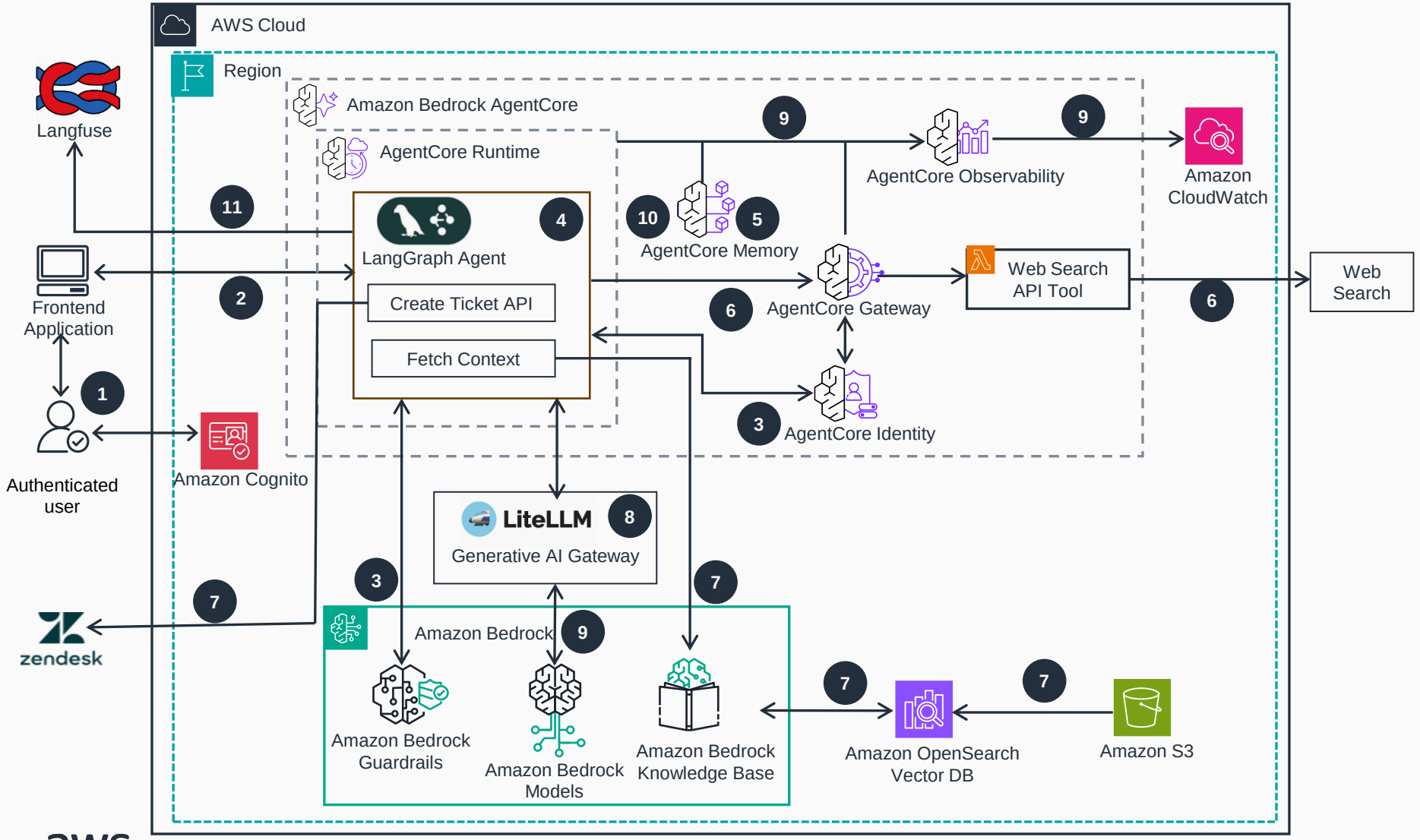
Guidance for Building Agentic AI Foundation Solutions on AWS

Agentic AI Operational Foundations

This architecture diagram illustrates how to effectively support applications using agentic AI on AWS. It shows the key components and their interactions, providing an overview of the architecture's structure and functionality. The guidance enables authenticated users to interact with AI-powered agents through a frontend application, where Amazon Bedrock AgentCore orchestrates LangGraph-based agents that access knowledge bases, perform web searches, and create support tickets. The architecture incorporates comprehensive security, scalable storage, external integrations, and monitoring capabilities to deliver intelligent, contextual customer support experiences.



1. An authenticated user interacts with the Frontend Application, which validates their identity through **Amazon Cognito.**
2. The frontend application forwards the user's request to the **Amazon Bedrock AgentCore runtime**.
3. **Amazon Bedrock AgentCore Identity** validates user permissions and establishes secure service-to-service authentication, while **Amazon Bedrock Guardrails** evaluate the incoming request to ensure it meets safety and compliance requirements before proceeding.
4. **Amazon Bedrock AgentCore Runtime** initializes the LangGraph Agent, which acts as the central orchestrator that will make decisions about the workflow path and coordinate all subsequent processing steps.
5. The LangGraph Agent queries **Amazon Bedrock AgentCore Memory** to retrieve conversation history and contextual information, enabling continuity across sessions and personalized interactions based on previous exchanges.
6. The LangGraph Agent analyzes request types and context to determine processing paths, then connects to **Amazon Bedrock AgentCore Gateway** to discover and invoke available tools such as the **Web Search API** for gathering external information.
7. The LangGraph Agent retrieves domain-specific information through the **Fetch Context** tool powered by **Amazon Bedrock Knowledge Base**, **Amazon S3**, and **Amazon OpenSearch** Vector DB, or creates support tickets via **Create Ticket API** based on request analysis.

Reviewed for technical accuracy January 29, 2026

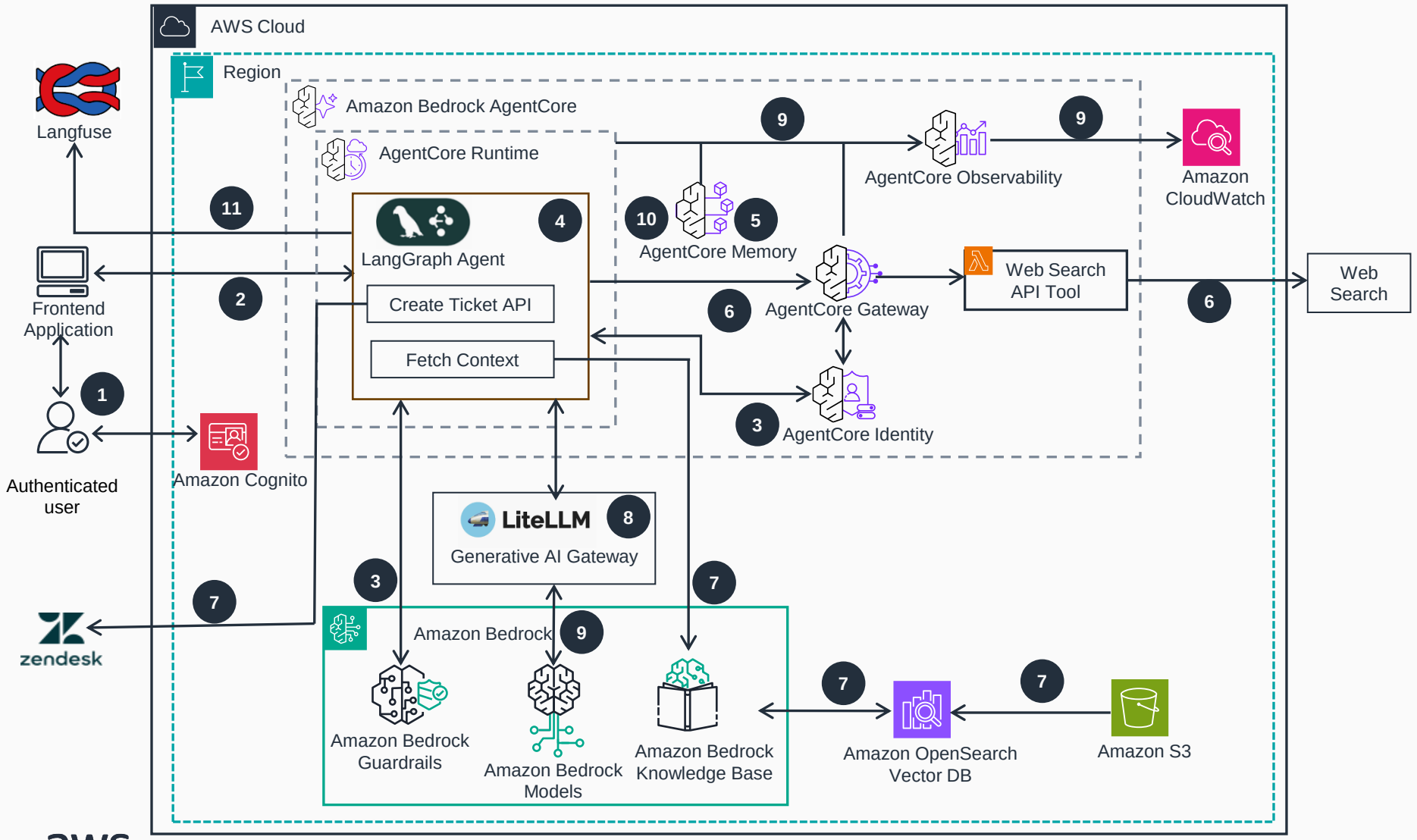
© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Reference Architecture

Guidance for Building Agentic AI Foundation Solutions on AWS

Agentic AI Operational Foundations

This architecture diagram illustrates how to effectively support applications using agentic AI on AWS. It shows the key components and their interactions, providing an overview of the architecture's structure and functionality. The guidance enables authenticated users to interact with AI-powered agents through a frontend application, where Amazon Bedrock AgentCore orchestrates LangGraph-based agents that access knowledge bases, perform web searches, and create support tickets. The architecture incorporates comprehensive security, scalable storage, external integrations, and monitoring capabilities to deliver intelligent, contextual customer support experiences.



8 The LangGraph Agent orchestrates response generation using **Amazon Bedrock Models**, via **Multi-Provider Generative AI Gateway** (Litellm) combining retrieved knowledge from the **Amazon Bedrock Knowledge Base**, tools outputs from external services, and conversation context to create intelligent, contextually appropriate responses.

9 Throughout the entire LangGraph workflow execution, **Amazon Bedrock AgentCore Observability** tracks system performance, logs decision paths, tool usage patterns, and interaction outcomes, while **Amazon CloudWatch** monitors the overall system health and collects metrics from all AWS services.

10 The LangGraph Agent updates **Amazon Bedrock AgentCore Memory** with new conversation state and user interaction data, then coordinates the final response delivery through the **Amazon Bedrock AgentCore runtime** back to the Frontend application and ultimately to the authenticated user.

11 The workflow state and observability data flows to **Langfuse** for comprehensive tracking and analysis of agent interactions, providing detailed insights into conversation quality and system performance for continuous improvement.

Reviewed for technical accuracy January 29, 2026

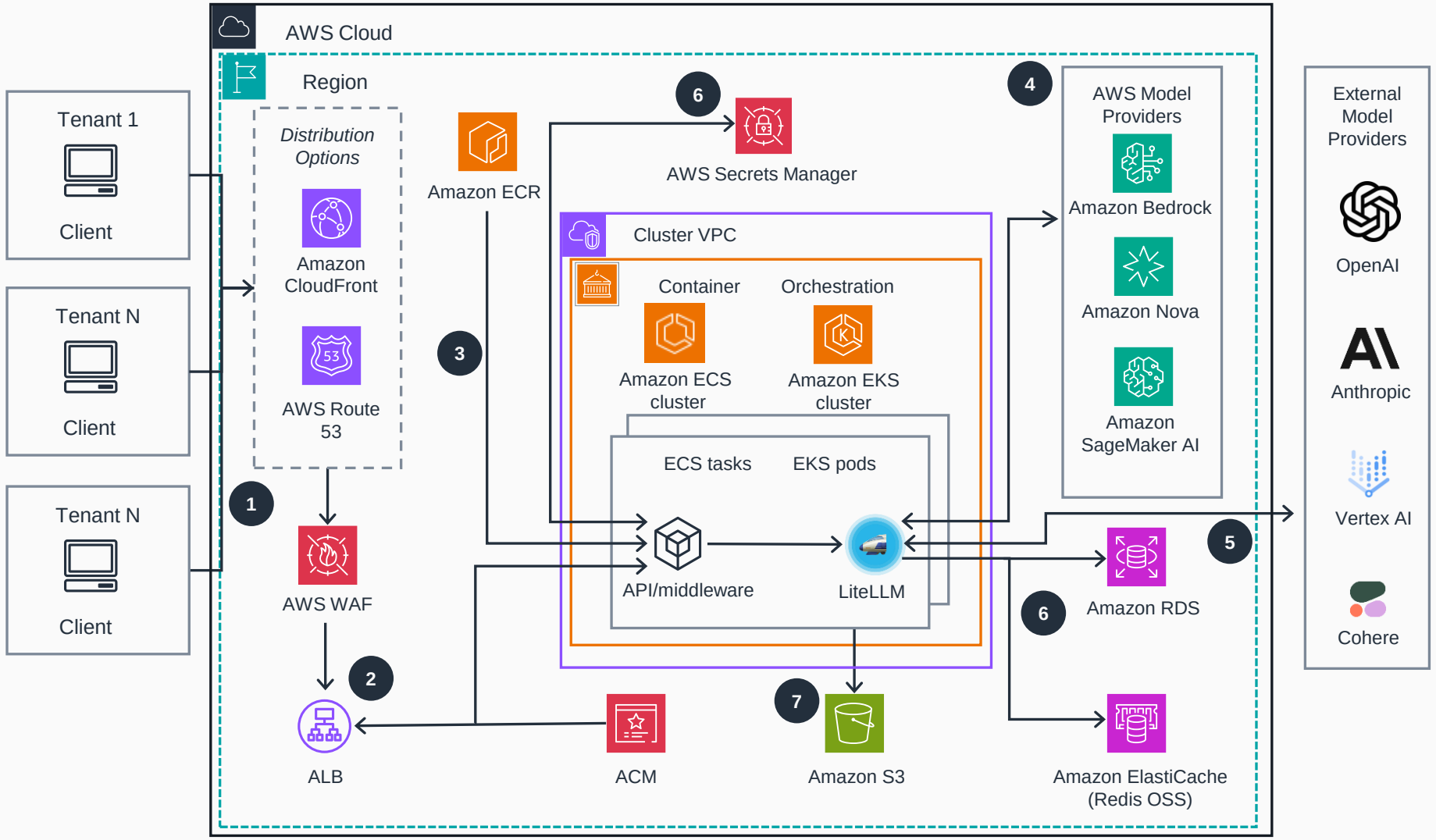
© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Reference Architecture

Guidance for Multi-Provider Generative AI Gateway on AWS

Multi-Provider Generative AI Gateway

This architecture diagram demonstrates how to deploy with Amazon ECS or Amazon EKS container orchestration running on AWS.



1. Tenants and client applications access the LiteLLM gateway proxy API through the **Amazon Route 53** URL endpoint or **Amazon CloudFront,** which is protected against common web exploits and bots using **AWS WAF**.

2. **AWS WAF** forwards requests to **Application Load Balancer (ALB)** to automatically distribute incoming application traffic to **Amazon Elastic Container Service (Amazon ECS)** tasks or **Amazon Elastic Kubernetes Service (Amazon EKS)** pods running generative AI gateway containers. TLS/SSL encryption secures traffic to the load balancer using a certificate issued by **AWS Certificate Manager (ACM).**

3. Container images for API/middleware and LiteLLM applications are built during guidance deployment and pushed to **Amazon Elastic Container Registry (Amazon ECR)**. They are used for deployment to **Amazon ECS** on **AWS Fargate** or **Amazon EKS** clusters that run these applications as containers in **ECS** tasks or **EKS** pods, respectively. LiteLLM provides a unified application interface for configuration and interacting with LLM providers. The API/middleware integrates natively with **Amazon Bedrock** to enable features not supported by the LiteLLM opensource project.

4. Models hosted on **Amazon Bedrock** and **Amazon Nova** provide model access, guardrails, prompt caching, and routing to enhance the AI gateway and additional controls for clients through a unified API. Model access is also available for models deployed on Amazon SageMaker AI. Access to required **Amazon Bedrock** models must be properly configured.

Reviewed for technical accuracy January 29, 2026

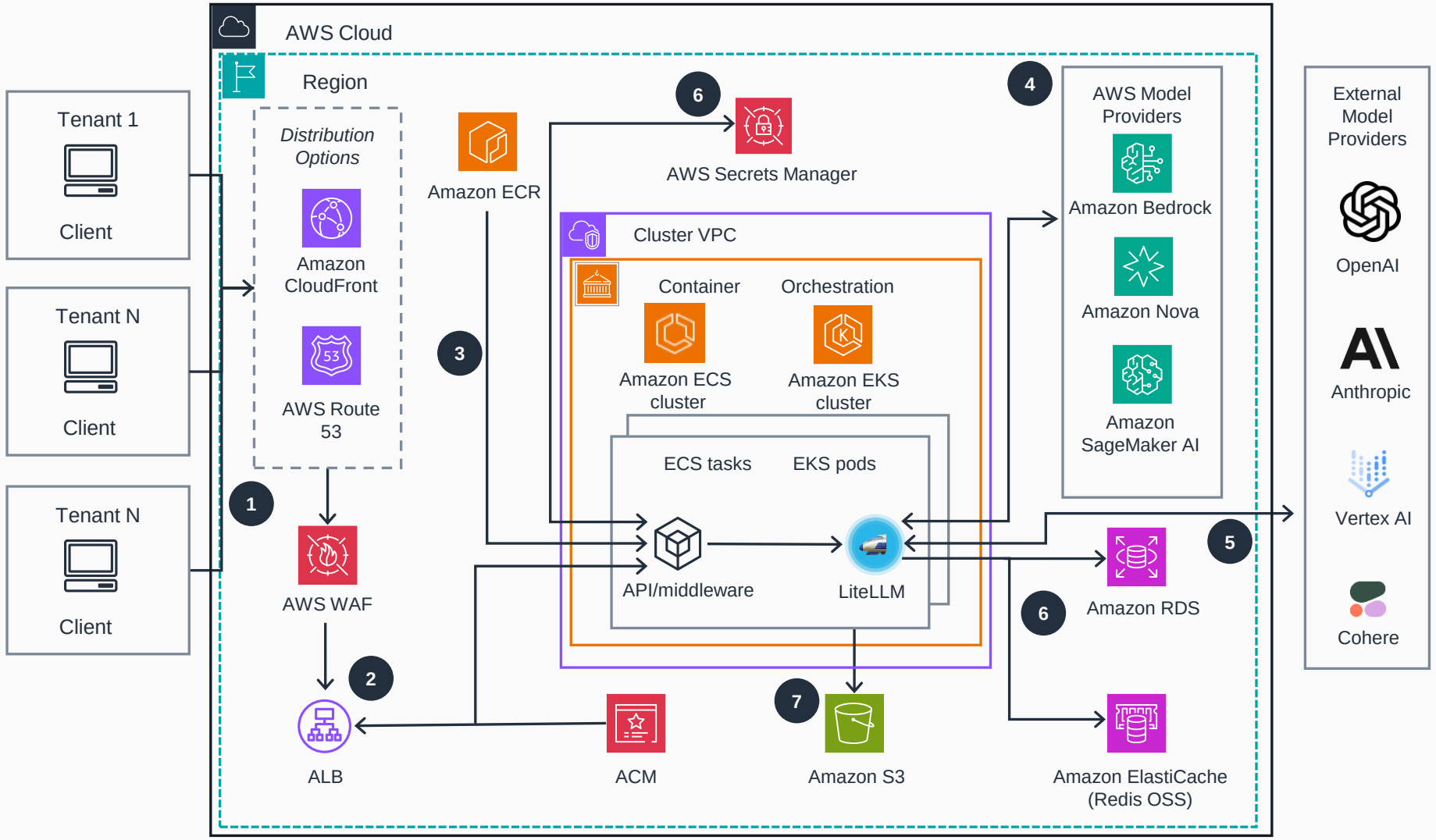
© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Reference Architecture

Guidance for Multi-Provider Generative AI Gateway on AWS

Multi-Provider Generative AI Gateway

This architecture diagram demonstrates how to deploy with Amazon ECS or Amazon EKS container orchestration running on AWS.



5 External model providers (such as OpenAI, Anthropic, or Vertex AI) are configured using the LiteLLM Admin UI to enable additional model access through LiteLLM's unified application interface. Integrate pre-existing configurations of third-party providers into the gateway using LiteLLM APIs.

6 LiteLLM integrates with **Amazon ElastiCache (Redis OSS), Amazon Relational Database Service (Amazon RDS)**, and **AWS Secrets Manager** services. **Amazon ElastiCache** enables multi-tenant distribution of application settings and prompt caching. **Amazon RDS** enables persistence of virtual API keys and other configuration settings provided by LiteLLM. **Secrets Manager** stores external model provider credentials and other sensitive settings securely.

7 LiteLLM and the API/middleware store application sends logs to the dedicated **Amazon Simple Storage Service (Amazon S3)** storage bucket for troubleshooting and access analysis.

Reviewed for technical accuracy January 29, 2026

© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Reference Architecture