Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Building AI Agents with Strands Agents and Amazon Bedrock AgentCore

Class Duration

21 hours of live training delivered over 3 days to accommodate your scheduling needs.

Student Prerequisites

- Professional Python development experience (type hints, virtual environments, and basic `async` code)
- Familiarity with LLM concepts such as prompts, tool calling, and retrieval-augmented generation
- Working knowledge of AWS fundamentals: IAM roles and policies, the AWS CLI, and CloudWatch
- An AWS account with access to Amazon Bedrock and Amazon Bedrock AgentCore
- Experience with Amazon Bedrock is helpful (see *Generative AI on AWS with Amazon Bedrock* for the platform overview)

Target Audience

Python developers, AI engineers, and platform teams building agents on AWS, including teams moving off Amazon Bedrock Agents Classic. For an AI-assisted workflow on AWS, pair it with *Spec-Driven Agentic Coding with Kiro*. Teams on other stacks can take *Building AI Agents with Agent SDKs* or *Building AI Agents in .NET with Microsoft Agent Framework*.

Description

Strands Agents is AWS's open-source agent SDK, and Amazon Bedrock AgentCore is its platform for running agents built with any framework. In this course, students build agents with Strands and then deploy and operate them on AgentCore. The first half covers the agent loop, tools, MCP, structured output, human-in-the-loop control, and multi-agent patterns. The second half covers AgentCore Runtime, Memory, Gateway, Identity, and Policy, plus evaluation, infrastructure as code, and cost. Python is the main language, with notes where the TypeScript SDK differs.

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Learning Outcomes

- Explain how Strands Agents and AgentCore fit together.
- Choose between the AgentCore harness and code-defined agents.
- Build Strands agents with model providers, Python tools, vended tools, and MCP.
- Return validated structured output and manage session state.
- Add human approval with hooks, interrupts, and interventions.
- Compose agents as tools, swarms, graphs, and workflows.
- Connect agents across services with A2A.
- Deploy agents to AgentCore Runtime with the AgentCore CLI.
- Add memory, gateway tools, and credentials with AgentCore services.
- Trace and evaluate agents with AgentCore Observability, Strands Evals, and AgentCore Evaluations.
- Define agent infrastructure with the AWS CDK and AWS CloudFormation.
- Secure agents and control cost with Guardrails, Policy, and least-privilege IAM.

Training Materials

Comprehensive courseware is distributed online at the start of class. All students receive a downloadable MP4 recording of the training.

Software Requirements

Python 3.10+ with uv, Node.js 22+ and npm for the AgentCore CLI, the AWS CLI configured for an AWS account with Amazon Bedrock and AgentCore access and permission to bootstrap the AWS CDK, a code editor such as VS Code, Git, and optionally Docker, Podman, or Finch for local container testing.

Training Topics

Strands Agents and Amazon Bedrock AgentCore

- Strands Agents as an open-source SDK
- The model-driven approach
- AgentCore as a modular agent platform
- AgentCore services at a glance
- The AgentCore harness versus code-defined agents

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

- AgentCore with LangGraph, CrewAI, and other frameworks
- Models in and outside Amazon Bedrock
- Reference architecture for an agent on AWS

The Strands Agent Loop and Model Providers

- Creating and invoking an `Agent`
- How the agent loop runs tools
- How the loop decides it is done
- Amazon Bedrock as the default provider
- Anthropic, OpenAI, Gemini, Ollama, and LiteLLM
- Streaming agent events
- Synchronous and async invocation
- The Strands harness versus the SDK

Tools and MCP in Strands

- The `@tool` decorator
- Docstrings and type hints as tool contracts
- TypeScript tools with `tool()` and Zod
- Vended tools for files and shell commands
- HTTP request, web fetch, and Python REPL tools
- `MCPClient` with stdio, Streamable HTTP, or SSE
- Tool filtering and name prefixes
- Elicitation callbacks
- Concurrent versus sequential tool execution

Structured Output, Context, and Sessions

- Structured output with Pydantic and Zod
- Validation retries and output errors
- Agent state versus conversation history
- Sliding window and summarizing managers
- Offloading large tool results
- Session storage in files, S3, or AgentCore Memory
- Persisting graph and swarm sessions

Hooks, Interrupts, and Human-in-the-Loop

- Hook providers and lifecycle events
- Interrupts that pause for human input
- Approving sensitive tool calls

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

- Interventions for gating and steering

Multi-Agent Patterns

- Agents as tools
- Swarms of specialist agents
- Graphs with model-chosen paths
- Workflows with parallel tasks
- Shared invocation state
- Choosing a pattern

Agent-to-Agent (A2A) Communication

- Agent cards, JSON-RPC, and streaming
- Serving agents with `A2AServer`
- `A2AExpressServer` in TypeScript
- Consuming remote agents with `A2AAgent`
- Context IDs and conversation isolation
- Context IDs are not authentication
- Hosting A2A servers on AgentCore Runtime

Deploying to AgentCore Runtime

- A microVM per session
- Long-running sessions
- `agentcore create` and `agentcore dev`
- Local testing in the Agent Inspector
- `agentcore deploy` and `agentcore invoke`
- Direct code and container deployments
- HTTP, MCP, A2A, and AG-UI protocols
- WebSocket bidirectional streaming
- Versions and endpoints for rollout
- Exporting a harness to Strands code

AgentCore Memory

- Short-term memory within a session
- Long-term memory extraction
- Built-in and self-managed strategies
- Namespaces and actor IDs
- Strands sessions on AgentCore Memory
- Sharing memory across agents

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

AgentCore Gateway and Identity

- Lambda, OpenAPI, and Smithy targets
- API Gateway and existing MCP servers
- One MCP endpoint for many tools
- Semantic tool selection
- Built-in integrations such as Slack and Jira
- Inbound JWT authorization
- Outbound OAuth 2.0 and API keys
- On-behalf-of user access
- Workload identities
- Gateway versus your own MCP server

AgentCore Code Interpreter and Browser

- Sandboxed Python, JavaScript, and TypeScript
- Managed browser sessions
- Playwright and Browser Use
- Custom resources with network settings
- Code Interpreter and Browser as Strands tools

Observability and Evaluation

- Strands OpenTelemetry traces and metrics
- AgentCore Observability dashboards
- Strands Evals evaluators
- User and tool simulators
- Failure detection and red teaming
- AgentCore Evaluations with LLM judges
- Online, batch, and dataset evaluation
- AgentCore Optimization recommendations
- A/B testing prompts and tool descriptions
- Gating releases on evaluation results

Infrastructure as Code

- The AgentCore CLI project layout
- `agentcore.json` and `aws-targets.json`
- The generated AWS CDK app
- AWS CDK constructs for AgentCore
- `AWS::BedrockAgentCore` CloudFormation resources

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

- Terraform as an alternative
- CI/CD for runtime versions

Security, Guardrails, and Cost

- Amazon Bedrock Guardrails on Strands models
- Shadow-mode guardrail checks with hooks
- AgentCore Policy with Cedar or natural language
- Least-privilege IAM execution roles
- Session isolation and VPC connectivity
- Prompt injection and tool abuse
- Sandboxing local agents with Strands Box
- Virtual shells with Strands Shell
- Model choice and token cost
- Prompt caching and tool fan-out
- AgentCore consumption-based pricing

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved