Acme Manufacturing Cloud Migration: Automation Scripts

This document lists the automation scripts used in the Acme Manufacturing cloud migration project. These scripts automate various tasks to improve efficiency, reduce manual effort, and ensure consistency. All scripts should be version-controlled and thoroughly tested before deployment. This document serves as a catalog and brief description; the scripts themselves reside in their respective folders within this repository.

I. Script Organization:

Automation scripts are organized into subdirectories based on their function and the phase of the migration they support:

- **assessment:** Scripts related to the assessment and planning phase.
- **poc:** Scripts used during the Proof of Concept phase.
- **pilot-migration:** Scripts used during the Pilot Migration phase.
- **full-migration:** Scripts used during the Full Migration phase.
- **post-migration:** Scripts used for post-migration activities (optimization, automation).

II. Script Descriptions:

This section provides a brief description of each script's purpose and functionality. The scripts are written primarily in Bash or Python or unless otherwise noted.

A. Assessment Phase:

- inventory_script.sh (Bash): This script automates the discovery and inventory of on-premises servers and applications. It gathers information such as operating system, software versions, CPU usage, memory usage, and network configuration. The output is a structured CSV file.

B. Proof of Concept (POC) Phase:

- poc_deploy.py (Python): Automates the deployment of the POC application to an EC2 instance, including the creation of the instance, configuration of security groups, and installation of necessary dependencies. It uses boto3 to interact with the AWS API.
- poc_test.sh (Bash): A script to run automated functional tests on the POC application after deployment. It uses a testing framework (specify framework used if applicable).

C. Pilot Migration Phase:

- pilot_migration_db.sql (SQL): SQL scripts to automate the migration of the database schema and data for the pilot applications to an RDS instance. Specific SQL commands will depend on the database technology used (e.g., MySQL, PostgreSQL).
- pilot_migrate_app.py (Python): This script automates the deployment of pilot applications to their respective AWS environments (EC2, ECS, or Lambda). It uses the appropriate AWS SDKs and handles dependency management.

D. Full Migration Phase:

- full_migration_app.py (Python): This script extends the functionality of pilot_migrate_app.py to handle the migration of all remaining applications based on the prioritized sequence.
- full_migration_db.sql (SQL): Similar to pilot_migration_db.sql, this script automates database migration for all remaining applications, appropriately handling dependencies and data transformations.

E. Post-Migration Phase:

- cost_optimization.py (Python): This script analyzes AWS costs and identifies opportunities for cost optimization (e.g., Reserved Instances, Savings Plans). It uses the AWS Cost Explorer API and outputs a report.
- monitoring_setup.sh (Bash): This script automates the setup of CloudWatch alarms and dashboards for ongoing monitoring of the migrated applications and infrastructure.

III. Script Usage and Best Practices:

- **Version Control:** All scripts are managed using Git for version control.
- **Testing:** Each script should be thoroughly tested in a non-production environment before deployment.
- **Error Handling:** Robust error handling should be implemented in all scripts.
- **Logging:** All scripts should include detailed logging to facilitate troubleshooting.
- **Documentation:** Each script should be well-documented, including a clear description of its purpose, parameters, and expected output.
- **Security:** Scripts should be reviewed for security vulnerabilities and adhere to secure coding practices. Avoid hardcoding sensitive information (access keys, passwords) directly into the scripts. Use AWS IAM roles and environment variables instead.

This document provides a high-level overview. Detailed documentation for each script is included within its respective file. Regular updates and revisions to these scripts will be made as the migration progresses.