**📊 Monitoring and Logging Guide**

Complete monitoring and observability setup for the DevOps Learning Project.

**🎛️ Monitoring Stack**

**Components**

* **AWS CloudWatch** - Metrics collection and alerting
* **CloudWatch Agent** - Custom metrics and log collection
* **SNS** - Email and SMS alerts
* **Custom Health Checks** - Application-specific monitoring
* **Real-time Dashboard** - Live monitoring interface

**🚀 Quick Start**

**1. Set Up Monitoring Infrastructure**

cd scripts

./setup-monitoring.sh

**2. Test Your Monitoring**

./test-alerts.sh

**3. Launch Real-time Dashboard**

./monitor-realtime.sh

**4. Analyze Logs**

./analyze-logs.sh

**📈 Available Scripts**

**setup-monitoring.sh**

**Complete monitoring infrastructure setup**

Creates:

* SNS topic for alerts
* CloudWatch alarms (CPU, Status Check)
* CloudWatch agent installation
* Automated health checks
* Custom metrics
* Monitoring dashboard

**Usage:**

./setup-monitoring.sh

**monitor-realtime.sh**

**Live monitoring dashboard**

Features:

* Real-time system metrics
* Web server status
* Traffic monitoring
* Error tracking
* Interactive controls

**Usage:**

./monitor-realtime.sh

./monitor-realtime.sh --interval 10 # Custom refresh rate

**Controls:**

* R - Refresh now
* Q - Quit
* L - View logs
* H - Run health check

**analyze-logs.sh**

**Comprehensive log analysis**

Analyzes:

* Apache access logs
* Apache error logs
* Application health logs
* Performance metrics
* Security patterns

**Usage:**

./analyze-logs.sh # Interactive menu

./analyze-logs.sh access # Access log analysis

./analyze-logs.sh error # Error log analysis

./analyze-logs.sh full # Complete report

./analyze-logs.sh export # Export logs locally

**test-alerts.sh**

**Monitoring system testing**

Tests:

* CloudWatch agent status
* Health check functionality
* Log collection
* Metrics reporting
* SNS alerts
* CloudWatch alarms

**Usage:**

./test-alerts.sh

./test-alerts.sh --load-test # Include load testing

**📊 Monitoring Dashboards**

**CloudWatch Dashboard**

Access your custom dashboard:

https://ap-southeast-1.console.aws.amazon.com/cloudwatch/home?region=ap-southeast-1#dashboards:name=DevOps-WebApp-Monitoring

**Widgets:**

* EC2 CPU Utilization
* Application Health Status
* Recent Access Logs
* Custom Metrics

**Real-time Terminal Dashboard**

Launch with: ./monitor-realtime.sh

**Displays:**

* System status (CPU, memory, disk)
* Web server status
* Traffic and errors
* Recent activity

**🚨 Alerts and Alarms**

**Configured Alarms**

**High CPU Usage**

* **Threshold:** >80% for 10 minutes
* **Action:** SNS notification
* **Recovery:** Automatic when CPU drops

**Instance Status Check Failed**

* **Threshold:** Status check failure
* **Action:** SNS notification
* **Use:** Detects instance-level issues

**Custom Health Check**

* **Frequency:** Every 5 minutes
* **Metrics:** Custom application health
* **Logs:** /var/log/webapp-health.log

**Setting Up Email Alerts**

1. **Run setup script:**
2. ./setup-monitoring.sh
3. **Enter your email when prompted**
4. **Check email and confirm subscription**
5. **Test alerts:**
6. ./test-alerts.sh

**📝 Logs**

**Log Locations**

**On EC2 Instance:**

/var/log/httpd/access\_log # Apache access logs

/var/log/httpd/error\_log # Apache error logs

/var/log/webapp-health.log # Health check logs

**In CloudWatch:**

* devops-webapp-access-logs - Access log group
* devops-webapp-error-logs - Error log group

**Log Analysis Examples**

**View real-time access logs:**

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

**Analyze traffic patterns:**

./analyze-logs.sh access

**Export logs for analysis:**

./analyze-logs.sh export

**📈 Metrics**

**Standard EC2 Metrics**

* CPU Utilization
* Network In/Out
* Disk Read/Write
* Status Check Failed

**Custom Application Metrics**

* Health Check Status
* Response Time
* Application Errors
* Custom Business Metrics

**Viewing Metrics**

**CloudWatch Console:**

https://ap-southeast-1.console.aws.amazon.com/cloudwatch/home?region=ap-southeast-1#metricsV2:

**Command Line:**

# View health check metrics

aws cloudwatch get-metric-statistics \

--namespace "DevOps/WebApp" \

--metric-name "HealthCheck" \

--start-time $(date -u -d '1 hour ago' +%Y-%m-%dT%H:%M:%S) \

--end-time $(date -u +%Y-%m-%dT%H:%M:%S) \

--period 300 \

--statistics Sum

**🔧 Troubleshooting**

**Common Issues**

**CloudWatch Agent Not Running**

# Check status

ssh -i my-devops-key.pem ec2-user@YOUR\_IP 'sudo systemctl status amazon-cloudwatch-agent'

# Restart agent

ssh -i my-devops-key.pem ec2-user@YOUR\_IP 'sudo systemctl restart amazon-cloudwatch-agent'

**No Metrics in CloudWatch**

1. Check agent configuration
2. Verify IAM permissions
3. Check agent logs:
4. ssh -i my-devops-key.pem ec2-user@YOUR\_IP 'sudo tail -f /opt/aws/amazon-cloudwatch-agent/logs/amazon-cloudwatch-agent.log'

**Health Checks Not Running**

# Check cron job

ssh -i my-devops-key.pem ec2-user@YOUR\_IP 'crontab -l'

# Run manual health check

ssh -i my-devops-key.pem ec2-user@YOUR\_IP '/usr/local/bin/health-check.sh'

**No Email Alerts**

1. Check SNS subscription confirmation
2. Verify email address
3. Check spam folder
4. Test with:
5. ./test-alerts.sh

**Log Analysis Issues**

**Missing Logs**

# Check Apache is running

./server-status.sh

# Check log file permissions

ssh -i my-devops-key.pem ec2-user@YOUR\_IP 'ls -la /var/log/httpd/'

**Real-time Monitor Connection Issues**

1. Verify SSH connectivity
2. Check instance is running
3. Ensure security group allows SSH
4. Test connection:
5. ssh -i my-devops-key.pem ec2-user@YOUR\_IP 'echo "Connection OK"'

**🎯 Advanced Monitoring**

**Custom Metrics**

Add custom business metrics to your application:

# Send custom metric

aws cloudwatch put-metric-data \

--namespace "DevOps/WebApp" \

--metric-data MetricName=PageViews,Value=1,Unit=Count

**Log Queries**

Use CloudWatch Insights for advanced log analysis:

fields @timestamp, @message

| filter @message like /ERROR/

| sort @timestamp desc

| limit 20

**Automated Responses**

Set up Lambda functions to automatically respond to alerts:

1. Create Lambda function
2. Subscribe to SNS topic
3. Implement auto-remediation

**📚 Best Practices**

**Monitoring Strategy**

* ✅ Monitor at multiple levels (infrastructure, application, business)
* ✅ Set meaningful alert thresholds
* ✅ Use both proactive and reactive monitoring
* ✅ Implement health checks for critical components
* ✅ Regular review and tuning of alerts

**Log Management**

* ✅ Centralized logging
* ✅ Structured log formats
* ✅ Log rotation and retention policies
* ✅ Regular log analysis
* ✅ Security log monitoring

**Alert Fatigue Prevention**

* ✅ Tune alert thresholds carefully
* ✅ Use alert aggregation
* ✅ Implement alert escalation
* ✅ Regular review of alert effectiveness
* ✅ Clear alert documentation

**🚀 Next Steps**

1. **Enhanced Monitoring:**
   * Add business metrics
   * Implement custom dashboards
   * Set up log aggregation
2. **Automation:**
   * Auto-remediation scripts
   * Capacity planning automation
   * Predictive alerting
3. **Security Monitoring:**
   * Intrusion detection
   * Security log analysis
   * Compliance monitoring
4. **Performance Optimization:**
   * Application performance monitoring
   * Database monitoring
   * User experience tracking

**Happy Monitoring! 📊**

*Remember: Good monitoring is about actionable insights, not just data collection.*