

AWS Certified Developer - Associate (DVA-C02) Exam Guide

Introduction

The AWS Certified Developer - Associate (DVA-C02) exam is intended for individuals who perform a developer role. The exam validates a candidate's ability to demonstrate proficiency in developing, testing, deploying, and debugging AWS Cloud-based applications.

The exam also validates a candidate's ability to complete the following tasks:

- Develop and optimize applications on AWS.
- Package and deploy by using continuous integration and continuous delivery (CI/CD) workflows.
- Secure application code and data.
- Identify and resolve application issues.

Target candidate description

The target candidate should have 1 or more years of hands-on experience in developing and maintaining applications by using AWS services.

Recommended general IT knowledge

The target candidate should have the following general IT knowledge:

- Proficiency in at least one high-level programming language
- Understanding of application lifecycle management
- Basic understanding of cloud-native applications to write code
- Ability to develop functional applications
- Experience in using development tools

Recommended AWS knowledge

The target candidate should be able to complete the following tasks:

- Develop and secure applications by using AWS service APIs, the AWS CLI, and SDKs.
- Use a CI/CD pipeline to deploy applications on AWS.

Job tasks that are out of scope for the target candidate

The following list contains job tasks that the target candidate is not expected to be able to perform. This list is non-exhaustive. These tasks are out of scope for the exam:

- Design architectures (for example, distributed systems, microservices, database schemas and modeling).
- Design and create CI/CD pipelines.
- Administer IAM users and groups.
- Administer servers and operating systems.
- Design AWS networking infrastructure (for example, Amazon VPC, AWS Direct Connect).

Refer to the Appendix for a list of technologies and concepts that might appear on the exam, a list of in-scope AWS services and features, and a list of out-of-scope AWS services and features.

Exam content

Response types

There are two types of questions on the exam:

- **Multiple choice:** Has one correct response and three incorrect responses (distractors)
- **Multiple response:** Has two or more correct responses out of five or more response options

Select one or more responses that best complete the statement or answer the question. Distractors, or incorrect answers, are response options that a candidate with incomplete knowledge or skill might choose. Distractors are generally plausible responses that match the content area.

Unanswered questions are scored as incorrect; there is no penalty for guessing. The exam includes 50 questions that affect your score.

Unscored content

The exam includes 15 unscored questions that do not affect your score. AWS collects information about performance on these unscored questions to evaluate these

questions for future use as scored questions. These unscored questions are not identified on the exam.

Exam results

The AWS Certified Developer - Associate (DVA-C02) exam has a pass or fail designation. The exam is scored against a minimum standard established by AWS professionals who follow certification industry best practices and guidelines.

Your results for the exam are reported as a scaled score of 100–1,000. The minimum passing score is 720. Your score shows how you performed on the exam as a whole and whether you passed. Scaled scoring models help equate scores across multiple exam forms that might have slightly different difficulty levels.

Your score report could contain a table of classifications of your performance at each section level. The exam uses a compensatory scoring model, which means that you do not need to achieve a passing score in each section. You need to pass only the overall exam.

Each section of the exam has a specific weighting, so some sections have more questions than other sections have. The table of classifications contains general information that highlights your strengths and weaknesses. Use caution when you interpret section-level feedback.

Content outline

This exam guide includes weightings, content domains, and task statements for the exam. This guide does not provide a comprehensive list of the content on the exam. However, additional context for each task statement is available to help you prepare for the exam.

The exam has the following content domains and weightings:

- Domain 1: Development with AWS Services (32% of scored content)
- Domain 2: Security (26% of scored content)
- Domain 3: Deployment (24% of scored content)
- Domain 4: Troubleshooting and Optimization (18% of scored content)

Domain 1: Development with AWS Services

Task Statement 1: Develop code for applications hosted on AWS.

Knowledge of:

- Architectural patterns (for example, event-driven, microservices, monolithic, choreography, orchestration, fanout)
- Idempotency
- Differences between stateful and stateless concepts
- Differences between tightly coupled and loosely coupled components
- Fault-tolerant design patterns (for example, retries with exponential backoff and jitter, dead-letter queues)
- Differences between synchronous and asynchronous patterns

Skills in:

- Creating fault-tolerant and resilient applications in a programming language (for example, Java, C#, Python, JavaScript, TypeScript, Go)
- Creating, extending, and maintaining APIs (for example, response/request transformations, enforcing validation rules, overriding status codes)
- Writing and running unit tests in development environments (for example, using AWS Serverless Application Model [AWS SAM])
- Writing code to use messaging services
- Writing code that interacts with AWS services by using APIs and AWS SDKs
- Handling data streaming by using AWS services

Task Statement 2: Develop code for AWS Lambda.

Knowledge of:

- Event source mapping
- Stateless applications
- Unit testing
- Event-driven architecture
- Scalability
- The access of private resources in VPCs from Lambda code

Skills in:

- Configuring Lambda functions by defining environment variables and parameters (for example, memory, concurrency, timeout, runtime, handler, layers, extensions, triggers, destinations)
- Handling the event lifecycle and errors by using code (for example, Lambda Destinations, dead-letter queues)
- Writing and running test code by using AWS services and tools
- Integrating Lambda functions with AWS services
- Tuning Lambda functions for optimal performance

Task Statement 3: Use data stores in application development.

Knowledge of:

- Relational and non-relational databases
- Create, read, update, and delete (CRUD) operations
- High-cardinality partition keys for balanced partition access
- Cloud storage options (for example, file, object, databases)
- Database consistency models (for example, strongly consistent, eventually consistent)
- Differences between query and scan operations
- Amazon DynamoDB keys and indexing
- Caching strategies (for example, write-through, read-through, lazy loading, TTL)
- Amazon S3 tiers and lifecycle management
- Differences between ephemeral and persistent data storage patterns

Skills in:

- Serializing and deserializing data to provide persistence to a data store
- Using, managing, and maintaining data stores
- Managing data lifecycles
- Using data caching services

Domain 2: Security

Task Statement 1: Implement authentication and/or authorization for applications and AWS services.

Knowledge of:

- Identity federation (for example, Security Assertion Markup Language [SAML], OpenID Connect [OIDC], Amazon Cognito)
- Bearer tokens (for example, JSON Web Token [JWT], OAuth, AWS Security Token Service [AWS STS])
- The comparison of user pools and identity pools in Amazon Cognito
- Resource-based policies, service policies, and principal policies
- Role-based access control (RBAC)
- Application authorization that uses ACLs
- The principle of least privilege
- Differences between AWS managed policies and customer-managed policies
- Identity and access management

Skills in:

- Using an identity provider to implement federated access (for example, Amazon Cognito, AWS Identity and Access Management [IAM])
- Securing applications by using bearer tokens
- Configuring programmatic access to AWS
- Making authenticated calls to AWS services
- Assuming an IAM role
- Defining permissions for principals

Task Statement 2: Implement encryption by using AWS services.

Knowledge of:

- Encryption at rest and in transit
- Certificate management (for example, AWS Private Certificate Authority)
- Key protection (for example, key rotation)
- Differences between client-side encryption and server-side encryption
- Differences between AWS managed and customer managed AWS Key Management Service (AWS KMS) keys

Skills in:

- Using encryption keys to encrypt or decrypt data
- Generating certificates and SSH keys for development purposes
- Using encryption across account boundaries
- Enabling and disabling key rotation

Task Statement 3: Manage sensitive data in application code.

Knowledge of:

- Data classification (for example, personally identifiable information [PII], protected health information [PHI])
- Environment variables
- Secrets management (for example, AWS Secrets Manager, AWS Systems Manager Parameter Store)
- Secure credential handling

Skills in:

- Encrypting environment variables that contain sensitive data
- Using secret management services to secure sensitive data
- Sanitizing sensitive data

Domain 3: Deployment

Task Statement 1: Prepare application artifacts to be deployed to AWS.

Knowledge of:

- Ways to access application configuration data (for example, AWS AppConfig, Secrets Manager, Parameter Store)
- Lambda deployment packaging, layers, and configuration options
- Git-based version control tools (for example, Git, AWS CodeCommit)
- Container images

Skills in:

- Managing the dependencies of the code module (for example, environment variables, configuration files, container images) within the package
- Organizing files and a directory structure for application deployment
- Using code repositories in deployment environments
- Applying application requirements for resources (for example, memory, cores)

Task Statement 2: Test applications in development environments.

Knowledge of:

- Features in AWS services that perform application deployment
- Integration testing that uses mock endpoints
- Lambda versions and aliases

Skills in:

- Testing deployed code by using AWS services and tools
- Performing mock integration for APIs and resolving integration dependencies
- Testing applications by using development endpoints (for example, configuring stages in Amazon API Gateway)
- Deploying application stack updates to existing environments (for example, deploying an AWS SAM template to a different staging environment)

Task Statement 3: Automate deployment testing.

Knowledge of:

- API Gateway stages
- Branches and actions in the continuous integration and continuous delivery (CI/CD) workflow
- Automated software testing (for example, unit testing, mock testing)

Skills in:

- Creating application test events (for example, JSON payloads for testing Lambda, API Gateway, AWS SAM resources)
- Deploying API resources to various environments
- Creating application environments that use approved versions for integration testing (for example, Lambda aliases, container image tags, AWS Amplify branches, AWS Copilot environments)
- Implementing and deploying infrastructure as code (IaC) templates (for example, AWS SAM templates, AWS CloudFormation templates)
- Managing environments in individual AWS services (for example, differentiating between development, test, and production in API Gateway)

Task Statement 4: Deploy code by using AWS CI/CD services.

Knowledge of:

- Git-based version control tools (for example, Git, AWS CodeCommit)
- Manual and automated approvals in AWS CodePipeline
- Access application configurations from AWS AppConfig and Secrets Manager
- CI/CD workflows that use AWS services
- Application deployment that uses AWS services and tools (for example, CloudFormation, AWS Cloud Development Kit [AWS CDK], AWS SAM, AWS CodeArtifact, AWS Copilot, Amplify, Lambda)
- Lambda deployment packaging options
- API Gateway stages and custom domains
- Deployment strategies (for example, canary, blue/green, rolling)

Skills in:

- Updating existing IaC templates (for example, AWS SAM templates, CloudFormation templates)
- Managing application environments by using AWS services
- Deploying an application version by using deployment strategies
- Committing code to a repository to invoke build, test, and deployment actions
- Using orchestrated workflows to deploy code to different environments
- Performing application rollbacks by using existing deployment strategies
- Using labels and branches for version and release management
- Using existing runtime configurations to create dynamic deployments (for example, using staging variables from API Gateway in Lambda functions)

Domain 4: Troubleshooting and Optimization

Task Statement 1: Assist in a root cause analysis.

Knowledge of:

- Logging and monitoring systems
- Languages for log queries (for example, Amazon CloudWatch Logs Insights)
- Data visualizations
- Code analysis tools
- Common HTTP error codes
- Common exceptions generated by SDKs
- Service maps in AWS X-Ray

Skills in:

- Debugging code to identify defects
- Interpreting application metrics, logs, and traces
- Querying logs to find relevant data
- Implementing custom metrics (for example, CloudWatch embedded metric format [EMF])
- Reviewing application health by using dashboards and insights
- Troubleshooting deployment failures by using service output logs

Task Statement 2: Instrument code for observability.

Knowledge of:

- Distributed tracing
- Differences between logging, monitoring, and observability
- Structured logging
- Application metrics (for example, custom, embedded, built-in)

Skills in:

- Implementing an effective logging strategy to record application behavior and state
- Implementing code that emits custom metrics
- Adding annotations for tracing services
- Implementing notification alerts for specific actions (for example, notifications about quota limits or deployment completions)
- Implementing tracing by using AWS services and tools

Task Statement 3: Optimize applications by using AWS services and features.

Knowledge of:

- Caching
- Concurrency
- Messaging services (for example, Amazon Simple Queue Service [Amazon SQS], Amazon Simple Notification Service [Amazon SNS])

Skills in:

- Profiling application performance
- Determining minimum memory and compute power for an application
- Using subscription filter policies to optimize messaging
- Caching content based on request headers

Appendix

Technologies and concepts that might appear on the exam

The following list contains technologies and concepts that might appear on the exam. This list is non-exhaustive and is subject to change. The order and placement of the items in this list is no indication of their relative weight or importance on the exam:

- Analytics
- Application integration
- Compute
- Containers
- Cost and capacity management
- Database
- Developer tools
- Management and governance
- Networking and content delivery
- Security, identity, and compliance
- Storage

In-scope AWS services and features

The following list contains AWS services and features that are in scope for the exam. This list is non-exhaustive and is subject to change. AWS offerings appear in categories that align with the offerings' primary functions:

Analytics:

- Amazon Athena
- Amazon Kinesis
- Amazon OpenSearch Service

Application Integration:

- AWS AppSync
- Amazon EventBridge
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Queue Service (Amazon SQS)
- AWS Step Functions

Compute:

- Amazon EC2
- AWS Elastic Beanstalk
- AWS Lambda
- AWS Serverless Application Model (AWS SAM)

Containers:

- AWS Copilot
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Elastic Container Service (Amazon ECS)
- Amazon Elastic Kubernetes Service (Amazon EKS)

Database:

- Amazon Aurora
- Amazon DynamoDB
- Amazon ElastiCache
- Amazon MemoryDB for Redis
- Amazon RDS

Developer Tools:

- AWS Amplify
- AWS Cloud9
- AWS CloudShell
- AWS CodeArtifact
- AWS CodeBuild
- AWS CodeCommit
- AWS CodeDeploy
- Amazon CodeGuru
- AWS CodePipeline
- AWS CodeStar
- Amazon CodeWhisperer
- AWS X-Ray

Management and Governance:

- AWS AppConfig
- AWS CLI
- AWS Cloud Development Kit (AWS CDK)
- AWS CloudFormation
- AWS CloudTrail
- Amazon CloudWatch
- Amazon CloudWatch Logs
- AWS Systems Manager

Networking and Content Delivery:

- Amazon API Gateway
- Amazon CloudFront
- Elastic Load Balancing (ELB)
- Amazon Route 53
- Amazon VPC

Security, Identity, and Compliance:

- AWS Certificate Manager (ACM)
- Amazon Cognito
- AWS Identity and Access Management (IAM)
- AWS Key Management Service (AWS KMS)
- AWS Private Certificate Authority
- AWS Secrets Manager
- AWS Security Token Service (AWS STS)
- AWS WAF

Storage:

- Amazon Elastic Block Store (Amazon EBS)
- Amazon Elastic File System (Amazon EFS)
- Amazon S3
- Amazon S3 Glacier

Out-of-scope AWS services and features

The following list contains AWS services and features that are out of scope for the exam. This list is non-exhaustive and is subject to change. AWS offerings that are entirely unrelated to the target job roles for the exam are excluded from this list:

Analytics:

- Amazon QuickSight

Business Applications:

- Amazon Chime
- Amazon Connect
- Amazon WorkMail

End User Computing:

- Amazon AppStream 2.0
- Amazon WorkSpaces

Frontend Web and Mobile:

- AWS Device Farm

Game Tech:

- Amazon GameLift

Machine Learning:

- Amazon Lex
- Amazon Machine Learning (Amazon ML)
- Amazon Polly
- Amazon Rekognition

Management and Governance:

- AWS Managed Services (AMS)
- AWS Service Catalog

Media Services:

- Amazon Elastic Transcoder

Migration and Transfer:

- AWS Application Discovery Service
- AWS Application Migration Service
- AWS Database Migration Service (AWS DMS)

Security, Identity, and Compliance:

- AWS Shield Advanced
- AWS Shield Standard

Storage:

- AWS Snow Family
- AWS Storage Gateway

Survey

How useful was this exam guide? Let us know by [taking our survey](#).