AfricanMarket Maintenance and Operations Guide

Table of Contents

Overview

Maintenance Philosophy

The AfricanMarket platform requires regular maintenance to ensure optimal performance, security, and reliability. This guide provides systematic procedures for maintaining the production environment.

Maintenance Windows

- **Standard Maintenance**: Sundays 2:00-4:00 AM UTC
- **Emergency Maintenance**: As needed with minimal notice
- **Planned Upgrades**: Scheduled 2 weeks in advance

Maintenance Team Roles

- **Site Reliability Engineer**: Overall system health and performance
- **Database Administrator**: Database optimization and maintenance
- **Security Engineer**: Security updates and monitoring
- **DevOps Engineer**: Infrastructure and deployment management

Daily Operations

Daily Checklist (15-30 minutes)

- [ ] **System Health Check**
  ```bash
  # Check overall system health
  curl -f https://africanmarket.com/api/health?detailed=true

# Check all containers
docker-compose ps
  ```

```
# Verify critical services
systemctl status nginx postgresql redis
```

- [ ] **Resource Monitoring**
  ```bash
  # Check disk space
  df -h

# Check memory usage
free -m

# Check CPU usage
top -bn1 | grep “Cpu(s)”
```

- [ ] **Application Metrics Review**
- Response times < 2 seconds average
- Error rate < 1%
- Uptime > 99.9%
- Active user count trends

- [ ] **Backup Verification**
  ```bash
  # Check latest backup
  ls -la backups/database/ | head -5

# Verify backup completion
grep “backup completed” /var/log/africanmarket/backup.log
```

- [ ] **Security Alerts Review**
- Check Sentry for new errors
- Review failed login attempts
- Monitor unusual traffic patterns
- Check SSL certificate status

- [ ] **Business Metrics Review**

- Daily active users
- Order completion rates
- Payment processing success
- Customer support ticket volume

Daily Automated Tasks

These tasks run automatically but should be monitored:

Backup Operations

```
# Database backup (runs at 2 AM)
0 2 * * * /opt/africanmarket/scripts/backup.sh production full

# Log rotation (runs at 3 AM)
0 3 * * * /usr/sbin/logrotate /etc/logrotate.conf
```

Monitoring and Alerts

```
# Health check monitoring (every 5 minutes)
*/5 * * * * /opt/africanmarket/scripts/health-check.sh

# Performance metrics collection (every 10 minutes)
*/10 * * * * /opt/africanmarket/scripts/collect-metrics.sh
```

Daily Log Review

```
# Application error logs
tail -100 /var/log/africanmarket/error.log

# Nginx access logs for unusual patterns
tail -1000 /var/log/nginx/access.log | awk '{print $1}' | sort | uniq -c | sort -nr | h
ead -20

# Database slow query log
docker exec africanmarket_postgres psql -U postgres -c "SELECT query, mean_time, calls
FROM pg_stat_statements ORDER BY mean_time DESC LIMIT 10;"
```

Weekly Maintenance

Weekly Checklist (1-2 hours)

- [ ] **Performance Analysis**
  ```bash
  # Generate weekly performance report
  node scripts/generate-performance-report.js –week

# Analyze slow queries
docker exec africanmarket_postgres psql -U postgres -f scripts/analyze-slow-queries.sql
```

- [ ] **Security Updates**
  ```bash
  # Update system packages
  sudo apt update && sudo apt list –upgradable

# Check for security advisories
sudo apt-get install unattended-upgrades
sudo unattended-upgrade –dry-run
```

- [ ] **Database Maintenance**
  ```bash
  ```

```
# Update statistics
docker exec africanmarket_postgres psql -U postgres -c “ANALYZE;”

# Check database size growth
docker exec africanmarket_postgres psql -U postgres -c “SELECT schemaname,tablename,attname,inherited,null_frac,avg_width,n_distinct FROM pg_stats WHERE schemaname=’public’;”
```

- [ ] **Backup Testing**

```
# Test backup restoration to staging
./scripts/restore.sh backups/latest.sql.gz staging

# Verify data integrity
./scripts/verify-backup-integrity.sh
```

- [ ] **Log Analysis**

```
# Analyze error patterns
grep -c “ERROR” /var/log/africanmarket/*.log

# Check for unusual activity
./scripts/analyze-access-logs.sh
```

Weekly Performance Optimization

```bash
# Clear temporary files
sudo find /tmp -type f -atime +7 -delete

# Optimize Docker images
docker system prune -f

# Clear Redis cache if needed
redis-cli FLUSHDB

# Restart application to clear memory leaks
docker-compose restart app
```

Monthly Tasks

Monthly Checklist (3-4 hours)

- [ ] **Full System Audit**
- Review all monitoring dashboards
- Analyze monthly performance trends
- Check capacity utilization
- Review cost optimization opportunities
- [ ] **Security Audit**

```bash
```

```
    # Run security scan
    nmap -sV -sC localhost

# Check for vulnerabilities
npm audit
yarn audit

# Review access logs for suspicious activity
./scripts/security-audit.sh
```

- [ ] **Database Optimization**
  ```bash
  # Full database vacuum
  docker exec africanmarket_postgres psql -U postgres -c "VACUUM FULL;"

# Reindex database
docker exec africanmarket_postgres psql -U postgres -c "REINDEX DATABASE africanmarket_prod;"

# Update statistics
docker exec africanmarket_postgres psql -U postgres -c "ANALYZE;"
```

- [ ] **SSL Certificate Management**
  ```bash
  # Check certificate expiration
  openssl x509 -in /etc/letsencrypt/live/africanmarket.com/cert.pem -text -noout | grep "Not After"

# Test certificate renewal
sudo certbot renew --dry-run
```

- [ ] **Dependency Updates**
  ```bash
  # Check for outdated packages
  yarn outdated

# Update dependencies (after testing)
yarn upgrade-interactive

# Update Docker images
docker-compose pull
```

Monthly Backup Verification

```
# Full backup restore test
./scripts/disaster-recovery.sh staging full

# Data integrity verification
./scripts/verify-data-integrity.sh

# Performance test on restored data
./scripts/performance-test.sh staging
```

Quarterly Reviews

Quarterly Checklist (Full day)

- [ ] **Infrastructure Review**
- Capacity planning assessment
- Cost optimization analysis
- Technology stack evaluation
- Scaling requirements review
- [ ] **Security Assessment**
- Penetration testing
- Security policy review
- Access control audit
- Compliance verification
- [ ] **Disaster Recovery Testing**
  ```bash
  # Full disaster recovery drill
  ./scripts/disaster-recovery.sh production full

# RTO/RPO verification
./scripts/verify-recovery-objectives.sh
```

- [ ] **Performance Benchmarking**
  ```bash
  # Load testing
  artillery run load-test.yml

# Stress testing
artillery run stress-test.yml

# Endurance testing
artillery run endurance-test.yml
```

Performance Optimization

Database Performance

```
-- Identify slow queries
SELECT query, mean_time, calls, total_time
FROM pg_stat_statements
ORDER BY mean_time DESC
LIMIT 20;

-- Check table sizes
SELECT schemaname,tablename,
       pg_size_pretty(size) as size,
       pg_size_pretty(total_size) as total_size
FROM (
  SELECT schemaname,tablename,
         pg_relation_size(schemaname||'.'||tablename) as size,
         pg_total_relation_size(schemaname||'.'||tablename) as total_size
  FROM pg_tables
  WHERE schemaname='public'
) t
ORDER BY total_size DESC;

-- Analyze index usage
SELECT schemaname,tablename,attname,n_distinct,correlation
FROM pg_stats
WHERE schemaname='public'
ORDER BY n_distinct DESC;
```

Application Performance

```
# Monitor Node.js performance
node --inspect scripts/performance-monitor.js

# Profile memory usage
node --heap-prof app.js

# Analyze bundle size
npx webpack-bundle-analyzer .next/static/chunks/*.js
```

Server Performance

```
# Monitor I/O performance
iostat -x 1 10

# Check network performance
iftop

# Monitor process performance
htop
```

Security Maintenance

Regular Security Tasks

```
# Update security definitions
sudo apt update && sudo apt upgrade -y

# Check for rootkits
sudo rkhunter --check

# Monitor failed login attempts
sudo grep "Failed password" /var/log/auth.log | tail -20

# Check open ports
nmap -sT -O localhost
```

Security Hardening

```
# Configure firewall rules
sudo ufw status verbose

# Check SSH configuration
sudo sshd -T | grep -E "(permitrootlogin|passwordauthentication|port)"

# Review sudo access
sudo cat /etc/sudoers

# Check file permissions
find /opt/africanmarket -type f -perm /o+w -ls
```

Vulnerability Scanning

```
# Scan for vulnerabilities
npm audit --audit-level moderate

# Check Docker images
docker scan africanmarket:latest

# Web application security scan
nikto -h https://africanmarket.com
```

Database Maintenance

Regular Database Tasks

```sql
-- Update table statistics
ANALYZE;

-- Vacuum to reclaim space
VACUUM VERBOSE;

-- Check for bloated tables
SELECT schemaname, tablename,
       n_tup_ins, n_tup_upd, n_tup_del,
       n_live_tup, n_dead_tup
FROM pg_stat_user_tables
WHERE n_dead_tup > 1000
ORDER BY n_dead_tup DESC;
```

Database Optimization

```sql
-- Optimize frequently queried tables
VACUUM ANALYZE users;
VACUUM ANALYZE orders;
VACUUM ANALYZE products;

-- Check index usage
SELECT schemaname,tablename,indexname,idx_tup_read,idx_tup_fetch
FROM pg_stat_user_indexes
ORDER BY idx_tup_read DESC;

-- Identify missing indexes
SELECT schemaname,tablename,seq_scan,seq_tup_read,
       seq_tup_read/seq_scan as avg_tup_per_scan
FROM pg_stat_user_tables
WHERE seq_scan > 0
ORDER BY seq_tup_read DESC;
```

Database Monitoring

```bash
# Connection monitoring
docker exec africanmarket_postgres psql -U postgres -c "SELECT count(*), state FROM
pg_stat_activity GROUP BY state;"

# Lock monitoring
docker exec africanmarket_postgres psql -U postgres -c "SELECT mode, count(*) FROM
pg_locks GROUP BY mode ORDER BY count DESC;"

# Query monitoring
docker exec africanmarket_postgres psql -U postgres -c
"SELECT query, state, wait_event FROM pg_stat_activity WHERE state != 'idle';"
```

Scaling Operations

Vertical Scaling

```
# Monitor resource usage trends
sar -u 1 60  # CPU usage
sar -r 1 60  # Memory usage
sar -d 1 60  # Disk I/O

# Identify scaling triggers
./scripts/check-scaling-metrics.sh
```

Horizontal Scaling Preparation

```
# Load balancer configuration
sudo nginx -t

# Session storage migration to Redis
./scripts/migrate-sessions-to-redis.sh

# Database connection pooling
./scripts/configure-pgbouncer.sh
```

Auto-scaling Configuration

```
# Docker Swarm scaling
version: '3.8'
services:
  app:
    deploy:
      replicas: 3
      update_config:
        parallelism: 1
        delay: 10s
      restart_policy:
        condition: on-failure
```

Troubleshooting Guide

Common Issues and Solutions

High CPU Usage

```
# Identify CPU-intensive processes
top -p $(pgrep -f "node")

# Check for infinite loops
strace -p <pid>

# Analyze Node.js performance
node --prof app.js
```

Memory Leaks

```
# Monitor memory usage
watch -n 1 'free -m'

# Generate heap dump
kill -USR2 <node_pid>

# Analyze heap dump
node --inspect-brk=0.0.0.0:9229 app.js
```

Database Performance Issues

```
-- Check active queries
SELECT pid, now() - pg_stat_activity.query_start AS duration, query
FROM pg_stat_activity
WHERE (now() - pg_stat_activity.query_start) > interval '5 minutes';

-- Check locks
SELECT blocked_locks.pid AS blocked_pid,
       blocked_activity.usename AS blocked_user,
       blocking_locks.pid AS blocking_pid,
       blocking_activity.usename AS blocking_user,
       blocked_activity.query AS blocked_statement,
       blocking_activity.query AS current_statement_in_blocking_process
FROM pg_catalog.pg_locks blocked_locks
JOIN pg_catalog.pg_stat_activity blocked_activity ON blocked_activity.pid = blocked_loc
ks.pid
JOIN pg_catalog.pg_locks blocking_locks ON blocking_locks.locktype = blocked_locks.lock
type
JOIN pg_catalog.pg_stat_activity blocking_activity ON blocking_activity.pid = block-
ing_locks.pid
WHERE NOT blocked_locks.granted;
```

Network Issues

```
# Check network connectivity
ping -c 4 google.com

# Test DNS resolution
nslookup africanmarket.com

# Check port accessibility
netstat -tuln | grep :3000
```

Emergency Procedures

Application Crash Recovery

```
# Quick restart
docker-compose restart app

# Full recovery with logs
docker-compose down
docker-compose up -d
docker-compose logs -f app
```

Database Corruption Recovery

```
# Stop application
docker-compose stop app

# Restore from backup
./scripts/disaster-recovery.sh production database

# Verify data integrity
./scripts/verify-data-integrity.sh

# Restart application
docker-compose start app
```

Security Incident Response

```
# Isolate affected systems
sudo ufw deny from <suspicious_ip>

# Collect evidence
./scripts/collect-security-logs.sh

# Rotate credentials
./scripts/rotate-api-keys.sh

# Notify stakeholders
./scripts/send-security-alert.sh
```

Maintenance Scripts

Automated Maintenance Scripts

Located in `/opt/africanmarket/scripts/maintenance/`:

- `daily-health-check.sh` - Daily system health verification
- `weekly-optimization.sh` - Weekly performance optimization
- `monthly-security-scan.sh` - Monthly security assessment
- `quarterly-dr-test.sh` - Quarterly disaster recovery testing

Custom Monitoring Scripts

```
# Resource monitoring
#!/bin/bash
# File: /opt/africanmarket/scripts/monitor-resources.sh

# Check disk space
DISK_USAGE=$(df / | grep -vE '^Filesystem' | awk '{print $5}' | sed 's/%//')
if [ $DISK_USAGE -gt 85 ]; then
    echo "ALERT: Disk usage is ${DISK_USAGE}%"
    # Send alert
fi

# Check memory usage
MEMORY_USAGE=$(free | grep Mem | awk '{printf("%.0f", $3/$2 * 100.0)}')
if [ $MEMORY_USAGE -gt 90 ]; then
    echo "ALERT: Memory usage is ${MEMORY_USAGE}%"
    # Send alert
fi
```

Documentation Maintenance

Keeping Documentation Updated

- [ ] Review and update procedures monthly
- [ ] Document any new issues and solutions
- [ ] Update contact information quarterly
- [ ] Review and test all scripts monthly
- [ ] Update dependency versions and commands

Change Management

- All maintenance procedures must be version controlled
- Changes require peer review before implementation
- Test all changes in staging environment first
- Document all changes with rationale and impact assessment

This maintenance guide should be reviewed and updated regularly to reflect changes in the application architecture, infrastructure, and operational procedures.