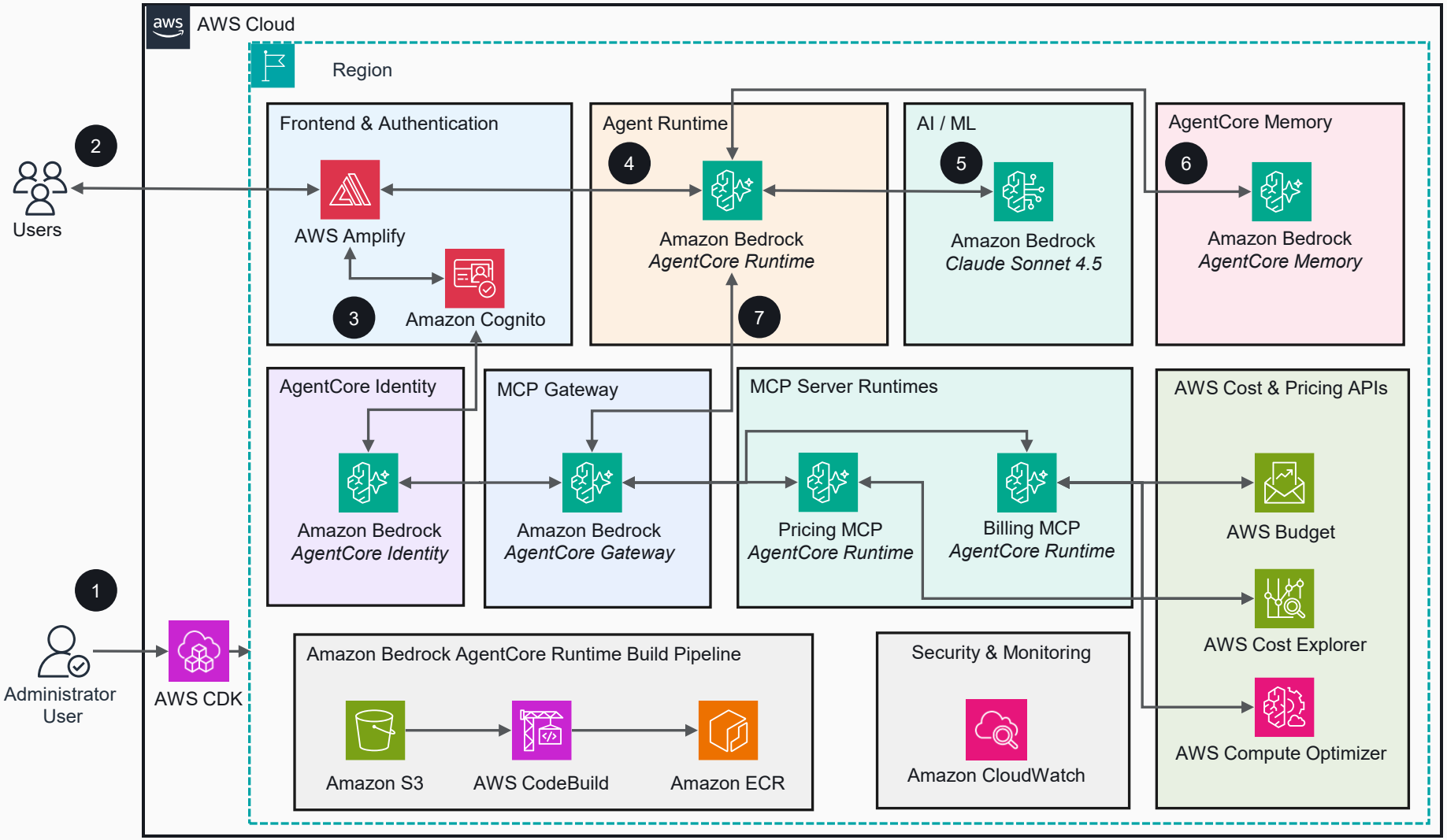
# Guidance for Cost Analysis and Optimization with Amazon Bedrock AgentCore on AWS

This architecture diagram shows how to build a conversational FinOps agent that consolidates AWS cost data using Bedrock AgentCore, MCP servers, and natural language.




1. Administrative users uses **AWS CDK** to deploy the the guidance with a single script, uploading application code to **Amazon S3** bucket and triggering **AWS CodeBuild** to build container images stored in **Amazon Elastic Container Registry (ECR)** for the **Amazon Bedrock AgentCore** runtime.
2. Users access the web application hosted on AWS Amplify, which serves the frontend interface.
3. Users authenticate with **Amazon Cognito**. Cognito validates your credentials and returns temporary AWS credentials from the Identity Pool.
4. The frontend sends the user's question to the **Amazon Bedrock AgentCore Runtime** — a secure, serverless environment that hosts and runs the agent with session isolation — using the temporary AWS credentials to call InvokeAgentRuntime via IAM SigV4 authentication.
5. The Strands agent — an open-source agent framework supported natively by AgentCore Runtime — sends the user's question with 24 tool definitions to Claude Sonnet 4.5 on Amazon Bedrock, a fully managed service providing secure access to foundation models. The model selects the appropriate cost analysis tool.
6. **Amazon Bedrock AgentCore** Memory — a fully managed service for session and long-term memory — maintains conversation context across interactions, enabling the agent to understand follow-up questions and provide coherent multi-turn cost analysis without users repeating context.
7. The agent routes the tool call to AgentCore Gateway using IAM SigV4 authentication via InvokeGateway.

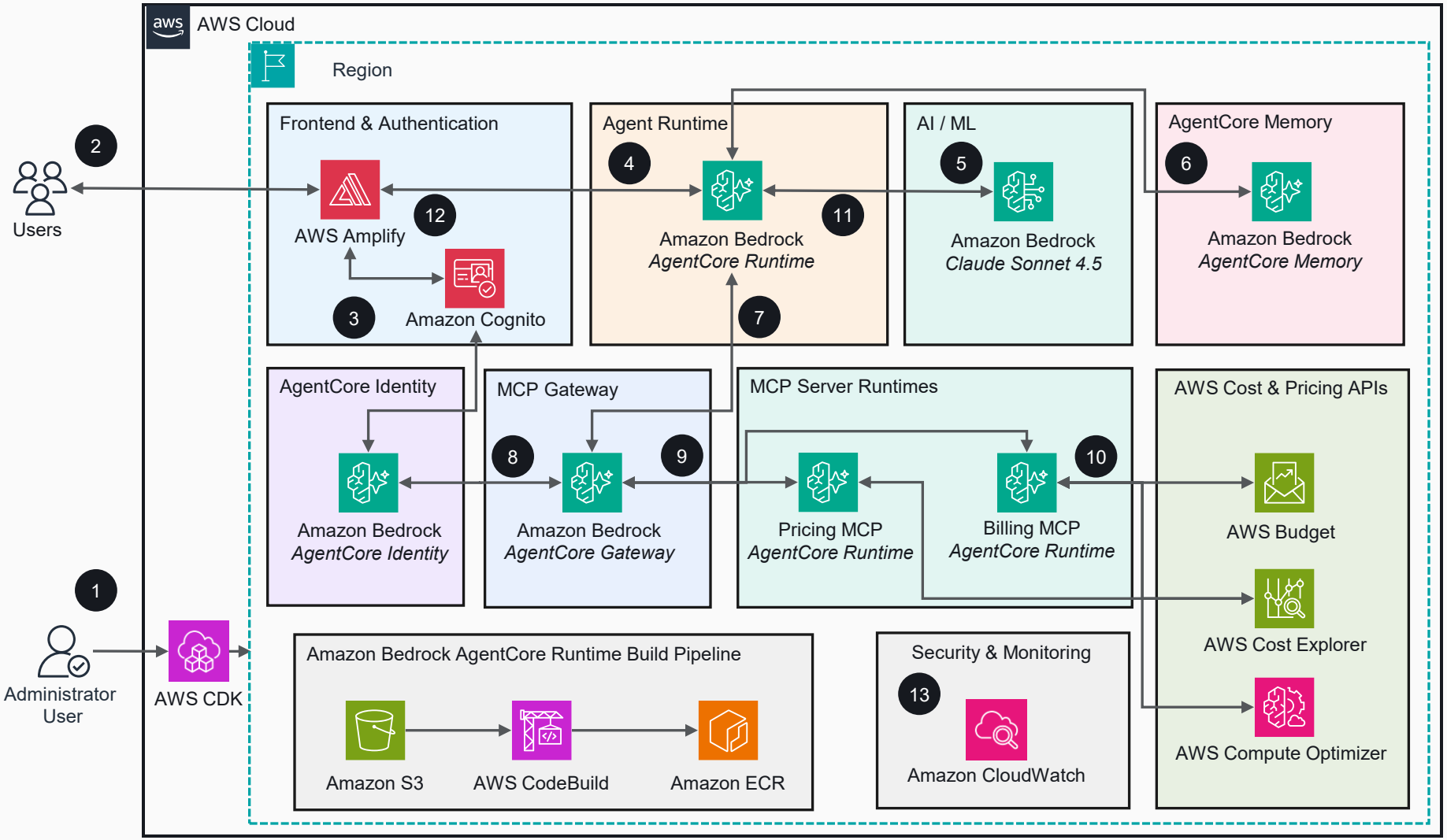
*Reviewed for technical accuracy June 15, 2026*



**AWS Reference Architecture**

# Guidance for Cost Analysis and Optimization with Amazon Bedrock AgentCore on AWS

This architecture diagram shows how to build a conversational FinOps agent that consolidates AWS cost data using Bedrock AgentCore, MCP servers, and natural language.




8 AgentCore Identity — a secure identity and credential management service purpose-built for AI agents — retrieves an OAuth 2.0 access token from the registered Cognito M2M credential provider (using the client credentials grant) and attaches it to the outbound MCP request, enabling the agent to securely access the billing tools.

9 The Gateway sends the Model Context Protocol (MCP) tool call request with the OAuth token to the Billing MCP Runtime.

10 The Billing MCP Runtime queries the appropriate AWS cost services: AWS Cost Explorer for historical cost and usage data, AWS Budgets for budget status and alerts, AWS Compute Optimizer for rightsizing recommendations, and AWS Cost & Pricing APIs for current service pricing — providing comprehensive FinOps coverage through a single conversational interface.

11 Cost data flows back through the chain. The agent sends it to Amazon Bedrock, where Claude generates a natural language summary of your costs.

12 The formatted response displays the cost breakdown in users' chat interface.

13 **Amazon CloudWatch** provides centralized monitoring, logging, and alerting across all guidance services for complete observability

*Reviewed for technical accuracy June 15, 2026*



**AWS Reference Architecture**