CLOUD AI DEVELOPER GUIDE SERIES — FINAL GUIDE

Guide #6

# Master Cheat Sheet

Every model · Every API endpoint · Every SDK command
Pricing reference · Decision flowchart · Migration guide

**AWS Bedrock** | **Azure OpenAI** | **GCP Vertex AI**

✓ Guide 1: First API Call
✓ Guide 2: RAG on Cloud
✓ Guide 3: Multi-Agent
✓ Guide 4: Guardrails
✓ Guide 5: Production Arch
★ Guide 6: Cheat Sheet

cloudai.dev · 2025 Edition · Print & keep

# AWS Bedrock

Complete reference — models · endpoints · SDK · pricing · CLI

## Foundation Models

| Model ID | Name | Context | Best For | In $/1K tokens | Out $/1K tokens |
|---|---|---|---|---|---|
| anthropic.claude-3-5-sonnet-20241022-v2:0 | Claude 3.5 Sonnet | 200K | Coding, analysis, agents | $0.003 | $0.015 |
| anthropic.claude-3-5-haiku-20241022-v1:0 | Claude 3.5 Haiku | 200K | Fast tasks, classification | $0.0008 | $0.004 |
| anthropic.claude-3-haiku-20240307-v1:0 | Claude 3 Haiku | 200K | Cost-efficient workloads | $0.00025 | $0.00125 |
| amazon.nova-pro-v1:0 | Amazon Nova Pro | 300K | Reasoning, agents, RAG | $0.0008 | $0.0032 |
| amazon.nova-lite-v1:0 | Amazon Nova Lite | 300K | Multimodal, fast, cheap | $0.00006 | $0.00024 |
| amazon.nova-micro-v1:0 | Amazon Nova Micro | 128K | Text-only, ultra-fast | $0.000035 | $0.00014 |
| meta.llama3-1-70b-instruct-v1:0 | Llama 3.1 70B | 128K | Open weights, general | $0.00099 | $0.00099 |
| mistral.mistral-large-2402-v1:0 | Mistral Large | 32K | European compliance | $0.004 | $0.012 |
| amazon.titan-embed-text-v2:0 | Titan Embed v2 | 8K | Embeddings for RAG | $0.00002/1K | N/A |

## Python SDK Quick Reference

| | |
|---|---|
| **Install** | `pip install boto3` |
| **Client** | `boto3.client('bedrock-runtime', region_name='us-east-1')` |
| **Invoke model** | `client.invoke_model(modelId=..., body=json.dumps({...}))` |
| **Invoke agent** | `boto3.client('bedrock-agent-runtime').invoke_agent(agentId=..., agentAliasId=.` |
| **Retrieve+Generate** | `boto3.client('bedrock-agent-runtime').retrieve_and_generate(input={'text': ...` |
| **Create guardrail** | `boto3.client('bedrock').create_guardrail(name=..., contentPolicyConfig={...})` |
| **List models** | `boto3.client('bedrock').list_foundation_models()` |
| **Batch inference** | `boto3.client('bedrock').create_model_invocation_job(modelId=..., inputDataConf` |

## CLI Commands & Auth

**Auth (recommended)**

**Auth (dev/test)**

```
# IAM Role (no keys in code)
aws configure sso
# Or attach execution role to Lambda/EC2
```

```
# Access keys (rotate every 90 days)
export AWS_ACCESS_KEY_ID=AKIA...
export AWS_SECRET_ACCESS_KEY=...
export AWS_DEFAULT_REGION=us-east-1
```

**Key Services at a Glance**

| Service | Purpose | CLI Prefix |
|---|---|---|
| Bedrock Runtime | Invoke foundation models | bedrock-runtime |
| Bedrock Agent RT | Invoke agents, RAG retrieve+generate | bedrock-agent-runtime |
| Bedrock (mgmt) | Create guardrails, knowledge bases, model jobs | bedrock |
| Step Functions | Orchestrate multi-agent / async pipelines | stepfunctions |
| SQS | Async job queue, decouples API from workers | sqs |
| DynamoDB | Job state store, session memory | dynamodb |
| OpenSearch Serverless | Default vector store for Knowledge Bases | opensearchserverless |
| CloudWatch | Metrics, logs, alarms for all AI workloads | cloudwatch / logs |

# Azure OpenAI Service

Complete reference — models · endpoints · SDK · pricing · CLI

## Models & Deployments

| Model | Version | Context | Best For | In $/1K tokens | Out $/1K tokens |
|---|---|---|---|---|---|
| gpt-4o | 2024-11-20 | 128K | Top quality, multi-modal | $0.0025 | $0.010 |
| gpt-4o-mini | 2024-07-18 | 128K | Fast & cheap, high volume | $0.00015 | $0.0006 |
| o1 | 2024-12-17 | 200K | Complex reasoning, STEM | $0.015 | $0.060 |
| o1-mini | 2024-09-12 | 128K | Reasoning, lower cost | $0.003 | $0.012 |
| o3-mini | 2025-01-31 | 200K | Efficient reasoning tasks | $0.0011 | $0.0044 |
| gpt-4-turbo | 2024-04-09 | 128K | GPT-4 class, vision support | $0.010 | $0.030 |
| text-embedding-3-large | 1 | 8K | High-accuracy embeddings | $0.00013 | N/A |
| text-embedding-3-small | 1 | 8K | Cost-efficient embeddings | $0.00002 | N/A |
| whisper-1 | — | — | Speech-to-text transcription | $0.006/min | N/A |
| dall-e-3 | 3.0 | — | Image generation | $0.04/img | N/A |

## Python SDK Quick Reference

| | |
|---|---|
| **Install** | `pip install openai azure-ai-projects azure-identity` |
| **Client** | `AzureOpenAI(api_key=..., api_version='2024-02-01', azure_endpoint=...)` |
| **Chat** | `client.chat.completions.create(model='gpt-4o', messages=[...])` |
| **Embeddings** | `client.embeddings.create(model='text-embedding-3-large', input='...')` |
| **RAG on data** | `client.chat.completions.create(..., extra_body={'data_sources': [{'type': 'azure ...}]})` |
| **Create agent** | `AIProjectClient.from_connection_string(...).agents.create_agent(model=..., instr` |
| **Content Safety** | `ContentSafetyClient(endpoint, AzureKeyCredential(key)).analyze_text(...)` |
| **Streaming** | `client.chat.completions.create(..., stream=True)` → `for chunk in stream: print(chunk.choices[0].delta.content)` |

## Auth & Environment Variables

**API Key (dev)**

**Managed Identity (prod)**

```
export AZURE_OPENAI_KEY=sk-...
export AZURE_OPENAI_ENDPOINT=https://your-resou
export AZURE_OPENAI_API_VERSION=2024-02-01
```

```
from azure.identity import DefaultAzureCredential
from openai import AzureOpenAI
client = AzureOpenAI(
azure_ad_token_provider=get_bearer_token_provider(
DefaultAzureCredential(), 'https://cognitiveservices.azure.com/
azure_endpoint=os.environ['AZURE_OPENAI_ENDPOINT'],
api_version='2024-02-01')
```

**Key Azure Services**

| Service | Purpose | SDK / Package |
|---|---|---|
| Azure OpenAI Service | GPT-4o, o1, DALL-E, Whisper, Embeddings | openai (azure provider) |
| Azure AI Search | Vector + full-text + semantic hybrid RAG | azure-search-documents |
| Azure AI Content Safety | Harm detection, Prompt Shield, groundedness | azure-ai-contentsafety |
| Azure AI Foundry | Agent Service, evaluations, prompt flow | azure-ai-projects |
| Azure Service Bus | Async queue for AI job pipelines | azure-servicebus |
| Azure Cosmos DB | Job state store, vector search (DiskANN) | azure-cosmos |
| Azure Monitor | Metrics, logs, distributed tracing | azure-monitor-opentelemetry |

# GCP Vertex AI

Complete reference — models · endpoints · SDK · pricing · CLI

## Gemini Models & Pricing

| Model ID | Name | Context | Best For | In $/1K tokens | Out $/1K tokens |
|---|---|---|---|---|---|
| gemini-1.5-pro-002 | Gemini 1.5 Pro | 2M | Long context, complex reasoning | $0.00125 | $0.005 |
| gemini-1.5-flash-002 | Gemini 1.5 Flash | 1M | Fast, cost-efficient, multi-modal | $0.000075 | $0.0003 |
| gemini-1.5-flash-8b | Flash 8B | 1M | Ultra-cheap, high volume tasks | $0.0000375 | $0.00015 |
| gemini-2.0-flash | Gemini 2.0 Flash | 1M | Next-gen speed, agentic tasks | $0.0001 | $0.0004 |
| gemini-2.0-flash-thinking | 2.0 Flash Thinking | 32K | Reasoning, math, STEM | $0.0035 | $0.0105 |
| text-embedding-004 | Text Embedding 004 | 2K | English embeddings, Vertex-native | $0.000025 | N/A |
| textembedding-gecko@003 | Gecko 003 | 3K | Balanced embedding quality | $0.000025 | N/A |
| text-multilingual-embedding-002 | Multilingual 002 | 2K | 100+ language embeddings | $0.000025 | N/A |
| imagegeneration@006 | Imagen 3 | — | Text-to-image generation | $0.04/img | N/A |

## Python SDK Quick Reference

| | |
|---|---|
| **Install** | `pip install google-cloud-aiplatform google-cloud-discoveryengine google-cloud-dl` |
| **Init** | `vertexai.init(project='your-project', location='us-central1')` |
| **Generate (chat)** | `GenerativeModel('gemini-1.5-flash').generate_content('Hello')` |
| **Safety settings** | `GenerativeModel(...).generate_content(..., safety_settings=[SafetySetting(...)])` |
| **Embed** | `TextEmbeddingModel.from_pretrained('text-embedding-004').get_embeddings(['text']` |
| **RAG query** | `discoveryengine.ConversationalSearchServiceClient().converse_conversation(reques` |
| **Agent Builder** | `AgentBuilder.create_agent(display_name=..., goal=..., instructions=...)` |
| **Reasoning Eng** | `reasoning_engines.ReasoningEngine.create(agent, requirements=[...])` |
| **Batch predict** | `aiplatform.BatchPredictionJob.create(model_name=..., gcs_source=..., gcs_dest=..` |

## Auth & gcloud CLI

**Application Default Credentials (ADC)**

```
# Login once locally
gcloud auth application-default login
# In code — ADC is auto-detected
import vertexai
vertexai.init(project='my-project', location='u
```

**Service Account (CI/CD / production)**

```
# Create SA + key
gcloud iam service-accounts create ai-sa
gcloud projects add-iam-policy-binding PROJECT \
--member='serviceAccount:ai-sa@PROJECT.iam.gserviceaccount.com'
--role='roles/aiplatform.user'
export GOOGLE_APPLICATION_CREDENTIALS=/path/to/key.json
```

**Key GCP Services**

| Service | Purpose | SDK / Package |
|---|---|---|
| Vertex AI (Generative) | Gemini, embeddings, fine-tuning, batch predict | google-cloud-aiplatform |
| Vertex AI Search | Managed RAG, conversational search, data stores | google-cloud-discoveryengine |
| Vertex AI Agent Builder | No-code agent creation, managed serving | google-cloud-aiplatform |
| Reasoning Engine | Deploy LangGraph / custom agents as managed API | google-cloud-aiplatform |
| Cloud DLP | PII detection & redaction for AI pipelines | google-cloud-dlp |
| Cloud Tasks | Async job queue for AI workers | google-cloud-tasks |
| Cloud Run | Serverless containers for AI API wrappers | google-cloud-run (gcloud) |
| Cloud Logging/Monitoring | Logs, metrics, traces for all AI workloads | google-cloud-logging |

## Master Platform Comparison

| | AWS Bedrock | Azure OpenAI | GCP Vertex AI |
|---|---|---|---|
| Top LLM | Claude 3.5 Sonnet | GPT-4o | Gemini 1.5 Pro |
| Cheapest LLM | Nova Micro ($0.000035/1K in) | GPT-4o mini ($0.00015) | Gemini Flash 8B ($0.0000375) |
| Max context | 300K (Nova Pro) | 200K (o1) | 2M (Gemini 1.5 Pro) |
| Best for reasoning | Claude 3.5 Sonnet | o1 / o3-mini | Gemini 2.0 Flash Thinking |
| Best for coding | Claude 3.5 Sonnet | GPT-4o | Gemini 1.5 Pro |
| Best embeddings | Titan Embed v2 (1024-dim) | text-embedding-3-large (3072) | text-embedding-004 (768) |
| RAG service | Knowledge Bases | Azure AI Search | Vertex AI Search |
| Agent service | Bedrock Agents | AI Agent Service | Agent Builder |
| Guardrails | Bedrock Guardrails | Azure Content Safety | Model Armor + Safety filters |
| Orchestration | Step Functions | AutoGen / Semantic Kernel | Reasoning Engine / LangGraph |
| Async queue | SQS + Lambda | Service Bus + Functions | Cloud Tasks + Cloud Run |
| Vector store | OpenSearch / pgvector / Pinecone | Azure AI Search (built-in) | Vertex AI Vector Search |
| Prompt caching | Yes (Claude, ~90% discount) | Yes (automatic, ~50%) | Yes (context cache, ~75%) |
| Batch inference | 50% discount | 50% discount | ~40% discount |
| Multi-modal | Yes (Nova, Claude) | Yes (GPT-4o, DALL-E 3) | Yes (Gemini native, Imagen 3) |
| Compliance | SOC2, HIPAA, PCI, FedRAMP High | SOC2, HIPAA, PCI, FedRAMP High | SOC2, HIPAA, PCI, FedRAMP Mod |
| Data residency | Region-locked inference | Region + Private Link | Region + VPC Service Controls |
| Free tier | No free tier | Free credits on signup | $300 credit, new accounts |
| Primary SDK | boto3 (Python) | openai (azure provider) | google-cloud-aiplatform |
| IaC support | CloudFormation / SAM / CDK | ARM / Bicep / Terraform | Terraform / Deployment Manager |

# Platform Selection Decision Flowchart

**Q1: Is your primary cloud AWS?**

- **YES → Use Bedrock — zero new accounts, IAM just works, Claude + Nova available** → **AWS Bedrock**
- **NO → Continue to Q2** → **Q2**

▼

**Q2: Is your org on Microsoft / Azure stack?**

- **YES → Use Azure OpenAI — GPT-4o, Managed Identity, SharePoint/M365 integration** → **Azure OpenAI**
- **NO → Continue to Q3** → **Q3**

▼

**Q3: Do you need massive context (500K+ tokens) or BigQuery integration?**

- **YES → Use GCP Vertex AI — Gemini 1.5 Pro has 2M context, native BigQuery tool** → **GCP Vertex AI**
- **NO → Task-based routing** → **Q4**

▼

**Q4: What is the primary task type?**

▼

| Task Type | Best Platform | Best Model | Reason |
|---|---|---|---|
| Complex reasoning / STEM | **Azure** | o1 / o3-mini | Dedicated reasoning models with chain-of-thought |
| Coding / debugging | **AWS** | Claude 3.5 Sonnet | Highest scores on SWE-bench coding benchmarks |
| Long document analysis | **GCP** | Gemini 1.5 Pro | 2M token context — entire codebases / books |
| High-volume classification | **Any** | Haiku / Flash 8B / GPT-4o mini | Cheapest models, still accurate for structured tasks |
| Multi-modal (image+text) | **GCP** | Gemini 1.5 Flash | Native multi-modal, fast, cheap |
| Enterprise RAG | **AWS or Azure** | Claude or GPT-4o | Bedrock KB or Azure AI Search — both production-grade |
| Multi-agent pipelines | **AWS** | Nova Pro + Claude Haiku | Step Functions + Bedrock Agents — deepest integration |
| EU data residency | **Azure or GCP** | GPT-4o or Gemini | Both have EU-only regions + EU BAAs |

| | | | |
|---|---|---|---|
| Real-time Google Search | GCP | Gemini 2.0 Flash | Native Google Search grounding via Grounding API |

# Migration Guide & Error Code Reference

## OpenAI API → Bedrock (Claude)

| | |
|---|---|
| `from openai import OpenAI` | `import boto3, json` |
| `client = OpenAI(api_key=...)` | `client = boto3.client('bedrock-runtime')` |
| `model='gpt-4o'` | `modelId='anthropic.claude-3-5-sonnet-20241022-v2` |
| `messages=[{'role':'user',...}]` | `body={'anthropic_version':'bedrock-2023-05-31',` |
| `response.choices[0].message.content` | `json.loads(resp['body'].read())['content'][0]['t` |
| `stream=True` | `client.invoke_model_with_response_stream(...)` |

## OpenAI API → Azure OpenAI

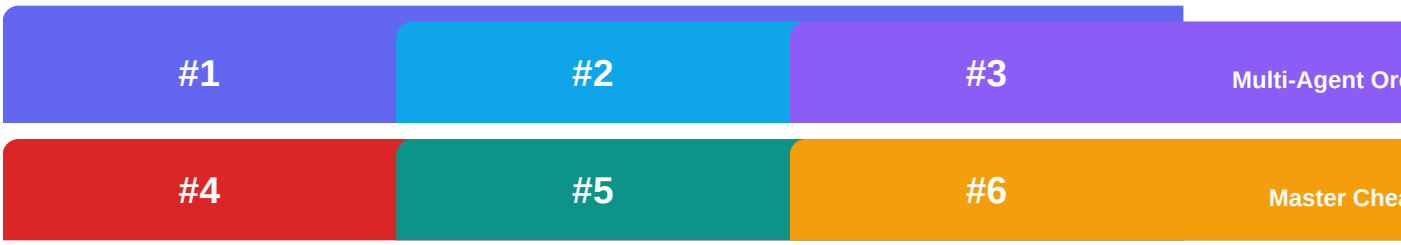
| | |
|---|---|
| `from openai import OpenAI` | `from openai import AzureOpenAI` |
| `OpenAI(api_key=...)` | `AzureOpenAI(api_key=..., api_version='2024-02-01` |
| `model='gpt-4o'` | `model='gpt-4o' # your deployment name, not model` |
| `messages=[...]` | `messages=[...] # identical schema` |
| `response.choices[0].message.content` | `response.choices[0].message.content # identical` |
| `# No RAG built-in` | `extra_body={'data_sources': [{'type': 'azure_sea` |

## Common Error Codes — All Platforms

| HTTP Code | Error | Platform | Fix |
|---|---|---|---|
| 400 | ValidationException | AWS | Check model ID, body schema, guardrail params. Common: wrong anthropic_version string. |
| 400 | Bad Request | Azure/GCP | Invalid model name, missing required field, or malformed messages array. |
| 401 | AuthenticationError | All | API key invalid/expired. Verify env var is set and key hasn't been rotated. |
| 403 | AccessDeniedException | AWS | IAM missing bedrock:InvokeModel or model access not enabled in console. |
| 404 | ResourceNotFound | All | Wrong deployment name (Azure), model ID (AWS), or project/location (GCP). |
| 429 | ThrottlingException | AWS | Rate limit hit. Exponential backoff. Request quota increase for production. |

| | | | |
|---|---|---|---|
| 429 | RateLimitError | Azure/GCP | RPM or TPM limit exceeded. Implement backoff. Consider PTU (Azure) or quota increase (GCP). |
| 500 | InternalServerError | All | Transient provider error. Retry with backoff. If persistent, check provider status page. |
| 503 | ServiceUnavailable | All | Provider overloaded or down. Trigger multi-cloud failover. Check status page. |
| — | guardrail_intervened | AWS | Bedrock Guardrail blocked request. Check stop_reason and amazon-bedrock-guardrailAction. |
| — | finish_reason: SAFETY | GCP | Vertex safety filter blocked. Check safety_ratings per category. Adjust thresholds. |
| — | content_filter | Azure | Azure content filter triggered. Check which category. Enable custom category if false positive. |

## Series Complete — Your Learning Path Forward




## What To Build Next

| | |
|---|---|
| **Beginner** | Build a CLI chatbot that queries all 3 platforms with the same prompt and compares outputs side-by-side. Use Guide #1 code as your starting point. |
| **Intermediate** | Add RAG to your chatbot: ingest 5-10 PDFs into Bedrock Knowledge Bases and give your bot access to your own documents. Guide #2 covers this end-to-end. |
| **Advanced** | Replace the single chatbot with a 3-agent pipeline: Researcher → Writer → Reviewer using the team.json config pattern from Guide #3. |
| **Production** | Add Bedrock Guardrails (Guide #4), implement async polling to bypass API timeouts (Guide #5), and set up CloudWatch cost anomaly alerts. |
| **Architect** | Build the MultiCloudLLMRouter from Guide #5 — connect all 3 platforms, test failover, measure P99 latency per provider, and publish your findings. |

**ted the Cloud AI Developer Guide Series.**

6 guides · AWS Bedrock · Azure OpenAI · GCP Verte