Service Name

AWS Serverless Application Repository

Service Overview

AWS Serverless Application Repository is a managed service that makes it easy for developers and enterprises to discover, deploy, and publish serverless applications in the AWS Cloud. The service allows users to publish applications publicly to the community or privately within teams and organizations. Applications are defined using AWS Serverless Application Model (AWS SAM) templates and can be deployed with a few clicks after configuring required fields. The repository is deeply integrated with the AWS Lambda console, enabling developers to browse applications by categories, keywords, publishers, or event sources. Users can share code samples, components, and complete applications for various serverless use cases, accelerating development and promoting reuse of proven patterns.

Key Use Cases

- Use case 1: Publishing and sharing serverless applications publicly with the developer community or privately within teams and organizations
- Use case 2: Discovering and deploying pre-built serverless applications for common patterns like web backends, data processing, chatbots, and IoT applications
- Use case 3: Creating nested applications to reuse common serverless components and patterns across multiple applications, reducing duplication and enhancing organization

Features

- **Application Publishing**: Publish serverless applications using AWS Management Console, AWS SAM CLI, or AWS SDKs with AWS SAM templates
- **Application Discovery and Deployment**: Browse and search applications by categories, keywords, publishers, or event sources with one-click deployment
- **Access Control and Sharing**: Control application visibility with private, privately shared, or public access levels including AWS Organization integration
- **Nested Applications**: Create and deploy applications that contain other applications as components for code reuse and modularity
- **Lambda Console Integration**: Deep integration with AWS Lambda console for seamless serverless application management
- **Verified Author Badge**: Author verification system to establish trust and credibility for published applications

Value Proposition

AWS Serverless Application Repository accelerates serverless development by providing a centralized marketplace for discovering, sharing, and deploying proven serverless patterns. The service eliminates the need to build common functionality from scratch, reducing development time and effort:

- With deep Lambda console integration, developers can deploy applications with minimal configuration
- The repository supports both public community sharing and private organizational use, enabling knowledge sharing while maintaining security
- Verified author badges and application validation provide trust and quality assurance
- The service supports complex architectures through nested applications, enabling modular design and component reuse across multiple projects

Service Integration

AWS Serverless Application Repository integrates primarily with AWS Lambda for serverless compute functions and AWS CloudFormation for infrastructure deployment and stack management. Applications are deployed as CloudFormation stacks, enabling unified resource management:

- The service works closely with AWS SAM (Serverless Application Model) for application definition and packaging
- Integration with Amazon API Gateway enables API-based serverless applications, while Amazon DynamoDB integration supports database requirements
- Amazon S3 provides storage for code packages and deployment artifacts
- Additional integrations include AWS IAM for access control, AWS CodePipeline for CI/CD workflows, and AWS Organizations for enterprise-level application sharing and governance

Onboarding and Offboarding

Customers begin by accessing the AWS Serverless Application Repository through the AWS Management Console or AWS Lambda console. For deploying applications, users browse or search the repository, select an application, configure required parameters, acknowledge any necessary permissions (IAM roles, resource policies), and deploy with a few clicks:

- For publishing applications, users need a valid AWS account, AWS SAM template, packaged application, source code URL, and README file
- Applications can be published using AWS Management Console, AWS SAM CLI, or AWS SDKs

- The service provides quick start tutorials and integration with GitHub for automated publishing workflows

Getting Started Guide: https://docs.aws.amazon.com/serverlessrepo/latest/devguide/serverlessrepo-quick-start.html

Data Removal

Applications deployed from the repository are managed as AWS CloudFormation stacks, allowing complete removal through standard CloudFormation deletion processes. Publishers can delete their applications using the AWS Management Console or AWS CLI, which removes the application from the repository. For deployed applications, users delete the corresponding CloudFormation stack to remove all associated resources. Application artifacts stored in S3 are subject to standard S3 deletion processes.

Backup/Restore and Disaster Recovery

AWS Serverless Application Repository does not provide specific backup and disaster recovery capabilities as it primarily serves as a distribution mechanism for serverless applications. Published applications and their metadata are stored across multiple AWS regions for public applications. The underlying infrastructure relies on AWS's standard durability and availability guarantees. Deployed applications inherit backup capabilities from their constituent services like Lambda, DynamoDB, and S3.

Backup Capabilities

The service itself does not offer traditional backup capabilities. Published applications are replicated across regions for public availability:

- Code packages are stored in S3 with standard S3 durability
- The repository maintains application metadata and versions, but does not integrate with AWS Backup service
- Backup responsibility lies with the individual AWS services used by deployed applications

Disaster Recovery

AWS Serverless Application Repository does not directly support AWS Elastic Disaster Recovery integration. The service provides regional availability for public applications, automatically copying deployment artifacts to destination regions when applications are deployed:

- Disaster recovery depends on the underlying services used by deployed applications
- The repository itself benefits from AWS's infrastructure resilience across multiple regions

Recovery Objectives

The service contributes to recovery objectives primarily through its regional replication of public applications and rapid deployment capabilities. Applications can be quickly redeployed in different regions to support disaster recovery scenarios. However, specific RPO and RTO objectives depend on the architecture of individual serverless applications deployed from the repository rather than the repository service itself.

Service Constraints

Service Quotas:
https://docs.aws.amazon.com/serverlessrepo/latest/devguide/quotas.html

FAQ: https://aws.amazon.com/serverless/serverlessrepo/faqs/

Technical Requirements

Service Documentation:
https://docs.aws.amazon.com/serverlessrepo/latest/devguide/what-is-serverlessrepo.html

User Guide: https://docs.aws.amazon.com/serverlessrepo/latest/devguide/what-is-serverlessrepo.html

API Reference:
https://docs.aws.amazon.com/serverlessrepo/latest/devguide/resources.html

Pricing Overview

Please see the AWS UK G Cloud 15 Pricing Document accompanying this service in the Digital Marketplace.

Public Pricing: https://aws.amazon.com/serverless/serverlessrepo/faqs/

Cost Optimization

AWS Serverless Application Repository incurs no direct charges for browsing, deploying, or publishing applications. Cost optimization focuses on the underlying AWS resources used by deployed applications:

- The service provides 5GB of free S3 storage per region for code packages
- Cost efficiency comes from reusing proven serverless patterns rather than building from scratch, reducing development time and resource consumption
- Publishers can optimize costs by sharing applications to avoid duplication across teams
- The pay-per-use nature of underlying serverless services like Lambda provides automatic cost optimization based on actual usage

Savings Considerations

Primary cost savings derive from accelerated development cycles through application reuse and proven patterns. Teams avoid duplicating development effort by leveraging existing applications from the repository:

- The serverless architecture of deployed applications provides cost efficiency through pay-per-execution billing models
- Organizations can reduce development costs by sharing applications internally, eliminating redundant development across teams
- The service's integration with serverless services enables automatic scaling and cost optimization based on demand
- No infrastructure management overhead reduces operational costs compared to traditional application deployment models