '2024-10-01',
    enterpriseFocus: true,
  },
};

// ============================================================================
// PERPLEXITY - Search-Augmented Models
// ============================================================================

const PERPLEXITY_MODELS: ExternalModel[] = [
  {
    id: 'perplexity-sonar-huge',
    modelId: 'sonar-huge-online',
    litellmId: 'perplexity/sonar-huge-online',
    name: 'sonar-huge',
    displayName: 'Sonar Huge',
    description: 'Most capable search-augmented model with real-time web access',
    category: 'search',
    capabilities: ['chat', 'search', 'citations', 'streaming'],
    contextWindow: 200000,
    maxOutput: 8192,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.005,
      outputCostPer1k: 0.005,
      searchCostPerRequest: 0.005,
      markup: MARKUP,
      billedInputPer1k: 0.005 * MARKUP,
      billedOutputPer1k: 0.005 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-09-01',
      family: 'sonar',
      version: 'huge',
      isLatest: true,
      searchEnabled: true,
    },
  },
  {
    id: 'perplexity-sonar-pro',
    modelId: 'sonar-pro-online',
    litellmId: 'perplexity/sonar-pro-online',
    name: 'sonar-pro',
    displayName: 'Sonar Pro',
    description: 'Balanced search model for general research and fact-checking',
```

```
    category: 'search',
    capabilities: ['chat', 'search', 'citations', 'streaming'],
    contextWindow: 200000,
    maxOutput: 8192,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.003,
      outputCostPer1k: 0.015,
      searchCostPerRequest: 0.005,
      markup: MARKUP,
      billedInputPer1k: 0.003 * MARKUP,
      billedOutputPer1k: 0.015 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-09-01',
      family: 'sonar',
      version: 'pro',
      searchEnabled: true,
    },
  },
  {
    id: 'perplexity-sonar',
    modelId: 'sonar-online',
    litellmId: 'perplexity/sonar-online',
    name: 'sonar',
    displayName: 'Sonar',
    description: 'Fast search-augmented model for quick lookups',
    category: 'search',
    capabilities: ['chat', 'search', 'citations', 'streaming'],
    contextWindow: 127000,
    maxOutput: 8192,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: {
      type: 'per_token',
      inputCostPer1k: 0.001,
      outputCostPer1k: 0.001,
      searchCostPerRequest: 0.005,
      markup: MARKUP,
      billedInputPer1k: 0.001 * MARKUP,
      billedOutputPer1k: 0.001 * MARKUP,
    },
    metadata: {
      releaseDate: '2024-09-01',
      family: 'sonar',
      version: 'base',
```

```
      searchEnabled: true,
    },
  },
];

export const PERPLEXITY_PROVIDER: ExternalProvider = {
  id: 'perplexity',
  name: 'perplexity',
  displayName: 'Perplexity',
  category: 'search',
  description: 'Search-augmented AI with real-time web access and automatic citations',
  website: 'https://perplexity.ai',
  apiBaseUrl: 'https://api.perplexity.ai',
  authType: 'bearer',
  secretName: 'radiant/providers/perplexity',
  enabled: true,
  regions: ['us-east-1'],
  models: PERPLEXITY_MODELS,
  rateLimit: { requestsPerMinute: 100 },
  features: ['streaming', 'search', 'citations', 'web_access'],
  compliance: ['SOC2'],
  metadata: {
    foundedYear: 2022,
    headquarters: 'San Francisco, CA',
    totalModels: PERPLEXITY_MODELS.length,
    lastUpdated: '2024-09-01',
    searchProvider: true,
  },
};

// ============================================================================
// EXPORT ALL TEXT PROVIDERS
// ============================================================================

export const TEXT_PROVIDERS: ExternalProvider[] = [
  OPENAI_PROVIDER,
  ANTHROPIC_PROVIDER,
  GOOGLE_PROVIDER,
  XAI_PROVIDER,
  DEEPSEEK_PROVIDER,
  MISTRAL_PROVIDER,
  COHERE_PROVIDER,
  PERPLEXITY_PROVIDER,
];

/**
 * Image Generation Providers
 * DALL-E, Stability, Flux, Ideogram, Midjourney
```

```
 */

import { ExternalProvider, ExternalModel } from './index';

const MARKUP = 1.40;

// ============================================================================
// OPENAI (DALL-E)
// ============================================================================

const DALLE_MODELS: ExternalModel[] = [
  {
    id: 'openai-dall-e-3',
    modelId: 'dall-e-3',
    litellmId: 'dall-e-3',
    name: 'dall-e-3',
    displayName: 'DALL-E 3',
    description: 'Latest image generation with superior quality',
    category: 'image_generation',
    capabilities: ['text_to_image', 'hd', 'wide_format'],
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.04, markup: MARKUP },
  },
  {
    id: 'openai-dall-e-3-hd',
    modelId: 'dall-e-3-hd',
    litellmId: 'dall-e-3',
    name: 'dall-e-3-hd',
    displayName: 'DALL-E 3 HD',
    description: 'High-definition DALL-E 3 images',
    category: 'image_generation',
    capabilities: ['text_to_image', 'hd', 'wide_format'],
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.08, markup: MARKUP },
  },
];

export const DALLE_PROVIDER: ExternalProvider = {
  id: 'openai-images',
  name: 'openai-images',
  displayName: 'OpenAI Images',
  category: 'image_generation',
  description: 'OpenAI DALL-E image generation',
  website: 'https://openai.com',
  apiBaseUrl: 'https://api.openai.com/v1',
  authType: 'bearer',
```

```
  secretName: 'radiant/providers/openai',
  enabled: true,
  regions: ['us-east-1'],
  models: DALLE_MODELS,
  features: ['text_to_image', 'image_editing', 'variations'],
};

// ============================================================================
// STABILITY AI
// ============================================================================

const STABILITY_MODELS: ExternalModel[] = [
  {
    id: 'stability-sdxl',
    modelId: 'stable-diffusion-xl-1024-v1-0',
    litellmId: 'stability/stable-diffusion-xl-1024-v1-0',
    name: 'sdxl',
    displayName: 'Stable Diffusion XL',
    description: 'High-quality 1024px image generation',
    category: 'image_generation',
    capabilities: ['text_to_image', 'image_to_image'],
    inputModalities: ['text', 'image'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.002, markup: MARKUP },
  },
  {
    id: 'stability-sd3-large',
    modelId: 'sd3-large',
    litellmId: 'stability/sd3-large',
    name: 'sd3-large',
    displayName: 'Stable Diffusion 3 Large',
    description: 'Latest SD3 with improved quality',
    category: 'image_generation',
    capabilities: ['text_to_image'],
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.065, markup: MARKUP },
  },
  {
    id: 'stability-ultra',
    modelId: 'stable-image-ultra',
    litellmId: 'stability/stable-image-ultra',
    name: 'stable-image-ultra',
    displayName: 'Stable Image Ultra',
    description: 'Premium image generation with maximum quality',
    category: 'image_generation',
    capabilities: ['text_to_image'],
    inputModalities: ['text'],
```

```
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.08, markup: MARKUP },
  },
];

export const STABILITY_PROVIDER: ExternalProvider = {
  id: 'stability',
  name: 'stability',
  displayName: 'Stability AI',
  category: 'image_generation',
  description: 'Open-source focused image generation',
  website: 'https://stability.ai',
  apiBaseUrl: 'https://api.stability.ai/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/stability',
  enabled: true,
  regions: ['us-east-1'],
  models: STABILITY_MODELS,
  features: ['text_to_image', 'image_to_image', 'inpainting', 'upscaling'],
};

// ============================================================================
// FLUX (BLACK FOREST LABS)
// ============================================================================

const FLUX_MODELS: ExternalModel[] = [
  {
    id: 'flux-pro-11',
    modelId: 'flux-pro-1.1',
    litellmId: 'replicate/black-forest-labs/flux-1.1-pro',
    name: 'flux-pro-1.1',
    displayName: 'FLUX 1.1 Pro',
    description: 'State-of-the-art text-to-image with 6x faster generation',
    category: 'image_generation',
    capabilities: ['text_to_image'],
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.04, markup: MARKUP },
  },
  {
    id: 'flux-pro-ultra',
    modelId: 'flux-pro-1.1-ultra',
    litellmId: 'replicate/black-forest-labs/flux-1.1-pro-ultra',
    name: 'flux-pro-1.1-ultra',
    displayName: 'FLUX 1.1 Pro Ultra',
    description: 'Ultra-high resolution 4MP images',
    category: 'image_generation',
    capabilities: ['text_to_image', '4mp'],
```

```
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.06, markup: MARKUP },
  },
  {
    id: 'flux-schnell',
    modelId: 'flux-schnell',
    litellmId: 'replicate/black-forest-labs/flux-schnell',
    name: 'flux-schnell',
    displayName: 'FLUX Schnell',
    description: 'Fast, efficient image generation',
    category: 'image_generation',
    capabilities: ['text_to_image', 'fast'],
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.003, markup: MARKUP },
  },
];

export const FLUX_PROVIDER: ExternalProvider = {
  id: 'flux',
  name: 'flux',
  displayName: 'FLUX (Black Forest Labs)',
  category: 'image_generation',
  description: 'State-of-the-art open image generation',
  website: 'https://blackforestlabs.ai',
  apiBaseUrl: 'https://api.replicate.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/replicate',
  enabled: true,
  regions: ['us-east-1'],
  models: FLUX_MODELS,
  features: ['text_to_image', 'fast_generation', 'high_resolution'],
};

// ============================================================================
// IDEOGRAM
// ============================================================================

const IDEOGRAM_MODELS: ExternalModel[] = [
  {
    id: 'ideogram-v2',
    modelId: 'ideogram-v2',
    litellmId: 'ideogram/ideogram-v2',
    name: 'ideogram-v2',
    displayName: 'Ideogram V2',
    description: 'Advanced text rendering in images',
    category: 'image_generation',
```

```
    capabilities: ['text_to_image', 'text_rendering'],
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.08, markup: MARKUP },
  },
  {
    id: 'ideogram-v2-turbo',
    modelId: 'ideogram-v2-turbo',
    litellmId: 'ideogram/ideogram-v2-turbo',
    name: 'ideogram-v2-turbo',
    displayName: 'Ideogram V2 Turbo',
    description: 'Fast with good text rendering',
    category: 'image_generation',
    capabilities: ['text_to_image', 'text_rendering', 'fast'],
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.05, markup: MARKUP },
  },
];

export const IDEOGRAM_PROVIDER: ExternalProvider = {
  id: 'ideogram',
  name: 'ideogram',
  displayName: 'Ideogram',
  category: 'image_generation',
  description: 'Specialized in accurate text rendering',
  website: 'https://ideogram.ai',
  apiBaseUrl: 'https://api.ideogram.ai/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/ideogram',
  enabled: true,
  regions: ['us-east-1'],
  models: IDEOGRAM_MODELS,
  features: ['text_to_image', 'text_rendering', 'logos'],
};

// ============================================================================
// MIDJOURNEY
// ============================================================================

const MIDJOURNEY_MODELS: ExternalModel[] = [
  {
    id: 'midjourney-v6',
    modelId: 'midjourney-v6',
    litellmId: 'midjourney/v6',
    name: 'midjourney-v6',
    displayName: 'Midjourney V6',
    description: 'Premium artistic image generation',
```

```
    category: 'image_generation',
    capabilities: ['text_to_image', 'artistic'],
    inputModalities: ['text'],
    outputModalities: ['image'],
    pricing: { type: 'per_image', costPerImage: 0.10, markup: MARKUP },
  },
];

export const MIDJOURNEY_PROVIDER: ExternalProvider = {
  id: 'midjourney',
  name: 'midjourney',
  displayName: 'Midjourney',
  category: 'image_generation',
  description: 'Premium artistic image generation',
  website: 'https://midjourney.com',
  apiBaseUrl: 'https://api.mymidjourney.ai/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/midjourney',
  enabled: true,
  regions: ['us-east-1'],
  models: MIDJOURNEY_MODELS,
  features: ['text_to_image', 'artistic', 'premium_quality'],
};

export const IMAGE_PROVIDERS: ExternalProvider[] = [
  DALLE_PROVIDER,
  STABILITY_PROVIDER,
  FLUX_PROVIDER,
  IDEOGRAM_PROVIDER,
  MIDJOURNEY_PROVIDER,
];
```

packages/infrastructure/lib/config/providers/video.providers.ts

```
/**
 * Video Generation Providers
 * Runway, Luma, Pika, Kling, HailuoAI, Veo
 */

import { ExternalProvider, ExternalModel } from './index';

const MARKUP = 1.40;

// =============================================================================
// RUNWAY
// =============================================================================

const RUNWAY_MODELS: ExternalModel[] = [
```

```
  {
    id: 'runway-gen3-alpha-turbo',
    modelId: 'gen3a_turbo',
    litellmId: 'runway/gen3a_turbo',
    name: 'gen3-alpha-turbo',
    displayName: 'Gen-3 Alpha Turbo',
    description: 'Fast, high-quality video generation',
    category: 'video_generation',
    capabilities: ['text_to_video', 'image_to_video'],
    inputModalities: ['text', 'image'],
    outputModalities: ['video'],
    pricing: { type: 'per_second', costPerSecond: 0.05, markup: MARKUP },
  },
  {
    id: 'runway-gen3-alpha',
    modelId: 'gen3a',
    litellmId: 'runway/gen3a',
    name: 'gen3-alpha',
    displayName: 'Gen-3 Alpha',
    description: 'Maximum quality video generation',
    category: 'video_generation',
    capabilities: ['text_to_video', 'image_to_video'],
    inputModalities: ['text', 'image'],
    outputModalities: ['video'],
    pricing: { type: 'per_second', costPerSecond: 0.10, markup: MARKUP },
  },
];

export const RUNWAY_PROVIDER: ExternalProvider = {
  id: 'runway',
  name: 'runway',
  displayName: 'Runway',
  category: 'video_generation',
  description: 'Professional AI video generation tools',
  website: 'https://runway.ml',
  apiBaseUrl: 'https://api.runway.ml/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/runway',
  enabled: true,
  regions: ['us-east-1'],
  models: RUNWAY_MODELS,
  features: ['text_to_video', 'image_to_video', 'motion_brush'],
};

// ============================================================================
// LUMA AI
// ============================================================================
```

```
const LUMA_MODELS: ExternalModel[] = [
  {
    id: 'luma-dream-machine',
    modelId: 'dream-machine',
    litellmId: 'luma/dream-machine',
    name: 'dream-machine',
    displayName: 'Dream Machine',
    description: 'Fast, cinematic video generation',
    category: 'video_generation',
    capabilities: ['text_to_video', 'image_to_video'],
    inputModalities: ['text', 'image'],
    outputModalities: ['video'],
    pricing: { type: 'per_second', costPerSecond: 0.032, markup: MARKUP },
  },
  {
    id: 'luma-ray-2',
    modelId: 'ray-2',
    litellmId: 'luma/ray-2',
    name: 'ray-2',
    displayName: 'Ray 2',
    description: 'Latest model with improved realism',
    category: 'video_generation',
    capabilities: ['text_to_video', 'image_to_video', 'camera_motion'],
    inputModalities: ['text', 'image'],
    outputModalities: ['video'],
    pricing: { type: 'per_second', costPerSecond: 0.05, markup: MARKUP },
  },
];

export const LUMA_PROVIDER: ExternalProvider = {
  id: 'luma',
  name: 'luma',
  displayName: 'Luma AI',
  category: 'video_generation',
  description: 'Cinematic AI video generation',
  website: 'https://lumalabs.ai',
  apiBaseUrl: 'https://api.lumalabs.ai/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/luma',
  enabled: true,
  regions: ['us-east-1'],
  models: LUMA_MODELS,
  features: ['text_to_video', 'image_to_video', 'camera_control'],
};

// ============================================================================
// PIKA LABS
// ============================================================================
```

```typescript
const PIKA_MODELS: ExternalModel[] = [
  {
    id: 'pika-v2',
    modelId: 'pika-2.0',
    litellmId: 'pika/pika-2.0',
    name: 'pika-2.0',
    displayName: 'Pika 2.0',
    description: 'Creative video generation with effects',
    category: 'video_generation',
    capabilities: ['text_to_video', 'image_to_video', 'video_editing'],
    inputModalities: ['text', 'image'],
    outputModalities: ['video'],
    pricing: { type: 'per_second', costPerSecond: 0.02, markup: MARKUP },
  },
];

export const PIKA_PROVIDER: ExternalProvider = {
  id: 'pika',
  name: 'pika',
  displayName: 'Pika Labs',
  category: 'video_generation',
  description: 'Creative AI video platform',
  website: 'https://pika.art',
  apiBaseUrl: 'https://api.pika.art/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/pika',
  enabled: true,
  regions: ['us-east-1'],
  models: PIKA_MODELS,
  features: ['text_to_video', 'lip_sync', 'effects'],
};

// ============================================================================
// KLING AI
// ============================================================================

const KLING_MODELS: ExternalModel[] = [
  {
    id: 'kling-v15-pro',
    modelId: 'kling-v1.5-pro',
    litellmId: 'kling/kling-v1.5-pro',
    name: 'kling-v1.5-pro',
    displayName: 'Kling 1.5 Pro',
    description: 'High-quality Chinese video model',
    category: 'video_generation',
    capabilities: ['text_to_video', 'image_to_video'],
    inputModalities: ['text', 'image'],
```

```
    outputModalities: ['video'],
    pricing: { type: 'per_second', costPerSecond: 0.025, markup: MARKUP },
  },
];

export const KLING_PROVIDER: ExternalProvider = {
  id: 'kling',
  name: 'kling',
  displayName: 'Kling AI',
  category: 'video_generation',
  description: 'Chinese AI video generation',
  website: 'https://kling.kuaishou.com',
  apiBaseUrl: 'https://api.klingai.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/kling',
  enabled: true,
  regions: ['ap-southeast-1'],
  models: KLING_MODELS,
  features: ['text_to_video', 'image_to_video'],
};

// ============================================================================
// GOOGLE VEO
// ============================================================================

const VEO_MODELS: ExternalModel[] = [
  {
    id: 'google-veo-2',
    modelId: 'veo-2.0',
    litellmId: 'gemini/veo-2.0',
    name: 'veo-2.0',
    displayName: 'Veo 2',
    description: 'Google DeepMind video generation',
    category: 'video_generation',
    capabilities: ['text_to_video', 'image_to_video'],
    inputModalities: ['text', 'image'],
    outputModalities: ['video'],
    pricing: { type: 'per_second', costPerSecond: 0.04, markup: MARKUP },
  },
];

export const VEO_PROVIDER: ExternalProvider = {
  id: 'veo',
  name: 'veo',
  displayName: 'Google Veo',
  category: 'video_generation',
  description: 'Google DeepMind video model',
  website: 'https://deepmind.google',
```

```
  apiBaseUrl: 'https://generativelanguage.googleapis.com/v1beta',
  authType: 'api_key',
  secretName: 'radiant/providers/google',
  enabled: true,
  regions: ['us-east-1'],
  models: VEO_MODELS,
  features: ['text_to_video', 'image_to_video'],
};

export const VIDEO_PROVIDERS: ExternalProvider[] = [
  RUNWAY_PROVIDER,
  LUMA_PROVIDER,
  PIKA_PROVIDER,
  KLING_PROVIDER,
  VEO_PROVIDER,
];
```

packages/infrastructure/lib/config/providers/audio.providers.ts

```
/**
 * Audio/Speech Providers
 * OpenAI TTS/Whisper, ElevenLabs, Deepgram, AssemblyAI
 */

import { ExternalProvider, ExternalModel } from './index';

const MARKUP = 1.40;

// ============================================================================
// OPENAI AUDIO
// ============================================================================

const OPENAI_AUDIO_MODELS: ExternalModel[] = [
  {
    id: 'openai-tts-1',
    modelId: 'tts-1',
    litellmId: 'tts-1',
    name: 'tts-1',
    displayName: 'TTS-1',
    description: 'Fast text-to-speech',
    category: 'text_to_speech',
    capabilities: ['text_to_speech', 'multiple_voices'],
    inputModalities: ['text'],
    outputModalities: ['audio'],
    pricing: { type: 'per_token', inputCostPer1k: 0.015, markup: MARKUP, billedInputPer1k: 0.0
  },
  {
    id: 'openai-tts-1-hd',
```

```
    modelId: 'tts-1-hd',
    litellmId: 'tts-1-hd',
    name: 'tts-1-hd',
    displayName: 'TTS-1 HD',
    description: 'High-definition text-to-speech',
    category: 'text_to_speech',
    capabilities: ['text_to_speech', 'multiple_voices', 'hd'],
    inputModalities: ['text'],
    outputModalities: ['audio'],
    pricing: { type: 'per_token', inputCostPer1k: 0.030, markup: MARKUP, billedInputPer1k: 0.03
  },
  {
    id: 'openai-whisper-1',
    modelId: 'whisper-1',
    litellmId: 'whisper-1',
    name: 'whisper-1',
    displayName: 'Whisper',
    description: 'Speech recognition and transcription',
    category: 'speech_to_text',
    capabilities: ['transcription', 'translation', 'timestamps'],
    inputModalities: ['audio'],
    outputModalities: ['text'],
    pricing: { type: 'per_minute', costPerMinute: 0.006, markup: MARKUP },
  },
];

export const OPENAI_AUDIO_PROVIDER: ExternalProvider = {
  id: 'openai-audio',
  name: 'openai-audio',
  displayName: 'OpenAI Audio',
  category: 'audio_generation',
  description: 'OpenAI speech and audio models',
  website: 'https://openai.com',
  apiBaseUrl: 'https://api.openai.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/openai',
  enabled: true,
  regions: ['us-east-1'],
  models: OPENAI_AUDIO_MODELS,
  features: ['tts', 'stt', 'audio_understanding'],
};

// ============================================================================
// ELEVENLABS
// ============================================================================

const ELEVENLABS_MODELS: ExternalModel[] = [
  {
```

```
    id: 'elevenlabs-multilingual-v2',
    modelId: 'eleven_multilingual_v2',
    litellmId: 'elevenlabs/eleven_multilingual_v2',
    name: 'multilingual-v2',
    displayName: 'Multilingual V2',
    description: 'Multi-language voice synthesis',
    category: 'text_to_speech',
    capabilities: ['text_to_speech', 'multilingual', 'voice_cloning'],
    inputModalities: ['text'],
    outputModalities: ['audio'],
    pricing: { type: 'per_token', inputCostPer1k: 0.18, markup: MARKUP, billedInputPer1k: 0.18
  },
  {
    id: 'elevenlabs-turbo-v2-5',
    modelId: 'eleven_turbo_v2_5',
    litellmId: 'elevenlabs/eleven_turbo_v2_5',
    name: 'turbo-v2.5',
    displayName: 'Turbo V2.5',
    description: 'Fast, low-latency voice synthesis',
    category: 'text_to_speech',
    capabilities: ['text_to_speech', 'low_latency', 'streaming'],
    inputModalities: ['text'],
    outputModalities: ['audio'],
    pricing: { type: 'per_token', inputCostPer1k: 0.09, markup: MARKUP, billedInputPer1k: 0.09
  },
];

export const ELEVENLABS_PROVIDER: ExternalProvider = {
  id: 'elevenlabs',
  name: 'elevenlabs',
  displayName: 'ElevenLabs',
  category: 'text_to_speech',
  description: 'Premium voice synthesis and cloning',
  website: 'https://elevenlabs.io',
  apiBaseUrl: 'https://api.elevenlabs.io/v1',
  authType: 'api_key',
  secretName: 'radiant/providers/elevenlabs',
  enabled: true,
  regions: ['us-east-1'],
  models: ELEVENLABS_MODELS,
  features: ['voice_cloning', 'multilingual', 'streaming', 'sound_effects'],
};

// ============================================================================
// DEEPGRAM
// ============================================================================

const DEEPGRAM_MODELS: ExternalModel[] = [
```

```
  {
    id: 'deepgram-nova-2',
    modelId: 'nova-2',
    litellmId: 'deepgram/nova-2',
    name: 'nova-2',
    displayName: 'Nova-2',
    description: 'Industry-leading speech recognition',
    category: 'speech_to_text',
    capabilities: ['transcription', 'diarization', 'sentiment', 'streaming'],
    inputModalities: ['audio'],
    outputModalities: ['text'],
    pricing: { type: 'per_minute', costPerMinute: 0.0043, markup: MARKUP },
  },
  {
    id: 'deepgram-nova-2-medical',
    modelId: 'nova-2-medical',
    litellmId: 'deepgram/nova-2-medical',
    name: 'nova-2-medical',
    displayName: 'Nova-2 Medical',
    description: 'Medical terminology optimized',
    category: 'speech_to_text',
    capabilities: ['transcription', 'medical', 'hipaa'],
    inputModalities: ['audio'],
    outputModalities: ['text'],
    pricing: { type: 'per_minute', costPerMinute: 0.0079, markup: MARKUP },
  },
];

export const DEEPGRAM_PROVIDER: ExternalProvider = {
  id: 'deepgram',
  name: 'deepgram',
  displayName: 'Deepgram',
  category: 'speech_to_text',
  description: 'Enterprise speech recognition',
  website: 'https://deepgram.com',
  apiBaseUrl: 'https://api.deepgram.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/deepgram',
  enabled: true,
  regions: ['us-east-1'],
  models: DEEPGRAM_MODELS,
  features: ['real_time', 'diarization', 'sentiment', 'medical'],
  compliance: ['HIPAA', 'SOC2'],
};

// ============================================================================
// ASSEMBLYAI
// ============================================================================
```

```typescript
const ASSEMBLYAI_MODELS: ExternalModel[] = [
  {
    id: 'assemblyai-best',
    modelId: 'best',
    litellmId: 'assemblyai/best',
    name: 'best',
    displayName: 'AssemblyAI Best',
    description: 'Highest accuracy transcription',
    category: 'speech_to_text',
    capabilities: ['transcription', 'diarization', 'summarization', 'chapters'],
    inputModalities: ['audio'],
    outputModalities: ['text'],
    pricing: { type: 'per_minute', costPerMinute: 0.0062, markup: MARKUP },
  },
];

export const ASSEMBLYAI_PROVIDER: ExternalProvider = {
  id: 'assemblyai',
  name: 'assemblyai',
  displayName: 'AssemblyAI',
  category: 'speech_to_text',
  description: 'AI-powered transcription and audio intelligence',
  website: 'https://assemblyai.com',
  apiBaseUrl: 'https://api.assemblyai.com/v2',
  authType: 'bearer',
  secretName: 'radiant/providers/assemblyai',
  enabled: true,
  regions: ['us-east-1'],
  models: ASSEMBLYAI_MODELS,
  features: ['lemur', 'summarization', 'chapters', 'entity_detection'],
  compliance: ['SOC2', 'GDPR'],
};

export const AUDIO_PROVIDERS: ExternalProvider[] = [
  OPENAI_AUDIO_PROVIDER,
  ELEVENLABS_PROVIDER,
  DEEPGRAM_PROVIDER,
  ASSEMBLYAI_PROVIDER,
];
```

packages/infrastructure/lib/config/providers/embedding.providers.ts

```typescript
/**
 * Embedding Providers
 * OpenAI, Cohere, Voyage, Google
 */
```

```
import { ExternalProvider, ExternalModel } from './index';

const MARKUP = 1.40;

const OPENAI_EMBEDDING_MODELS: ExternalModel[] = [
  {
    id: 'openai-text-embedding-3-small',
    modelId: 'text-embedding-3-small',
    litellmId: 'text-embedding-3-small',
    name: 'text-embedding-3-small',
    displayName: 'Text Embedding 3 Small',
    description: 'Efficient embedding with 1536 dimensions',
    category: 'embedding',
    capabilities: ['text_embedding', 'variable_dimensions'],
    contextWindow: 8191,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: { type: 'per_token', inputCostPer1k: 0.00002, markup: MARKUP, billedInputPer1k: 0
  },
  {
    id: 'openai-text-embedding-3-large',
    modelId: 'text-embedding-3-large',
    litellmId: 'text-embedding-3-large',
    name: 'text-embedding-3-large',
    displayName: 'Text Embedding 3 Large',
    description: 'High-performance with 3072 dimensions',
    category: 'embedding',
    capabilities: ['text_embedding', 'variable_dimensions'],
    contextWindow: 8191,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: { type: 'per_token', inputCostPer1k: 0.00013, markup: MARKUP, billedInputPer1k: 0
  },
];

export const OPENAI_EMBEDDING_PROVIDER: ExternalProvider = {
  id: 'openai-embeddings',
  name: 'openai-embeddings',
  displayName: 'OpenAI Embeddings',
  category: 'embedding',
  description: 'OpenAI text embedding models',
  website: 'https://openai.com',
  apiBaseUrl: 'https://api.openai.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/openai',
  enabled: true,
  regions: ['us-east-1'],
  models: OPENAI_EMBEDDING_MODELS,
```

```
  features: ['variable_dimensions', 'batch'],
};

const VOYAGE_MODELS: ExternalModel[] = [
  {
    id: 'voyage-3',
    modelId: 'voyage-3',
    litellmId: 'voyage/voyage-3',
    name: 'voyage-3',
    displayName: 'Voyage 3',
    description: 'State-of-the-art general-purpose embeddings',
    category: 'embedding',
    capabilities: ['text_embedding'],
    contextWindow: 32000,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: { type: 'per_token', inputCostPer1k: 0.00006, markup: MARKUP, billedInputPer1k: 0
  },
  {
    id: 'voyage-code-3',
    modelId: 'voyage-code-3',
    litellmId: 'voyage/voyage-code-3',
    name: 'voyage-code-3',
    displayName: 'Voyage Code 3',
    description: 'Code-optimized embeddings',
    category: 'embedding',
    capabilities: ['text_embedding', 'code'],
    contextWindow: 32000,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: { type: 'per_token', inputCostPer1k: 0.00006, markup: MARKUP, billedInputPer1k: 0
  },
];

export const VOYAGE_PROVIDER: ExternalProvider = {
  id: 'voyage',
  name: 'voyage',
  displayName: 'Voyage AI',
  category: 'embedding',
  description: 'Premium embedding models',
  website: 'https://voyageai.com',
  apiBaseUrl: 'https://api.voyageai.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/voyage',
  enabled: true,
  regions: ['us-east-1'],
  models: VOYAGE_MODELS,
  features: ['long_context', 'code', 'multimodal'],
```

```
};

export const EMBEDDING_PROVIDERS: ExternalProvider[] = [
  OPENAI_EMBEDDING_PROVIDER,
  VOYAGE_PROVIDER,
];
```

packages/infrastructure/lib/config/providers/search.providers.ts

```
/**
 * Search Providers
 * Perplexity, Exa, Tavily
 */

import { ExternalProvider, ExternalModel } from './index';

const MARKUP = 1.40;

const PERPLEXITY_MODELS: ExternalModel[] = [
  {
    id: 'perplexity-sonar-pro',
    modelId: 'sonar-pro',
    litellmId: 'perplexity/sonar-pro',
    name: 'sonar-pro',
    displayName: 'Sonar Pro',
    description: 'Premium search-augmented model',
    category: 'search',
    capabilities: ['search', 'citations', 'reasoning'],
    contextWindow: 200000,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: { type: 'per_token', inputCostPer1k: 0.003, outputCostPer1k: 0.015, baseCostPerRe
  },
  {
    id: 'perplexity-sonar',
    modelId: 'sonar',
    litellmId: 'perplexity/sonar',
    name: 'sonar',
    displayName: 'Sonar',
    description: 'Fast search-augmented model',
    category: 'search',
    capabilities: ['search', 'citations'],
    contextWindow: 128000,
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: { type: 'per_token', inputCostPer1k: 0.001, outputCostPer1k: 0.001, baseCostPerRe
  },
];
```

```typescript
export const PERPLEXITY_PROVIDER: ExternalProvider = {
  id: 'perplexity',
  name: 'perplexity',
  displayName: 'Perplexity',
  category: 'search',
  description: 'AI search engine with real-time web access',
  website: 'https://perplexity.ai',
  apiBaseUrl: 'https://api.perplexity.ai',
  authType: 'bearer',
  secretName: 'radiant/providers/perplexity',
  enabled: true,
  regions: ['us-east-1'],
  models: PERPLEXITY_MODELS,
  features: ['web_search', 'citations', 'deep_research'],
};

const EXA_MODELS: ExternalModel[] = [
  {
    id: 'exa-search',
    modelId: 'exa-search',
    litellmId: 'exa/search',
    name: 'exa-search',
    displayName: 'Exa Search',
    description: 'Neural search engine for precise results',
    category: 'search',
    capabilities: ['neural_search', 'content_extraction'],
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: { type: 'per_request', baseCostPerRequest: 0.001, markup: MARKUP },
  },
];

export const EXA_PROVIDER: ExternalProvider = {
  id: 'exa',
  name: 'exa',
  displayName: 'Exa',
  category: 'search',
  description: 'Neural search API',
  website: 'https://exa.ai',
  apiBaseUrl: 'https://api.exa.ai/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/exa',
  enabled: true,
  regions: ['us-east-1'],
  models: EXA_MODELS,
  features: ['neural_search', 'similarity', 'keyword'],
};
```

```
const TAVILY_MODELS: ExternalModel[] = [
  {
    id: 'tavily-search',
    modelId: 'tavily-search',
    litellmId: 'tavily/search',
    name: 'tavily-search',
    displayName: 'Tavily Search',
    description: 'AI-optimized search API',
    category: 'search',
    capabilities: ['search', 'answer_generation'],
    inputModalities: ['text'],
    outputModalities: ['text'],
    pricing: { type: 'per_request', baseCostPerRequest: 0.001, markup: MARKUP },
  },
];

export const TAVILY_PROVIDER: ExternalProvider = {
  id: 'tavily',
  name: 'tavily',
  displayName: 'Tavily',
  category: 'search',
  description: 'Search API for AI agents',
  website: 'https://tavily.com',
  apiBaseUrl: 'https://api.tavily.com/v1',
  authType: 'bearer',
  secretName: 'radiant/providers/tavily',
  enabled: true,
  regions: ['us-east-1'],
  models: TAVILY_MODELS,
  features: ['search', 'extract', 'answer'],
};

export const SEARCH_PROVIDERS: ExternalProvider[] = [
  PERPLEXITY_PROVIDER,
  EXA_PROVIDER,
  TAVILY_PROVIDER,
];
```

packages/infrastructure/lib/config/providers/3d.providers.ts

```
/**
 * 3D Generation Providers
 * Meshy, Rodin, Tripo
 */

import { ExternalProvider, ExternalModel } from './index';
```

```typescript
const MARKUP = 1.40;

const MESHY_MODELS: ExternalModel[] = [
  {
    id: 'meshy-v3',
    modelId: 'meshy-v3',
    litellmId: 'meshy/meshy-v3',
    name: 'meshy-v3',
    displayName: 'Meshy V3',
    description: 'Text-to-3D and image-to-3D generation',
    category: '3d_generation',
    capabilities: ['text_to_3d', 'image_to_3d'],
    inputModalities: ['text', 'image'],
    outputModalities: ['3d'],
    pricing: { type: 'per_request', baseCostPerRequest: 0.20, markup: MARKUP },
  },
  {
    id: 'meshy-v3-turbo',
    modelId: 'meshy-v3-turbo',
    litellmId: 'meshy/meshy-v3-turbo',
    name: 'meshy-v3-turbo',
    displayName: 'Meshy V3 Turbo',
    description: 'Fast 3D generation (preview quality)',
    category: '3d_generation',
    capabilities: ['text_to_3d', 'fast'],
    inputModalities: ['text'],
    outputModalities: ['3d'],
    pricing: { type: 'per_request', baseCostPerRequest: 0.05, markup: MARKUP },
  },
];

export const MESHY_PROVIDER: ExternalProvider = {
  id: 'meshy',
  name: 'meshy',
  displayName: 'Meshy',
  category: '3d_generation',
  description: 'AI 3D model generation',
  website: 'https://meshy.ai',
  apiBaseUrl: 'https://api.meshy.ai/v2',
  authType: 'bearer',
  secretName: 'radiant/providers/meshy',
  enabled: true,
  regions: ['us-east-1'],
  models: MESHY_MODELS,
  features: ['text_to_3d', 'image_to_3d', 'texturing'],
};

const TRIPO_MODELS: ExternalModel[] = [
```

```
  {
    id: 'tripo-v2',
    modelId: 'tripo-v2',
    litellmId: 'tripo/tripo-v2',
    name: 'tripo-v2',
    displayName: 'Tripo V2',
    description: 'Fast text and image to 3D',
    category: '3d_generation',
    capabilities: ['text_to_3d', 'image_to_3d'],
    inputModalities: ['text', 'image'],
    outputModalities: ['3d'],
    pricing: { type: 'per_request', baseCostPerRequest: 0.10, markup: MARKUP },
  },
];

export const TRIPO_PROVIDER: ExternalProvider = {
  id: 'tripo',
  name: 'tripo',
  displayName: 'Tripo AI',
  category: '3d_generation',
  description: 'AI 3D generation platform',
  website: 'https://tripo3d.ai',
  apiBaseUrl: 'https://api.tripo3d.ai/v2',
  authType: 'bearer',
  secretName: 'radiant/providers/tripo',
  enabled: true,
  regions: ['us-east-1'],
  models: TRIPO_MODELS,
  features: ['text_to_3d', 'image_to_3d', 'animation'],
};

export const THREE_D_PROVIDERS: ExternalProvider[] = [
  MESHY_PROVIDER,
  TRIPO_PROVIDER,
];
```

PART 2: LITELLM CONFIGURATION

packages/infrastructure/litellm/config/external.yaml

```
# LiteLLM Configuration - External Providers
# RADIANT v2.2.0 - December 2024

model_list:
  # ============================================================================
  # OPENAI MODELS
  # ============================================================================
```

```yaml
- model_name: gpt-4o
  litellm_params:
    model: gpt-4o
    api_key: os.environ/OPENAI_API_KEY
  model_info:
    mode: chat
    input_cost_per_token: 0.0000025
    output_cost_per_token: 0.00001
    max_tokens: 16384
    max_input_tokens: 128000
    supports_function_calling: true
    supports_vision: true

- model_name: gpt-4o-mini
  litellm_params:
    model: gpt-4o-mini
    api_key: os.environ/OPENAI_API_KEY
  model_info:
    mode: chat
    input_cost_per_token: 0.00000015
    output_cost_per_token: 0.0000006
    max_tokens: 16384
    max_input_tokens: 128000

- model_name: o1
  litellm_params:
    model: o1
    api_key: os.environ/OPENAI_API_KEY
  model_info:
    mode: chat
    input_cost_per_token: 0.000015
    output_cost_per_token: 0.00006
    max_tokens: 100000
    max_input_tokens: 200000

- model_name: o3-mini
  litellm_params:
    model: o3-mini
    api_key: os.environ/OPENAI_API_KEY
  model_info:
    mode: chat
    input_cost_per_token: 0.0000011
    output_cost_per_token: 0.0000044

# ============================================================================
# ANTHROPIC MODELS
# ============================================================================
- model_name: claude-opus-4
```

```yaml
    litellm_params:
      model: anthropic/claude-opus-4-20250514
      api_key: os.environ/ANTHROPIC_API_KEY
    model_info:
      mode: chat
      input_cost_per_token: 0.000015
      output_cost_per_token: 0.000075
      max_tokens: 32000
      max_input_tokens: 200000

  - model_name: claude-sonnet-4
    litellm_params:
      model: anthropic/claude-sonnet-4-20250514
      api_key: os.environ/ANTHROPIC_API_KEY
    model_info:
      mode: chat
      input_cost_per_token: 0.000003
      output_cost_per_token: 0.000015
      max_tokens: 64000
      max_input_tokens: 200000

  - model_name: claude-3.5-haiku
    litellm_params:
      model: anthropic/claude-3-5-haiku-20241022
      api_key: os.environ/ANTHROPIC_API_KEY
    model_info:
      mode: chat
      input_cost_per_token: 0.0000008
      output_cost_per_token: 0.000004

  # ============================================================================
  # GOOGLE MODELS
  # ============================================================================
  - model_name: gemini-2.0-flash
    litellm_params:
      model: gemini/gemini-2.0-flash
      api_key: os.environ/GOOGLE_API_KEY
    model_info:
      mode: chat
      input_cost_per_token: 0.0000001
      output_cost_per_token: 0.0000004
      max_input_tokens: 1000000

  - model_name: gemini-1.5-pro
    litellm_params:
      model: gemini/gemini-1.5-pro
      api_key: os.environ/GOOGLE_API_KEY
    model_info:
```

```yaml
    mode: chat
    input_cost_per_token: 0.00000125
    output_cost_per_token: 0.000005
    max_input_tokens: 2000000

# ================================================================================
# XAI (GROK) MODELS
# ================================================================================
- model_name: grok-3
  litellm_params:
    model: xai/grok-3
    api_key: os.environ/XAI_API_KEY
    api_base: https://api.x.ai/v1
  model_info:
    mode: chat
    input_cost_per_token: 0.000003
    output_cost_per_token: 0.000015

# ================================================================================
# DEEPSEEK MODELS
# ================================================================================
- model_name: deepseek-chat
  litellm_params:
    model: deepseek/deepseek-chat
    api_key: os.environ/DEEPSEEK_API_KEY
    api_base: https://api.deepseek.com/v1
  model_info:
    mode: chat
    input_cost_per_token: 0.00000027
    output_cost_per_token: 0.0000011

- model_name: deepseek-reasoner
  litellm_params:
    model: deepseek/deepseek-reasoner
    api_key: os.environ/DEEPSEEK_API_KEY
    api_base: https://api.deepseek.com/v1
  model_info:
    mode: chat
    input_cost_per_token: 0.00000055
    output_cost_per_token: 0.00000219

# ================================================================================
# MISTRAL MODELS
# ================================================================================
- model_name: mistral-large
  litellm_params:
    model: mistral/mistral-large-latest
    api_key: os.environ/MISTRAL_API_KEY
```

```yaml
    model_info:
      mode: chat
      input_cost_per_token: 0.000002
      output_cost_per_token: 0.000006

  - model_name: mistral-small
    litellm_params:
      model: mistral/mistral-small-latest
      api_key: os.environ/MISTRAL_API_KEY
    model_info:
      mode: chat
      input_cost_per_token: 0.0000002
      output_cost_per_token: 0.0000006

  # ============================================================================
  # EMBEDDING MODELS
  # ============================================================================
  - model_name: text-embedding-3-small
    litellm_params:
      model: text-embedding-3-small
      api_key: os.environ/OPENAI_API_KEY
    model_info:
      mode: embedding
      input_cost_per_token: 0.00000002

  - model_name: text-embedding-3-large
    litellm_params:
      model: text-embedding-3-large
      api_key: os.environ/OPENAI_API_KEY
    model_info:
      mode: embedding
      input_cost_per_token: 0.00000013

  - model_name: voyage-3
    litellm_params:
      model: voyage/voyage-3
      api_key: os.environ/VOYAGE_API_KEY
    model_info:
      mode: embedding
      input_cost_per_token: 0.00000006

  # ============================================================================
  # PERPLEXITY SEARCH MODELS
  # ============================================================================
  - model_name: sonar-pro
    litellm_params:
      model: perplexity/sonar-pro
      api_key: os.environ/PERPLEXITY_API_KEY
```

```yaml
    model_info:
      mode: chat
      input_cost_per_token: 0.000003
      output_cost_per_token: 0.000015

  - model_name: sonar
    litellm_params:
      model: perplexity/sonar
      api_key: os.environ/PERPLEXITY_API_KEY
    model_info:
      mode: chat
      input_cost_per_token: 0.000001
      output_cost_per_token: 0.000001

# ============================================================================
# LITELLM SETTINGS
# ============================================================================
litellm_settings:
  success_callback: ["langfuse"]
  failure_callback: ["langfuse"]
  request_timeout: 300
  num_retries: 3
  retry_after: 5
  cache: true
  cache_params:
    type: redis
    host: os.environ/REDIS_HOST
    port: 6379
    ttl: 3600
  enable_rate_limiting: true
  set_verbose: false
  json_logs: true

# ============================================================================
# ROUTER SETTINGS
# ============================================================================
router_settings:
  routing_strategy: "least-busy"
  enable_pre_call_checks: true
  allowed_fails: 3
  cooldown_time: 60
  fallbacks:
    - model_name: gpt-4o
      fallback_models: ["claude-sonnet-4", "gemini-2.0-flash"]
    - model_name: claude-opus-4
      fallback_models: ["gpt-4o", "gemini-1.5-pro"]

general_settings:
```

```yaml
  master_key: os.environ/LITELLM_MASTER_KEY
  database_url: os.environ/LITELLM_DATABASE_URL
  alerting: ["slack"]
  alerting_threshold: 300
```

PART 3: POSTGRESQL SCHEMA

packages/infrastructure/migrations/001_initial_schema.sql

```sql
-- Migration: 001_initial_schema
-- RADIANT v2.2.0 - Base tables

CREATE EXTENSION IF NOT EXISTS "uuid-ossp";
CREATE EXTENSION IF NOT EXISTS "pgcrypto";
CREATE EXTENSION IF NOT EXISTS "pg_trgm";

-- ============================================================================
-- TENANTS
-- ============================================================================

CREATE TABLE tenants (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    app_id VARCHAR(50) NOT NULL,
    name VARCHAR(255) NOT NULL,
    slug VARCHAR(100) NOT NULL UNIQUE,
    domain VARCHAR(255),
    tier INTEGER NOT NULL DEFAULT 1 CHECK (tier BETWEEN 1 AND 5),
    status VARCHAR(20) NOT NULL DEFAULT 'active' CHECK (status IN ('active', 'suspended', 'can
    stripe_customer_id VARCHAR(100),
    billing_email VARCHAR(255),
    settings JSONB DEFAULT '{}',
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    deleted_at TIMESTAMPTZ
);

CREATE INDEX idx_tenants_app_id ON tenants(app_id);
CREATE INDEX idx_tenants_slug ON tenants(slug);
CREATE INDEX idx_tenants_status ON tenants(status);

-- ============================================================================
-- USERS
-- ============================================================================

CREATE TABLE users (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
```

```
    cognito_sub VARCHAR(100) UNIQUE,
    email VARCHAR(255) NOT NULL,
    email_verified BOOLEAN DEFAULT false,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    display_name VARCHAR(200),
    avatar_url TEXT,
    preferences JSONB DEFAULT '{}',
    timezone VARCHAR(50) DEFAULT 'UTC',
    locale VARCHAR(10) DEFAULT 'en-US',
    status VARCHAR(20) NOT NULL DEFAULT 'active' CHECK (status IN ('active', 'inactive', 'suspe
    last_login_at TIMESTAMPTZ,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    deleted_at TIMESTAMPTZ,
    UNIQUE(tenant_id, email)
);

CREATE INDEX idx_users_tenant ON users(tenant_id);
CREATE INDEX idx_users_email ON users(email);
CREATE INDEX idx_users_cognito ON users(cognito_sub);

-- ============================================================================
-- API KEYS
-- ============================================================================

CREATE TABLE api_keys (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    user_id UUID REFERENCES users(id) ON DELETE SET NULL,
    name VARCHAR(100) NOT NULL,
    key_prefix VARCHAR(10) NOT NULL,
    key_hash VARCHAR(255) NOT NULL,
    scopes TEXT[] DEFAULT ARRAY['chat', 'models', 'embeddings'],
    rate_limit_rpm INTEGER DEFAULT 60,
    rate_limit_tpm INTEGER DEFAULT 100000,
    last_used_at TIMESTAMPTZ,
    usage_count BIGINT DEFAULT 0,
    status VARCHAR(20) NOT NULL DEFAULT 'active' CHECK (status IN ('active', 'revoked', 'expire
    expires_at TIMESTAMPTZ,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

CREATE INDEX idx_api_keys_tenant ON api_keys(tenant_id);
CREATE INDEX idx_api_keys_prefix ON api_keys(key_prefix);
CREATE UNIQUE INDEX idx_api_keys_hash ON api_keys(key_hash);
```

```sql
-- ============================================================================
-- SCHEMA MIGRATIONS TRACKING
-- ============================================================================

CREATE TABLE schema_migrations (
    version VARCHAR(20) PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    applied_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    applied_by VARCHAR(100),
    checksum VARCHAR(64)
);

INSERT INTO schema_migrations (version, name, applied_by)
VALUES ('001', 'initial_schema', 'system');
```

packages/infrastructure/migrations/002_tenant_isolation.sql

```sql
-- Migration: 002_tenant_isolation
-- Row-Level Security policies

ALTER TABLE tenants ENABLE ROW LEVEL SECURITY;
ALTER TABLE users ENABLE ROW LEVEL SECURITY;
ALTER TABLE api_keys ENABLE ROW LEVEL SECURITY;

CREATE OR REPLACE FUNCTION current_tenant_id() RETURNS UUID AS $$
BEGIN
    RETURN NULLIF(current_setting('app.current_tenant_id', true), '')::UUID;
EXCEPTION WHEN OTHERS THEN RETURN NULL;
END;
$$ LANGUAGE plpgsql SECURITY DEFINER;

CREATE POLICY tenant_isolation_select ON tenants FOR SELECT USING (id = current_tenant_id());
CREATE POLICY tenant_isolation_update ON tenants FOR UPDATE USING (id = current_tenant_id());

CREATE POLICY users_tenant_isolation ON users FOR ALL USING (tenant_id = current_tenant_id() O
CREATE POLICY api_keys_tenant_isolation ON api_keys FOR ALL USING (tenant_id = current_tenant_

-- System role for admin operations
DO $$ BEGIN IF NOT EXISTS (SELECT FROM pg_roles WHERE rolname = 'radiant_system') THEN CREATE
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO radiant_system;

INSERT INTO schema_migrations (version, name, applied_by) VALUES ('002', 'tenant_isolation', '
```

packages/infrastructure/migrations/003_ai_models.sql

```sql
-- Migration: 003_ai_models
-- Providers and models registry
```

```sql
CREATE TABLE providers (
    id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    display_name VARCHAR(100) NOT NULL,
    category VARCHAR(50) NOT NULL,
    description TEXT,
    website VARCHAR(255),
    api_base_url VARCHAR(255),
    auth_type VARCHAR(20) NOT NULL DEFAULT 'bearer',
    secret_name VARCHAR(100),
    enabled BOOLEAN DEFAULT true,
    features TEXT[],
    regions TEXT[],
    compliance TEXT[],
    rate_limit_rpm INTEGER,
    rate_limit_tpm INTEGER,
    metadata JSONB DEFAULT '{}',
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

CREATE INDEX idx_providers_category ON providers(category);
CREATE INDEX idx_providers_enabled ON providers(enabled);

CREATE TABLE models (
    id VARCHAR(100) PRIMARY KEY,
    provider_id VARCHAR(50) NOT NULL REFERENCES providers(id) ON DELETE CASCADE,
    model_id VARCHAR(100) NOT NULL,
    litellm_id VARCHAR(100) NOT NULL UNIQUE,
    name VARCHAR(100) NOT NULL,
    display_name VARCHAR(100) NOT NULL,
    description TEXT,
    category VARCHAR(50) NOT NULL,
    capabilities TEXT[],
    context_window INTEGER,
    max_output INTEGER,
    input_modalities TEXT[],
    output_modalities TEXT[],
    input_cost_per_1k NUMERIC(10, 8),
    output_cost_per_1k NUMERIC(10, 8),
    cost_per_request NUMERIC(10, 6),
    cost_per_second NUMERIC(10, 6),
    cost_per_image NUMERIC(10, 6),
    cost_per_minute NUMERIC(10, 6),
    pricing_type VARCHAR(20) NOT NULL DEFAULT 'per_token',
    markup_rate NUMERIC(4, 2) NOT NULL DEFAULT 1.40,
    enabled BOOLEAN DEFAULT true,
    deprecated BOOLEAN DEFAULT false,
```

```
    is_self_hosted BOOLEAN DEFAULT false,
    self_hosted_config JSONB,
    metadata JSONB DEFAULT '{}',
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

CREATE INDEX idx_models_provider ON models(provider_id);
CREATE INDEX idx_models_category ON models(category);
CREATE INDEX idx_models_enabled ON models(enabled);
CREATE INDEX idx_models_litellm ON models(litellm_id);

CREATE TABLE tenant_model_access (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    model_id VARCHAR(100) NOT NULL REFERENCES models(id) ON DELETE CASCADE,
    enabled BOOLEAN DEFAULT true,
    custom_input_cost_per_1k NUMERIC(10, 8),
    custom_output_cost_per_1k NUMERIC(10, 8),
    custom_markup_rate NUMERIC(4, 2),
    rate_limit_rpm INTEGER,
    rate_limit_tpm INTEGER,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    UNIQUE(tenant_id, model_id)
);

ALTER TABLE tenant_model_access ENABLE ROW LEVEL SECURITY;
CREATE POLICY tenant_model_access_isolation ON tenant_model_access FOR ALL USING (tenant_id = c

INSERT INTO schema_migrations (version, name, applied_by) VALUES ('003', 'ai_models', 'system')
```

packages/infrastructure/migrations/004_usage_billing.sql

```
-- Migration: 004_usage_billing
-- Usage tracking and billing

CREATE TABLE usage_events (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    user_id UUID REFERENCES users(id) ON DELETE SET NULL,
    api_key_id UUID REFERENCES api_keys(id) ON DELETE SET NULL,
    request_id VARCHAR(100) NOT NULL,
    model_id VARCHAR(100) NOT NULL REFERENCES models(id),
    provider_id VARCHAR(50) NOT NULL REFERENCES providers(id),
    input_tokens INTEGER NOT NULL DEFAULT 0,
    output_tokens INTEGER NOT NULL DEFAULT 0,
    total_tokens INTEGER GENERATED ALWAYS AS (input_tokens + output_tokens) STORED,
```

```
    image_count INTEGER DEFAULT 0,
    video_seconds NUMERIC(10, 2) DEFAULT 0,
    audio_minutes NUMERIC(10, 2) DEFAULT 0,
    request_count INTEGER DEFAULT 1,
    base_cost NUMERIC(12, 8) NOT NULL DEFAULT 0,
    billed_cost NUMERIC(12, 8) NOT NULL DEFAULT 0,
    endpoint VARCHAR(50),
    latency_ms INTEGER,
    status VARCHAR(20) NOT NULL DEFAULT 'success',
    error_code VARCHAR(50),
    ip_address INET,
    metadata JSONB DEFAULT '{}',
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

CREATE INDEX idx_usage_events_tenant ON usage_events(tenant_id);
CREATE INDEX idx_usage_events_model ON usage_events(model_id);
CREATE INDEX idx_usage_events_created ON usage_events(created_at DESC);
CREATE INDEX idx_usage_events_request ON usage_events(request_id);

ALTER TABLE usage_events ENABLE ROW LEVEL SECURITY;
CREATE POLICY usage_events_tenant_isolation ON usage_events FOR ALL USING (tenant_id = current_

CREATE TABLE usage_rollups (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    rollup_date DATE NOT NULL,
    model_id VARCHAR(100) REFERENCES models(id),
    provider_id VARCHAR(50) REFERENCES providers(id),
    user_id UUID REFERENCES users(id) ON DELETE SET NULL,
    request_count BIGINT NOT NULL DEFAULT 0,
    success_count BIGINT NOT NULL DEFAULT 0,
    error_count BIGINT NOT NULL DEFAULT 0,
    input_tokens BIGINT NOT NULL DEFAULT 0,
    output_tokens BIGINT NOT NULL DEFAULT 0,
    total_tokens BIGINT NOT NULL DEFAULT 0,
    image_count INTEGER DEFAULT 0,
    video_seconds NUMERIC(12, 2) DEFAULT 0,
    audio_minutes NUMERIC(12, 2) DEFAULT 0,
    base_cost NUMERIC(14, 6) NOT NULL DEFAULT 0,
    billed_cost NUMERIC(14, 6) NOT NULL DEFAULT 0,
    avg_latency_ms NUMERIC(10, 2),
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    UNIQUE(tenant_id, rollup_date, model_id, provider_id, user_id)
);

CREATE INDEX idx_usage_rollups_tenant ON usage_rollups(tenant_id);
```

```
CREATE INDEX idx_usage_rollups_date ON usage_rollups(rollup_date DESC);

ALTER TABLE usage_rollups ENABLE ROW LEVEL SECURITY;
CREATE POLICY usage_rollups_tenant_isolation ON usage_rollups FOR ALL USING (tenant_id = curre

CREATE TABLE invoices (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    invoice_number VARCHAR(50) NOT NULL UNIQUE,
    billing_period_start DATE NOT NULL,
    billing_period_end DATE NOT NULL,
    subtotal NUMERIC(14, 2) NOT NULL DEFAULT 0,
    discount_amount NUMERIC(14, 2) NOT NULL DEFAULT 0,
    tax_amount NUMERIC(14, 2) NOT NULL DEFAULT 0,
    total_amount NUMERIC(14, 2) NOT NULL DEFAULT 0,
    status VARCHAR(20) NOT NULL DEFAULT 'draft' CHECK (status IN ('draft', 'pending', 'paid',
    stripe_invoice_id VARCHAR(100),
    paid_at TIMESTAMPTZ,
    line_items JSONB NOT NULL DEFAULT '[]',
    notes TEXT,
    metadata JSONB DEFAULT '{}',
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

CREATE INDEX idx_invoices_tenant ON invoices(tenant_id);
CREATE INDEX idx_invoices_status ON invoices(status);

ALTER TABLE invoices ENABLE ROW LEVEL SECURITY;
CREATE POLICY invoices_tenant_isolation ON invoices FOR ALL USING (tenant_id = current_tenant_

INSERT INTO schema_migrations (version, name, applied_by) VALUES ('004', 'usage_billing', 'syst
```

packages/infrastructure/migrations/005_admin_approval.sql

```
-- Migration: 005_admin_approval
-- Administrator management and approval workflows

CREATE TABLE administrators (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    cognito_sub VARCHAR(100) UNIQUE,
    email VARCHAR(255) NOT NULL,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL,
    display_name VARCHAR(200),
    phone VARCHAR(20),
    role VARCHAR(20) NOT NULL CHECK (role IN ('super_admin', 'admin', 'operator', 'auditor')),
```

```
    custom_permissions TEXT[],
    mfa_enabled BOOLEAN DEFAULT false,
    mfa_method VARCHAR(20) CHECK (mfa_method IN ('authenticator', 'sms', 'email')),
    mfa_verified_at TIMESTAMPTZ,
    preferences JSONB DEFAULT '{}',
    notification_settings JSONB DEFAULT '{"email": true, "slack": false, "approval_required": t
    status VARCHAR(20) NOT NULL DEFAULT 'active' CHECK (status IN ('pending', 'active', 'inacti
    last_login_at TIMESTAMPTZ,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    deleted_at TIMESTAMPTZ,
    UNIQUE(tenant_id, email)
);

CREATE INDEX idx_admins_tenant ON administrators(tenant_id);
CREATE INDEX idx_admins_email ON administrators(email);
CREATE INDEX idx_admins_role ON administrators(role);

ALTER TABLE administrators ENABLE ROW LEVEL SECURITY;
CREATE POLICY admins_tenant_isolation ON administrators FOR ALL USING (tenant_id = current_tena

CREATE TABLE invitations (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    email VARCHAR(255) NOT NULL,
    role VARCHAR(20) NOT NULL CHECK (role IN ('super_admin', 'admin', 'operator', 'auditor')),
    message TEXT,
    token_hash VARCHAR(255) NOT NULL,
    invited_by UUID NOT NULL REFERENCES administrators(id),
    invited_by_name VARCHAR(200) NOT NULL,
    status VARCHAR(20) NOT NULL DEFAULT 'pending' CHECK (status IN ('pending', 'accepted', 'exp
    expires_at TIMESTAMPTZ NOT NULL,
    accepted_at TIMESTAMPTZ,
    accepted_by_ip INET,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

CREATE INDEX idx_invitations_tenant ON invitations(tenant_id);
CREATE INDEX idx_invitations_email ON invitations(email);
CREATE INDEX idx_invitations_status ON invitations(status);
CREATE INDEX idx_invitations_token ON invitations(token_hash);

ALTER TABLE invitations ENABLE ROW LEVEL SECURITY;
CREATE POLICY invitations_tenant_isolation ON invitations FOR ALL USING (tenant_id = current_te

CREATE TABLE approval_requests (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
```

```
    app_id VARCHAR(50) NOT NULL,
    environment VARCHAR(20) NOT NULL CHECK (environment IN ('dev', 'staging', 'prod')),
    type VARCHAR(50) NOT NULL CHECK (type IN ('deployment', 'promotion', 'model_activation', '
    title VARCHAR(255) NOT NULL,
    description TEXT,
    payload JSONB NOT NULL DEFAULT '{}',
    risk_level VARCHAR(20) NOT NULL DEFAULT 'medium' CHECK (risk_level IN ('low', 'medium', 'hi
    impact_analysis TEXT,
    rollback_plan TEXT,
    initiated_by UUID NOT NULL REFERENCES administrators(id),
    initiated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    status VARCHAR(20) NOT NULL DEFAULT 'pending' CHECK (status IN ('pending', 'approved', 're
    approved_by UUID REFERENCES administrators(id),
    approved_at TIMESTAMPTZ,
    approval_notes TEXT,
    executed_at TIMESTAMPTZ,
    execution_result JSONB,
    expires_at TIMESTAMPTZ NOT NULL DEFAULT (NOW() + INTERVAL '24 hours'),
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    CONSTRAINT different_approver CHECK (approved_by IS NULL OR approved_by != initiated_by)
);

CREATE INDEX idx_approval_tenant ON approval_requests(tenant_id);
CREATE INDEX idx_approval_status ON approval_requests(status);
CREATE INDEX idx_approval_environment ON approval_requests(environment);

ALTER TABLE approval_requests ENABLE ROW LEVEL SECURITY;
CREATE POLICY approval_tenant_isolation ON approval_requests FOR ALL USING (tenant_id = current

CREATE TABLE audit_logs (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    user_id UUID REFERENCES users(id) ON DELETE SET NULL,
    admin_id UUID REFERENCES administrators(id) ON DELETE SET NULL,
    action VARCHAR(100) NOT NULL,
    resource_type VARCHAR(100) NOT NULL,
    resource_id VARCHAR(255),
    ip_address INET,
    user_agent TEXT,
    request_id VARCHAR(100),
    old_values JSONB,
    new_values JSONB,
    metadata JSONB DEFAULT '{}',
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

CREATE INDEX idx_audit_tenant ON audit_logs(tenant_id);
```

```
CREATE INDEX idx_audit_action ON audit_logs(action);
CREATE INDEX idx_audit_resource ON audit_logs(resource_type, resource_id);
CREATE INDEX idx_audit_created ON audit_logs(created_at DESC);

ALTER TABLE audit_logs ENABLE ROW LEVEL SECURITY;
CREATE POLICY audit_logs_tenant_select ON audit_logs FOR SELECT USING (tenant_id = current_tena

INSERT INTO schema_migrations (version, name, applied_by) VALUES ('005', 'admin_approval', 'sys
```

packages/infrastructure/migrations/007_external_providers.sql

```
-- Migration: 007_external_providers
-- External provider configuration

CREATE TABLE provider_health (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    provider_id VARCHAR(50) NOT NULL REFERENCES providers(id) ON DELETE CASCADE,
    region VARCHAR(20) NOT NULL,
    status VARCHAR(20) NOT NULL DEFAULT 'healthy' CHECK (status IN ('healthy', 'degraded', 'unh
    latency_ms INTEGER,
    error_rate NUMERIC(5, 2),
    success_rate NUMERIC(5, 2),
    last_check_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    last_success_at TIMESTAMPTZ,
    last_failure_at TIMESTAMPTZ,
    last_error TEXT,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    UNIQUE(provider_id, region)
);

CREATE INDEX idx_provider_health_provider ON provider_health(provider_id);
CREATE INDEX idx_provider_health_status ON provider_health(status);

CREATE TABLE routing_rules (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID REFERENCES tenants(id) ON DELETE CASCADE,
    name VARCHAR(100) NOT NULL,
    description TEXT,
    priority INTEGER NOT NULL DEFAULT 100,
    enabled BOOLEAN DEFAULT true,
    conditions JSONB NOT NULL DEFAULT '{}',
    action VARCHAR(50) NOT NULL CHECK (action IN ('route', 'fallback', 'block', 'rate_limit'))
    target_provider VARCHAR(50),
    target_model VARCHAR(100),
    fallback_models TEXT[],
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
```

```
);

CREATE INDEX idx_routing_rules_tenant ON routing_rules(tenant_id);
CREATE INDEX idx_routing_rules_priority ON routing_rules(priority);
CREATE INDEX idx_routing_rules_enabled ON routing_rules(enabled);

-- Populate initial providers
INSERT INTO providers (id, name, display_name, category, description, website, api_base_url, au
('openai', 'openai', 'OpenAI', 'text_generation', 'GPT and O-series models', 'https://openai.co
('anthropic', 'anthropic', 'Anthropic', 'text_generation', 'Claude models', 'https://anthropic
('google', 'google', 'Google AI', 'text_generation', 'Gemini models', 'https://ai.google.dev',
('xai', 'xai', 'xAI', 'text_generation', 'Grok models', 'https://x.ai', 'https://api.x.ai/v1',
('deepseek', 'deepseek', 'DeepSeek', 'text_generation', 'Affordable models', 'https://deepseek
('mistral', 'mistral', 'Mistral AI', 'text_generation', 'European models', 'https://mistral.ai
('cohere', 'cohere', 'Cohere', 'text_generation', 'Enterprise AI', 'https://cohere.com', 'https
('stability', 'stability', 'Stability AI', 'image_generation', 'Open-source images', 'https://s
('runway', 'runway', 'Runway', 'video_generation', 'AI video tools', 'https://runway.ml', 'http
('luma', 'luma', 'Luma AI', 'video_generation', 'Cinematic video', 'https://lumalabs.ai', 'http
('elevenlabs', 'elevenlabs', 'ElevenLabs', 'text_to_speech', 'Voice synthesis', 'https://elever
('deepgram', 'deepgram', 'Deepgram', 'speech_to_text', 'Speech recognition', 'https://deepgram
('perplexity', 'perplexity', 'Perplexity', 'search', 'AI search engine', 'https://perplexity.ai
('voyage', 'voyage', 'Voyage AI', 'embedding', 'Premium embeddings', 'https://voyageai.com', 'h
('meshy', 'meshy', 'Meshy', '3d_generation', 'AI 3D models', 'https://meshy.ai', 'https://api.m

INSERT INTO schema_migrations (version, name, applied_by) VALUES ('007', 'external_providers',
```

PART 4: DYNAMODB TABLES

packages/infrastructure/dynamodb/tables.ts

```typescript
/**
 * DynamoDB Table Definitions
 * For session, cache, and real-time data
 *
 * RADIANT v2.2.0 - December 2024
 */

import * as cdk from 'aws-cdk-lib';
import * as dynamodb from 'aws-cdk-lib/aws-dynamodb';
import { Construct } from 'constructs';

// ============================================================================
// TABLE CONFIGURATIONS
// ============================================================================

export interface TableConfig {
  tableName: string;
```

```
  partitionKey: { name: string; type: dynamodb.AttributeType };
  sortKey?: { name: string; type: dynamodb.AttributeType };
  gsis?: GlobalSecondaryIndexConfig[];
  ttlAttribute?: string;
  stream?: dynamodb.StreamViewType;
  billingMode: dynamodb.BillingMode;
  pointInTimeRecovery: boolean;
}

export interface GlobalSecondaryIndexConfig {
  indexName: string;
  partitionKey: { name: string; type: dynamodb.AttributeType };
  sortKey?: { name: string; type: dynamodb.AttributeType };
  projectionType: dynamodb.ProjectionType;
}

// ============================================================================
// SESSIONS TABLE
// ============================================================================

export const SESSIONS_TABLE: TableConfig = {
  tableName: 'radiant-sessions',
  partitionKey: { name: 'pk', type: dynamodb.AttributeType.STRING },
  sortKey: { name: 'sk', type: dynamodb.AttributeType.STRING },
  gsis: [
    {
      indexName: 'gsi-session-token',
      partitionKey: { name: 'sessionToken', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.ALL,
    },
    {
      indexName: 'gsi-tenant-created',
      partitionKey: { name: 'tenantId', type: dynamodb.AttributeType.STRING },
      sortKey: { name: 'createdAt', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.KEYS_ONLY,
    },
  ],
  ttlAttribute: 'expiresAt',
  stream: dynamodb.StreamViewType.NEW_AND_OLD_IMAGES,
  billingMode: dynamodb.BillingMode.PAY_PER_REQUEST,
  pointInTimeRecovery: true,
};

// Session item structure:
// {
//   pk: 'tenant#<tenant-id>#user#<user-id>',
//   sk: 'session#<session-id>',
//   sessionToken: '<jwt-token-hash>',
```

```
//   tenantId: '<tenant-id>',
//   userId: '<user-id>',
//   createdAt: '<iso-timestamp>',
//   expiresAt: <unix-timestamp>,
//   lastActivityAt: '<iso-timestamp>',
//   ipAddress: '<ip>',
//   userAgent: '<user-agent>',
//   metadata: { ... }
// }

// ============================================================================
// CHAT HISTORY TABLE
// ============================================================================

export const CHAT_HISTORY_TABLE: TableConfig = {
  tableName: 'radiant-chat-history',
  partitionKey: { name: 'pk', type: dynamodb.AttributeType.STRING },
  sortKey: { name: 'sk', type: dynamodb.AttributeType.STRING },
  gsis: [
    {
      indexName: 'gsi-conversation',
      partitionKey: { name: 'conversationId', type: dynamodb.AttributeType.STRING },
      sortKey: { name: 'createdAt', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.ALL,
    },
    {
      indexName: 'gsi-tenant-recent',
      partitionKey: { name: 'tenantId', type: dynamodb.AttributeType.STRING },
      sortKey: { name: 'createdAt', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.KEYS_ONLY,
    },
  ],
  ttlAttribute: 'expiresAt',
  stream: dynamodb.StreamViewType.NEW_IMAGE,
  billingMode: dynamodb.BillingMode.PAY_PER_REQUEST,
  pointInTimeRecovery: true,
};

// ============================================================================
// MODEL THERMAL STATE TABLE
// ============================================================================

export const MODEL_STATE_TABLE: TableConfig = {
  tableName: 'radiant-model-state',
  partitionKey: { name: 'pk', type: dynamodb.AttributeType.STRING },
  sortKey: { name: 'sk', type: dynamodb.AttributeType.STRING },
  gsis: [
    {
```

```
      indexName: 'gsi-thermal-state',
      partitionKey: { name: 'thermalState', type: dynamodb.AttributeType.STRING },
      sortKey: { name: 'lastRequestAt', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.ALL,
    },
  ],
  billingMode: dynamodb.BillingMode.PAY_PER_REQUEST,
  pointInTimeRecovery: true,
};

// ============================================================================
// CACHE TABLE
// ============================================================================

export const CACHE_TABLE: TableConfig = {
  tableName: 'radiant-cache',
  partitionKey: { name: 'pk', type: dynamodb.AttributeType.STRING },
  sortKey: { name: 'sk', type: dynamodb.AttributeType.STRING },
  ttlAttribute: 'expiresAt',
  billingMode: dynamodb.BillingMode.PAY_PER_REQUEST,
  pointInTimeRecovery: false,
};

// ============================================================================
// WEBSOCKET CONNECTIONS TABLE
// ============================================================================

export const CONNECTIONS_TABLE: TableConfig = {
  tableName: 'radiant-connections',
  partitionKey: { name: 'connectionId', type: dynamodb.AttributeType.STRING },
  gsis: [
    {
      indexName: 'gsi-tenant-user',
      partitionKey: { name: 'tenantId', type: dynamodb.AttributeType.STRING },
      sortKey: { name: 'userId', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.ALL,
    },
    {
      indexName: 'gsi-subscription',
      partitionKey: { name: 'subscription', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.ALL,
    },
  ],
  ttlAttribute: 'expiresAt',
  billingMode: dynamodb.BillingMode.PAY_PER_REQUEST,
  pointInTimeRecovery: false,
};
```

```typescript
// ============================================================================
// APPROVAL QUEUE TABLE
// ============================================================================

export const APPROVAL_QUEUE_TABLE: TableConfig = {
  tableName: 'radiant-approval-queue',
  partitionKey: { name: 'pk', type: dynamodb.AttributeType.STRING },
  sortKey: { name: 'sk', type: dynamodb.AttributeType.STRING },
  gsis: [
    {
      indexName: 'gsi-approver',
      partitionKey: { name: 'approverPool', type: dynamodb.AttributeType.STRING },
      sortKey: { name: 'createdAt', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.ALL,
    },
    {
      indexName: 'gsi-initiator',
      partitionKey: { name: 'initiatedBy', type: dynamodb.AttributeType.STRING },
      sortKey: { name: 'createdAt', type: dynamodb.AttributeType.STRING },
      projectionType: dynamodb.ProjectionType.ALL,
    },
  ],
  ttlAttribute: 'expiresAt',
  stream: dynamodb.StreamViewType.NEW_AND_OLD_IMAGES,
  billingMode: dynamodb.BillingMode.PAY_PER_REQUEST,
  pointInTimeRecovery: true,
};

// ============================================================================
// TABLE FACTORY
// ============================================================================

export function createDynamoDBTable(
  scope: Construct,
  id: string,
  config: TableConfig,
  props: {
    resourcePrefix: string;
    environment: string;
    encryptionKey?: cdk.aws_kms.IKey;
    removalPolicy?: cdk.RemovalPolicy;
  }
): dynamodb.Table {
  const tableName = `${props.resourcePrefix}-${config.tableName}`;

  const table = new dynamodb.Table(scope, id, {
    tableName,
    partitionKey: config.partitionKey,
```

```
    sortKey: config.sortKey,
    billingMode: config.billingMode,
    pointInTimeRecovery: config.pointInTimeRecovery,
    encryption: props.encryptionKey
      ? dynamodb.TableEncryption.CUSTOMER_MANAGED
      : dynamodb.TableEncryption.AWS_MANAGED,
    encryptionKey: props.encryptionKey,
    removalPolicy: props.removalPolicy ?? cdk.RemovalPolicy.RETAIN,
    stream: config.stream,
    timeToLiveAttribute: config.ttlAttribute,
  });

  if (config.gsis) {
    for (const gsi of config.gsis) {
      table.addGlobalSecondaryIndex({
        indexName: gsi.indexName,
        partitionKey: gsi.partitionKey,
        sortKey: gsi.sortKey,
        projectionType: gsi.projectionType,
      });
    }
  }

  return table;
}

// ============================================================================
// ALL TABLE CONFIGS
// ============================================================================

export const ALL_TABLES: TableConfig[] = [
  SESSIONS_TABLE,
  CHAT_HISTORY_TABLE,
  MODEL_STATE_TABLE,
  CACHE_TABLE,
  CONNECTIONS_TABLE,
  APPROVAL_QUEUE_TABLE,
];
```

PART 5: MIGRATION RUNNER

packages/infrastructure/migrations/migrations.ts

```
/**
 * Database Migration Runner
 * RADIANT v2.2.0 - December 2024
 */
```

```typescript
import { Client } from 'pg';
import * as fs from 'fs';
import * as path from 'path';
import * as crypto from 'crypto';

interface Migration {
  version: string;
  name: string;
  filename: string;
  sql: string;
  checksum: string;
}

interface MigrationResult {
  version: string;
  name: string;
  success: boolean;
  duration: number;
  error?: string;
}

export class MigrationRunner {
  private client: Client;
  private migrationsDir: string;

  constructor(connectionString: string, migrationsDir?: string) {
    this.client = new Client({ connectionString });
    this.migrationsDir = migrationsDir ?? path.join(__dirname, '.');
  }

  async connect(): Promise<void> {
    await this.client.connect();
  }

  async disconnect(): Promise<void> {
    await this.client.end();
  }

  async loadMigrations(): Promise<Migration[]> {
    const files = fs.readdirSync(this.migrationsDir)
      .filter(f => f.endsWith('.sql'))
      .sort();

    const migrations: Migration[] = [];

    for (const filename of files) {
      const match = filename.match(/^(\d{3})_(.+)\.sql$/);
```

```
    if (!match) continue;

    const [, version, name] = match;
    const filepath = path.join(this.migrationsDir, filename);
    const sql = fs.readFileSync(filepath, 'utf-8');
    const checksum = crypto.createHash('sha256').update(sql).digest('hex');

    migrations.push({ version, name, filename, sql, checksum });
  }

  return migrations;
}

async getAppliedMigrations(): Promise<Map<string, { name: string; checksum: string }>> {
  const result = await this.client.query(`
    SELECT version, name, checksum FROM schema_migrations ORDER BY version
  `);

  const applied = new Map<string, { name: string; checksum: string }>();
  for (const row of result.rows) {
    applied.set(row.version, { name: row.name, checksum: row.checksum });
  }

  return applied;
}

async runMigrations(options: { dryRun?: boolean; force?: boolean } = {}): Promise<MigrationR
  const results: MigrationResult[] = [];
  const migrations = await this.loadMigrations();
  const applied = await this.getAppliedMigrations();

  for (const migration of migrations) {
    if (applied.has(migration.version)) {
      const existing = applied.get(migration.version)!;
      if (existing.checksum && existing.checksum !== migration.checksum && !options.force) {
        throw new Error(`Migration ${migration.version} checksum mismatch`);
      }
      continue;
    }

    const startTime = Date.now();

    try {
      if (options.dryRun) {
        console.log(`[DRY RUN] Would apply: ${migration.version} - ${migration.name}`);
      } else {
        console.log(`Applying: ${migration.version} - ${migration.name}...`);
```

```
        await this.client.query('BEGIN');
        await this.client.query(migration.sql);
        await this.client.query(`
          INSERT INTO schema_migrations (version, name, checksum, applied_by)
          VALUES ($1, $2, $3, $4)
          ON CONFLICT (version) DO UPDATE SET checksum = EXCLUDED.checksum
        `, [migration.version, migration.name, migration.checksum, 'migration-runner']);
        await this.client.query('COMMIT');
      }

      results.push({
        version: migration.version,
        name: migration.name,
        success: true,
        duration: Date.now() - startTime,
      });

    } catch (error) {
      await this.client.query('ROLLBACK');
      results.push({
        version: migration.version,
        name: migration.name,
        success: false,
        duration: Date.now() - startTime,
        error: error instanceof Error ? error.message : String(error),
      });
      throw error;
    }
  }

  return results;
}

async getStatus(): Promise<{
  applied: { version: string; name: string; appliedAt: string }[];
  pending: { version: string; name: string }[];
}> {
  const migrations = await this.loadMigrations();
  const applied = await this.getAppliedMigrations();

  const result = await this.client.query(`
    SELECT version, name, applied_at FROM schema_migrations ORDER BY version
  `);

  const pending = migrations
    .filter(m => !applied.has(m.version))
    .map(m => ({ version: m.version, name: m.name }));
```

```
    return {
      applied: result.rows.map(r => ({
        version: r.version,
        name: r.name,
        appliedAt: r.applied_at,
      })),
      pending,
    };
  }
}

// CLI
async function main() {
  const command = process.argv[2];
  const connectionString = process.env.DATABASE_URL;

  if (!connectionString) {
    console.error('DATABASE_URL environment variable is required');
    process.exit(1);
  }

  const runner = new MigrationRunner(connectionString);

  try {
    await runner.connect();

    switch (command) {
      case 'status': {
        const status = await runner.getStatus();
        console.log('\n=== Applied Migrations ===');
        for (const m of status.applied) {
          console.log(`  Ã¢Å"â€œ ${m.version} - ${m.name} (${m.appliedAt})`);
        }
        console.log('\n=== Pending Migrations ===');
        for (const m of status.pending) {
          console.log(`  Ã¢â€"â€¹ ${m.version} - ${m.name}`);
        }
        break;
      }

      case 'migrate': {
        const dryRun = process.argv.includes('--dry-run');
        const force = process.argv.includes('--force');
        const results = await runner.runMigrations({ dryRun, force });

        console.log('\n=== Migration Results ===');
        for (const r of results) {
          const status = r.success ? 'Ã¢Å"â€œ' : 'Ã¢Å"â€"';
```

```
          console.log(`  ${status} ${r.version} - ${r.name} (${r.duration}ms)`);
          if (r.error) console.log(`    Error: ${r.error}`);
        }
        break;
      }

      default:
        console.log('Usage: migrations.ts <command>');
        console.log('Commands: status, migrate');
        console.log('Options: --dry-run, --force');
    }

  } finally {
    await runner.disconnect();
  }
}

if (require.main === module) {
  main().catch(err => {
    console.error('Migration error:', err);
    process.exit(1);
  });
}
```

PART 6: ENHANCED PRICING METADATA SYSTEM (v4.12)

NEW in v4.12: Comprehensive pricing metadata for Brain cost optimization.

Terminology: - **COST** = What RADIANT pays to providers (internal, admin-visible only) - **PRICE** = What customers pay (cost × markup) - visible to clients/Think Tank

Key Behavior: - Metadata service syncs **costs** from providers during model registry updates - Brain optimizes on **PRICE** when users request cost optimization - Admin sees: Cost + Markup % + Price + Estimated flags - Client/Think Tank sees: Price only + Estimated price flag

packages/infrastructure/lib/config/providers/pricing.config.ts

```
/**
 * Pricing Configuration - RADIANT v4.12
 *
 * TERMINOLOGY:
 *   COST  = What RADIANT pays (internal)
 *   PRICE = What customers pay (cost × markup)
 *
 * December 2024
 */
```

```
// ============================================================================
// MARKUP CONFIGURATION
// ============================================================================

export const DEFAULT_MARKUP_RATES = {
  EXTERNAL_PERCENT: 40,       // 40% markup on external providers
  SELF_HOSTED_PERCENT: 75,    // 75% markup on self-hosted models
};

export const MARKUP_RATES = {
  EXTERNAL: 1.40,             // Multiplier for external (1 + 40%)
  SELF_HOSTED: 1.75,          // Multiplier for self-hosted (1 + 75%)
};

// ============================================================================
// VOLUME DISCOUNTS
// ============================================================================

export const VOLUME_DISCOUNTS = [
  { minSpend: 100, discount: 0.05 },    // 5% off at $100/month
  { minSpend: 500, discount: 0.10 },    // 10% off at $500/month
  { minSpend: 1000, discount: 0.15 },   // 15% off at $1,000/month
  { minSpend: 5000, discount: 0.20 },   // 20% off at $5,000/month
  { minSpend: 10000, discount: 0.25 },  // 25% off at $10,000/month
];

// ============================================================================
// FREE TIER LIMITS (per month)
// ============================================================================

export const FREE_TIER = {
  tokens: 100_000,
  requests: 1_000,
  images: 10,
  videoSeconds: 60,
  audioMinutes: 10,
  embeddings: 10_000,
};

// ============================================================================
// TIER RATE LIMITS
// ============================================================================

export const TIER_LIMITS = {
  1: { rpm: 60, tpm: 100_000 },       // SEED
  2: { rpm: 200, tpm: 500_000 },      // STARTUP
  3: { rpm: 1000, tpm: 2_000_000 },   // GROWTH
  4: { rpm: 5000, tpm: 10_000_000 },  // SCALE
```

```
  5: { rpm: 20000, tpm: 50_000_000 }, // ENTERPRISE
};

// ============================================================================
// COST → PRICE CONVERSION
// ============================================================================

/**
 * Convert provider cost to customer price
 */
export function costToPrice(cost: number, markupPercent: number): number {
  return cost * (1 + markupPercent / 100);
}

/**
 * Convert customer price back to provider cost (admin reference)
 */
export function priceToCost(price: number, markupPercent: number): number {
  return price / (1 + markupPercent / 100);
}

/**
 * Calculate billed price from base cost
 */
export function calculateBilledPrice(
  baseCost: number,
  isSelfHosted: boolean
): number {
  const markupPercent = isSelfHosted
    ? DEFAULT_MARKUP_RATES.SELF_HOSTED_PERCENT
    : DEFAULT_MARKUP_RATES.EXTERNAL_PERCENT;
  return costToPrice(baseCost, markupPercent);
}

// Legacy alias for backward compatibility
export const calculateBilledCost = calculateBilledPrice;

export function applyVolumeDiscount(
  totalSpend: number,
  billedAmount: number
): number {
  let discount = 0;
  for (const tier of VOLUME_DISCOUNTS) {
    if (totalSpend >= tier.minSpend) {
      discount = tier.discount;
    }
  }
  return billedAmount * (1 - discount);
```

```
}

export function isWithinFreeTier(
  usage: {
    tokens?: number;
    requests?: number;
    images?: number;
    videoSeconds?: number;
    audioMinutes?: number;
  }
): boolean {
  if ((usage.tokens ?? 0) > FREE_TIER.tokens) return false;
  if ((usage.requests ?? 0) > FREE_TIER.requests) return false;
  if ((usage.images ?? 0) > FREE_TIER.images) return false;
  if ((usage.videoSeconds ?? 0) > FREE_TIER.videoSeconds) return false;
  if ((usage.audioMinutes ?? 0) > FREE_TIER.audioMinutes) return false;
  return true;
}
```

packages/infrastructure/lib/config/providers/cost-metadata.types.ts

```
/**
 * Cost Metadata Types - RADIANT v4.12
 *
 * Internal cost data collected by metadata service.
 * VISIBLE TO: Administrators only
 */

// ============================================================================
// COST METADATA (Admin-Only)
// ============================================================================

export interface ModelCostMetadata {
  modelId: string;
  providerId: string;

  // Cost type
  costType: 'per_token' | 'per_request' | 'per_second' | 'per_image' | 'per_minute';

  // Base token costs (what RADIANT pays)
  baseCosts: {
    inputPer1kTokens: number;
    outputPer1kTokens: number;
  };

  // Context caching cost discounts
  cachingCosts: {
    supported: boolean;
```

```
    cachedInputPer1kTokens?: number;
    discountPercent?: number;           // e.g., 90 = 90% off
    minTokensForCaching?: number;
    cacheTTLSeconds?: number;
  };

  // Batch API cost discounts
  batchCosts: {
    supported: boolean;
    batchInputPer1kTokens?: number;
    batchOutputPer1kTokens?: number;
    discountPercent?: number;           // e.g., 50 = 50% off
    maxLatencyHours?: number;
  };

  // Long context costs
  longContextCosts?: {
    thresholdTokens: number;
    inputPer1kTokens: number;
    outputPer1kTokens: number;
  };

  // Special costs
  specialCosts: {
    perRequest?: number;
    perSecond?: number;
    perImage?: number;
    perMinute?: number;
    perSearchRequest?: number;
    perToolCall?: number;
  };

  // Source and freshness
  source: CostSource;
  lastVerifiedAt: string;
  lastUpdatedAt: string;
  nextSyncDueAt: string;

  // Estimated cost flag
  isEstimated: boolean;
  estimatedInfo?: EstimatedCostInfo;
}

export type CostSource =
  | 'provider_api'
  | 'provider_documentation'
  | 'admin_manual_entry'
  | 'registry_sync'
```

```
  | 'estimated_average'
  | 'historical_fallback';

export interface EstimatedCostInfo {
  reason: EstimatedCostReason;
  confidence: number;                    // 0.0 - 1.0

  sourceModels: Array<{
    modelId: string;
    displayName: string;
    providerId: string;
    inputCostPer1k: number;
    outputCostPer1k: number;
    similarityScore: number;
    weight: number;
  }>;

  calculationMethod: 'weighted_average' | 'simple_average' | 'nearest_model';
  similarityFactors: string[];
  estimatedAt: string;

  // Admin tracking
  adminNotified: boolean;
  adminNotifiedAt?: string;
  adminAdjusted: boolean;
  adminAdjustedBy?: string;
  adminAdjustedAt?: string;

  awaitingRealCost: boolean;
  expectedResolutionDate?: string;
}

export type EstimatedCostReason =
  | 'model_temporarily_unavailable'
  | 'cost_not_published'
  | 'new_model_pending'
  | 'provider_api_error'
  | 'provider_rate_limited'
  | 'deprecated_model'
  | 'preview_beta_model';

// ============================================================================
// THERMAL COST FACTORS (Self-Hosted)
// ============================================================================

export interface ThermalCostFactors {
  modelId: string;
  currentState: ThermalState;
```

```
  warmupCosts: {
    estimatedCost: number;
    estimatedDurationSeconds: number;
  };

  hourlyInfrastructureCosts: {
    OFF: number;
    COLD: number;
    WARM: number;
    HOT: number;
  };

  instanceDetails: {
    type: string;
    costPerHour: number;
  };

  perRequestCost: {
    amortizedCostPerRequest: number;
  };
}

export type ThermalState = 'OFF' | 'COLD' | 'WARM' | 'HOT' | 'AUTOMATIC';
```

packages/infrastructure/lib/config/providers/pricing-views.ts

```
/**
 * Role-Based Pricing Views - RADIANT v4.12
 *
 * Admin View:  Shows COST + MARKUP % + PRICE + Estimated flags
 * Client View: Shows PRICE only + Estimated price flag
 */

import { ModelCostMetadata, CostSource } from './cost-metadata.types';
import { costToPrice, DEFAULT_MARKUP_RATES } from './pricing.config';

// ============================================================================
// MARKUP CONFIGURATION
// ============================================================================

export interface MarkupConfiguration {
  defaults: {
    externalProvidersPercent: number;
    selfHostedModelsPercent: number;
  };

  providerOverrides: Record<string, {
```

```
    markupPercent: number;
    setBy: string;
    setAt: string;
    reason?: string;
  }>;

  modelOverrides: Record<string, {
    markupPercent: number;
    setBy: string;
    setAt: string;
    reason?: string;
    expiresAt?: string;
  }>;
}

export function getEffectiveMarkup(
  modelId: string,
  providerId: string,
  isExternal: boolean,
  config: MarkupConfiguration
): { percent: number; source: 'model' | 'provider' | 'default' } {
  // Priority: Model > Provider > Default
  const modelOverride = config.modelOverrides[modelId];
  if (modelOverride && (!modelOverride.expiresAt || new Date(modelOverride.expiresAt) > new Da
    return { percent: modelOverride.markupPercent, source: 'model' };
  }

  const providerOverride = config.providerOverrides[providerId];
  if (providerOverride) {
    return { percent: providerOverride.markupPercent, source: 'provider' };
  }

  return {
    percent: isExternal
      ? config.defaults.externalProvidersPercent
      : config.defaults.selfHostedModelsPercent,
    source: 'default'
  };
}

// ============================================================================
// ADMIN PRICING VIEW (Full Visibility)
// ============================================================================

export interface AdminPricingView {
  modelId: string;
  modelDisplayName: string;
  providerId: string;
```

```
providerDisplayName: string;

// Our COST (what RADIANT pays) - ADMIN ONLY
cost: {
  inputPer1M: string;                  // "$3.00"
  outputPer1M: string;                 // "$15.00"
  source: CostSource;
  lastUpdated: string;
  cachedInputPer1M?: string;
  batchInputPer1M?: string;
  batchOutputPer1M?: string;
};

// MARKUP configuration
markup: {
  percent: number;
  source: 'model' | 'provider' | 'default';
  isOverridden: boolean;
  overriddenBy?: string;
  overriddenAt?: string;
};

// Customer PRICE (what they pay)
price: {
  inputPer1M: string;                  // "$4.20"
  outputPer1M: string;                 // "$21.00"
  cachedInputPer1M?: string;
  batchInputPer1M?: string;
  batchOutputPer1M?: string;
};

// MARGIN (price - cost)
margin: {
  inputPer1M: string;
  outputPer1M: string;
  percentOfPrice: number;
};

// ESTIMATED FLAG
estimatedCost: {
  isEstimated: boolean;
  reason?: string;
  confidence?: number;
  sourceModels?: string[];
  adminCanAdjust: boolean;
  awaitingRealCost: boolean;
};
```

```
  // Admin actions
  actions: {
    canOverrideMarkup: boolean;
    canAdjustEstimatedCost: boolean;
    canForceSync: boolean;
  };
}

// ============================================================================
// CLIENT PRICING VIEW (Price Only)
// ============================================================================

export interface ClientPricingView {
  modelId: string;
  modelDisplayName: string;
  providerDisplayName: string;

  // PRICES only (what customer pays) - NO COST/MARGIN VISIBLE
  prices: {
    inputPer1M: string;                 // "$4.20 / 1M tokens"
    outputPer1M: string;                // "$21.00 / 1M tokens"
  };

  // Savings options
  savingsOptions?: {
    caching?: {
      available: boolean;
      cachedInputPer1M: string;
      savingsPercent: number;
      description: string;
    };
    batch?: {
      available: boolean;
      batchInputPer1M: string;
      batchOutputPer1M: string;
      savingsPercent: number;
      description: string;
    };
  };

  // ESTIMATED PRICE FLAG (shown to customer)
  estimatedPrice: {
    isEstimated: boolean;
    badge?: {
      text: string;                     // "Estimated"
      tooltip: string;                  // "Price is estimated and may change"
      variant: 'warning' | 'info';
    };
```

```
  };

  // Price comparison
  priceTier: 'budget' | 'standard' | 'premium';
  comparedToAverage: 'cheaper' | 'average' | 'expensive';
}

// ============================================================================
// VIEW FORMATTERS
// ============================================================================

export function formatAdminPricingView(
  costMetadata: ModelCostMetadata,
  markupConfig: MarkupConfiguration,
  isExternal: boolean,
  modelDisplayName: string,
  providerDisplayName: string
): AdminPricingView {
  const markup = getEffectiveMarkup(
    costMetadata.modelId,
    costMetadata.providerId,
    isExternal,
    markupConfig
  );

  const formatMoney = (amount: number) => `$${(amount * 1000).toFixed(2)}`;
  const toPrice = (cost: number) => costToPrice(cost, markup.percent);

  const inputCost = costMetadata.baseCosts.inputPer1kTokens;
  const outputCost = costMetadata.baseCosts.outputPer1kTokens;
  const inputPrice = toPrice(inputCost);
  const outputPrice = toPrice(outputCost);

  return {
    modelId: costMetadata.modelId,
    modelDisplayName,
    providerId: costMetadata.providerId,
    providerDisplayName,

    cost: {
      inputPer1M: formatMoney(inputCost),
      outputPer1M: formatMoney(outputCost),
      source: costMetadata.source,
      lastUpdated: costMetadata.lastUpdatedAt,
      cachedInputPer1M: costMetadata.cachingCosts.cachedInputPer1kTokens
        ? formatMoney(costMetadata.cachingCosts.cachedInputPer1kTokens)
        : undefined,
      batchInputPer1M: costMetadata.batchCosts.batchInputPer1kTokens
```

```
      ? formatMoney(costMetadata.batchCosts.batchInputPer1kTokens)
      : undefined,
    batchOutputPer1M: costMetadata.batchCosts.batchOutputPer1kTokens
      ? formatMoney(costMetadata.batchCosts.batchOutputPer1kTokens)
      : undefined,
  },

  markup: {
    percent: markup.percent,
    source: markup.source,
    isOverridden: markup.source !== 'default',
  },

  price: {
    inputPer1M: formatMoney(inputPrice),
    outputPer1M: formatMoney(outputPrice),
    cachedInputPer1M: costMetadata.cachingCosts.cachedInputPer1kTokens
      ? formatMoney(toPrice(costMetadata.cachingCosts.cachedInputPer1kTokens))
      : undefined,
    batchInputPer1M: costMetadata.batchCosts.batchInputPer1kTokens
      ? formatMoney(toPrice(costMetadata.batchCosts.batchInputPer1kTokens))
      : undefined,
    batchOutputPer1M: costMetadata.batchCosts.batchOutputPer1kTokens
      ? formatMoney(toPrice(costMetadata.batchCosts.batchOutputPer1kTokens))
      : undefined,
  },

  margin: {
    inputPer1M: formatMoney(inputPrice - inputCost),
    outputPer1M: formatMoney(outputPrice - outputCost),
    percentOfPrice: ((inputPrice - inputCost) / inputPrice) * 100,
  },

  estimatedCost: {
    isEstimated: costMetadata.isEstimated,
    reason: costMetadata.estimatedInfo?.reason,
    confidence: costMetadata.estimatedInfo?.confidence,
    sourceModels: costMetadata.estimatedInfo?.sourceModels.map(m => m.displayName),
    adminCanAdjust: costMetadata.isEstimated,
    awaitingRealCost: costMetadata.estimatedInfo?.awaitingRealCost || false,
  },

  actions: {
    canOverrideMarkup: true,
    canAdjustEstimatedCost: costMetadata.isEstimated,
    canForceSync: true,
  },
};
```

```
}

export function formatClientPricingView(
  costMetadata: ModelCostMetadata,
  markupConfig: MarkupConfiguration,
  isExternal: boolean,
  modelDisplayName: string,
  providerDisplayName: string,
  averagePrices: { input: number; output: number }
): ClientPricingView {
  const markup = getEffectiveMarkup(
    costMetadata.modelId,
    costMetadata.providerId,
    isExternal,
    markupConfig
  );

  const formatPrice = (amount: number) => `$${(amount * 1000).toFixed(2)} / 1M tokens`;
  const toPrice = (cost: number) => costToPrice(cost, markup.percent);

  const inputPrice = toPrice(costMetadata.baseCosts.inputPer1kTokens);

  // Determine price tier
  let priceTier: 'budget' | 'standard' | 'premium';
  let comparedToAverage: 'cheaper' | 'average' | 'expensive';

  if (inputPrice < averagePrices.input * 0.7) {
    priceTier = 'budget';
    comparedToAverage = 'cheaper';
  } else if (inputPrice > averagePrices.input * 1.3) {
    priceTier = 'premium';
    comparedToAverage = 'expensive';
  } else {
    priceTier = 'standard';
    comparedToAverage = 'average';
  }

  return {
    modelId: costMetadata.modelId,
    modelDisplayName,
    providerDisplayName,

    // ONLY PRICES - NO COST OR MARGIN
    prices: {
      inputPer1M: formatPrice(inputPrice),
      outputPer1M: formatPrice(toPrice(costMetadata.baseCosts.outputPer1kTokens)),
    },
```

```
    savingsOptions: {
      caching: costMetadata.cachingCosts.supported ? {
        available: true,
        cachedInputPer1M: formatPrice(toPrice(costMetadata.cachingCosts.cachedInputPer1kTokens
        savingsPercent: costMetadata.cachingCosts.discountPercent || 0,
        description: `${costMetadata.cachingCosts.discountPercent}% off with context caching`,
      } : undefined,
      batch: costMetadata.batchCosts.supported ? {
        available: true,
        batchInputPer1M: formatPrice(toPrice(costMetadata.batchCosts.batchInputPer1kTokens!)),
        batchOutputPer1M: formatPrice(toPrice(costMetadata.batchCosts.batchOutputPer1kTokens!)
        savingsPercent: costMetadata.batchCosts.discountPercent || 0,
        description: `${costMetadata.batchCosts.discountPercent}% off with batch API (24h)`,
      } : undefined,
    },

    // ESTIMATED FLAG - Customer sees this but doesn't know it's from estimated COST
    estimatedPrice: {
      isEstimated: costMetadata.isEstimated,
      badge: costMetadata.isEstimated ? {
        text: 'Estimated',
        tooltip: getClientEstimatedPriceTooltip(costMetadata.estimatedInfo?.reason),
        variant: 'warning',
      } : undefined,
    },

    priceTier,
    comparedToAverage,
  };
}

function getClientEstimatedPriceTooltip(reason?: string): string {
  const messages: Record<string, string> = {
    'model_temporarily_unavailable': 'Price is estimated - final pricing coming soon',
    'cost_not_published': 'Price not yet confirmed by provider',
    'new_model_pending': 'New model - pricing being finalized',
    'provider_api_error': 'Price temporarily unavailable',
    'provider_rate_limited': 'Price temporarily unavailable',
    'deprecated_model': 'Legacy model - estimated pricing',
    'preview_beta_model': 'Preview model - pricing may change',
  };
  return messages[reason || ''] || 'Price is estimated and may change';
}
```

packages/infrastructure/lib/config/providers/brain-price-optimizer.ts

```
/**
 * RADIANT Brain Price Optimizer - v4.12
```

```
 *
 * When users request cost optimization, the Brain optimizes based on PRICE
 * (what the customer pays), not our internal cost.
 */

import { ModelCostMetadata, ThermalCostFactors } from './cost-metadata.types';
import { getEffectiveMarkup, MarkupConfiguration } from './pricing-views';
import { costToPrice } from './pricing.config';

// ============================================================================
// BRAIN PRICE OPTIMIZATION REQUEST
// ============================================================================

export interface BrainPriceOptimizationRequest {
  request: {
    estimatedInputTokens: number;
    estimatedOutputTokens: number;
    canUseCaching: boolean;
    cachedTokenCount?: number;
    canUseBatch: boolean;
    isLongContext: boolean;
    requiresToolCalls: boolean;
    estimatedToolCalls?: number;
  };

  requiredCapabilities: string[];
  pricePreference: 'minimize' | 'balance' | 'ignore';
  minQualityScore?: number;
  includeEstimatedPricing: boolean;

  budgetConstraints?: {
    maxPricePerRequest?: number;        // Customer's max price
    remainingMonthlyBudget?: number;    // Customer's remaining budget
  };
}

// ============================================================================
// MODEL PRICE ANALYSIS (Brain uses this)
// ============================================================================

export interface ModelPriceAnalysis {
  modelId: string;
  providerId: string;
  displayName: string;
  capabilities: string[];
  qualityScore: number;

  // === CALCULATED PRICES (What customer pays) - Brain optimizes on these ===
```

```
  calculatedPrices: {
    standardPrice: number;
    withCachingPrice?: number;
    withBatchPrice?: number;
    thermalOverheadPrice?: number;      // Self-hosted warm-up (marked up)
    totalPrice: number;

    breakdown: {
      inputTokensPrice: number;
      outputTokensPrice: number;
      cachingDiscount: number;
      batchDiscount: number;
      thermalPrice: number;
      toolCallsPrice: number;
    };
  };

  // === ADMIN-ONLY COST INFO (Not used for optimization) ===
  adminCostInfo: {
    totalCost: number;
    markupPercent: number;
    marginAmount: number;
  };

  // Flags
  flags: {
    isEstimatedPrice: boolean;
    isAvailable: boolean;
    requiresWarmup: boolean;
    warmupWaitSeconds?: number;
  };

  // Rankings (based on PRICE)
  rankings: {
    priceRank: number;
    valueRank: number;
    isCheapest: boolean;
    isBestValue: boolean;
    isRecommended: boolean;
  };
}

// ============================================================================
// BRAIN OPTIMIZATION RESULT
// ============================================================================

export interface BrainPriceOptimizationResult {
  selectedModel: ModelPriceAnalysis;
```

```
  selectionReason: string;
  allCandidates: ModelPriceAnalysis[];

  // All in PRICE terms (what customer pays)
  summary: {
    selectedPrice: number;
    cheapestPrice: number;
    averagePrice: number;
    savingsVsAverage: number;
    savingsPercent: number;
  };

  // Warnings for client (price-focused)
  clientWarnings: Array<{
    type: 'estimated_price' | 'warmup_delay' | 'near_budget';
    message: string;
  }>;

  // Warnings for admin (cost/margin-focused)
  adminWarnings: Array<{
    type: 'low_margin' | 'estimated_cost' | 'high_volume_discount';
    message: string;
    modelId?: string;
  }>;
}

// ============================================================================
// PRICE OPTIMIZATION FUNCTION
// ============================================================================

export function optimizeForPrice(
  candidates: ModelPriceAnalysis[],
  request: BrainPriceOptimizationRequest
): BrainPriceOptimizationResult {
  // Filter by capabilities
  let eligible = candidates.filter(c =>
    request.requiredCapabilities.every(cap => c.capabilities.includes(cap))
  );

  // Filter by availability
  eligible = eligible.filter(c => c.flags.isAvailable);

  // Filter estimated if not allowed
  if (!request.includeEstimatedPricing) {
    eligible = eligible.filter(c => !c.flags.isEstimatedPrice);
  }

  // Filter by quality
```

```
if (request.minQualityScore) {
  eligible = eligible.filter(c => c.qualityScore >= request.minQualityScore!);
}

// Filter by budget (PRICE constraint)
if (request.budgetConstraints?.maxPricePerRequest) {
  eligible = eligible.filter(c =>
    c.calculatedPrices.totalPrice <= request.budgetConstraints!.maxPricePerRequest!
  );
}

// Sort by PRICE preference
if (request.pricePreference === 'minimize') {
  // Sort by lowest PRICE
  eligible.sort((a, b) => a.calculatedPrices.totalPrice - b.calculatedPrices.totalPrice);
} else if (request.pricePreference === 'balance') {
  // Sort by value (quality per dollar of PRICE)
  eligible.sort((a, b) =>
    (b.qualityScore / b.calculatedPrices.totalPrice) -
    (a.qualityScore / a.calculatedPrices.totalPrice)
  );
}
// If 'ignore', keep original quality-based order

const selected = eligible[0];
const prices = eligible.map(c => c.calculatedPrices.totalPrice);
const cheapestPrice = Math.min(...prices);
const averagePrice = prices.reduce((a, b) => a + b, 0) / prices.length;

// Build client warnings
const clientWarnings: BrainPriceOptimizationResult['clientWarnings'] = [];
if (selected.flags.isEstimatedPrice) {
  clientWarnings.push({
    type: 'estimated_price',
    message: 'Price is estimated and may change',
  });
}
if (selected.flags.requiresWarmup) {
  clientWarnings.push({
    type: 'warmup_delay',
    message: `Model requires ~${selected.flags.warmupWaitSeconds}s warm-up`,
  });
}

// Build admin warnings
const adminWarnings: BrainPriceOptimizationResult['adminWarnings'] = [];
const marginPercent = (selected.adminCostInfo.marginAmount / selected.calculatedPrices.totall
if (marginPercent < 20) {
```

```
    adminWarnings.push({
      type: 'low_margin',
      message: `Low margin (${marginPercent.toFixed(1)}%) on ${selected.displayName}`,
      modelId: selected.modelId,
    });
  }
  if (selected.flags.isEstimatedPrice) {
    adminWarnings.push({
      type: 'estimated_cost',
      message: `Cost for ${selected.displayName} is estimated - verify margin`,
      modelId: selected.modelId,
    });
  }

  return {
    selectedModel: selected,
    selectionReason: buildSelectionReason(selected, request),
    allCandidates: eligible,
    summary: {
      selectedPrice: selected.calculatedPrices.totalPrice,
      cheapestPrice,
      averagePrice,
      savingsVsAverage: averagePrice - selected.calculatedPrices.totalPrice,
      savingsPercent: ((averagePrice - selected.calculatedPrices.totalPrice) / averagePrice) *
    },
    clientWarnings,
    adminWarnings,
  };
}

function buildSelectionReason(
  model: ModelPriceAnalysis,
  request: BrainPriceOptimizationRequest
): string {
  const reasons: string[] = [];

  if (model.rankings.isCheapest) {
    reasons.push('lowest price');
  } else if (model.rankings.isBestValue) {
    reasons.push('best value (quality per dollar)');
  }

  if (request.pricePreference === 'minimize') {
    reasons.push('optimized for minimum price');
  } else if (request.pricePreference === 'balance') {
    reasons.push('balanced price and quality');
  }
```

```
  if (model.flags.isEstimatedPrice) {
    reasons.push('note: estimated pricing');
  }

  return `Selected ${model.displayName}: ${reasons.join(', ')}`;
}

/**
 * Calculate prices for a model given request parameters
 */
export function calculateModelPrices(
  costMetadata: ModelCostMetadata,
  thermalCosts: ThermalCostFactors | null,
  markupConfig: MarkupConfiguration,
  isExternal: boolean,
  request: BrainPriceOptimizationRequest['request']
): ModelPriceAnalysis['calculatedPrices'] & { adminCostInfo: ModelPriceAnalysis['adminCostInfo
  const markup = getEffectiveMarkup(
    costMetadata.modelId,
    costMetadata.providerId,
    isExternal,
    markupConfig
  );

  const toPrice = (cost: number) => costToPrice(cost, markup.percent);

  // Calculate base costs
  let inputCost = costMetadata.baseCosts.inputPer1kTokens * (request.estimatedInputTokens / 10
  let outputCost = costMetadata.baseCosts.outputPer1kTokens * (request.estimatedOutputTokens /

  // Apply caching discount to cost
  let cachingDiscount = 0;
  if (request.canUseCaching && costMetadata.cachingCosts.supported && request.cachedTokenCount
    const cachedCost = costMetadata.cachingCosts.cachedInputPer1kTokens! * (request.cachedToke
    const standardCost = costMetadata.baseCosts.inputPer1kTokens * (request.cachedTokenCount /
    cachingDiscount = standardCost - cachedCost;
    inputCost -= cachingDiscount;
  }

  // Apply batch discount to cost
  let batchDiscount = 0;
  if (request.canUseBatch && costMetadata.batchCosts.supported) {
    const discountPercent = costMetadata.batchCosts.discountPercent || 0;
    batchDiscount = (inputCost + outputCost) * (discountPercent / 100);
    inputCost *= (1 - discountPercent / 100);
    outputCost *= (1 - discountPercent / 100);
  }
```

```
  // Thermal cost for self-hosted
  let thermalCost = 0;
  if (thermalCosts && thermalCosts.currentState === 'COLD') {
    thermalCost = thermalCosts.warmupCosts.estimatedCost;
  }

  // Tool calls cost
  let toolCallsCost = 0;
  if (request.requiresToolCalls && costMetadata.specialCosts.perToolCall) {
    toolCallsCost = costMetadata.specialCosts.perToolCall * (request.estimatedToolCalls || 1);
  }

  const totalCost = inputCost + outputCost + thermalCost + toolCallsCost;
  const totalPrice = toPrice(totalCost);

  return {
    standardPrice: toPrice(inputCost + outputCost),
    withCachingPrice: cachingDiscount > 0 ? toPrice(inputCost + outputCost) : undefined,
    withBatchPrice: batchDiscount > 0 ? toPrice(inputCost + outputCost) : undefined,
    thermalOverheadPrice: thermalCost > 0 ? toPrice(thermalCost) : undefined,
    totalPrice,

    breakdown: {
      inputTokensPrice: toPrice(inputCost),
      outputTokensPrice: toPrice(outputCost),
      cachingDiscount: toPrice(cachingDiscount),
      batchDiscount: toPrice(batchDiscount),
      thermalPrice: toPrice(thermalCost),
      toolCallsPrice: toPrice(toolCallsCost),
    },

    adminCostInfo: {
      totalCost,
      markupPercent: markup.percent,
      marginAmount: totalPrice - totalCost,
    },
  };
}
```

packages/infrastructure/lib/config/providers/estimated-cost-alerts.ts

```
/**
 * Estimated Cost Alert System - RADIANT v4.12
 *
 * Alerts administrators when costs are estimated.
 * Admin can adjust estimated costs until real costs are found.
 */
```

```typescript
import { ModelCostMetadata, EstimatedCostReason, CostSource } from './cost-metadata.types';

export interface EstimatedCostAlert {
  id: string;
  createdAt: string;
  severity: 'info' | 'warning' | 'critical';

  modelId: string;
  modelDisplayName: string;
  providerId: string;
  providerDisplayName: string;

  reason: EstimatedCostReason;
  estimatedCost: {
    inputPer1k: number;
    outputPer1k: number;
  };
  confidence: number;

  sourceModels: Array<{
    modelId: string;
    displayName: string;
    inputCostPer1k: number;
    outputCostPer1k: number;
    similarityScore: number;
  }>;

  // Impact on customer price
  affectedPrice: {
    inputPer1k: number;
    outputPer1k: number;
    markupPercent: number;
  };

  recommendation: string;

  // Status workflow
  status: 'pending' | 'acknowledged' | 'adjusted' | 'resolved';
  acknowledgedBy?: string;
  acknowledgedAt?: string;
  adjustedBy?: string;
  adjustedAt?: string;
  adjustedCost?: {
    inputPer1k: number;
    outputPer1k: number;
  };
  resolvedBy?: string;
  resolvedAt?: string;
```

```
  resolutionSource?: CostSource;

  notifications: Array<{
    adminId: string;
    adminEmail: string;
    channel: 'email' | 'slack' | 'dashboard';
    sentAt: string;
  }>;
}

export function createEstimatedCostAlert(
  costMetadata: ModelCostMetadata,
  affectedPrice: { inputPer1k: number; outputPer1k: number; markupPercent: number },
  modelDisplayName: string,
  providerDisplayName: string
): EstimatedCostAlert {
  const info = costMetadata.estimatedInfo!;

  // Determine severity based on confidence
  let severity: 'info' | 'warning' | 'critical';
  if (info.confidence > 0.8) severity = 'info';
  else if (info.confidence > 0.5) severity = 'warning';
  else severity = 'critical';

  return {
    id: `alert_${costMetadata.modelId}_${Date.now()}`,
    createdAt: new Date().toISOString(),
    severity,

    modelId: costMetadata.modelId,
    modelDisplayName,
    providerId: costMetadata.providerId,
    providerDisplayName,

    reason: info.reason,
    estimatedCost: {
      inputPer1k: costMetadata.baseCosts.inputPer1kTokens,
      outputPer1k: costMetadata.baseCosts.outputPer1kTokens,
    },
    confidence: info.confidence,

    sourceModels: info.sourceModels.map(m => ({
      modelId: m.modelId,
      displayName: m.displayName,
      inputCostPer1k: m.inputCostPer1k,
      outputCostPer1k: m.outputCostPer1k,
      similarityScore: m.similarityScore,
    })),
```

```
    affectedPrice,

    recommendation: buildRecommendation(info.confidence, severity),

    status: 'pending',
    notifications: [],
  };
}

function buildRecommendation(confidence: number, severity: string): string {
  if (severity === 'critical') {
    return 'Low confidence estimate. Recommend manually verifying cost with provider or adjusti
  } else if (severity === 'warning') {
    return 'Moderate confidence estimate. Consider verifying with provider documentation.';
  }
  return 'High confidence estimate based on similar models. Will auto-update when real cost is
}

/**
 * Admin action: Adjust estimated cost
 */
export function adjustEstimatedCost(
  alert: EstimatedCostAlert,
  newCost: { inputPer1k: number; outputPer1k: number },
  adminId: string
): EstimatedCostAlert {
  return {
    ...alert,
    status: 'adjusted',
    adjustedBy: adminId,
    adjustedAt: new Date().toISOString(),
    adjustedCost: newCost,
  };
}

/**
 * Resolve alert when real cost is found
 */
export function resolveAlert(
  alert: EstimatedCostAlert,
  source: CostSource,
  adminId?: string
): EstimatedCostAlert {
  return {
    ...alert,
    status: 'resolved',
    resolvedBy: adminId || 'system',
```

```
    resolvedAt: new Date().toISOString(),
    resolutionSource: source,
  };
}
```

PART 7: ESTIMATED PRICES

External Provider Costs (Example Monthly Usage)

Provider	Model	1M Tokens	10M Tokens	100M Tokens
OpenAI	GPT-4o	$17.50	$175	$1,750
OpenAI	GPT-4o-mini	$1.05	$10.50	$105
Anthropic	Claude Opus 4	$126	$1,260	$12,600
Anthropic	Claude Sonnet 4	$25.20	$252	$2,520
Anthropic	Claude 3.5 Haiku	$6.72	$67.20	$672
Google	Gemini 2.0 Flash	$0.70	$7.00	$70
DeepSeek	DeepSeek Chat	$1.92	$19.20	$192
DeepSeek	DeepSeek R1	$3.84	$38.40	$384

All prices include 40% markup

Database Costs (Monthly)

Component	Tier 1	Tier 2	Tier 3	Tier 4	Tier 5
Aurora (Primary)	~$29	~$75	~$200	~$500	~$1,500
DynamoDB	~$5	~$15	~$50	~$150	~$500
ElastiCache	N/A	~$25	~$75	~$200	~$500
Database Total	**~$34**	**~$115**	**~$325**	**~$850**	**~$2,500**

PROVIDER SUMMARY

Text Generation (7 providers, 15+ models)

- **OpenAI**: GPT-4o, GPT-4o-mini, O1, O1-mini, O3-mini
- **Anthropic**: Claude Opus 4, Sonnet 4, Haiku 3.5
- **Google**: Gemini 2.0 Flash, 1.5 Pro
- **xAI**: Grok 3, Grok 3 Mini
- **DeepSeek**: Chat, Reasoner (R1)
- **Mistral**: Large, Small, Codestral
- **Cohere**: Command R+, Command R

Image Generation (5 providers, 10+ models)

- **OpenAI**: DALL-E 3, DALL-E 3 HD
- **Stability**: SDXL, SD3 Large, Ultra
- **FLUX**: Pro 1.1, Pro Ultra, Schnell
- **Ideogram**: V2, V2 Turbo
- **Midjourney**: V6

Video Generation (5 providers, 7+ models)

- **Runway**: Gen-3 Alpha, Turbo
- **Luma**: Dream Machine, Ray 2
- **Pika**: Pika 2.0
- **Kling**: V1.5 Pro
- **Google Veo**: Veo 2

Audio (4 providers, 7+ models)

- **OpenAI**: TTS-1, TTS-1-HD, Whisper
- **ElevenLabs**: Multilingual V2, Turbo V2.5
- **Deepgram**: Nova-2, Nova-2 Medical
- **AssemblyAI**: Best

Embeddings (2 providers, 5 models)

- **OpenAI**: text-embedding-3-small/large
- **Voyage**: voyage-3, voyage-code-3

Search (3 providers)

- **Perplexity**: Sonar, Sonar Pro
- **Exa**: Neural Search
- **Tavily**: Search API

3D Generation (2 providers)

- **Meshy**: V3, V3 Turbo
- **Tripo**: V2

Total: 21 external providers, 50+ models

NEXT PROMPTS

Continue with: - **Prompt 8**: Admin Web Dashboard (Next.js) - **Prompt 9**: Assembly & Deployment Guide

---

End of Prompt 7: External Providers & Database Schema/Migrations RADIANT v2.2.0 - December 2024

â• â

END OF SECTION 7

â• â

â• â