🚀 Deployment Guide for Home Server

Network Setup

This guide assumes the following network configuration:
- **nginx Proxy Manager**: 192.168.1.236
- **Pop Culture App Server**: 192.168.1.142

Prerequisites on Your Home Computer

✅ You already have:
- Docker installed on 192.168.1.142
- nginx Proxy Manager installed on 192.168.1.236
- DuckDNS configured
- n8n for automations (highly recommended)

Step-by-Step Deployment

Step 1: Transfer Files to Your Server

Option A: Using Git (Recommended)

```
# On your server (192.168.1.142)
cd /home/your-username/
git clone <repository-url> pop-culture-news
cd pop-culture-news
```

Option B: Using SCP/SFTP

```
# From another computer, transfer the files
scp -r pop_culture_news_app/ user@192.168.1.142:/home/user/pop-culture-news/

# Or use FileZilla/WinSCP for GUI-based transfer
```

Step 2: Configure Environment

```
cd /home/your-username/pop-culture-news
cp .env.example .env
nano .env  # or use vim, or edit via file manager
```

Critical Settings for Your Setup:

```
# Database
DATABASE_URL="postgresql://popculture:SecurePassword123@postgres:5432/popculture_db?
schema=public"
POSTGRES_PASSWORD="SecurePassword123"  # Change this!

# Email Configuration for Gmail
SMTP_HOST="smtp.gmail.com"
SMTP_PORT="587"
SMTP_USER="your-email@gmail.com"
SMTP_PASSWORD="your-16-char-app-password"  # Get from Google Account settings
SMTP_FROM="your-email@gmail.com"

# App Configuration
NEXT_PUBLIC_APP_URL="http://192.168.1.142:3000"  # Local access
# Or use your DuckDNS domain:
# NEXT_PUBLIC_APP_URL="https://popculture.yourdomain.duckdns.org"

# Email Recipients
DEFAULT_EMAIL_RECIPIENTS="dain.franklyn@icloud.com,jherria@hotmail.com"

# Schedule Settings
DAILY_EMAIL_TIME="08:00"  # 8:00 AM
REFRESH_INTERVAL="6"  # Fetch news every 6 hours
```

Step 3: Launch Application

```
# Start all services
docker-compose up -d

# Wait for services to be ready (about 30-60 seconds)
docker-compose logs -f

# Press Ctrl+C when you see "ready" messages
```

Step 4: Verify Installation

1. **Check Services are Running:**
   `bash`
   `  docker-compose ps`
   You should see:
   - `popculture-db` (postgres)
   - `popculture-app` (next.js)
   - `popculture-cron` (cron jobs)

2. **Access the Application:**
   - Local: http://192.168.1.142:3000
   - Or: http://localhost:3000 (if on the server)

3. **Initial Setup:**
   - Go to Settings: http://192.168.1.142:3000/settings
   - Click "🔄 Fetch News Now" to load initial articles
   - Send a test email to verify email setup

Step 5: Configure nginx Proxy Manager (for external access)

1. **Open nginx Proxy Manager** (at http://192.168.1.236:81)

2. **Add Proxy Host:**
   - **Domain Names**: `popculture.yourdomain.duckdns.org`
   - **Scheme**: `http`
   - **Forward Hostname/IP**: `192.168.1.142` ← Note: This is the pop culture app server
   - **Forward Port**: `3000`
   - **Websockets Support**: ✅ On

3. **SSL Tab:**
   - **SSL Certificate**: Request a new SSL certificate
   - **Force SSL**: ✅ On
   - **HTTP/2 Support**: ✅ On
   - **HSTS Enabled**: ✅ On

4. **Update .env:**
   `env`
   `   NEXT_PUBLIC_APP_URL="https://popculture.yourdomain.duckdns.org"`

5. **Restart Docker:**
   `bash`
   `   docker-compose restart app`

Step 6: Configure Router Port Forwarding (Optional - for external access)

If using nginx Proxy Manager, forward port 443 (HTTPS):

1. Access your router admin panel
2. Go to Port Forwarding section
3. Add rule:
   - **Service Name**: nginx Proxy Manager
   - **External Port**: 443
   - **Internal IP**: 192.168.1.236 ← nginx Proxy Manager host
   - **Internal Port**: 443 (nginx proxy manager handles SSL)
   - **Protocol**: TCP

Note: The nginx Proxy Manager (192.168.1.236) will proxy requests to the Pop Culture App (192.168.1.142) internally on your network.

Step 7: Set Up Auto-Start on Boot

```
# Create systemd service file
sudo nano /etc/systemd/system/popculture-news.service
```

Add this content:

```
[Unit]
Description=Pop Culture News App
Requires=docker.service
After=docker.service

[Service]
Type=oneshot
RemainAfterExit=yes
WorkingDirectory=/home/your-username/pop-culture-news
ExecStart=/usr/bin/docker-compose up -d
ExecStop=/usr/bin/docker-compose down

[Install]
WantedBy=multi-user.target
```

Enable and start:

```
sudo systemctl enable popculture-news.service
sudo systemctl start popculture-news.service
```

Gmail Setup (Detailed)

Generate App Password for Gmail

1. Go to your Google Account: https://myaccount.google.com/
2. Click **Security** in the left sidebar
3. Enable **2-Step Verification** if not already enabled
4. Go back to Security, find **2-Step Verification**
5. Scroll down to **App passwords**
6. Click **App passwords**
7. Select:
   - **App**: Mail
   - **Device**: Other (Custom name) → "Pop Culture News"
8. Click **Generate**
9. Copy the 16-character password (format: `xxxx xxxx xxxx xxxx`)
10. Use this in `SMTP_PASSWORD` (without spaces)

Maintenance & Monitoring

Daily Checks

```
# Check if services are running
docker-compose ps

# View recent logs
docker-compose logs --tail=50 app
docker-compose logs --tail=50 cron
```

Weekly Maintenance

```
# Backup database
docker-compose exec postgres pg_dump -U popculture popculture_db > backup_$(date +
%Y%m%d).sql

# Check disk usage
df -h
docker system df

# Clean old Docker images (if needed)
docker system prune -a
```

Monthly Updates

```
# Pull latest code (if using git)
git pull

# Rebuild and restart
docker-compose down
docker-compose up -d --build

# Update dependencies
docker-compose exec app npm update
```

Troubleshooting

Issue: App won't start

```
# Check if ports are already in use
sudo lsof -i :3000
sudo lsof -i :5432

# If ports are in use, either:
# 1. Stop the conflicting service
# 2. Change port in docker-compose.yml:

services:
  app:
    ports:
      - "3001:3000"  # Use port 3001 instead
```

Issue: Database connection errors

```
# Check database is running
docker-compose ps postgres

# Check database logs
docker-compose logs postgres

# Restart database
docker-compose restart postgres

# Wait 30 seconds, then restart app
docker-compose restart app
```

Issue: No articles loading

```
# Manually trigger news fetch
curl -X POST http://192.168.1.142:3000/api/cron/fetch-news

# Check cron logs
docker-compose logs cron

# Restart cron service
docker-compose restart cron
```

Issue: Emails not sending

```
# Test email configuration
curl -X POST http://192.168.1.142:3000/api/email/test \
  -H "Content-Type: application/json" \
  -d '{"email":"your-test@email.com"}'

# Check app logs for email errors
docker-compose logs app | grep -i "email\|smtp"

# Verify SMTP settings in .env
cat .env | grep SMTP
```

Integration with n8n (Optional)

You can use n8n to create additional automations:

Example Workflows:

1. **Backup Automation**:
   - Schedule: Daily at 3 AM
   - Execute Command: Backup database
   - Upload to cloud storage

2. **Monitoring**:
   - Check if app is responding
   - Send notification if down

3. **Custom Notifications**:
   - Webhook when specific celebrities are mentioned
   - Custom email formatting

n8n HTTP Request Node:

```
{
  "method": "GET",
  "url": "http://192.168.1.142:3000/api/articles"
}
```

Performance Optimization

For Better Performance:

1. **Increase Docker Resources** (if needed):

```
bash
   # Edit docker-compose.yml to add resource limits
   services:
     app:
       deploy:
         resources:
           limits:
             memory: 1G
           reservations:
             memory: 512M
```

2. **Database Optimization**:

~~~
```bash
# Access PostgreSQL
docker-compose exec postgres psql -U popculture popculture_db

# Create indexes (if needed)
CREATE INDEX IF NOT EXISTS idx_articles_celebrities ON “Article” USING GIN (celebrities);
CREATE INDEX IF NOT EXISTS idx_articles_categories ON “Article” USING GIN (categories);
```
~~~

1. **Cleanup Old Articles** (optional):

```
sql
   -- Delete articles older than 90 days
   DELETE FROM "Article" WHERE "publishDate" < NOW() - INTERVAL '90 days';
```

Security Best Practices

1. **Change Default Passwords**:

```
bash
   # Generate strong password
   openssl rand -base64 32
   # Use in POSTGRES_PASSWORD
```

2. **Firewall Rules**:

```
bash
   # If using UFW
   sudo ufw allow 3000/tcp  # Only if accessing locally
   sudo ufw allow 443/tcp   # For HTTPS through nginx
```

3. **Regular Updates**:

~~~
```bash
# Update system packages
sudo apt update && sudo apt upgrade

# Update Docker images
docker-compose pull
~~~

docker-compose up -d
```

Accessing from Other Devices

Local Network Access:

- **From any device on your home network**: http://192.168.1.142:3000
- **From iPhone/Android**: http://192.168.1.142:3000

External Access (with DuckDNS):

- **From anywhere**: https://popculture.yourdomain.duckdns.org (via nginx Proxy Manager)

Support Checklist

Before asking for help, verify:

- ✅ Docker is running: `docker --version`
- ✅ Docker Compose is running: `docker-compose --version`
- ✅ Services are up: `docker-compose ps`
- ✅ No port conflicts: `sudo lsof -i :3000 :5432`
- ✅ Environment variables are set: `cat .env`
- ✅ Logs show no errors: `docker-compose logs`

Ready to deploy! Follow these steps and you'll have your pop culture news app running in under 15 minutes. 🚀
```