**DevOps Learning Project - Deployment Scripts**

This directory contains automated deployment and management scripts for the DevOps learning web application.

**Scripts Overview**

**🚀 deploy.sh**

**Main deployment script** - Automates the entire deployment process.

**Usage:**

./deploy.sh

**What it does:**

* Finds your EC2 instance automatically
* Tests SSH connectivity
* Creates backup of current deployment
* Deploys new version
* Verifies deployment success
* Shows deployment information

**Prerequisites:**

* AWS CLI configured
* EC2 instance running with tag Name=my-webapp-server
* SSH key file my-devops-key.pem in scripts directory
* Security group allowing SSH (port 22) and HTTP (port 80)

**📊 server-status.sh**

**Server monitoring script** - Checks health and status of your web server.

**Usage:**

./server-status.sh

**What it shows:**

* EC2 instance information
* System status (uptime, memory, disk usage)
* Apache web server status
* Web content verification
* External accessibility test
* Useful management commands

**⏪ rollback.sh**

**Deployment rollback script** - Restore previous version in case of issues.

**Usage:**

# Interactive mode (recommended)

./rollback.sh

# Direct rollback to specific backup

./rollback.sh index.html.20241201\_143022

**Features:**

* Lists all available backups
* Interactive backup selection
* Automatic backup before rollback
* Verification of rollback success

**Quick Start Guide**

**1. Initial Setup**

# Make scripts executable

chmod +x \*.sh

# Test AWS connectivity

aws sts get-caller-identity

**2. Deploy Your Application**

./deploy.sh

**3. Monitor Your Server**

./server-status.sh

**4. Rollback if Needed**

./rollback.sh

**File Structure**

scripts/

├── deploy.sh # Main deployment automation

├── server-status.sh # Server monitoring

├── rollback.sh # Deployment rollback

├── my-devops-key.pem # SSH private key

└── README.md # This documentation

**Configuration**

All scripts use these default configurations:

* **Instance Tag:** my-webapp-server
* **SSH User:** ec2-user
* **SSH Key:** my-devops-key.pem
* **Local HTML:** ../src/index.html
* **Remote Path:** /var/www/html/
* **Backup Path:** /var/www/html/backups/

**Troubleshooting**

**Common Issues**

**1. "AWS CLI not configured"**

aws configure

**2. "Key file not found"**

* Ensure my-devops-key.pem is in the scripts directory
* Check file permissions: chmod 400 my-devops-key.pem

**3. "Cannot connect to instance via SSH"**

* Check instance is running: aws ec2 describe-instances --filters "Name=tag:Name,Values=my-webapp-server"
* Verify security group allows SSH on port 22
* Confirm you're using the correct region

**4. "HTTP test failed"**

* Check Apache status: ./server-status.sh
* Verify security group allows HTTP on port 80
* Restart Apache: ssh -i my-devops-key.pem ec2-user@INSTANCE\_IP 'sudo systemctl restart httpd'

**Useful Commands**

# Check AWS configuration

aws configure list

# Find your instance

aws ec2 describe-instances --filters "Name=tag:Name,Values=my-webapp-server"

# Connect to server manually

ssh -i my-devops-key.pem ec2-user@INSTANCE\_IP

# View Apache logs

ssh -i my-devops-key.pem ec2-user@INSTANCE\_IP 'sudo tail -f /var/log/httpd/access\_log'

**Security Notes**

* Keep your SSH key file (my-devops-key.pem) secure and never commit it to git
* The key file should have restrictive permissions (400)
* Scripts include connection timeouts and error handling
* Backups are created automatically before deployments

**Next Steps**

After mastering these scripts, consider:

* Setting up CI/CD pipelines (GitHub Actions, Jenkins)
* Implementing infrastructure as code (Terraform, CloudFormation)
* Adding monitoring and alerting (CloudWatch, Grafana)
* Using container orchestration (Docker, Kubernetes)

**Happy DevOps Learning! 🚀**