AWS Certified Developer - Associate Cheat Sheet

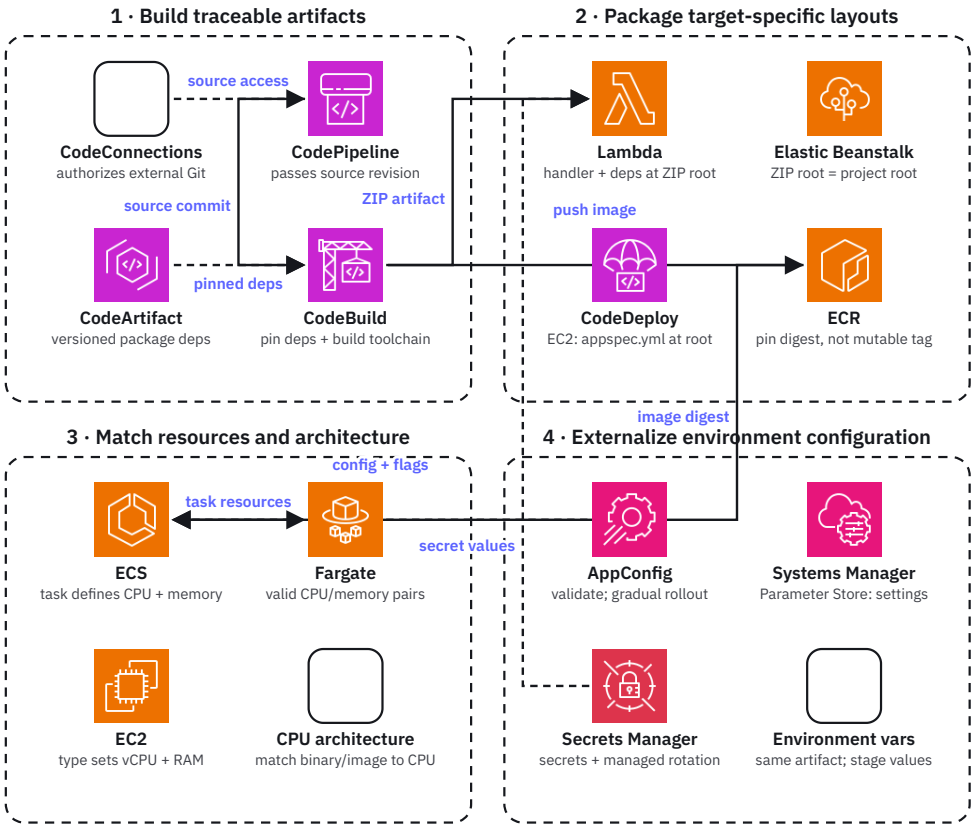
DVA-C02 · Domain 3 of 4

BlowStack

DOMAIN 3 · DEPLOYMENT · 24% OF THE EXAM · TASK STATEMENT 3.1

Prepare application artifacts to be deployed to AWS

Pin, package, size, configure



1 Build traceable artifacts

Use **pinned inputs** (lockfiles, toolchain, base-image digest) for reproducible builds. CodeConnections authorizes external Git; CodeCommit is a direct source. CodeArtifact stores packages, not container images.

2 Package target-specific layouts

Use **target-specific layouts**: Lambda handler and dependencies at ZIP root; Elastic Beanstalk .ebextensions at bundle root. CodeDeploy EC2 requires root appspec.yml; Lambda/ECS use YAML or JSON AppSpec.

3 Match resources and architecture

Lambda **memory** also determines CPU allocation; CPU is not independently selectable. Fargate requires supported task CPU/memory pairs. Benchmarks, expected concurrency, and quotas drive capacity planning.

4 Externalize environment configuration

Use **AWS AppConfig** for validated flags/config with gradual rollout and alarm-triggered rollback; Parameter Store for settings, Secrets Manager for rotating credentials. Environment variables are deployment-time inputs.

Exam traps

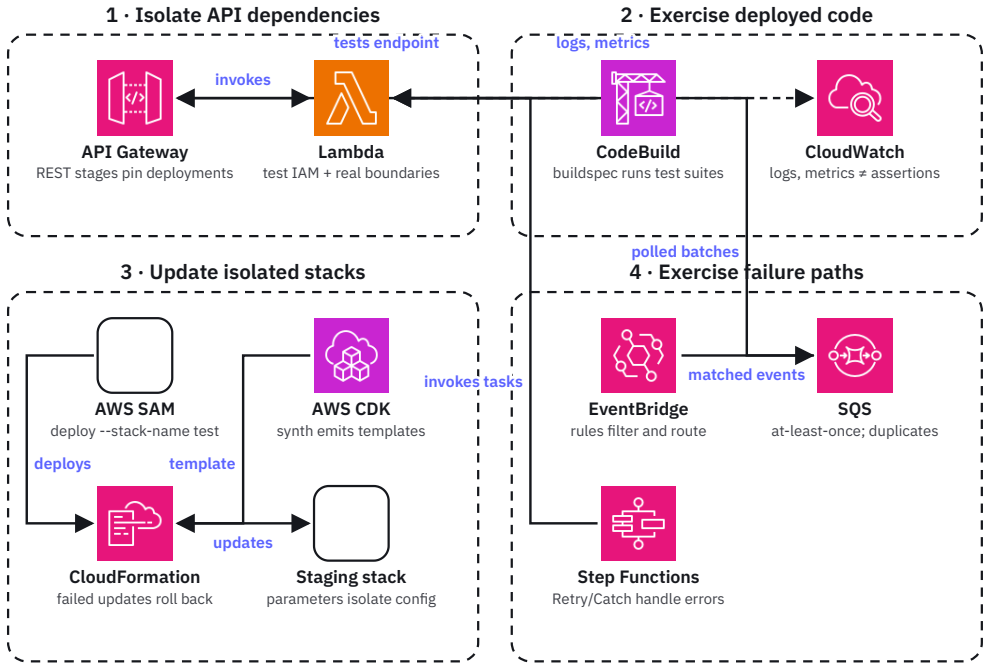
- An extra enclosing folder can break deployment: **archive root** determines Lambda imports, Elastic Beanstalk layout, and EC2 CodeDeploy's appspec.yml discovery.
- Changing only the runtime to arm64 does not convert dependencies: **native binaries and container images** must match the target CPU architecture.
- **Environment variables** do not provide live feature-flag delivery or automatic credential rotation. Applications must fetch configuration updates and retrieve refreshed secrets.

Get practice exams and lessons for AWS Certified Developer - Associate

DOMAIN 3 · DEPLOYMENT · TASK STATEMENT 3.2

Test applications in development environments

Isolate, test, redeploy, retry



1 Isolate API dependencies

Use **MOCK integration** for fixed responses without a backend; contract tests check schemas/status codes. REST API stages select deployments; real integration tests still verify permissions and configuration.

2 Exercise deployed code

Use **AWS CodeBuild** to run end-to-end tests against deployed endpoints and rerun regression suites after changes. Amazon CloudWatch supplies diagnostic evidence; test assertions determine pass/fail.

3 Update isolated stacks

Use **separate stacks** for staging isolation and the existing stack name for updates. Parameterize environment differences; change sets preview changes. Failed updates roll back by default, unless rollback is disabled.

4 Exercise failure paths

Test **duplicate delivery** and retries; handlers must be idempotent. For SQS batches, enable ReportBatchItemFailures and return failed item IDs so successful records are not retried.

Exam traps

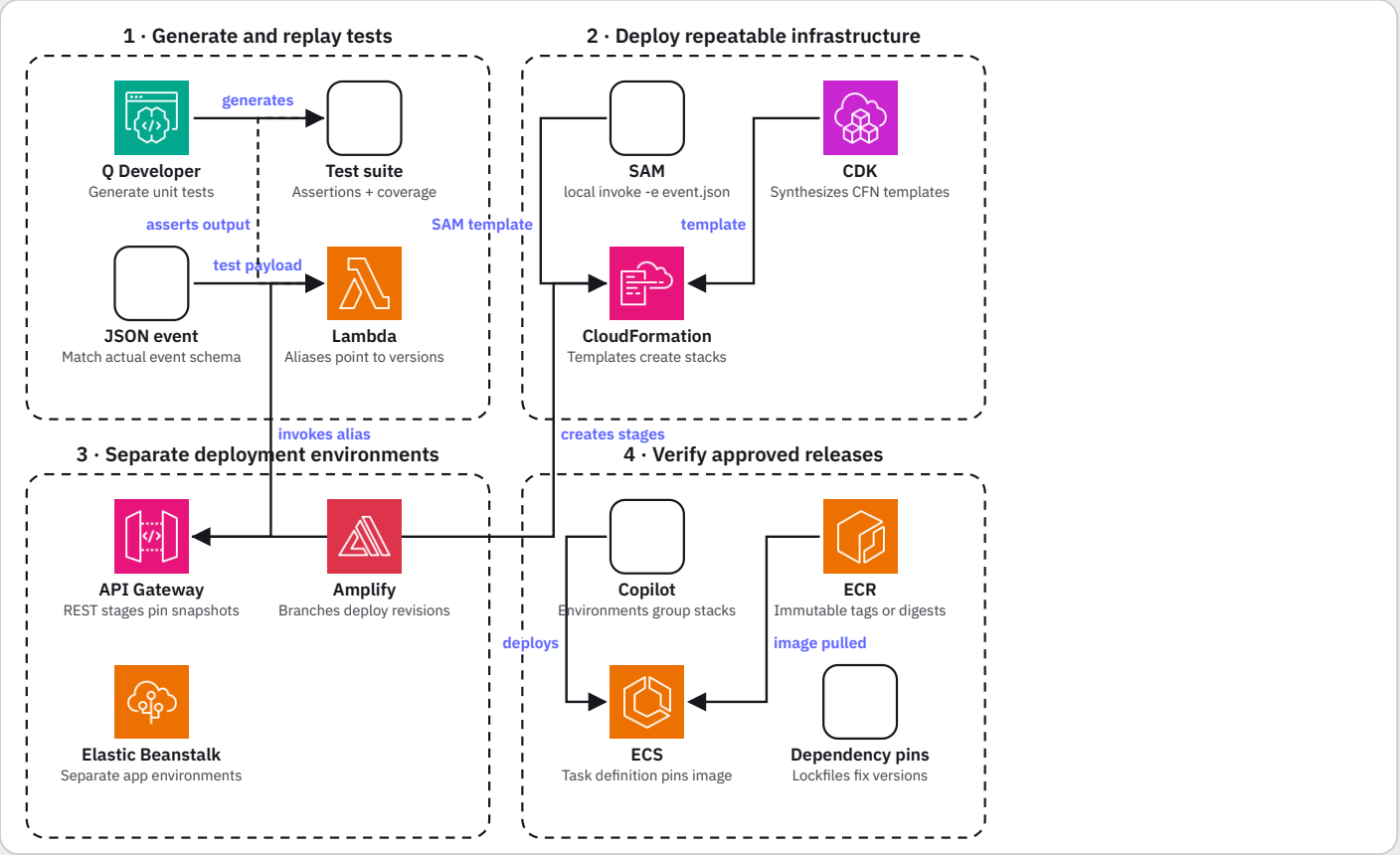
- **Passing mocks** does not prove integration works: test doubles cannot verify real IAM permissions, network access, or provider behavior. Contract tests reduce mock drift, not all integration risk.
- SAM **--config-env** selects saved CLI settings, not a new environment; a different stack name creates a separate stack. CloudFormation rollback does not undo external data writes.
- **SQS is polled**, not a Lambda asynchronous invocation source. Test visibility timeouts, redelivery, and queue redrive policies; Lambda asynchronous retry settings do not govern SQS processing.

Get practice exams and lessons for AWS Certified Developer - Associate

DOMAIN 3 · DEPLOYMENT · TASK STATEMENT 3.3

Automate deployment testing

Generate, deploy, isolate, verify



1 Generate and replay tests

Use **Amazon Q Developer**, which includes CodeWhisperer capabilities, to generate unit tests. Review assertions and mock dependencies; coverage measures exercised code, not correctness.

2 Deploy repeatable infrastructure

Use **IaC** for repeatable environments: CDK synthesizes CloudFormation templates; CloudFormation transforms SAM resources. sam deploy creates or updates a stack; local invocation does not deploy resources.

3 Separate deployment environments

Use REST API **stages** for dev/test/prod deployment snapshots; stage variables can select Lambda aliases. Amplify supports branch deployments; Beanstalk environments can run different application versions.

4 Verify approved releases

Run **integration tests** against deployed dependencies using approved Lambda versions via aliases and digest-pinned container images. Copilot environments separate deployments; lockfiles pin libraries.

Exam traps

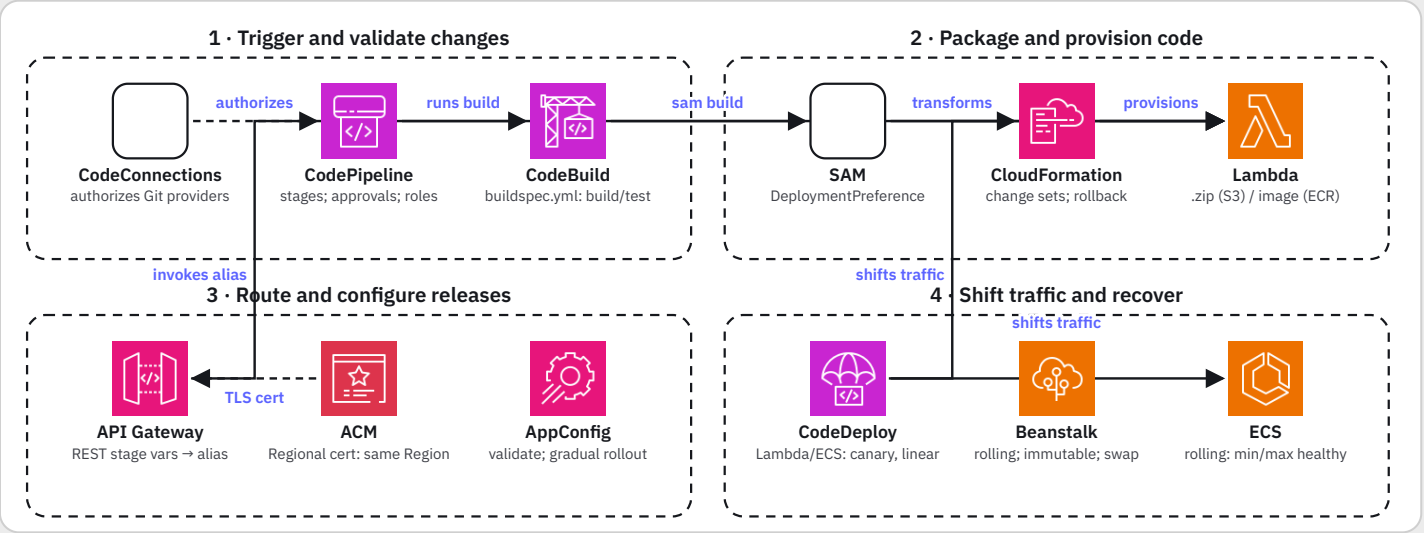
- Editing a REST API does not update its deployed endpoint: **redeploy to the stage**. Stage variables configure integrations; they do not deploy API changes.
- **Approved versions require controlled references**: Lambda aliases are movable pointers to published versions; $LATEST and mutable image tags do not pin releases. Use approved versions or image digests.
- **Local tests are not deployed integration tests**: SAM local invocation and mocked unit tests cannot prove deployed IAM permissions, network access, or real dependency compatibility.

Get practice exams and lessons for AWS Certified Developer - Associate

DOMAIN 3 · DEPLOYMENT · TASK STATEMENT 3.4

Deploy code by using AWS Continuous Integration and Continuous Delivery (CI/CD) services

Validate, provision, route, release



1 Trigger and validate changes

A **source event** triggers configured branch/tag filters. Promote the same tested artifact through isolated environments and approval gates; cross-account actions assume roles. Step Functions supports custom orchestration.

2 Package and provision code

A **change set** previews template updates; execution applies them. SAM transforms into CloudFormation; AutoPublishAlias plus DeploymentPreference enables managed Lambda traffic shifts. Separate stacks isolate environments.

3 Route and configure releases

A **stage** names an API deployment; REST stage variables can select a Lambda alias. Regional custom domains use same-Region ACM certificates and Route 53 aliases. AppConfig externalizes settings without rebuilding code.

4 Shift traffic and recover

Use **blue/green** to validate a replacement before switching traffic; Beanstalk swaps environment CNAMEs. Canary shifts a small share first; linear shifts equal increments. CloudWatch alarms trigger rollback only when configured.

Exam traps

- An ECR image does not make Lambda an ECS task. Lambda accepts **.zip or container image**, but an existing function cannot switch package type; create another function.
- **Rolling is not immutable**: rolling updates existing capacity in batches; Beanstalk immutable launches replacement instances. Lambda weighted aliases route published versions, not $LATEST.
- **Rollback does not undo data migrations**. Preserve known-good artifacts and backward-compatible schemas. Git branches select development lines; tags label revisions, not deployed environments.

Get practice exams and lessons for AWS Certified Developer - Associate

DOMAIN 3 · DEPLOYMENT · WHAT TO REMEMBER

Concepts and services of Deployment

Concepts to know

CI/CD

CI automates builds and tests; delivery pipelines promote validated releases across environments. **Promote the same tested artifact**, using automated tests and approval gates rather than rebuilding separately for production.

Infrastructure as code (IaC)

Versioned templates define repeatable infrastructure. **Parameterize environment differences** to reuse templates across isolated stacks; change sets preview proposed updates but do not guarantee successful deployment.

Deployment Rollback

Rollback restores a known-good release when health checks or alarms detect failure. **Traffic shifting enables rapid recovery**, but old code must remain compatible with changed data and dependencies; infrastructure rollback does not undo database writes.

Core services



AWS Lambda

Deploy code as ZIP archives or container images. **Aliases target published versions** and can split traffic between two versions; published versions are immutable, unlike $LATEST.



AWS CodeDeploy

Automates releases with hooks and alarm-based rollback. **Canary or linear traffic shifting** applies to Lambda aliases; ECS deployments use blue/green, while EC2 supports in-place or blue/green.



AWS CodePipeline

Orchestrates source, build, test, and deployment actions. **Manual approval gates** pause promotion; AWS CodeBuild runs build and test commands, while deployment actions release the artifacts.

Get practice exams and lessons for AWS Certified Developer - Associate