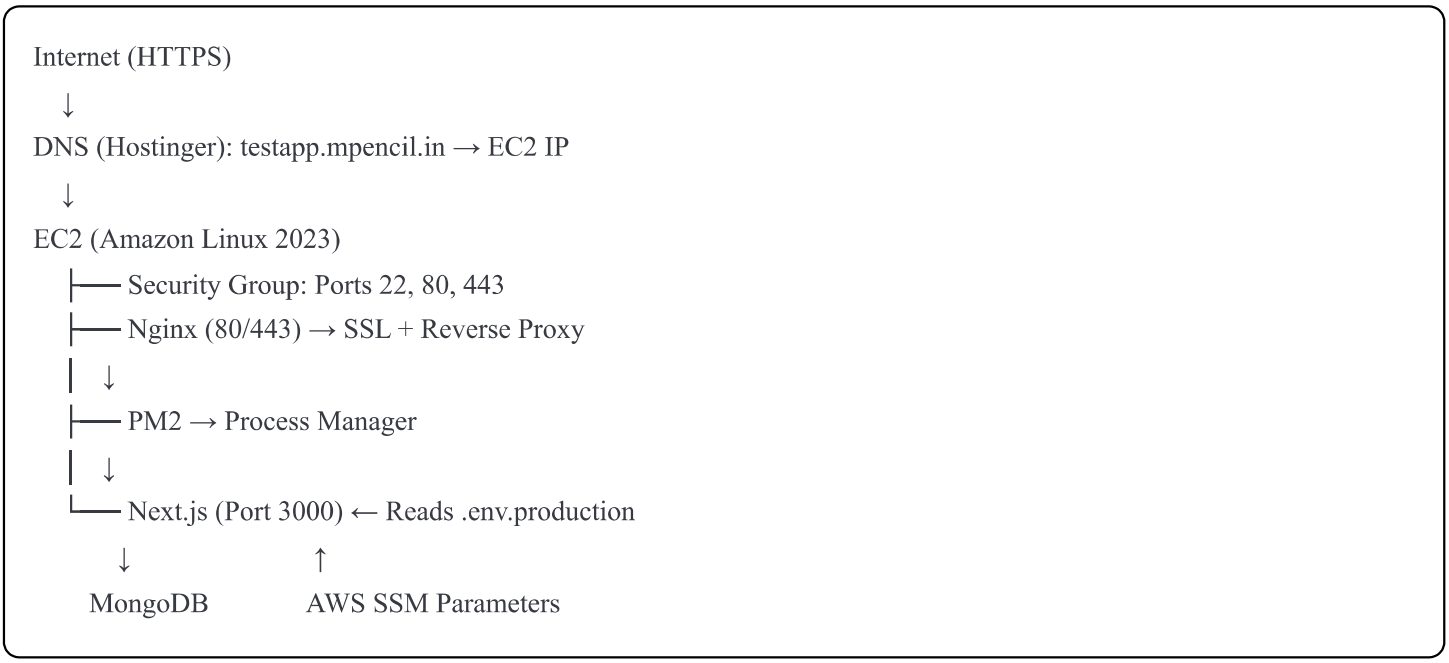
Next.js to EC2 - Complete Deployment Guide

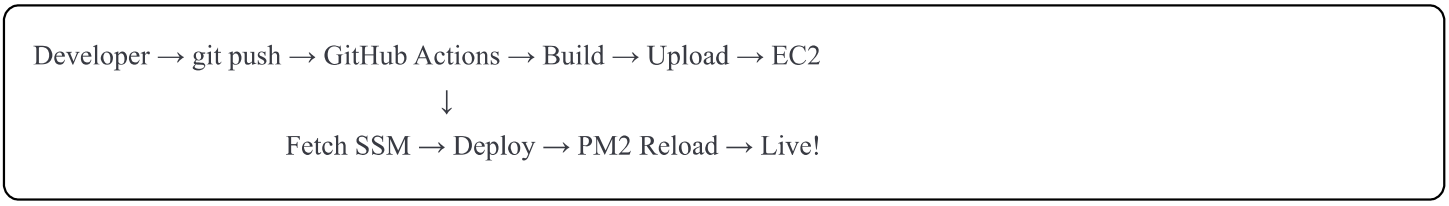
Automated CI/CD with SSL and Zero Downtime

Repository: https://github.com/Tusharc11/new-nextjs-mpen.git
Domain: https://testapp.mpencil.in
Time: ~45 minutes | Cost: ~$20/month

Architecture



Deployment Flow:



Why This Stack?

Tool	Why	Without It
PM2	Keeps app running 24/7, auto-restart, zero-downtime reload	App stops when you close terminal
Nginx	Handles HTTPS, custom domain, better performance	Can't easily use domain or HTTPS

Tool	Why	Without It
Amazon Linux	AWS-optimized, pre-installed tools, faster updates	Need to install AWS CLI, SSM agent manually
SSM	Secure encrypted secrets, easy updates	Hardcoded secrets (security risk)
GitHub Actions	Auto-deploy on push, no manual work	Manual SSH and deployment every time

---

Phase 1: AWS SSM Parameters (5 min)

Create Parameters

AWS Console → Systems Manager → Parameter Store → Create parameter

Create these 6 parameters:

```
/mpencil-app/test/MONGODB_URI          (SecureString) = mongodb+srv://...
/mpencil-app/test/JWT_SECRET           (SecureString) = [openssl rand -base64 32]
/mpencil-app/test/S3_REGION            (String)       = us-east-1
/mpencil-app/test/S3_ACCESS_KEY_ID     (SecureString) = AKIAXXXXX
/mpencil-app/test/S3_SECRET_ACCESS_KEY (SecureString) = wJalrXXXXX
/mpencil-app/test/S3_BUCKET_NAME       (String)       = your-bucket-name
```

Create IAM Role

IAM → Roles → Create role

- Trusted entity: AWS service → EC2
- Attach policies:
  - `AmazonSSMManagedInstanceCore`
  - `CloudWatchAgentServerPolicy`
- Name: `EC2-NextJS-SSM-Role`

Phase 2: Launch EC2 (10 min)

EC2 → Launch Instance

Settings:

Name: nextjs-erp-production

AMI: Amazon Linux 2023

Instance: t2.small (or t2.medium)

Key pair: Create new → nextjs-erp-key → Download .pem

Security group:

- SSH (22) - My IP
- HTTP (80) - 0.0.0.0/0
- HTTPS (443) - 0.0.0.0/0

Storage: 20 GB gp3

IAM role: EC2-NextJS-SSM-Role

User Data (paste this):

bash

```
#!/bin/bash
set -e

# Update system
dnf update -y

# Install Node.js 20
curl -fsSL https://rpm.nodesource.com/setup_20.x | bash -
dnf install -y nodejs

# Install PM2
npm install -g pm2

# Install Nginx
dnf install -y nginx
systemctl enable nginx
systemctl start nginx

# Create app directory
mkdir -p /var/www/nextjs-app/logs
chown -R ec2-user:ec2-user /var/www/nextjs-app

# Configure Nginx
cat > /etc/nginx/conf.d/nextjs-app.conf << 'EOF'
server {
    listen 80;
    server_name testapp.mpencil.in;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
        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;
    }
}
EOF

nginx -t && systemctl restart nginx
```

```
# Setup PM2 auto-start
env PATH=$PATH:/usr/bin /usr/local/lib/node_modules/pm2/bin/pm2 startup systemd -u ec2-user --hp /home/ec2-user

# Allow Nginx → Node.js
setsebool -P httpd_can_network_connect 1

echo "✅ Bootstrap completed!"
```

Launch → Wait for "Running" → Copy **Public IP**: _____

Phase 3: Domain Setup (5 min)

Hostinger → Domains → mpencil.in → DNS Records → Add Record

```
Type: A
Name: testapp
Points to: YOUR_EC2_PUBLIC_IP
TTL: 14400
```

Wait 10-30 minutes, then verify:

```bash
nslookup testapp.mpencil.in
# Should return your EC2 IP
```

Phase 4: SSL Certificate (5 min)

SSH into EC2:

```bash
ssh -i nextjs-erp-key.pem ec2-user@YOUR_EC2_IP
```

Install Certbot:

```bash
```

```
sudo dnf install -y python3 augeas-libs
sudo python3 -m venv /opt/certbot/
sudo /opt/certbot/bin/pip install certbot certbot-nginx
sudo ln -s /opt/certbot/bin/certbot /usr/bin/certbot
```

Get SSL certificate:

```bash
sudo certbot --nginx -d testapp.mpencil.in
# Email: your@email.com
# Agree: Y
# Share email: N
```

Setup auto-renewal:

```bash
sudo certbot renew --dry-run
echo "0 0,12 * * * root /opt/certbot/bin/python -c 'import random; import time; time.sleep(random.random() * 3600)' && sud
```

Exit:

```bash
exit
```

Test: `https://testapp.mpencil.in` (should show green lock)

Phase 5: Repository Files (10 min)

Navigate to repo:

```bash
cd new-nextjs-mpen
```

File 1: `.github/workflows/deploy.yml`

```yaml
```

```yaml
name: Deploy to EC2

on:
  push:
    branches: [main, master]
  workflow_dispatch:

jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4

      - uses: actions/setup-node@v4
        with:
          node-version: '20'
          cache: 'npm'

      - run: npm ci
      - run: npm run build
        env:
          NODE_ENV: production

      - run: |
          mkdir -p deploy-package
          cp -r .next deploy-package/
          cp -r public deploy-package/ 2>/dev/null || true
          cp package*.json deploy-package/
          cp next.config.* deploy-package/ 2>/dev/null || true
          cp ecosystem.config.js deploy-package/ 2>/dev/null || true
          tar -czf deploy.tar.gz -C deploy-package .

      - uses: appleboy/scp-action@master
        with:
          host: ${{ secrets.EC2_HOST }}
          username: ec2-user
          key: ${{ secrets.EC2_SSH_KEY }}
          source: "deploy.tar.gz"
          target: "/home/ec2-user/"

      - uses: appleboy/ssh-action@master
        with:
          host: ${{ secrets.EC2_HOST }}
```

```
username: ec2-user
key: ${{ secrets.EC2_SSH_KEY }}
command_timeout: 10m
script: |
  set -e
  APP_DIR="/var/www/nextjs-app"
  SSM_BASE="/mpencil-app/test"
  REGION=$(curl -s http://169.254.169.254/latest/meta-data/placement/region)

  cd $APP_DIR
  sudo tar -xzf /home/ec2-user/deploy.tar.gz
  sudo rm /home/ec2-user/deploy.tar.gz
  sudo chown -R ec2-user:ec2-user $APP_DIR

  npm ci --omit=dev --prefer-offline --no-audit

  get_ssm() {
     aws ssm get-parameter --name "$1" --with-decryption --region "$REGION" --query 'Parameter.Value' --output text
  }

  cat > $APP_DIR/.env.production << EOF
  NODE_ENV=production
  PORT=3000
  MONGODB_URI=$(get_ssm "$SSM_BASE/MONGODB_URI")
  JWT_SECRET=$(get_ssm "$SSM_BASE/JWT_SECRET")
  S3_REGION=$(get_ssm "$SSM_BASE/S3_REGION")
  S3_ACCESS_KEY_ID=$(get_ssm "$SSM_BASE/S3_ACCESS_KEY_ID")
  S3_SECRET_ACCESS_KEY=$(get_ssm "$SSM_BASE/S3_SECRET_ACCESS_KEY")
  S3_BUCKET_NAME=$(get_ssm "$SSM_BASE/S3_BUCKET_NAME")
  EOF

  chmod 600 $APP_DIR/.env.production

  if pm2 describe nextjs-app > /dev/null 2>&1; then
    pm2 reload ecosystem.config.js --update-env
  else
    pm2 start ecosystem.config.js
  fi

  pm2 save
  echo "✅ Deployment successful!"
```

File 2: `ecosystem.config.js` (root)

```javascript
module.exports = {
  apps: [{
    name: 'nextjs-app',
    script: 'npm',
    args: 'start',
    cwd: '/var/www/nextjs-app',
    instances: 1,
    autorestart: true,
    watch: false,
    max_memory_restart: '1G',
    env_file: '/var/www/nextjs-app/.env.production',
    error_file: '/var/www/nextjs-app/logs/err.log',
    out_file: '/var/www/nextjs-app/logs/out.log',
    log_date_format: 'YYYY-MM-DD HH:mm:ss',
    merge_logs: true,
    kill_timeout: 5000,
    listen_timeout: 3000,
  }]
};
```

File 3: Health Check

App Router: `app/api/health/route.js`

```javascript
import { NextResponse } from 'next/server';

export async function GET() {
  return NextResponse.json({
    status: 'healthy',
    timestamp: new Date().toISOString(),
    uptime: process.uptime(),
  });
}
```

Pages Router: `pages/api/health.js`

```javascript
```

```
export default function handler(req, res) {
  res.status(200).json({
    status: 'healthy',
    timestamp: new Date().toISOString(),
    uptime: process.uptime(),
  });
}
```

Phase 6: GitHub Secrets (5 min)

Go to: https://github.com/Tusharc11/new-nextjs-mpen/settings/secrets/actions

Add 2 secrets:

```
Name: EC2_HOST
Value: testapp.mpencil.in

Name: EC2_SSH_KEY
Value: [Paste entire content of nextjs-erp-key.pem]
```

To get key content:

```bash
cat nextjs-erp-key.pem
# Copy EVERYTHING including BEGIN/END lines
```

Phase 7: Deploy (5 min)

```bash
git add .
git commit -m "Add deployment configuration"
git push origin main
```

Monitor: https://github.com/Tusharc11/new-nextjs-mpen/actions

Wait 3-5 minutes for deployment to complete.

Phase 8: Verify

Browser: `https://testapp.mpencil.in` (should show your app)

Health check: `https://testapp.mpencil.in/api/health`

SSH check:

```bash
ssh -i nextjs-erp-key.pem ec2-user@testapp.mpencil.in
pm2 status  # Should show: nextjs-app | online
pm2 logs nextjs-app
exit
```

Troubleshooting

Issue	Solution
502 Bad Gateway	`ssh` in → `pm2 restart nextjs-app`
App won't start	`pm2 logs nextjs-app` → check errors
DNS not working	Wait longer (up to 48h), check Hostinger
SSL error	`sudo certbot renew --nginx`
GitHub Actions fails	Check which step failed, verify secrets

View logs:

```bash
ssh -i nextjs-erp-key.pem ec2-user@testapp.mpencil.in
pm2 logs nextjs-app --lines 100
```

Restart app:

```bash
```

```
pm2 reload nextjs-app    # Zero downtime
pm2 restart nextjs-app   # With downtime
```

Future Updates

To deploy changes:

```bash
git add .
git commit -m "Your changes"
git push origin main
```

✅ Automatic deployment in 3-5 minutes!
✅ Zero downtime!
✅ No manual work needed!

Maintenance Commands

Update environment variables:

1. Update SSM parameter in AWS
2. Push any code change (or manually SSH and reload)

View logs:

```bash
pm2 logs nextjs-app
pm2 logs nextjs-app --err  # Errors only
```

Restart services:

```bash
pm2 restart nextjs-app
sudo systemctl restart nginx
```

Check status:

```bash
pm2 status
sudo systemctl status nginx
```

Update system:

```bash
sudo dnf update -y
```

Quick Reference

SSH into server:

```bash
ssh -i nextjs-erp-key.pem ec2-user@testapp.mpencil.in
```

PM2 commands:

```bash
pm2 status            # View all apps
pm2 logs nextjs-app   # View logs
pm2 restart nextjs-app  # Restart
pm2 reload nextjs-app   # Zero downtime reload
pm2 monit             # Monitor resources
```

File locations:

```
App: /var/www/nextjs-app/
Logs: /var/www/nextjs-app/logs/
Env: /var/www/nextjs-app/.env.production
Nginx config: /etc/nginx/conf.d/nextjs-app.conf
SSL certs: /etc/letsencrypt/live/testapp.mpencil.in/
```

Complete Checklist

- [ ] Phase 1: Create 6 SSM parameters + IAM role
- [ ] Phase 2: Launch EC2 with bootstrap script
- [ ] Phase 3: Add DNS A record in Hostinger
- [ ] Phase 4: SSH in, install Certbot, get SSL
- [ ] Phase 5: Add 3 files to repository
- [ ] Phase 6: Add 2 GitHub secrets
- [ ] Phase 7: Commit and push
- [ ] Phase 8: Verify app works at https://testapp.mpencil.in

Total time: ~45 minutes
Monthly cost: ~$20-25

Summary

What you built:

- ✅ Production Next.js SSR app on EC2
- ✅ HTTPS with auto-renewal
- ✅ Custom domain
- ✅ Automated CI/CD
- ✅ Zero-downtime deployments
- ✅ Secure environment variables
- ✅ Auto-restart on crash/reboot

Tech stack: Amazon Linux 2023 | Node.js 20 | PM2 | Nginx | Let's Encrypt | AWS SSM | GitHub Actions

To deploy updates: `git push origin main` - That's it! 🚀

Need help? Check GitHub Actions logs or SSH in and run `pm2 logs nextjs-app`