CloudNinja.pro

ASSOCIATE · DVA-C03

AWS Certified Developer – Associate

DVA-C03 Cheat Sheet

CloudNinja · cloudninja.pro · Free — no signup required

Every in-scope service and decision rule for DVA-C03, organised by the four exam domains: development with AWS services, security (now including AI services), testing and deployment, and troubleshooting and optimization. Lambda, API Gateway, DynamoDB, messaging and CI/CD, plus the new C03 topics: AI-assisted development with Kiro, Amazon Bedrock and AgentCore in your code, presigned URLs, certificates, and container images and deployments. Assumes about a year of hands-on AWS development.

EXAM SNAPSHOT

- **65 questions** — 50 scored + 15 unscored
- **130 minutes** total time
- **720 / 1000** to pass
- **$150 USD** exam fee
- **Development with AWS Services** — 30%
- **Security** — 26%
- **Testing and Deployment** — 22%
- **Troubleshooting and Optimization** — 22%

FREE DVA-C03 practice exam →

DVA-C03 replaces DVA-C02 on 1 December 2026. The last day to take DVA-C02 is 30 November 2026. Same four domains, new weights (30 / 26 / 22 / 22; C02 was 32 / 26 / 24 / 18). New in C03: a whole security task on AI services (Task 2.3), AI-assisted coding, testing, CI/CD and troubleshooting, managed AI services in your code, presigned URLs, container images (ECR, Dockerfiles, tags, lifecycle policies) and container deployments (ECS, EKS, Fargate). Added to the in-scope list: Amazon Bedrock, Bedrock AgentCore, Amazon Quick, Kiro, Amazon Data Firehose, AWS SAM, CloudWatch Logs, AWS PrivateLink, AWS Certificate Manager, AWS Private CA and S3 Glacier. Dropped: AWS AppSync and Amazon Q Developer. DEPRECATED = closed to new customers, in maintenance mode or out of support; RETIRED = shut down. Know them, but do not pick them for a new design.

1 · Developer Toolkit & AI-Assisted Development

SDKs, CLI & credentials

Tool	What it does
AWS SDKs	Language libraries (Python/boto3, JavaScript, Java, Go, .NET, Ruby, PHP, C++, Rust). Sign requests with SigV4, retry throttling and 5xx errors with exponential backoff, paginate list calls, and offer waiters.
AWS CLI v2	Every service from the shell. `--query` (JMESPath) filters output; `--output json|yaml|table|text`; named profiles in `~/.aws/config`.
AWS CloudShell	Browser shell with the CLI pre-installed and the console user's credentials. 1 GB persistent home directory per Region.

The SDK/CLI default credential chain stops at the first hit: explicit credentials → environment variables (`AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, `AWS_SESSION_TOKEN`) → shared config/credentials files (profiles, IAM Identity Center SSO) → container credentials (ECS task role, EKS Pod Identity) → EC2 instance profile via IMDS.

> **Exam Tip:** Hard-coded access keys in code on EC2, Lambda or ECS are always wrong. The fix is an IAM role (instance profile, execution role, task role), never "rotate the keys" or "store the keys in Secrets Manager".

AI-assisted development tools

Tool	What it is and when to pick it
Kiro	AWS's agentic IDE and CLI. **Specs** turn a prompt into `requirements.md` (user stories + acceptance criteria), `design.md` (architecture, error handling, testing strategy) and `tasks.md` (trackable implementation tasks) before code is written — "spec-driven development". **Steering** files in `.kiro/steering/` (`product.md`, `tech.md`, `structure.md`) give the agent persistent project rules. **Hooks** in `.kiro/hooks/` run a prompt or shell command on events such as a file save or tool use (for example, generate or update unit tests whenever a source file is saved). Supports MCP servers.
Amazon Quick	AI-powered workspace for business users: chat agents over connected data (Quick Index), research reports (Quick Research), automations (Quick Flows, Quick Automate), apps, and BI dashboards (Amazon Quick Sight, formerly QuickSight). A developer integrates it (embedding, connectors, MCP/OpenAPI actions); it is not a coding assistant.
Amazon Q Developer DEPRECATED	No longer on the C03 in-scope list. AWS ends support for the Q Developer IDE plugins on 30 April 2027 and points users to Kiro (inline suggestions, chat, code generation).
CloudWatch investigations	Generative AI assistant in CloudWatch: start from an alarm, metric or Logs Insights query; it surfaces related metrics, logs, deployment and CloudTrail change events, X-Ray traces and root-cause hypotheses.
AWS DevOps Agent	AI operations teammate that investigates incidents, correlates telemetry to find root causes, and suggests reliability and performance improvements.

C03 skill	Typical AI-tool answer
1.1.9 Generate, review, optimise code	Kiro spec (requirements → design → tasks) and agent code review / refactoring suggestions; steering files keep the agent on house conventions.
3.2.5 Generate tests, automate testing	Kiro hook on file save that writes or updates tests; ask the agent for regression tests that cover the acceptance criteria in `requirements.md`.
3.3.9 AI in CI/CD	Agent-generated IaC and pipeline changes, AI summaries of test results before a manual approval, post-deployment validation scripts. Humans still approve production.

C03 skill	Typical AI-tool answer
4.1.5 Analyse errors	CloudWatch investigations from the firing alarm; AWS DevOps Agent for incident investigation; paste the stack trace into Kiro for a fix suggestion.
4.3.6 Find optimisation opportunities	AI analysis of CloudWatch metrics and traces for bottlenecks; Kiro refactoring for code efficiency; then verify with the metrics.

Exam Tip: "Agree requirements and design before the AI writes code" = Kiro **spec**. "Agent must always follow our coding standards" = Kiro **steering** file. "Run tests or linting automatically when a file changes" = Kiro **hook**. AI output is reviewed and tested like any other change — never auto-merged to production without a gate.

2 · Domain 1 — Development with AWS Services (30%)

Application patterns (Task 1.1)

- **Loose coupling** — put a queue (SQS), topic (SNS) or event bus (EventBridge) between components so each scales and fails independently.
- **Sync vs async** — synchronous when the caller needs the answer now (API Gateway → Lambda, 29 s default integration timeout on REST APIs); asynchronous for long or spiky work (accept, return 202, process from SQS, notify on completion).
- **Resilience** — retries with *exponential backoff and jitter* (the SDKs do this for throttling), timeouts on every call, idempotency keys so retries are safe, graceful degradation (serve cached or partial data), DLQs for poison messages.
- **APIs** — request validation (models + validators) and request/response mapping templates in API Gateway REST APIs; version through stages or base paths.
- **Serialisation** — JSON is the default for Lambda events, SDK payloads and DynamoDB documents; base64 for binary in JSON; Parquet/ORC for analytics in S3.

AWS Lambda — limits that get tested

Setting	Value
Timeout	Up to 900 s (15 min). Default 3 s.
Memory	128–10,240 MB in 1 MB steps. CPU scales with memory; 1,769 MB ≈ 1 vCPU.
`/tmp`	512–10,240 MB ephemeral storage. Survives between warm invocations of the same environment only.
Package	.zip: 50 MB zipped upload, 250 MB unzipped including layers. Container image: 10 GB. Up to 5 layers.
Environment variables	4 KB total per function.
Payload	Synchronous 6 MB request and 6 MB response; streamed response up to 200 MB; asynchronous 1 MB.
Concurrency	1,000 per Region by default (raisable). Each function scales by up to 1,000 execution environments every 10 s.

Invocation models, errors and destinations

Model	Sources	Error behaviour
Synchronous	API Gateway, ALB, function URLs, SDK `Invoke`	Error returns to the caller; the caller retries.
Asynchronous	S3, SNS, EventBridge, `InvocationType=Event`	Lambda queues the event and retries twice; then the on-failure destination or DLQ. Configure max retry attempts and max event age.

Model	Sources	Error behaviour
Event source mapping (poll)	SQS, Kinesis, DynamoDB Streams, MSK, Amazon MQ	SQS: failed messages return after the visibility timeout, then the queue's DLQ (redrive policy). Streams: retry until success or record expiry — use bisect batch on error, max retry attempts, max record age, an on-failure destination, and `ReportBatchItemFailures` (partial batch response).

- **Destinations** (async and stream sources) — on-success / on-failure targets: SQS, SNS, Lambda, EventBridge. Carry the invocation record with request and response. Prefer them over a DLQ, which only captures failures.
- **VPC access** (1.2.1) — attach the function to private subnets + security groups (needs `AWSLambdaVPCAccessExecutionRole` permissions for ENIs). It then has no internet: add a NAT gateway for internet, or VPC endpoints (gateway for S3/DynamoDB, interface/PrivateLink for others).
- **Layers** share libraries; **extensions** run agents (telemetry, secret caching) beside the function; **environment variables** hold config (encrypted at rest with KMS).
- **Durable functions** — checkpointed multi-step Lambda workflows that can wait for callbacks without paying for idle compute (up to a year).

Exam Tip: Work longer than 15 min → Step Functions, durable functions, ECS/Fargate. "Lambda in a VPC can't reach S3" → S3 gateway endpoint (or NAT). "Know which async events failed, with the payload" → on-failure destination. "One bad record blocks the Kinesis shard" → bisect on error + partial batch response + on-failure destination.

Amazon API Gateway

Type	Use when
REST API	Need caching, usage plans + API keys, request validation, mapping templates, WAF, private endpoints, canary on a stage, mock integrations. Payload 10 MB. Integration timeout 50 ms–29 s (raisable for Regional and private APIs).
HTTP API	Cheaper, lower latency proxy to Lambda/HTTP; native JWT authorizers and CORS; auto-deploy stages. No caching or usage plans.
WebSocket API	Persistent two-way connections (chat, live dashboards); routes on a message key; `@connections` API to push to clients.

- **Integrations** — Lambda proxy (default; Lambda returns status, headers, body), Lambda custom (mapping templates in VTL), HTTP/HTTP proxy, AWS service (SQS, Step Functions, DynamoDB without Lambda), mock.
- **Errors** — 502 = malformed Lambda proxy response; 504 = integration timeout; 429 = throttled; 403 = auth or WAF denied.
- **Throttling** — account/Region limit, stage and method limits, per-client limits via usage plans. Returns 429.

Messaging & orchestration

Service	Key facts
Amazon SQS	Standard: at-least-once, best-effort order, near-unlimited throughput. FIFO: ordered per message group, exactly-once processing with deduplication ID, 300 TPS per action (3,000 msg/s with batches of 10; high-throughput mode for more). Message up to 1 MiB (Extended Client Library + S3 for up to 2 GB). Retention 1 min–14 days (default 4 days). Visibility timeout default 30 s, max 12 h. Long polling up to 20 s. Delay up to 15 min. DLQ via redrive policy + `maxReceiveCount`.
Amazon SNS	Pub/sub push to SQS, Lambda, Firehose, HTTP/S, email, SMS, mobile. Fan-out = one topic, many SQS queues. Subscription filter policies. FIFO topics for ordered fan-out. Messages 256 KiB by default, up to 1 MiB with the topic's `MaximumMessageSize` (SQS, Lambda, Firehose subscribers only).
Amazon EventBridge	Event bus with JSON-pattern rules to 20+ target types; AWS service and SaaS partner events; schema registry; archive + replay; Pipes (source → filter → enrich → target); Scheduler (cron, rate, one-time).
AWS Step Functions	Standard workflows: up to 1 year, exactly-once, auditable history. Express: up to 5 min, at-least-once, high volume. `Retry` + `Catch` per state; Map for parallel items; `.waitForTaskToken` callback for human approval or external work; direct SDK integrations.

Exam Tip: Decouple → SQS. One event, many consumers → SNS fan-out or EventBridge rules. Route by event content or react to AWS service events → EventBridge. Ordered per customer → FIFO with `MessageGroupId` = customer ID. Multi-step with branching, retries and waits → Step Functions.

Streaming data (Skill 1.1.6)

Service	Developer view
Kinesis Data Streams	Ordered records per shard (by partition key), replayable. Retention 24 h default, up to 365 days. On-demand or provisioned capacity. Consumers: Lambda (event source mapping, batch size, batch window, parallelization factor), KCL apps, Firehose. Enhanced fan-out gives each consumer dedicated read throughput. `ProvisionedThroughputExceeded` → better partition key spread, more shards, backoff.
Amazon Data Firehose	Fully managed delivery (no consumers to write) to S3, Redshift, OpenSearch, Splunk, HTTP endpoints, Iceberg tables. Buffers by size/interval (interval can be 0 s for near-immediate delivery). Optional Lambda transform and format conversion to Parquet/ORC. Renamed from Kinesis Data Firehose.
DynamoDB Streams	Item-level change log kept 24 h; trigger Lambda for change data capture.
Lambda response streaming	Send partial responses as they are ready (function URLs, `InvokeWithResponseStream`, API Gateway proxy). Better time to first byte; up to 200 MB. Natively on Node.js runtimes.

Exam Tip: "Replay", "multiple consumers reading the same records", "ordering per key" → Kinesis Data Streams. "Just land the stream in S3 / OpenSearch with no code" → Data Firehose. "Stream tokens of an LLM answer to the browser" → Bedrock `ConverseStream` + Lambda response streaming or WebSocket API.

Data stores & caching (Task 1.3)

Store	Pick it for
DynamoDB	Key-value / document at any scale, single-digit ms, serverless. Known access patterns.
RDS / Aurora	Relational data, joins, transactions. Aurora: up to 15 replicas; Aurora Serverless v2 scales in ACUs and can auto-pause at 0 ACUs. **RDS Proxy** pools connections for Lambda.
ElastiCache	In-memory cache or session store (Valkey, Redis OSS, Memcached). Sorted sets for leaderboards.
S3	Objects up to 5 TB; static assets, uploads, data lake. Lifecycle rules transition (Standard-IA, Glacier) and expire objects.
OpenSearch Service · Athena	Full-text search and log analytics · serverless SQL over S3 (pay per data scanned; partition and use Parquet).

- **Caching strategies** — lazy loading / cache-aside (cache on miss; can be stale), write-through (update cache on write; never stale, caches unread data), TTL on both to bound staleness.
- **Data lifecycle** (1.3.1) — S3 Lifecycle, DynamoDB TTL (free deletes, appear in the stream), CloudWatch Logs retention, SQS retention, Kinesis retention.

Amazon DynamoDB (Skill 1.3.4)

Concept	Details
Keys	Partition key (high cardinality to avoid hot partitions) + optional sort key (range queries, `begins_with`).
Capacity	On-demand (pay per request) or provisioned RCU/WCU + auto scaling. 1 RCU = 1 strongly consistent 4 KB read/s (or 2 eventually consistent). 1 WCU = one 1 KB write/s. Transactions cost 2×.

Concept	Details
Consistency	Eventually consistent by default; `ConsistentRead=true` on table/LSI reads. GSIs are eventually consistent only.
Query vs Scan	Query reads one partition key (efficient); Scan reads the whole table (filters apply after reading — you still pay). Paginate with `LastEvaluatedKey` / `ExclusiveStartKey`; projection expressions trim attributes.
Indexes	GSI: any keys, add any time, own capacity. LSI: same partition key, different sort key, created with the table, strong reads allowed.
Writes	Condition expressions (`attribute_not_exists` for idempotency, version attribute for optimistic locking). `BatchWriteItem` 25 items, `BatchGetItem` 100 items — retry `UnprocessedItems`. `TransactWriteItems` for all-or-nothing.
DAX · Global tables	Microsecond read cache (in VPC, API-compatible) · multi-Region, multi-active replicas.

Exam Tip: "ProvisionedThroughputExceededException on one key" → better partition key (add a suffix / write sharding). "Avoid double-processing a retried request" → conditional put on an idempotency key. "Microsecond reads" → DAX. "Lambda exhausts RDS connections" → RDS Proxy.

Managed AI services in code (Skill 1.1.10)

Service / API	What a developer does with it
Amazon Bedrock	Serverless access to foundation models through one API. `Converse` / `ConverseStream` give a model-agnostic chat interface with tool use; `InvokeModel` / `InvokeModelWithResponseStream` take the model's native body. Knowledge Bases for managed RAG; Guardrails for safety; batch inference and prompt caching to cut cost; cross-Region inference profiles for throughput.
Amazon Bedrock AgentCore	Run and operate agents built with any framework or model: **Runtime** (serverless, session-isolated hosting), **Gateway** (turn APIs and Lambda functions into MCP tools), **Identity** (inbound auth + outbound credentials), **Memory** (short- and long-term), **Code Interpreter**, **Browser**, **Observability** (OpenTelemetry traces in CloudWatch).
Amazon Quick	Business-user AI workspace and BI (Quick Sight). Embed dashboards or chat in apps; connect data through connectors, MCP or OpenAPI actions.

Exam Tip: "Swap models without rewriting the request code" → Bedrock **Converse API**. "Answer from our documents" → Knowledge Bases. "Host an agent with per-user session isolation" → AgentCore Runtime. "Let the agent call our REST API safely" → AgentCore Gateway. Selecting models, prompt engineering and designing RAG are out of scope for DVA — integrating them is in.

3 · Domain 2 — Security (26%)

Authentication & authorization (Task 2.1)

- **IAM policies** — identity-based (user, group, role) and resource-based (S3 bucket, SQS queue, KMS key, Lambda function, API Gateway). Explicit Deny wins; no Allow = implicit Deny. Permission boundaries and SCPs cap the maximum.
- **Roles everywhere** — Lambda execution role, ECS task role (app) vs task execution role (pull image, send logs, read secrets at start), EC2 instance profile, EKS Pod Identity / IRSA.
- **Cross-account** — role in account B trusts account A; code calls `sts:AssumeRole` and uses the temporary credentials (optionally with an external ID).
- **MFA in policy** — condition `aws:MultiFactorAuthPresent`; CLI users get MFA-backed session credentials with `GetSessionToken`.

STS / federation	Use case
`AssumeRole`	Cross-account or elevated access for an AWS identity.
`AssumeRoleWithWebIdentity`	OIDC tokens (Cognito, Google, GitHub Actions OIDC) → AWS credentials.
`AssumeRoleWithSAML`	Corporate SAML 2.0 IdP federation.
Cognito user pools	Sign-up / sign-in, MFA, social and SAML/OIDC federation; issue ID, access and refresh JWTs. "Who is the user?"
Cognito identity pools	Exchange a user pool or external token for temporary AWS credentials (with guest access option). "What can the user do in AWS?"

API Gateway authorizer	Use case
IAM (SigV4)	Callers with AWS credentials (services, identity pool users).
Cognito user pool	REST API validates user pool JWTs; scopes for OAuth.
JWT authorizer	HTTP API, any OIDC/OAuth 2.0 issuer, no code.
Lambda authorizer	Custom logic (token or request parameters); returns an IAM policy; results cached by TTL.
API keys + usage plans	Identify and throttle clients — not authentication.

Presigned URLs (Skill 2.1.4)

Option	Facts
S3 presigned URL	Time-limited GET or PUT on one object, signed with the creator's credentials — the URL can do only what the signer may do. Console: 1 min–12 h; SDK/CLI: up to 7 days. Signed with temporary credentials (role, STS) → expires when they expire, whichever is first. Used for direct browser uploads/downloads without proxying through Lambda.
CloudFront signed URL	One file, through the CDN; canned or custom policy (expiry, start time, IP range). Sign with a key in a **trusted key group** (recommended; no root user needed).
CloudFront signed cookies	Many files (a whole course, an HLS video's segments) without changing URLs.

> **Exam Tip:** "Users upload large files straight to a private bucket" → S3 presigned PUT (or presigned multipart). "Paid content through the CDN, many files per user" → CloudFront signed cookies. "Presigned URL stops working after an hour" → it was signed with role/session credentials that expired.

Encryption & keys (Task 2.2)

- **At rest vs in transit** — KMS-backed encryption on S3, DynamoDB, EBS, RDS, SQS, SNS, Secrets Manager; TLS for in transit (enforce with `aws:SecureTransport` in a bucket policy).
- **S3 server-side** — SSE-S3 (default), SSE-KMS (key policy + CloudTrail audit; use S3 Bucket Keys to cut KMS calls), DSSE-KMS (two layers), SSE-C (you send the key each request). **Client-side** — encrypt before upload (AWS Encryption SDK) when AWS must never see plaintext.
- **KMS** — `Encrypt` / `Decrypt` directly only up to 4 KB; larger data uses **envelope encryption**: `GenerateDataKey` → encrypt locally with the plaintext data key → store the encrypted data key with the data. Key policies are the primary control; grants give temporary, scoped use. Automatic rotation of customer managed keys: default 365 days, configurable 90–2,560 days. `ThrottlingException` → data key caching, S3 Bucket Keys, or a quota increase.
- **Sensitive data** (2.2.6) — CloudWatch Logs data protection policies audit and mask PII/PHI/financial data at ingestion (only `logs:Unmask` can see it); Bedrock Guardrails redact PII in prompts and responses; least-privilege IAM; tokenise or mask in application code.

Certificates & SSH keys (Skills 2.2.2, 2.2.5)

Service	Facts
AWS Certificate Manager (ACM)	Free public TLS certificates for integrated services (ALB, CloudFront, API Gateway, App Runner…); DNS validation lets ACM renew automatically. Public certificates are valid 198 days and renew 45 days before expiry. Certificates for **CloudFront** (and edge-optimised API Gateway) must be in **us-east-1**; others in the resource's Region. **Exportable public certificates** (paid) give you the private key for EC2, containers or on-premises. Imported certificates are not renewed by ACM. Email validation is being phased out in 2027 — use DNS.
AWS Private CA	Your own private certificate authority for internal TLS and mutual TLS (service-to-service, IoT, private APIs). ACM can issue private certificates from it for integrated services. **Short-lived certificate mode**: certificates valid at most 7 days, cheaper CA, no revocation needed (ACM can't issue from a short-lived CA).
Self-signed for dev	`openssl` certificate for local HTTPS testing — never in production.
EC2 key pairs	EC2 creates ED25519 or 2048-bit RSA key pairs (or import a public key you generated with `ssh-keygen`); EC2 keeps the public key, you download the private key once. Prefer Systems Manager Session Manager or EC2 Instance Connect over long-lived SSH keys.

Secrets & configuration (Skill 2.2.7)

Feature	Secrets Manager	SSM Parameter Store
Rotation	Built in (RDS, Aurora, Redshift, DocumentDB) + custom Lambda rotation	None built in
Size	64 KB	4 KB Standard, 8 KB Advanced
Cost	Per secret per month + API calls	Standard tier free; Advanced tier paid
Use for	Database credentials, third-party API keys that rotate, cross-account secrets	Config values, feature settings, hierarchy per environment (`/prod/app/db-url`), `SecureString` via KMS

Lambda reads secrets at init and caches them (the AWS Parameters and Secrets Lambda Extension caches both). ECS injects them as environment variables through the task definition `secrets` field.

Security for AI services (Task 2.3 — new in C03)

Skill	How to do it on AWS
2.3.1 Request and manage access	In commercial Regions most Bedrock serverless models are enabled on first invocation — no access request; Bedrock auto-subscribes models sold through AWS Marketplace, which needs `aws-marketplace:Subscribe` the first time. Anthropic models need a one-time first-time-use form (an organisation's management account can submit it for all). Control use with IAM on `bedrock:InvokeModel` scoped to model ARNs (deny in an SCP to block a model) and `aws-marketplace:ProductId` conditions. Bedrock API keys exist for quick tests — prefer IAM roles.
2.3.2 Data privacy in transit	Interface VPC endpoints (AWS PrivateLink) for `bedrock-runtime` / AgentCore so traffic never crosses the internet; endpoint policies restrict actions. Model providers have no access to Bedrock prompts, completions or logs; Bedrock data retention settings include zero data retention mode. TLS 1.2+, KMS keys for stored artefacts.
2.3.3 Filter inputs and outputs	**Bedrock Guardrails**: content filters (hate, insults, sexual, violence, misconduct), **prompt attack** filter (jailbreaks, prompt injection), denied topics, word filters, sensitive information filters (block or mask PII, custom regex), contextual grounding checks. Attach to `Converse` / `InvokeModel`, or call `ApplyGuardrail` for any model or text.

Skill	How to do it on AWS
2.3.4 Secure agent actions	Least-privilege tool permissions (AgentCore Gateway + Identity, scoped IAM roles); session isolation (AgentCore Runtime microVM per session); human-in-the-loop for sensitive actions (Step Functions `.waitForTaskToken` approval, or the agent asks the user to confirm before calling the tool); validate tool inputs server-side.
2.3.5 Protect logs	Bedrock model invocation logging (to CloudWatch Logs and/or S3) captures prompts and responses — mask PII with a CloudWatch Logs data protection policy, encrypt log groups and buckets with KMS, restrict `logs:Unmask`, set retention.

Exam Tip: "Keep Bedrock traffic off the internet" → interface VPC endpoint (PrivateLink). "Stop users extracting the system prompt or jailbreaking" → Guardrails prompt attack filter. "Redact card numbers in answers" → Guardrails sensitive information filter. "Prompts in logs contain PII" → CloudWatch Logs data protection policy. "Agent may refund only after a person approves" → human-in-the-loop approval step.

4 · Domain 3 — Testing and Deployment (22%)

Prepare artifacts (Task 3.1)

Topic	Facts
Lambda packaging (3.3.1)	.zip (fast, 50/250 MB limits) · layers for shared dependencies · container image up to 10 GB for big ML libraries or custom runtimes (must implement the Lambda Runtime API; base images do).
Container images (3.1.2)	Dockerfile best practice: small base image, multi-stage builds, pin versions, non-root user. Tag with immutable release tags or git SHAs — not only `latest`.
Amazon ECR	Private registry. **Tag immutability** stops tags being overwritten (with exclusion filters, e.g. allow `latest` to move). **Lifecycle policies** expire images by count or age, filtered by tag prefix. Image scanning (basic or Amazon Inspector enhanced), cross-Region/cross-account replication, pull-through cache.
AWS CodeArtifact	Private npm, PyPI, Maven, NuGet repositories with upstream proxies to public registries.
Environment config (3.1.4)	AppConfig (feature flags, validated rollouts with alarm-based rollback), Parameter Store hierarchies, Secrets Manager, Lambda environment variables per alias/version, API Gateway **stage variables**.
API Gateway stages (3.1.5)	Each stage has its own URL (`…/dev`, `…/prod`), settings, logs and stage variables; custom domain + base path mappings for clean URLs.

Testing (Task 3.2)

- **AWS SAM** — `sam local invoke` (with `sam local generate-event` sample events), `sam local start-api`, `sam sync` for fast cloud iteration, `sam remote invoke` against the deployed function. SAM templates transform to CloudFormation.
- **AWS CDK** — assertions library (fine-grained and snapshot tests on the synthesised template); `cdk diff`. **CloudFormation** — change sets, linting with cfn-lint, validate before deploy.
- **Mock APIs** (3.2.2) — API Gateway **mock integration** returns a fixed response from a mapping template so front-end teams can build before the backend exists.
- **Test events** (3.2.3) — JSON payloads in the Lambda console (shareable test events), API Gateway test invoke, sample S3/SQS/API events.
- **Environments** (3.2.4) — Lambda **aliases** (`dev`, `test`, `prod` pointing at versions), container image tags per environment, **Amplify branches** (each Git branch = its own environment and URL; pull request previews), API Gateway stages, separate stacks or accounts. AWS Copilot environments DEPRECATED — Copilot CLI reached end of support on 12 June 2026.
- **AI-generated tests** (3.2.5) — Kiro hooks and agent-written unit/regression tests, run in CodeBuild; review coverage reports.

CI/CD services (Task 3.3)

Service	Facts
CodePipeline	Orchestrates source → build → test → approve → deploy stages across environments. Triggers from GitHub, GitLab, Bitbucket (via CodeConnections), CodeCommit, S3, ECR; V2 pipelines filter triggers by branch, tag or file path. Manual approval actions with SNS notification.
CodeBuild	`buildspec.yml` phases `install` → `pre_build` → `build` → `post_build`; `artifacts`, `reports` (test reports, coverage), `cache`; env vars from Parameter Store / Secrets Manager. Build logs to CloudWatch Logs or S3.
CodeDeploy	`appspec.yml`. EC2/on-premises: in-place or blue/green; hooks `ApplicationStop` → `BeforeInstall` → `AfterInstall` → `ApplicationStart` → `ValidateService`. Lambda and ECS: traffic shifting (Canary, Linear, AllAtOnce) with Lambda hook functions and alarm-based automatic rollback. Lambda hooks: `BeforeAllowTraffic` / `AfterAllowTraffic`. ECS hooks: `BeforeInstall` → `AfterInstall` → `AfterAllowTestTraffic` → `BeforeAllowTraffic` → `AfterAllowTraffic`.
CodeCommit	Managed private Git — open to new customers again (generally available since November 2025).
Amplify Hosting	Git-connected CI/CD for web front ends and full-stack apps; per-branch environments, PR previews, custom domains with managed certificates.

- **Source control** (3.3.8) — feature branches + pull requests, protected `main`, release **tags** (e.g. `v1.4.2`) to mark what shipped and to trigger release pipelines, semantic versioning, commit SHA as the image tag for traceability.

Deployment strategies & rollback (Skills 3.3.6, 3.3.7)

Target	Options
Lambda	Publish version → shift alias weights (manual, or CodeDeploy / SAM `DeploymentPreference`: `Canary10Percent5Minutes`, `Linear10PercentEvery1Minute`, `AllAtOnce`…). Rollback = point the alias back; CodeDeploy does it on alarm.
Amazon ECS	ECS deployment controller: `ROLLING` (default; `minimumHealthyPercent` / `maximumPercent`), plus native `BLUE_GREEN`, `LINEAR` and `CANARY` (need ALB/NLB, Service Connect or VPC Lattice). **Deployment circuit breaker** with rollback returns to the last good deployment when tasks keep failing; CloudWatch alarms can also stop and roll back. CodeDeploy blue/green is still an option.
Amazon EKS	Kubernetes manifests (`Deployment` with `RollingUpdate`: `maxSurge`, `maxUnavailable`), readiness probes gate traffic; `kubectl rollout undo` to roll back. Fargate profiles run pods without nodes.
Elastic Beanstalk	All at once · Rolling · Rolling with additional batch · Immutable (new instances, safest) · Traffic splitting (canary) · blue/green by CNAME swap. `.ebextensions`, `.platform/hooks`, Procfile.
CloudFormation	Failed create/update rolls back automatically; rollback triggers on alarms; change sets to preview; stack policies; `DeletionPolicy: Retain/Snapshot`.
API Gateway	Stage canary settings — send a percentage of traffic to a new deployment, then promote.

Containers (Skill 3.3.2)

- **ECS task definition** — image, CPU/memory, port mappings, environment and `secrets`, log driver (`awslogs`), health check, task role + task execution role. A new revision per change; the service points at a revision.
- **Launch types** — Fargate (serverless, per-task vCPU/memory) or EC2 (you manage instances; capacity providers).
- **EKS** — manifests or Helm; Pod Identity for AWS permissions; probes for health.

Infrastructure as code (Skill 3.3.3)

Tool	Facts
CloudFormation	Parameters, Mappings, Conditions, Outputs/Exports (`!ImportValue`), intrinsic functions (`!Ref` , `!GetAtt` , `!Sub` , `!If`), nested stacks, StackSets, drift detection, `cfn-init` / `cfn-signal` + `CreationPolicy` .
AWS SAM	`Transform: AWS::Serverless-2016-10-31` ; `AWS::Serverless::Function` , `Api` , `HttpApi` , `SimpleTable` ; `Globals` ; policy templates; `AutoPublishAlias` + `DeploymentPreference` for safe Lambda deployments.
AWS CDK	IaC in TypeScript, Python, Java, C#, Go; constructs L1/L2/L3; `cdk bootstrap` once per account/Region; `cdk synth` → CloudFormation; `cdk diff` , `cdk deploy` .

Exam Tip: buildspec.yml = CodeBuild; appspec.yml = CodeDeploy; template.yaml = SAM / CloudFormation. "Gradual Lambda rollout with automatic rollback" = CodeDeploy canary/linear + CloudWatch alarm. "ECS service keeps failing after a bad task definition" = deployment circuit breaker with rollback. "Old images pile up in ECR" = lifecycle policy. "Release tag must never point to a different image" = tag immutability.

5 · Domain 4 — Troubleshooting and Optimization (22%)

Root cause analysis (Task 4.1)

- **Read the signal** — metrics show *that* something is wrong (Errors, Throttles, Duration, IteratorAge, ApproximateAgeOfOldestMessage, 4XX/5XX, latency); logs show *what*; traces show *where* across services.
- **CloudWatch Logs Insights** — `fields @timestamp, @message | filter @message like /ERROR/ | stats count(*) by bin(5m) | sort @timestamp desc` . Lambda `REPORT` lines give duration, billed duration, max memory used, init duration.
- **Deployment failures** (4.1.4) — CodeBuild build logs, CodeDeploy deployment and lifecycle event logs (on the instance under `/opt/codedeploy-agent/`), CloudFormation stack events (first `CREATE_FAILED` is the cause), ECS service events and stopped-task reasons, Beanstalk environment events and logs.
- **Common Lambda errors** — `Task timed out` (raise timeout or fix slow dependency), `Runtime.ImportModuleError` (missing package/layer or wrong handler), `AccessDeniedException` (execution role), throttling (429, `TooManyRequestsException` : reserved concurrency or account limit).

Logging, metrics, alerts & tracing (Task 4.2)

Need	Answer
Structured logging	JSON logs with request ID, correlation/trace ID, user/tenant, event type; Lambda JSON log format and log levels; Powertools for AWS Lambda Logger. Set log retention.
Custom metrics	`PutMetricData` (standard 1 min or high-resolution 1 s), or **Embedded Metric Format** — metrics inside log lines, no extra API calls from Lambda. Metric filters turn log patterns into metrics.
Alerts (4.2.3)	CloudWatch alarm → SNS topic (email, SMS, Lambda, chat). Quota limits: alarm on Service Quotas usage metrics (`AWS/Usage`). Deployment completion/failure: EventBridge rules on CodePipeline / CodeDeploy / CloudFormation / ECS state-change events, or notification rules → SNS. Composite alarms reduce noise; anomaly detection alarms for seasonal metrics.
Tracing (4.2.4)	**AWS X-Ray** traces, trace map, segments/subsegments, annotations (indexed, filterable) vs metadata (not indexed), sampling rules. Turn on active tracing for Lambda and API Gateway. Instrument code with **OpenTelemetry** (AWS Distro for OpenTelemetry, or the AWS Lambda layer for OpenTelemetry) and collect with the CloudWatch agent or OpenTelemetry Collector. CloudWatch Application Signals for service-level health.

Need	Answer
Health (4.2.5)	ALB/NLB target group health checks; ECS container `healthCheck`; Kubernetes **liveness** (restart a stuck container), **readiness** (take it out of traffic), **startup** (slow starters — hold other probes until ready) probes; Route 53 health checks; CloudWatch Synthetics canaries.

AWS X-Ray SDKs and X-Ray daemon DEPRECATED — in maintenance mode since 25 February 2026 (security fixes only). AWS recommends OpenTelemetry SDKs / ADOT for instrumentation and the CloudWatch agent or OpenTelemetry Collector instead of the daemon. X-Ray itself (traces, trace map, APIs) is current and in scope; exam questions may still mention the SDK and daemon (UDP port 2000, needed on EC2/ECS, not on Lambda).

Optimization (Task 4.3)

Lever	Details
Concurrency	Reserved concurrency caps a function (and guarantees it); provisioned concurrency pre-warms environments (no cold starts, paid); **SnapStart** snapshots the initialised environment (Java 11+, Python 3.12+, .NET 8+; published versions only; not with provisioned concurrency). EC2 Auto Scaling target tracking; thread pools and connection pooling (RDS Proxy) in code.
Memory / compute	More Lambda memory = more CPU = often cheaper total duration; tune with the open-source Lambda Power Tuning tool; Graviton (arm64) for price-performance. Initialise SDK clients outside the handler.
Messaging	Batch send/receive/delete (10 per call), long polling, Lambda batch size + batching window, SNS fan-out instead of point-to-point copies, filter early (SNS filter policies, EventBridge patterns, ESM event filtering).
Caching	ElastiCache (app data, sessions), DAX (DynamoDB reads), CloudFront (edge, cache keys/TTLs), API Gateway stage cache (REST), Lambda `/tmp` and global variables for warm reuse.
Find bottlenecks	CloudWatch metrics and dashboards, Lambda Insights, Container Insights, X-Ray latency by segment, Logs Insights on `REPORT` lines; AI-assisted analysis with CloudWatch investigations / AWS DevOps Agent / Kiro.

Exam Tip: "Cold starts on a Java API" → SnapStart (or provisioned concurrency). "Function must never use more than 50 concurrent executions" → reserved concurrency = 50. "Find which downstream call is slow" → X-Ray / OpenTelemetry traces. "Alert when we approach a service quota" → CloudWatch alarm on usage metric. "Emit business metrics from Lambda cheaply" → Embedded Metric Format.

6 · Networking, Compute & Storage Reference

Service	What a developer needs
Amazon VPC	Public subnets route to an internet gateway; private subnets use a NAT gateway for outbound. Security groups (stateful, allow only) vs NACLs (stateless, subnet).
AWS PrivateLink	Interface VPC endpoints to AWS services (Bedrock, Secrets Manager, SQS…) or your own services, with private IPs; gateway endpoints for S3 and DynamoDB are free.
Elastic Load Balancing	ALB (HTTP/HTTPS, path/host routing, Lambda targets, Cognito/OIDC auth, sticky sessions) · NLB (TCP/UDP/TLS, static IPs, client IP preserved) · deregistration delay for draining.
Amazon CloudFront	CDN; origin access control (OAC) for private S3; signed URLs/cookies; CloudFront Functions (lightweight viewer request/response) vs Lambda@Edge (heavier, origin events too).
Amazon Route 53	DNS + health checks; alias records to AWS resources; routing: simple, weighted, latency, failover, geolocation, multivalue.

Service	What a developer needs
AWS WAF	Web ACLs on CloudFront, ALB, API Gateway, Cognito: managed rule groups (SQLi, XSS), rate-based rules, geo match.
Amazon EC2	User data runs once at first boot; IMDSv2 (session token; require it with `HttpTokens=required`); instance profile for credentials; launch templates + Auto Scaling.
Elastic Beanstalk	Upload code; Beanstalk provisions EC2, ALB, Auto Scaling; deployment policies above.
EBS · EFS	Block storage for one instance (gp3 default) · shared NFS for many EC2/ECS/Lambda.
S3 Glacier	Archive classes: Glacier Instant Retrieval (ms access), Flexible Retrieval (minutes–hours, restore first), Deep Archive (hours, cheapest). Move data there with lifecycle rules.
AWS CloudTrail	Who called which API, when, from where. Management events on by default; data events (S3 objects, Lambda invokes) optional. CloudWatch = performance; CloudTrail = audit.
AWS Systems Manager	Parameter Store, Session Manager (no SSH port), Run Command, AppConfig.

7 · Scenario → Answer Patterns

Scenario	Answer
Code on EC2/Lambda needs AWS access	IAM role (instance profile / execution role), never access keys.
Lambda hits RDS connection limits	RDS Proxy.
Java Lambda cold starts	SnapStart on a published version (or provisioned concurrency).
Async Lambda failures must be captured with payload	On-failure destination (SQS/SNS/EventBridge/Lambda).
SQS message processed twice	Visibility timeout ≥ processing time (6× function timeout for Lambda) + idempotent handler.
Strict order per customer	SQS FIFO with `MessageGroupId` = customer ID.
Message larger than 1 MiB	Store payload in S3; send a pointer (Extended Client Library).
Replayable clickstream, several consumers	Kinesis Data Streams (+ enhanced fan-out).
Load streaming logs into S3 as Parquet, no code	Amazon Data Firehose with format conversion.
Browser uploads big files to a private bucket	S3 presigned URL (PUT / multipart).
Private videos through the CDN, many files	CloudFront signed cookies with a trusted key group.
HTTPS on a CloudFront custom domain	ACM public certificate in us-east-1, DNS validation.
Mutual TLS between internal services	AWS Private CA (short-lived certificate mode for frequent rotation).
Rotate database password automatically	Secrets Manager rotation.
Encrypt a 5 MB file with KMS	`GenerateDataKey` + envelope encryption.
Call Bedrock from private subnets with no internet	Interface VPC endpoint for `bedrock-runtime` (PrivateLink).
Block prompt injection and mask PII in an AI chat	Bedrock Guardrails (prompt attack + sensitive information filters).
Agent action needs human approval	Human-in-the-loop step (Step Functions task token or confirmation before the tool call).

Scenario	Answer
Define requirements before AI writes code	Kiro spec (requirements.md → design.md → tasks.md).
Front end needs the API before the backend exists	API Gateway mock integration.
Separate dev/test/prod Lambda without new functions	Versions + aliases (+ API Gateway stage variables).
Preview every pull request of a web app	Amplify Hosting pull request previews / branch environments.
Gradual Lambda release, roll back on errors	CodeDeploy canary or linear + CloudWatch alarm.
ECS service stuck on a bad revision	Deployment circuit breaker with rollback.
Slow-starting container killed by health checks on EKS	Startup probe (and a readiness probe for traffic).
Find the slow call across microservices	X-Ray traces via OpenTelemetry / ADOT.
Notify the team when a deployment finishes	EventBridge rule on the deployment state-change event → SNS.
Who deleted the S3 object?	CloudTrail data events.
Roll out a feature flag to 10% with automatic rollback	AWS AppConfig with a gradual deployment strategy + alarm.

8 · Closed, Retired & Removed

Service / feature	Status and replacement
X-Ray SDKs & daemon DEPRECATED	Maintenance mode since 25 Feb 2026 → OpenTelemetry / ADOT + CloudWatch agent or OpenTelemetry Collector. X-Ray service stays current.
Amazon Q Developer IDE plugins DEPRECATED	End of support 30 Apr 2027 → Kiro. Removed from the DVA-C03 in-scope list.
AWS Copilot CLI DEPRECATED	End of support 12 Jun 2026 (no updates or fixes; existing deployments keep running) → ECS console/CLI, CDK, CloudFormation.
Amazon CodeGuru Reviewer DEPRECATED	No new repository associations since 7 Nov 2025 → AI code review in Kiro, Amazon Inspector code scanning.
AWS Cloud9 DEPRECATED	Closed to new customers → CloudShell, local IDE / Kiro + SAM.
Amazon CodeCatalyst DEPRECATED	Closed to new customers → CodePipeline, CodeBuild, GitHub/GitLab.
Bedrock Agents (Classic) DEPRECATED	Closed to new customers from 30 Jul 2026 → Bedrock AgentCore.
Amazon Q Business DEPRECATED	Closed to new customers → Amazon Quick.
AWS CodeStar RETIRED	Shut down 25 Jul 2024 → CodePipeline + CodeBuild + CodeDeploy.
CloudWatch Evidently RETIRED	Shut down 17 Oct 2025 → AWS AppConfig feature flags.
Back as active	AWS CodeCommit — open to new customers again since Nov 2025.
Removed from the DVA-C03 guide	AWS AppSync (still an active service, just out of scope) and Amazon Q Developer. Out of scope: Quick Sight, SageMaker AI, Strands Agents, Polly, Rekognition, Shield, Snow Family, Storage Gateway, Transfer Family.
Renamed	QuickSight → Amazon Quick Sight (part of Amazon Quick); Kinesis Data Firehose → Amazon Data Firehose; CloudWatch Events → Amazon EventBridge; Amazon Q Developer → Kiro (upgrade path).

DVA-C03 default answers: Credentials → IAM role. Decouple → SQS / SNS / EventBridge. Long workflow → Step Functions. Hot DynamoDB key → better partition key. Secrets → Secrets Manager. Big data + KMS → envelope encryption. Temporary file access → presigned URL. AI safety → Guardrails. Private AI traffic → PrivateLink. Safe Lambda release → aliases + CodeDeploy. Traces → X-Ray with OpenTelemetry. Spec before code → Kiro.

Built by **CloudNinja** · **cloudninja.pro** · Free AWS practice exams & cheat sheets. No signup, no paywall.