# ASMC Next - Deployment Guide

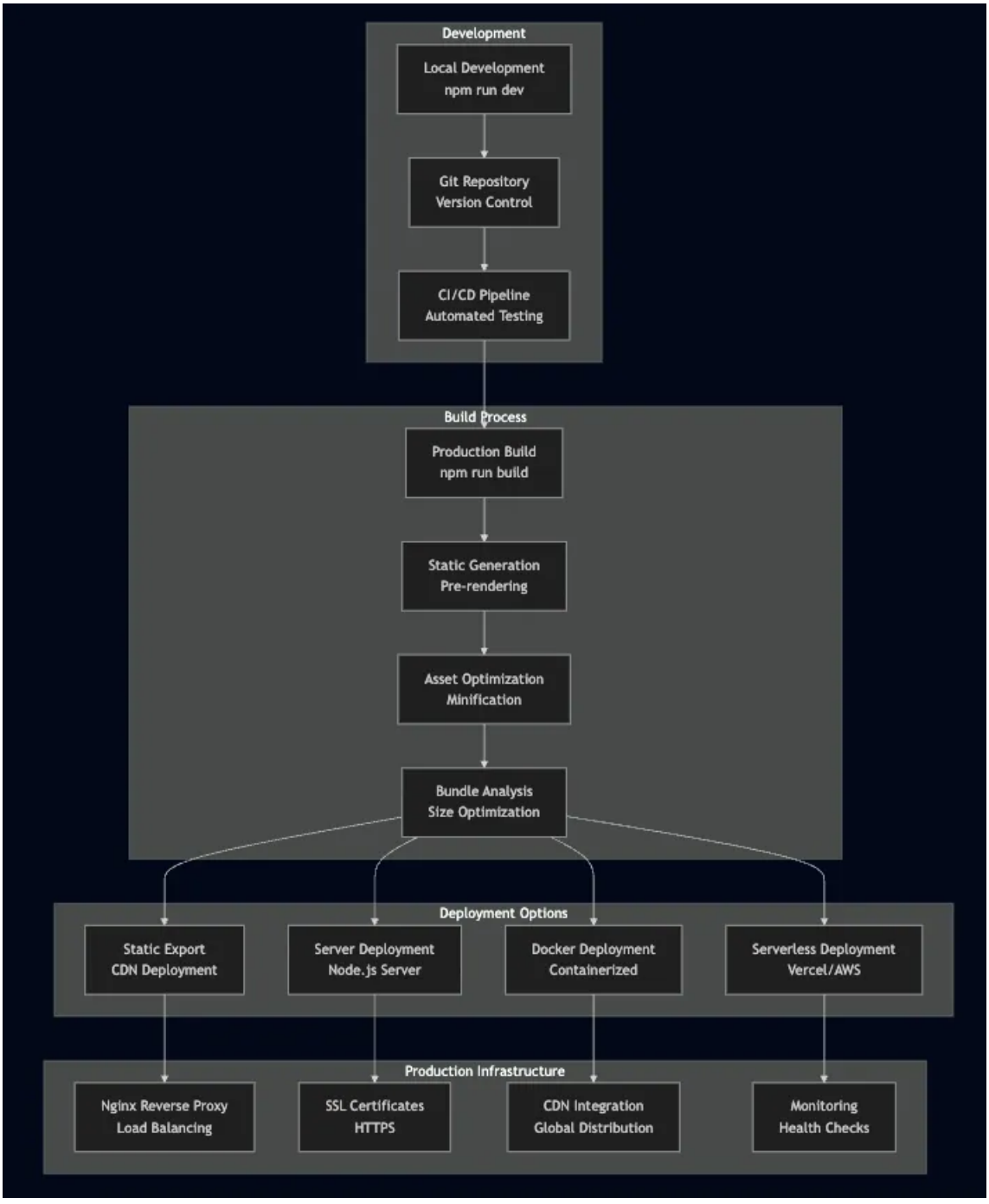
Comprehensive deployment guide for the ASMC Next.js frontend application, covering various deployment strategies, server configuration, monitoring, and maintenance.

## Table of Contents


- Deployment Overview
- Pre-deployment Checklist
- Build Configuration
- Deployment Strategies
- Server Configuration
- Environment Setup
- SSL & Security
- Performance Optimization
- Monitoring & Logging
- Maintenance & Updates
- Troubleshooting


## Deployment Overview

### Deployment Architecture




## Deployment Environments

### Development Environment

- **Purpose**: Local development and testing
- **URL**: `http://localhost:3000`
- **Features**: Hot reload, debug tools, source maps

### Staging Environment

- **Purpose**: Pre-production testing

- **URL**: `https://staging.asmcdae.in`
- **Features**: Production-like setup, testing data

**Production Environment**

- **Purpose**: Live application
- **URL**: `https://asmcdae.in`
- **Features**: Optimized build, monitoring, backups

## Pre-deployment Checklist

### Code Quality Checks

- [ ] **Code Review**: All code reviewed and approved
- [ ] **Linting**: No ESLint errors or warnings
- [ ] **Formatting**: Code formatted with Prettier
- [ ] **Testing**: All tests passing
- [ ] **Type Checking**: No TypeScript errors (if applicable)

### Build Verification

- [ ] **Build Success**: Production build completes successfully
- [ ] **Bundle Size**: Bundle size within acceptable limits
- [ ] **Performance**: Lighthouse scores meet requirements
- [ ] **Accessibility**: WCAG compliance verified

### Environment Configuration

- [ ] **Environment Variables**: All required variables set
- [ ] **API Endpoints**: Correct API URLs configured
- [ ] **Feature Flags**: Production flags enabled/disabled
- [ ] **Secrets**: Secure secrets properly configured

### Security Checks

- [ ] **Dependencies**: No known vulnerabilities
- [ ] **HTTPS**: SSL certificates configured
- [ ] **Headers**: Security headers implemented
- [ ] **CORS**: Cross-origin policies configured

## Build Configuration

### Production Build

**1. Build Scripts**

**package.json**:

```
{
    "scripts": {
        "build": "next build",
        "start": "next start",
        "build:analyze": "ANALYZE=true next build",
```

```
        "build:static": "next build && next export",
        "build:docker": "docker build -t asmc-next .",
        "deploy:staging": "npm run build && npm run deploy:staging:upload",
        "deploy:production": "npm run build && npm run deploy:production:upload"
    }
}
```

**2. Build Optimization**

**next.config.mjs**:

```
const nextConfig = {
    // Production optimizations
    reactStrictMode: true,
    swcMinify: true,
    compress: true,
    poweredByHeader: false,

    // Image optimization
    images: {
        domains: ['api.asmcdae.in', 'ik.imagekit.io'],
        formats: ['image/webp', 'image/avif'],
        deviceSizes: [640, 750, 828, 1080, 1200, 1920, 2048, 3840],
        imageSizes: [16, 32, 48, 64, 96, 128, 256, 384],
    },

    // Performance optimizations
    experimental: {
        optimizeCss: true,
        optimizePackageImports: [
            '@fortawesome/react-fontawesome',
            '@fortawesome/free-solid-svg-icons',
        ],
    },

    // Bundle analyzer
    webpack: (config, { isServer, dev }) => {
        // Bundle analyzer
        if (process.env.ANALYZE === 'true') {
            const { BundleAnalyzerPlugin } = require('webpack-bundle-analyzer');
            config.plugins.push(
                new BundleAnalyzerPlugin({
                    analyzerMode: 'server',
                    openAnalyzer: true,
                }),
            );
        }

        // Production optimizations
        if (!dev && !isServer) {
            config.optimization.splitChunks.cacheGroups = {
                ...config.optimization.splitChunks.cacheGroups,
                vendor: {
```

```
                    test: /[\\/]node_modules[\\/]/,
                    name: 'vendors',
                    chunks: 'all',
                },
            };
        }

        return config;
    },

    // Security headers
    async headers() {
        return [
            {
                source: '/(.*)',
                headers: [
                    {
                        key: 'X-Frame-Options',
                        value: 'DENY',
                    },
                    {
                        key: 'X-Content-Type-Options',
                        value: 'nosniff',
                    },
                    {
                        key: 'X-XSS-Protection',
                        value: '1; mode=block',
                    },
                    {
                        key: 'Referrer-Policy',
                        value: 'strict-origin-when-cross-origin',
                    },
                    {
                        key: 'Content-Security-Policy',
                        value: "default-src 'self'; script-src 'self' 'unsafe-eval' 'unsafe-inline'; style-src 'self' 'unsafe-inline'; img-src 'self' data: https:; font-src 'self' data:; connect-src 'self' https://api.asmcdae.in;",
                    },
                ],
            },
        ];
    },

    // Redirects
    async redirects() {
        return [
            {
                source: '/home',
                destination: '/',
                permanent: true,
            },
        ];
```

```
    },

    // Rewrites
    async rewrites() {
        return [
            {
                source: '/api/:path*',
                destination: 'https://api.asmcdae.in/api/:path*',
            },
        ];
    },
};

export default nextConfig;
```

**3. Environment Configuration**

**.env.production**:

```
# Production Environment Variables
NODE_ENV=production
NEXT_PUBLIC_API_URL=https://api.asmcdae.in
NEXT_PUBLIC_APP_NAME=ASMC
NEXT_PUBLIC_APP_VERSION=1.0.0
NEXT_PUBLIC_ENABLE_DEBUG=false
NEXT_PUBLIC_ENABLE_ANALYTICS=true

# Performance
NEXT_TELEMETRY_DISABLED=1
```

## Deployment Strategies

### 1. Static Export Deployment

**Static Export Configuration**

**next.config.mjs**:

```
const nextConfig = {
    output: 'export',
    trailingSlash: true,
    images: {
        unoptimized: true,
    },
    // ... other config
};

export default nextConfig;
```

**Build and Deploy**

```
# Build static export
npm run build
```

```
# Files will be in 'out' directory
ls -la out/

# Deploy to CDN or static hosting
# Upload 'out' directory contents to your hosting provider
```

**CDN Deployment**

```
# Example: Deploy to AWS S3
aws s3 sync out/ s3://asmc-next-bucket --delete

# Example: Deploy to Netlify
netlify deploy --prod --dir=out

# Example: Deploy to Vercel
vercel --prod
```

### 2. Server Deployment

**PM2 Deployment**

**ecosystem.config.js**:

```
module.exports = {
    apps: [
        {
            name: 'asmc-next',
            script: 'npm',
            args: 'start',
            instances: 'max',
            exec_mode: 'cluster',
            env: {
                NODE_ENV: 'development',
                PORT: 3000,
            },
            env_staging: {
                NODE_ENV: 'staging',
                PORT: 3000,
                NEXT_PUBLIC_API_URL: 'https://staging-api.asmcdae.in',
            },
            env_production: {
                NODE_ENV: 'production',
                PORT: 3000,
                NEXT_PUBLIC_API_URL: 'https://api.asmcdae.in',
            },
            error_file: './logs/err.log',
            out_file: './logs/out.log',
            log_file: './logs/combined.log',
            time: true,
            max_memory_restart: '1G',
            node_args: '--max-old-space-size=4096',
            watch: false,
```

```
            ignore_watch: ['node_modules', 'logs'],
            restart_delay: 4000,
            max_restarts: 10,
            min_uptime: '10s',
        },
    ],
};
```

**Deployment Script**

**deploy.sh**:

```
#!/bin/bash

# Deployment script for ASMC Next.js application

set -e

echo "🚀 Starting deployment process..."

# Environment variables
ENVIRONMENT=${1:-production}
APP_NAME="asmc-next"
APP_DIR="/var/www/asmc-next"
BACKUP_DIR="/var/backups/asmc-next"
LOG_DIR="$APP_DIR/logs"

# Create directories if they don't exist
mkdir -p $APP_DIR
mkdir -p $BACKUP_DIR
mkdir -p $LOG_DIR

# Backup current deployment
echo "📦 Creating backup..."
if [ -d "$APP_DIR/.next" ]; then
    tar -czf "$BACKUP_DIR/backup-$(date +%Y%m%d-%H%M%S).tar.gz" -C $APP_DIR .
fi

# Pull latest code
echo "📥 Pulling latest code..."
cd $APP_DIR
git fetch origin
git reset --hard origin/main

# Install dependencies
echo "📦 Installing dependencies..."
npm ci --production

# Build application
echo "🔨 Building application..."
npm run build

# Restart application
```

```
echo "🔄 Restarting application..."
pm2 restart $APP_NAME --env $ENVIRONMENT

# Health check
echo "🏥 Performing health check..."
sleep 10
if curl -f http://localhost:3000/api/health; then
    echo "✅ Deployment successful!"
else
    echo "❌ Deployment failed! Rolling back..."
    # Rollback logic here
    exit 1
fi

echo "🎉 Deployment completed successfully!"
```

### 3. Docker Deployment

**Dockerfile**

**Dockerfile**:

```
# Multi-stage build for production
FROM node:18-alpine AS base

# Install dependencies only when needed
FROM base AS deps
RUN apk add --no-cache libc6-compat
WORKDIR /app

# Install dependencies based on the preferred package manager
COPY package.json package-lock.json* ./
RUN npm ci --only=production && npm cache clean --force

# Rebuild the source code only when needed
FROM base AS builder
WORKDIR /app
COPY --from=deps /app/node_modules ./node_modules
COPY . .

# Set environment variables for build
ENV NEXT_TELEMETRY_DISABLED 1
ENV NODE_ENV production

# Build the application
RUN npm run build

# Production image, copy all the files and run next
FROM base AS runner
WORKDIR /app

ENV NODE_ENV production
ENV NEXT_TELEMETRY_DISABLED 1
```

```
RUN addgroup --system --gid 1001 nodejs
RUN adduser --system --uid 1001 nextjs

# Copy built application
COPY --from=builder /app/public ./public

# Set the correct permission for prerender cache
RUN mkdir .next
RUN chown nextjs:nodejs .next

# Automatically leverage output traces to reduce image size
COPY --from=builder --chown=nextjs:nodejs /app/.next/standalone ./
COPY --from=builder --chown=nextjs:nodejs /app/.next/static ./.next/static

USER nextjs

EXPOSE 3000

ENV PORT 3000
ENV HOSTNAME "0.0.0.0"

CMD ["node", "server.js"]
```

**Docker Compose**

**docker-compose.yml**:

```
version: '3.8'

services:
    asmc-next:
        build: .
        ports:
            - '3000:3000'
        environment:
            - NODE_ENV=production
            - NEXT_PUBLIC_API_URL=https://api.asmcdae.in
        volumes:
            - ./logs:/app/logs
        restart: unless-stopped
        healthcheck:
            test: ['CMD', 'curl', '-f', 'http://localhost:3000/api/health']
            interval: 30s
            timeout: 10s
            retries: 3
        networks:
            - asmc-network

    nginx:
        image: nginx:alpine
        ports:
            - '80:80'
```

```
            - '443:443'
        volumes:
            - ./nginx.conf:/etc/nginx/nginx.conf
            - ./ssl:/etc/nginx/ssl
        depends_on:
            - asmc-next
        restart: unless-stopped
        networks:
            - asmc-network

networks:
    asmc-network:
        driver: bridge
```

**Docker Deployment Commands**

```
# Build Docker image
docker build -t asmc-next:latest .

# Run container
docker run -d -p 3000:3000 --name asmc-next asmc-next:latest

# Using Docker Compose
docker-compose up -d

# Update deployment
docker-compose pull
docker-compose up -d
```

### 4. Serverless Deployment

**Vercel Deployment**

**vercel.json**:

```
{
    "version": 2,
    "builds": [
        {
            "src": "package.json",
            "use": "@vercel/next"
        }
    ],
    "routes": [
        {
            "src": "/api/(.*)",
            "dest": "https://api.asmcdae.in/api/$1"
        }
    ],
    "env": {
        "NEXT_PUBLIC_API_URL": "https://api.asmcdae.in"
    }
}
```

**AWS Lambda Deployment**

**serverless.yml**:

```
service: asmc-next

provider:
    name: aws
    runtime: nodejs18.x
    region: us-east-1
    stage: ${opt:stage, 'dev'}
    environment:
        NEXT_PUBLIC_API_URL: ${env:NEXT_PUBLIC_API_URL}

plugins:
    - serverless-nextjs-plugin

custom:
    nextjs:
        domain: asmcdae.in
        certificateName: asmcdae.in

functions:
    nextjs:
        handler: index.handler
        events:
            - http:
                  path: /{proxy+}
                  method: ANY
            - http:
                  path: /
                  method: ANY
```

## Server Configuration

### Nginx Configuration

**Basic Nginx Config**

**nginx.conf**:

```
user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log warn;
pid /var/run/nginx.pid;

events {
    worker_connections 1024;
    use epoll;
    multi_accept on;
}

http {
```

```
    include /etc/nginx/mime.types;
    default_type application/octet-stream;

    # Logging
    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
                    '$status $body_bytes_sent "$http_referer" '
                    '"$http_user_agent" "$http_x_forwarded_for"';

    access_log /var/log/nginx/access.log main;

    # Basic settings
    sendfile on;
    tcp_nopush on;
    tcp_nodelay on;
    keepalive_timeout 65;
    types_hash_max_size 2048;
    server_tokens off;

    # Gzip compression
    gzip on;
    gzip_vary on;
    gzip_min_length 1024;
    gzip_proxied any;
    gzip_comp_level 6;
    gzip_types
        text/plain
        text/css
        text/xml
        text/javascript
        application/json
        application/javascript
        application/xml+rss
        application/atom+xml
        image/svg+xml;

    # Rate limiting
    limit_req_zone $binary_remote_addr zone=api:10m rate=10r/s;
    limit_req_zone $binary_remote_addr zone=login:10m rate=1r/s;

    # Upstream servers
    upstream asmc_next {
        server 127.0.0.1:3000;
        keepalive 32;
    }

    # Main server block
    server {
        listen 80;
        server_name asmcdae.in www.asmcdae.in;
        return 301 https://$server_name$request_uri;
    }
```

```
    server {
        listen 443 ssl http2;
        server_name asmcdae.in www.asmcdae.in;

        # SSL configuration
        ssl_certificate /etc/letsencrypt/live/asmcdae.in/fullchain.pem;
        ssl_certificate_key /etc/letsencrypt/live/asmcdae.in/privkey.pem;
        ssl_protocols TLSv1.2 TLSv1.3;
        ssl_ciphers ECDHE-RSA-AES256-GCM-SHA512:DHE-RSA-AES256-GCM-SHA512:ECDHE-RSA-
AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384;
        ssl_prefer_server_ciphers off;
        ssl_session_cache shared:SSL:10m;
        ssl_session_timeout 10m;

        # Security headers
        add_header X-Frame-Options "SAMEORIGIN" always;
        add_header X-Content-Type-Options "nosniff" always;
        add_header X-XSS-Protection "1; mode=block" always;
        add_header Referrer-Policy "strict-origin-when-cross-origin" always;
        add_header Strict-Transport-Security "max-age=31536000; includeSubDomains"
always;

        # Static files
        location /_next/static/ {
            alias /var/www/asmc-next/.next/static/;
            expires 1y;
            add_header Cache-Control "public, immutable";
        }

        # API routes
        location /api/ {
            limit_req zone=api burst=20 nodelay;
            proxy_pass http://asmc_next;
            proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Connection 'upgrade';
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
            proxy_cache_bypass $http_upgrade;
            proxy_read_timeout 86400;
        }

        # Main application
        location / {
            proxy_pass http://asmc_next;
            proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Connection 'upgrade';
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
```

```
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
            proxy_cache_bypass $http_upgrade;
            proxy_read_timeout 86400;
        }
    }
}
```

**Nginx Setup Commands**

```
# Install Nginx
sudo apt update
sudo apt install nginx

# Copy configuration
sudo cp nginx.conf /etc/nginx/nginx.conf

# Test configuration
sudo nginx -t

# Restart Nginx
sudo systemctl restart nginx
sudo systemctl enable nginx
```

## 🔒 SSL & Security

### SSL Certificate Setup

**Let's Encrypt SSL**

```
# Install Certbot
sudo apt install certbot python3-certbot-nginx

# Obtain SSL certificate
sudo certbot --nginx -d asmcdae.in -d www.asmcdae.in

# Auto-renewal
sudo crontab -e
# Add: 0 12 * * * /usr/bin/certbot renew --quiet
```

**SSL Configuration**

```
# SSL configuration in Nginx
ssl_protocols TLSv1.2 TLSv1.3;
ssl_ciphers ECDHE-RSA-AES256-GCM-SHA512:DHE-RSA-AES256-GCM-SHA512:ECDHE-RSA-AES256-
GCM-SHA384:DHE-RSA-AES256-GCM-SHA384;
ssl_prefer_server_ciphers off;
ssl_session_cache shared:SSL:10m;
ssl_session_timeout 10m;

# HSTS
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
```

### Security Headers

#### Content Security Policy

```
// In next.config.mjs
async headers() {
    return [
        {
            source: '/(.*)',
            headers: [
                {
                    key: 'Content-Security-Policy',
                    value: [
                        "default-src 'self'",
                        "script-src 'self' 'unsafe-eval' 'unsafe-inline'",
                        "style-src 'self' 'unsafe-inline'",
                        "img-src 'self' data: https:",
                        "font-src 'self' data:",
                        "connect-src 'self' https://api.asmcdae.in",
                        "frame-ancestors 'none'",
                        "base-uri 'self'",
                        "form-action 'self'"
                    ].join('; '),
                },
            ],
        },
    ];
}
```

## Performance Optimization

### CDN Configuration

#### Cloudflare Setup

```
# DNS Configuration
# A record: asmcdae.in -> SERVER_IP
# CNAME record: www -> asmcdae.in

# Page Rules
# asmcdae.in/_next/static/* -> Cache Level: Cache Everything, Edge Cache TTL: 1 month
# asmcdae.in/api/* -> Cache Level: Bypass
```

#### CDN Headers

```
# Cache static assets
location ~* \.(js|css|png|jpg|jpeg|gif|ico|svg|woff|woff2|ttf|eot)$ {
    expires 1y;
    add_header Cache-Control "public, immutable";
    add_header Vary "Accept-Encoding";
}
```

### Performance Monitoring

#### Web Vitals

```
// In _app.js
import { getCLS, getFID, getFCP, getLCP, getTTFB } from 'web-vitals';

function sendToAnalytics(metric) {
    // Send to Google Analytics
    gtag('event', metric.name, {
        event_category: 'Web Vitals',
        event_label: metric.id,
        value: Math.round(metric.name === 'CLS' ? metric.value * 1000 : metric.value),
        non_interaction: true,
    });
}

getCLS(sendToAnalytics);
getFID(sendToAnalytics);
getFCP(sendToAnalytics);
getLCP(sendToAnalytics);
getTTFB(sendToAnalytics);
```

## Monitoring & Logging

### Application Monitoring

#### PM2 Monitoring

```
# Install PM2 monitoring
pm2 install pm2-server-monit

# View monitoring dashboard
pm2 monit

# View logs
pm2 logs asmc-next

# View real-time metrics
pm2 show asmc-next
```

#### Health Check Endpoint

**pages/api/health.js**:

```
export default function handler(req, res) {
    const healthCheck = {
        uptime: process.uptime(),
        message: 'OK',
        timestamp: Date.now(),
        environment: process.env.NODE_ENV,
        version: process.env.npm_package_version,
    };
```

```
    try {
        res.status(200).json(healthCheck);
    } catch (error) {
        healthCheck.message = error;
        res.status(503).json(healthCheck);
    }
}
```

### Log Management

#### Log Rotation

**/etc/logrotate.d/asmc-next**:

```
/var/www/asmc-next/logs/*.log {
    daily
    missingok
    rotate 52
    compress
    delaycompress
    notifempty
    create 644 www-data www-data
    postrotate
        pm2 reloadLogs
    endscript
}
```

#### Log Analysis

```
# View error logs
tail -f /var/www/asmc-next/logs/err.log

# Search for specific errors
grep "ERROR" /var/www/asmc-next/logs/combined.log

# Monitor real-time logs
pm2 logs asmc-next --lines 100
```

## Maintenance & Updates

### Update Process

#### Application Updates

```
#!/bin/bash
# update.sh - Application update script

set -e

echo " Starting application update..."

# Pull latest code
```

```
git pull origin main

# Install dependencies
npm ci --production

# Build application
npm run build

# Restart application
pm2 restart asmc-next

# Health check
sleep 10
if curl -f http://localhost:3000/api/health; then
    echo "✅ Update successful!"
else
    echo "❌ Update failed!"
    exit 1
fi
```

**Dependency Updates**

```
# Check for outdated packages
npm outdated

# Update dependencies
npm update

# Update specific package
npm install package-name@latest

# Security audit
npm audit
npm audit fix
```

## Backup Strategy

**Application Backup**

```
#!/bin/bash
# backup.sh - Application backup script

BACKUP_DIR="/var/backups/asmc-next"
APP_DIR="/var/www/asmc-next"
DATE=$(date +%Y%m%d-%H%M%S)

# Create backup directory
mkdir -p $BACKUP_DIR

# Backup application files
tar -czf "$BACKUP_DIR/app-backup-$DATE.tar.gz" -C $APP_DIR .
```

```
# Keep only last 7 days of backups
find $BACKUP_DIR -name "app-backup-*.tar.gz" -mtime +7 -delete

echo "✅ Backup completed: app-backup-$DATE.tar.gz"
```

**Database Backup**

```
#!/bin/bash
# db-backup.sh - Database backup script

BACKUP_DIR="/var/backups/asmc-db"
DATE=$(date +%Y%m%d-%H%M%S)

# Create backup directory
mkdir -p $BACKUP_DIR

# Backup MongoDB
mongodump --uri="mongodb://localhost:27017/asmc" --out="$BACKUP_DIR/db-backup-$DATE"

# Compress backup
tar -czf "$BACKUP_DIR/db-backup-$DATE.tar.gz" -C $BACKUP_DIR "db-backup-$DATE"

# Clean up uncompressed backup
rm -rf "$BACKUP_DIR/db-backup-$DATE"

echo "✅ Database backup completed: db-backup-$DATE.tar.gz"
```

## 🔧 Troubleshooting

### Common Issues

#### 1. Build Failures

**Problem**: Build fails with memory errors

```
# Solution: Increase Node.js memory
export NODE_OPTIONS="--max-old-space-size=4096"
npm run build
```

**Problem**: Build fails with dependency errors

```
# Solution: Clear cache and reinstall
rm -rf node_modules package-lock.json
npm cache clean --force
npm install
npm run build
```

#### 2. Runtime Issues

**Problem**: Application crashes on startup

```
# Check PM2 logs
pm2 logs asmc-next
```

```
# Check system resources
htop
df -h

# Restart application
pm2 restart asmc-next
```

**Problem**: High memory usage

```
# Monitor memory usage
pm2 monit

# Restart application
pm2 restart asmc-next

# Check for memory leaks
node --inspect server.js
```

#### 3. Performance Issues

**Problem**: Slow page loads

```
# Check server resources
htop
iotop

# Check Nginx logs
tail -f /var/log/nginx/access.log

# Test API response times
curl -w "@curl-format.txt" -o /dev/null -s "https://asmcdae.in/api/health"
```

#### 4. SSL Issues

**Problem**: SSL certificate errors

```
# Check certificate status
sudo certbot certificates

# Renew certificate
sudo certbot renew

# Test SSL configuration
openssl s_client -connect asmcdae.in:443
```

### Debug Commands

#### Application Debugging

```
# Enable debug mode
DEBUG=* pm2 restart asmc-next

# View detailed logs
```

```
pm2 logs asmc-next --lines 1000

# Monitor in real-time
pm2 monit
```

#### System Debugging

```
# Check system resources
htop
iotop
nethogs

# Check disk space
df -h
du -sh /var/www/asmc-next

# Check network connections
netstat -tulpn | grep :3000
```

#### Nginx Debugging

```
# Test Nginx configuration
sudo nginx -t

# Check Nginx status
sudo systemctl status nginx

# View Nginx logs
sudo tail -f /var/log/nginx/error.log
sudo tail -f /var/log/nginx/access.log
```

### Recovery Procedures

#### Application Recovery

```
# Restore from backup
cd /var/www/asmc-next
tar -xzf /var/backups/asmc-next/app-backup-YYYYMMDD-HHMMSS.tar.gz

# Restart application
pm2 restart asmc-next
```

#### Database Recovery

```
# Restore database from backup
mongorestore --uri="mongodb://localhost:27017/asmc" /var/backups/asmc-db/db-backup-YYYYMMDD-HHMMSS
```

---

## Additional Resources

### Documentation

- Next.js Deployment
- PM2 Documentation
- Nginx Documentation

### Monitoring Tools

- PM2 Plus
- New Relic
- DataDog

### Security Tools

- SSL Labs
- Security Headers
- Mozilla Observatory

---

 **Deployment Complete!**

Your ASMC Next.js application is now successfully deployed and ready for production use.

**Last Updated**: January 2025
**Maintainer**: ASMC Development Team