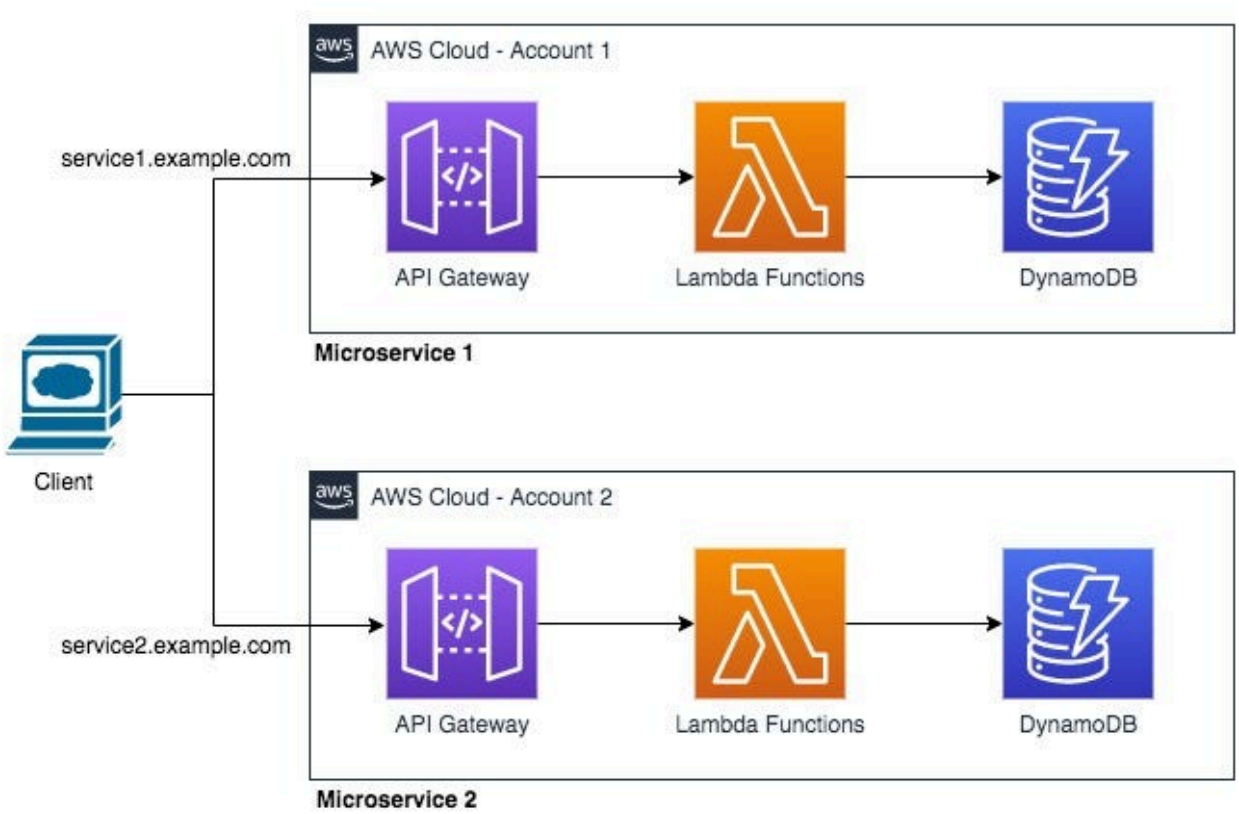
AWS
Serverless
Concepts

Concept Overview:

About Serverless:

AWS Serverless is a way to build and run applications without managing any servers. You focus on writing code for your application, and AWS handles the infrastructure, server management, capacity provisioning, and maintenance tasks.

Serveless architecture diagram



About Serverless:

Features of Serverless:

No Server Management: You don't need to provision, patch, or manage any underlying servers. AWS handles all the operational overhead.

Automatic Scaling: AWS services automatically scale your application up or down based on demand, from a few requests per day to thousands per second.

Pay-for-Use: You only pay for the compute resources your code consumes, not for idle servers.

Event-Driven: Serverless functions often run in response to events, such as an API request, a file upload to storage, or a database change.

Work Process:

Trigger: An event (such as HTTP request, file upload) activates a Lambda function.

Execution: AWS runs the function in a managed container.

Integration: Services like DynamoDB, S3, and SNS connect seamlessly.

Scale: Automatically adjusts to demand—no manual intervention.

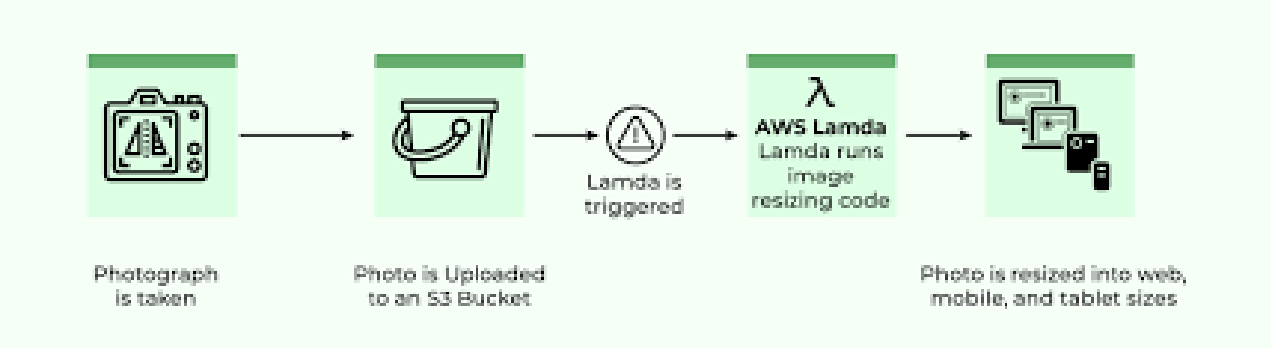
Serverless Services:

AWS Lambda, Amazon API Gateway, Amazon S3, Amazon DynamoDB, Amazon EventBridge, Amazon SNS/SQS etc.

AWS Lambda:

AWS Lambda is a serverless compute service provided by Amazon Web Services (AWS) that allows users to run code without provisioning or managing servers. It is an event-driven service, meaning it executes code in response to specific events or triggers.

Lambda Architecture



Work process:

- Lambda Functions: Code is organized into functions, the basic building blocks of a Lambda application.
- Permissions & Security: Execution roles and resource policies control access to AWS services and who can invoke functions.
- Event Sources & Triggers: AWS services or custom event sources trigger functions, passing event data in JSON format (via event source mappings).
- Execution Environment: Lambda runs code inside language-specific runtimes (like Node.js, Python), along with layers and extensions.

AWS Lambda:

Features of Lambda:

- Event-Driven Execution: Runs code automatically in response to events (API calls, file uploads, DB updates).
- No Server Management: Fully managed service; no need to provision or manage servers.
- Auto Scaling: Scales automatically with the number of incoming requests.
- Pay-as-You-Go: Charged only for the compute time used, not idle time.
- Multiple Language Support: Supports Python, Node.js, Java, Go, .NET, Ruby, etc.
- Integrations with AWS Services: Natively integrates with S3, DynamoDB, API Gateway, EventBridge, and more.
- High Availability: Built-in fault tolerance across multiple Availability Zones.
- Environment Variables: Store configuration and secrets securely for Lambda functions.
- Concurrency Controls: Limit or reserve concurrent executions for better performance management.

Amazon API Gateway:

Amazon API Gateway is an AWS service for creating, publishing, maintaining, monitoring, and securing REST, HTTP, and WebSocket APIs at any scale.

API developers can create APIs that access AWS or other web services, as well as data stored in the AWS cloud.

As an API Gateway API developer, you can create APIs for use in your own client applications. Or you can make your APIs available to third-party app developers.

Architecture of API Gateway

Amazon API Gateway:

Features of API Gateway:

- API Types: Supports stateful (WebSocket) and stateless (HTTP and REST) APIs.
- Authentication & Authorization: Offers IAM policies, Lambda authorizers, and Amazon Cognito user pools.
- Deployment Control: Provides canary release deployments for safe rollouts.
- Logging & Monitoring: Supports AWS CloudTrail for API changes and usage tracking.
- CloudWatch Integration: Provides access and execution logging with alarms and metrics for REST and WebSocket APIs.
- Infrastructure as Code: Enables API creation through AWS CloudFormation templates.
- Custom Domains: Allows mapping APIs to custom domain names.
- Security Protection: Integrates with AWS WAF to guard against common web exploits.
- Performance Insights: Integrates with AWS X-Ray for latency analysis and debugging.

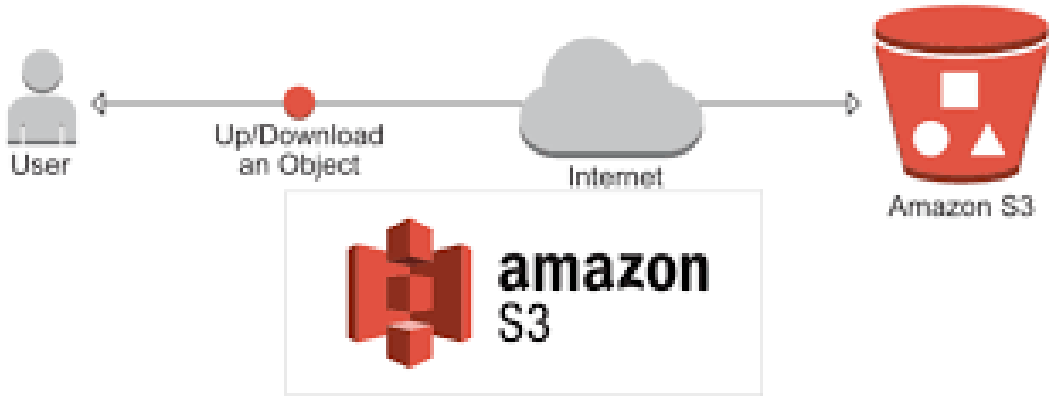
Amazon S3:

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance.

Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics.

Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

Simple Diagram of S3



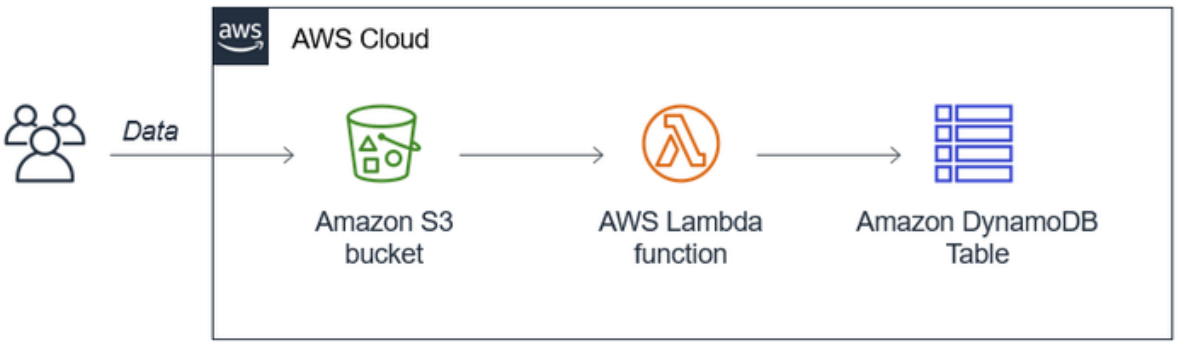
Amazon DynamoDB:

Amazon DynamoDB is a serverless, NoSQL, fully managed database with single-digit millisecond performance at any scale.

DynamoDB addresses your needs to overcome scaling and operational complexities of relational databases. DynamoDB is purpose-built and optimized for operational workloads that require consistent performance at any scale.

Customers across all sizes, industries, and geographies use DynamoDB to build modern, serverless applications that can start small and scale globally. DynamoDB scales to support tables of virtually any size while providing consistent single-digit millisecond performance and high availability.

Simple Illustration of DynamoDB



Amazon DynamoDB:

Features of DynamoDB:

- Serverless: No server provisioning or maintenance required, zero downtime, no patching, and no maintenance windows.
- On-Demand Capacity Mode: Pay-as-you-go pricing for read/write requests with instant auto-scaling, scales down to zero with no cold starts.
- NoSQL Database: Supports key-value and document data models, designed for performance, scalability, and flexibility.
- Data Modeling: Encourages denormalization (no JOINs) for efficiency, supports ACID transactions and strong consistency.
- Fully Managed: Handles setup, provisioning, security, backups, monitoring, and high availability automatically.
- High Performance: Delivers consistent single-digit millisecond latency at any scale, from hundreds to millions of users.

Amazon DynamoDB:

Use case of DynamoDB:

- Financial Services Applications: Supports live trading, loan management, transaction ledgers, and token generation with global tables for low-latency, multi-Region access.

- High Availability Workloads: Removes the need for manual scaling, patching, or licensing, suitable for mission-critical applications with strict uptime needs.

- ACID Transactions: Provides atomicity, consistency, isolation, and durability for financial transactions, order processing, and other enterprise workloads.

- Gaming Applications: Stores game state, player data, session history, and leaderboards, serverless scalability ensures performance during peak traffic and cost-efficiency during low usage.

- Streaming Applications: Powers metadata indexing, content management, watchlists, bookmarks, real-time sports stats, and recommendation engines with high scalability and resiliency.

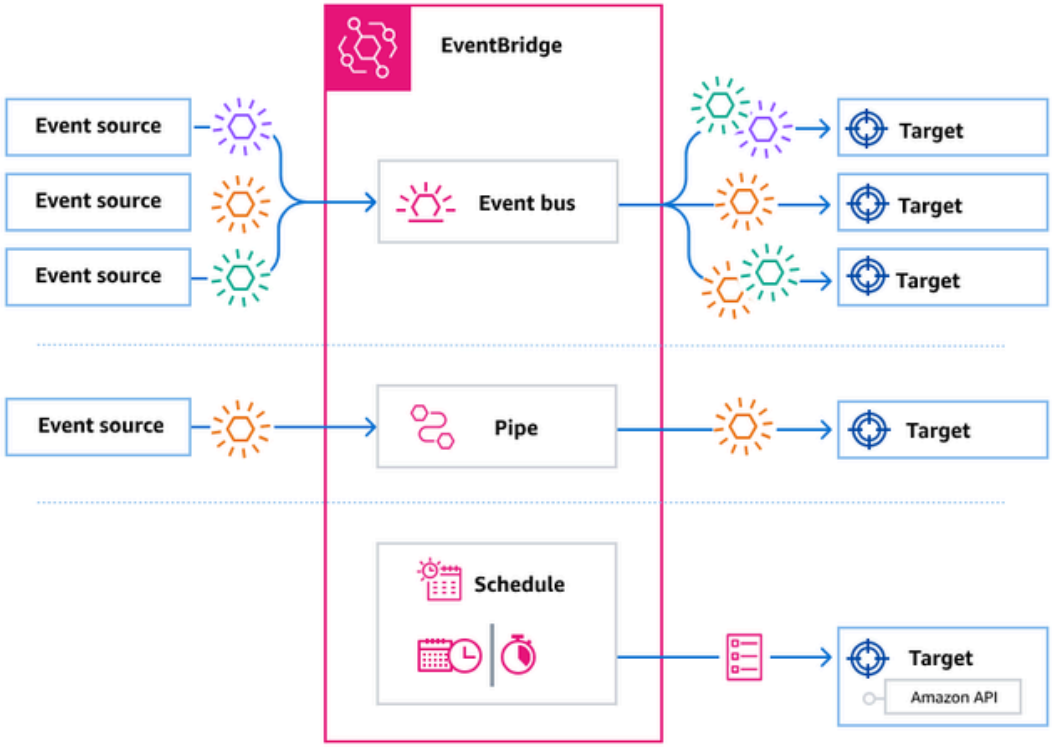
Amazon EventBridge:

EventBridge is a serverless service that uses events to connect application components together, making it easier for you to build scalable event-driven applications. Event-driven architecture is a style of building loosely-coupled software systems that work together by emitting and responding to events. Event-driven architecture can help you boost agility and build reliable, scalable applications.

EventBridge provides simple and consistent ways to ingest, filter, transform, and deliver events so you can build applications quickly.

You can create it using current distribution or previous distribution too.

EventBridge Architecture



Amazon EventBridge:

Event Processing Models:

- Event Buses: Act as routers to receive events from AWS services, custom apps, or third-party software and deliver them to one or more targets.
- Pipes: Enable point-to-point integrations between a single source and a single target, with support for advanced event transformations and enrichment.

Event Buses Use Case:

- Best for routing events from many sources to many targets.
- Supports optional event transformation before delivery.

Pipes Use Case:

- Ideal for direct, point-to-point connections.
- Adds filtering, transformation, and enrichment before sending events to the target.

Combined Usage:

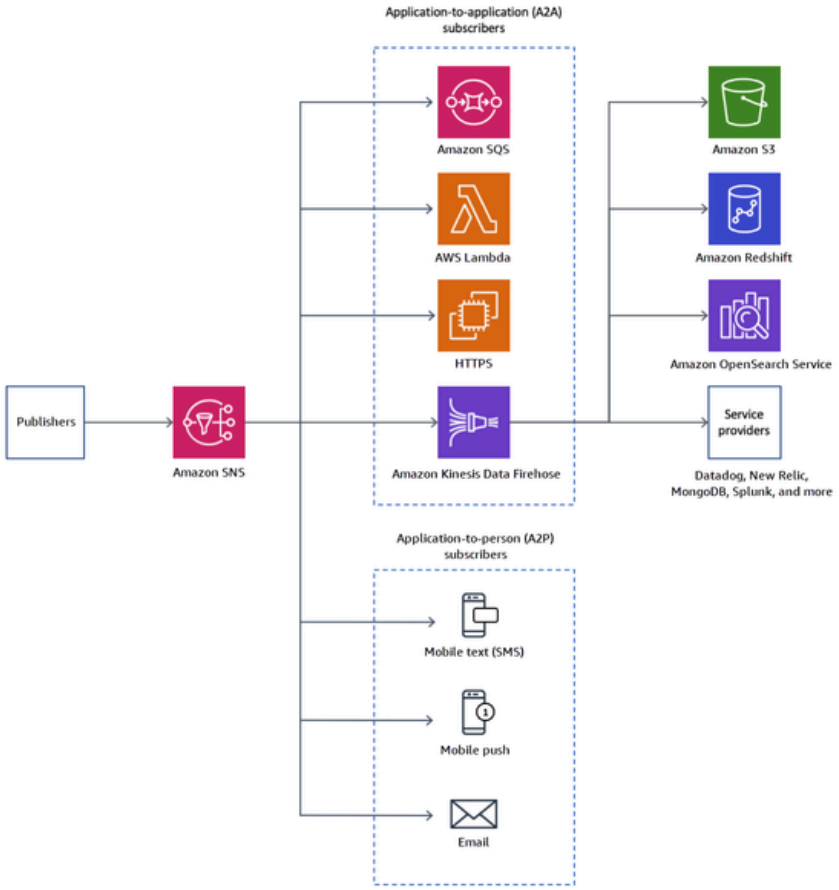
- Pipes and event buses are often used together.
- Example: A pipe connects a DynamoDB stream (source) to an event bus (target). The event bus then routes those events to multiple targets as defined by rules.

Amazon SNS/SQS:

SNS(Simple Notification Service):

Amazon Simple Notification Service (Amazon SNS) is a fully managed service that provides message delivery from publishers (producers) to subscribers (consumers). Publishers communicate asynchronously with subscribers by sending messages to a topic, which is a logical access point and communication channel.

SNS Architecture



Amazon SNS/SQS:

SNS Work Process:

In SNS, publishers send messages to a topic, which acts as a communication channel. The topic acts as a logical access point, ensuring messages are delivered to multiple subscribers across different platforms. Subscribers to an SNS topic can receive messages through different endpoints, depending on their use case, such as:

- Amazon SQS
- Lambda
- HTTP(S) endpoints
- Email
- Mobile push notifications
- Mobile text messages (SMS)
- Amazon Data Firehose
- Service providers (For example, Datadog, MongoDB, Splunk)

SNS supports both Application-to-Application (A2A) and Application-to-Person (A2P) messaging, giving flexibility to send messages between different applications or directly to mobile phones, email addresses, and more.

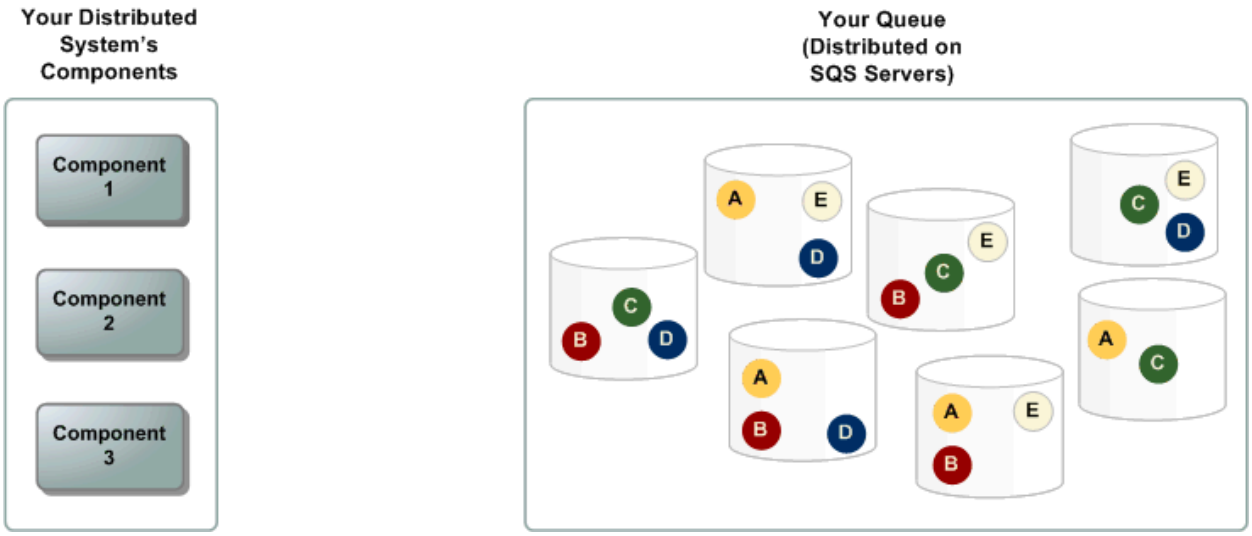
Amazon SNS/SQS:

SQS(Simple Queue Service):

Amazon Simple Queue Service (Amazon SQS) offers a secure, durable, and available hosted queue that lets you integrate and decouple distributed software systems and components.

Amazon SQS offers common constructs such as dead-letter queues and cost allocation tags. It provides a generic web services API that you can access using any programming language that the AWS SDK supports.

SQS Architecture



Amazon SNS/SQS:

Advantages of using SQS:

- Security: Control who can send and receive messages; protect sensitive data using Amazon SQS managed server-side encryption (SSE) or custom SSE keys via AWS KMS.

- Durability: Messages are stored on multiple servers; Standard queues provide at-least-once delivery, FIFO queues provide exactly-once processing and high throughput.

- Availability: Redundant infrastructure ensures high availability and concurrent access for producing and consuming messages.

- Scalability: Automatically scales to handle any load or traffic spikes without manual provisioning.

- Reliability: Locks messages during processing to allow multiple producers and consumers to interact with the queue safely.

- Customization: Configure queues individually, set default delays, store messages larger than 1 MiB using S3 or DynamoDB, or split large messages into smaller ones.

.

Thank You

Stay Connect:

/in/alamgirweb11

/alamgirweb11