The 7Rs Strategy

AWS Cloud Migration

Align your migration with AWS best practices.

Why Follow AWS's 7Rs?

- Minimize downtime and costs
- Prioritize scalability and compliance
- Match strategies to workload needs

1 Retire

Decommission unused apps.

- Zombie apps (<5% CPU/memory usage)
- Apps with no inbound traffic for 90+ days
- Legacy systems with unsupported OS

Best for: Cutting costs and security risks.

2 Retain

Keep apps on-prem for now.

- Data residency/compliance needs
- Apps with unresolved hardware dependencies
- Recently upgraded systems

Pro Tip: Revisit during future migration waves.

3 Rehost

Migrate apps “as-is” to AWS (e.g. EC2).

- **Tools**: AWS Application Migration Service, VM Import/Export.
- **Best for**: Large-scale migrations with minimal downtime.

AWS Tip: Optimize apps after migration.

4 Relocate

Migrate server fleets from on-prem platforms to cloud environment.

Examples:

- Bulk migration of VMware, physical, or non-x86 workloads to AWS.

- Moving RDS instances or EC2 fleets to new regions/accounts.

Why? Zero architectural changes + rapid migration.

5 Repurchase

Replace with SaaS/cloud-native tools (e.g., QuickSight).

- Migrating from legacy CRM to Salesforce
- Avoiding custom app redevelopment

AWS Advice: Partner with vendors for smooth data migration.

6 Replatform

Optimize apps for AWS with minor tweaks (not full modernization).

Use Cases:

- Migrate databases to Amazon RDS
- Switch to AWS Graviton Processors
- Move Windows→Linux (Porting Assistant for .NET)
- Upgrade outdated OS (AWS EMP for Windows)

7 Refactor

Rebuild cloud-native (e.g., serverless, microservices).

Use Cases: Legacy mainframes, monolithic apps.

AWS Note: Avoid Refactor for large migrations; modernize later!

Key Takeaway

- Start with Retire/Retain to simplify your portfolio.
- Use Rehost/Relocate for speed in large migrations.
- Replatform for quick wins; Refactor for innovation.

Ready to Migrate?

Follow for more engineering tips.