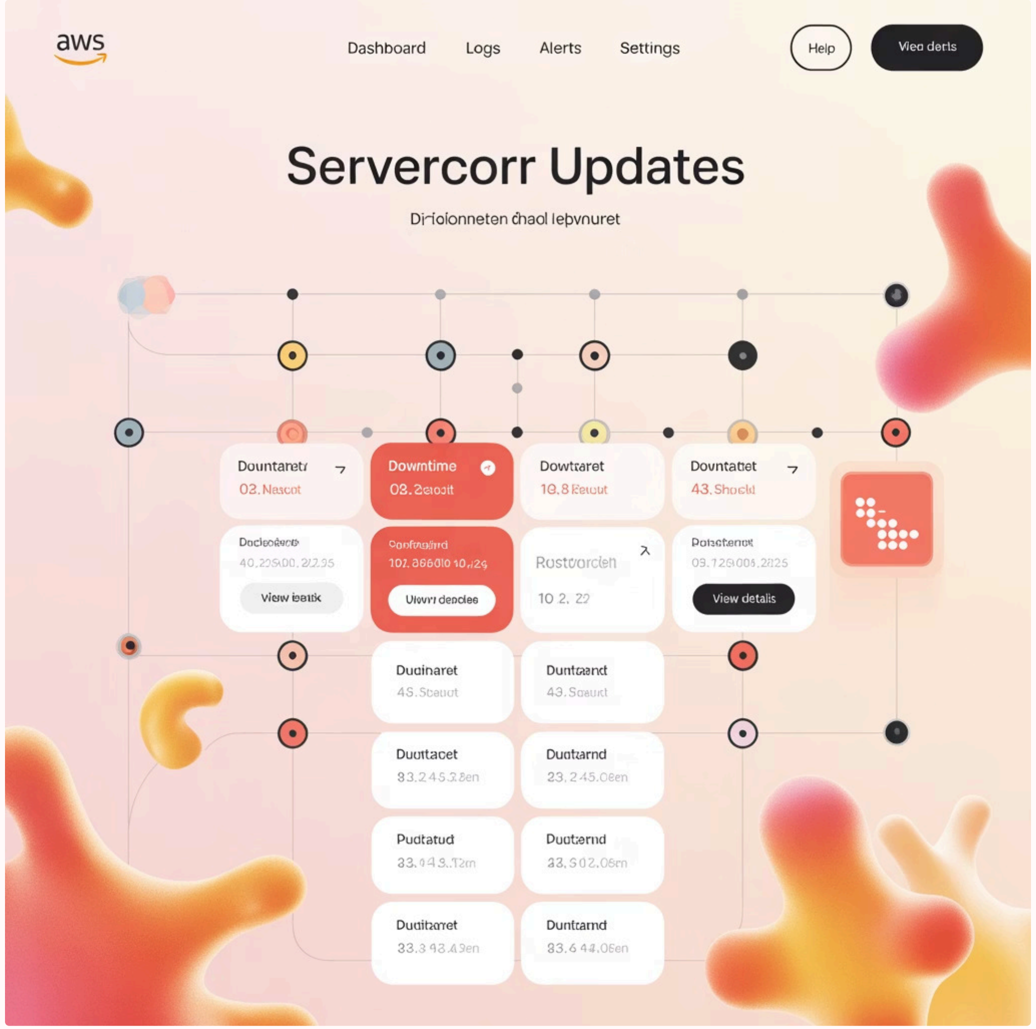
# AWS Cloud Deployment Methods: Mastering the Art of Cloud Operations

The AWS Cloud offers unprecedented flexibility in how organizations deploy and operate their applications. With multiple strategic approaches available, companies can balance speed, safety, and scalability while minimizing downtime. This presentation explores the various deployment methods in AWS and provides guidance on selecting the right approach for your specific needs.

# In-Place (All-at-Once) Deployment

The in-place deployment method updates all server instances simultaneously with new code. While this approach offers the fastest deployment time, it comes with a significant drawback: service downtime during the update process.

- Most straightforward deployment approach
- All instances receive updates simultaneously
- No additional infrastructure required
- Rollback requires redeploying previous code version



**Best for:** Non-critical applications where downtime is acceptable, development environments, or systems with maintenance windows.

# Rolling Deployment: Balancing Speed and Availability

### Batch 1 Deployment

First subset of servers receives the new code while others continue running the old version.

### Verification

Confirm successful deployment and proper functioning before proceeding.

### Continue Batches

Systematically update remaining server groups until the entire fleet runs the new version.

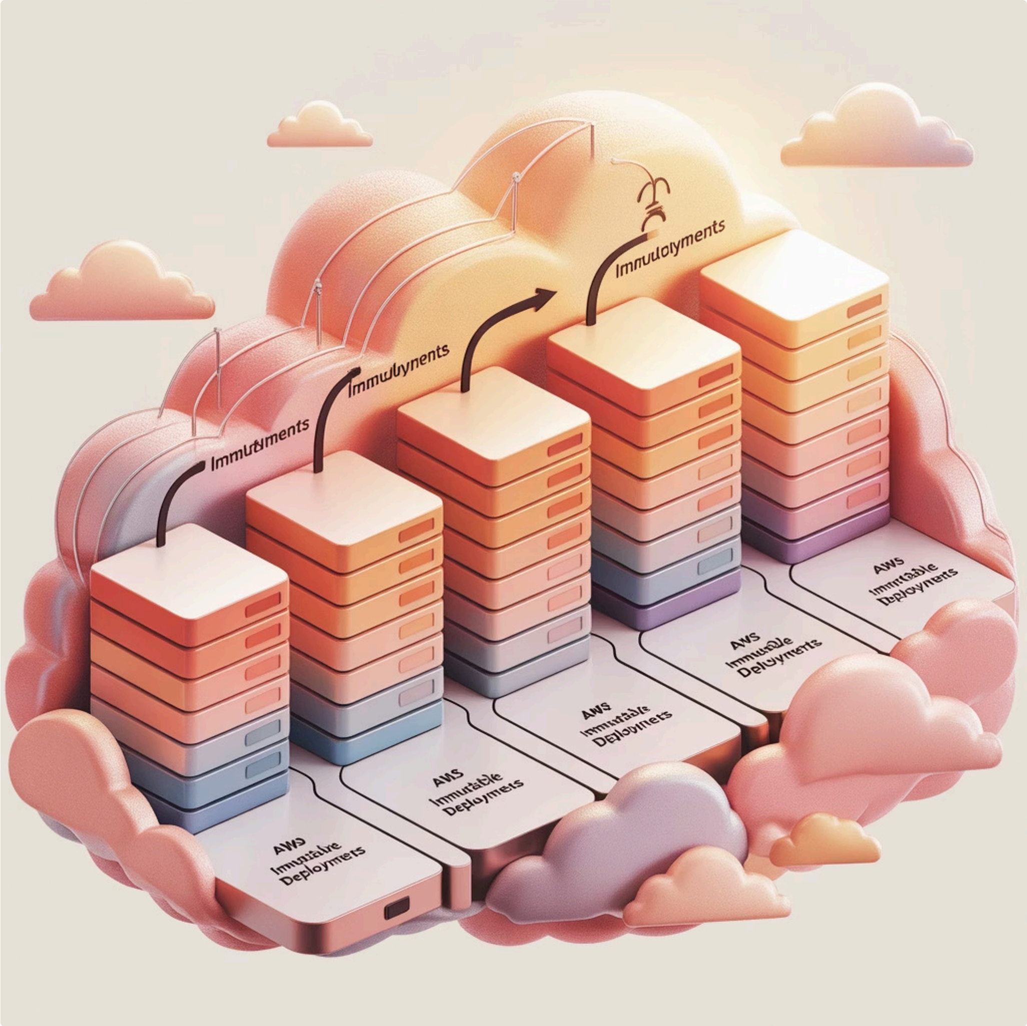
This strategy enables zero-downtime deployments since some servers remain operational throughout the process. If a failure occurs, only the affected batch requires fixing, minimizing the impact on overall system availability.

**Canary variant:** Initially deploys to a small subset (e.g., 5-10% of servers) to validate performance before proceeding with the full rolling deployment.

# Immutable Deployment: Maximum Safety

Immutable deployments create entirely new server instances with the updated code while keeping the old environment intact until the switch is made.

- Launch new infrastructure with the updated application
- Test thoroughly in isolation
- Shift traffic to new environment once validated
- Decommission old instances after successful transition


Immutyments
Immulbynents
Immuulolyments
AWS Immutable Deployments
AWS Immutable Deployments
AWS Immutable Deployments
AWS Immutable Deployments
AWS Immutable Deployments

This approach offers zero downtime and simple rollback by redirecting traffic back to the original environment if issues arise. The tradeoff is higher infrastructure usage during the transition period.

# Blue/Green Deployment: Parallel Environments

### Blue Environment (Current)

The existing production environment continues serving traffic while the new version is prepared.

### Green Environment (New)

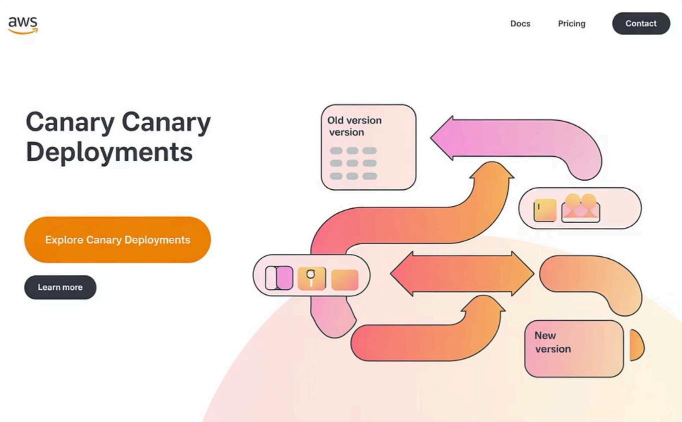
An identical infrastructure stack is provisioned with the new application version and thoroughly tested.

### Traffic Switch

Once verified, all user traffic is redirected from blue to green environment, typically through DNS or load balancer changes.

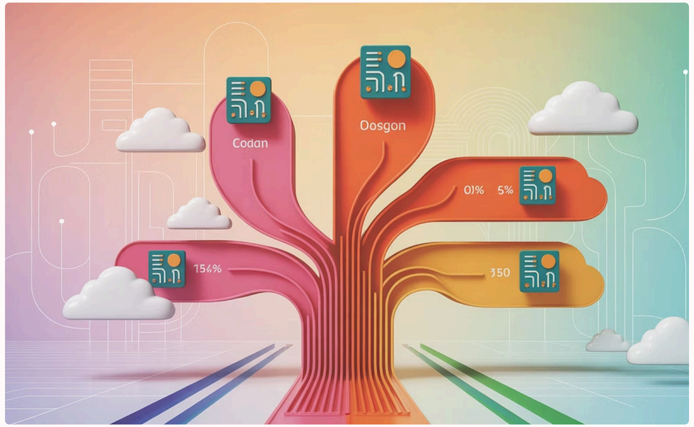
This method minimizes deployment risk with zero downtime and enables immediate rollback by switching traffic back to the blue environment if issues arise. The primary consideration is maintaining two complete environments simultaneously, which increases resource costs.

# Additional AWS Deployment Strategies



## Canary Deployment

Routes a small percentage of traffic to the new version first, allowing for real-world testing with limited risk. Gradually increase traffic as confidence grows.



## Traffic Splitting

Directs a specific percentage of requests to the new version while maintaining the rest on the current version, enabling controlled exposure.

## Linear Deployment

Shifts traffic in equal increments over a predefined timeframe, providing a predictable, gradual transition from old to new versions.

# Deployment Methods Comparison

| Method | Downtime | Rollback | Traffic Shift | Infrastructure Use |
|---|---|---|---|---|
| In-Place | Yes | Redeploy | No | Low |
| Rolling | No | Partial | No | Low |
| Immutable | No | Easy | Yes | Temporary increase |
| Blue/Green | No | Easy | Yes | Two full environments |
| Canary | No | Easy | Partial | Slight increase |

Each deployment method offers different tradeoffs between risk, speed, and resource consumption. Your application requirements and business priorities should guide your selection of the most appropriate strategy.

# Choosing the Right AWS Deployment Method

1 Application Criticality Assessment

Evaluate your application's availability requirements, acceptable downtime, and risk tolerance. Mission-critical applications warrant safer deployment methods despite higher resource costs.

2 Resource Considerations

Assess your infrastructure budget and capacity. Blue/green deployments require maintaining two complete environments, while in-place updates minimize resource usage but increase risk.

3 Leverage AWS Tools

Utilize AWS CodeDeploy, Elastic Beanstalk, and CloudFormation to automate your chosen deployment strategy. These services provide built-in support for various deployment patterns with minimal configuration.

Online Training Course

Free Practice Test