🚀 Portal NewLoc - Guia Completo de Deploy

📋 Índice

---

1. Pré-requisitos

1.1. Servidor

Requisitos Mínimos:
- CPU: 2 cores
- RAM: 4 GB
- Storage: 20 GB SSD
- SO: Ubuntu 20.04+ / Debian 11+ / CentOS 8+

Requisitos Recomendados:
- CPU: 4 cores
- RAM: 8 GB
- Storage: 50 GB SSD
- SO: Ubuntu 22.04 LTS

1.2. Software Necessário

```
# Atualizar sistema
sudo apt update && sudo apt upgrade -y

# Instalar Docker
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
sudo usermod -aG docker $USER

# Instalar Docker Compose
sudo apt install docker-compose-plugin -y

# Verificar instalação
docker --version
docker compose version
```

1.3. Portas Necessárias

Porta	Serviço	Descrição
80	HTTP	Tráfego web (redireciona para 443)
443	HTTPS	Tráfego web seguro
3000	Next.js	Aplicação (interno)
5432	PostgreSQL	Banco de dados (interno)
8080	Adminer	Gerenciamento DB (opcional)

Configurar Firewall:

```
# UFW (Ubuntu/Debian)
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp
sudo ufw enable

# Firewalld (CentOS/RHEL)
sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-service=https
sudo firewall-cmd --reload
```

---

2. Estrutura do Projeto

```
portal_newloc/
├── nextjs_space/              # Código-fonte Next.js
│   ├── app/                   # Páginas e APIs
│   ├── components/            # Componentes React
│   ├── lib/                   # Utilitários
│   ├── prisma/                # Schema do banco
│   ├── public/                # Assets estáticos
│   ├── scripts/               # Scripts auxiliares
│   └── package.json
├── Dockerfile                 # Build da aplicação
├── docker-compose.yml         # Orquestração dos serviços
├── .dockerignore              # Arquivos ignorados no build
├── .env.example               # Template de variáveis
├── nginx.conf                 # Configuração do NGINX
├── deploy.sh                  # Script de deploy
├── backup.sh                  # Script de backup
├── API_DOCUMENTATION.md       # Documentação da API
├── N8N_INTEGRATION.md         # Guia de integração n8n
└── DEPLOY_GUIDE.md           # Este guia
```

---

3. Configuração Inicial

3.1. Clonar/Upload do Projeto

```
# Opção 1: Git (recomendado)
git clone https://seu-repositorio.git
cd portal_newloc

# Opção 2: Upload via SCP
scp -r portal_newloc user@server:/opt/
ssh user@server
cd /opt/portal_newloc
```

3.2. Configurar Variáveis de Ambiente

```
# Copiar template
cp .env.example .env

# Editar com suas configurações
nano .env
```

Variáveis Obrigatórias:

```
# Database
DATABASE_URL="postgresql://postgres:SUA_SENHA_SEGURA@db:5432/portal_newloc?
schema=public"
DB_PASSWORD="SUA_SENHA_SEGURA"

# Autenticação (gerar com: openssl rand -base64 32)
NEXTAUTH_SECRET="gere_um_secret_aqui_32_caracteres"
JWT_SECRET="outro_secret_aqui_32_caracteres"

# URLs
NEXTAUTH_URL="https://app.newloc.com"
NEXT_PUBLIC_BASE_URL="https://app.newloc.com"

# Production
NODE_ENV="production"
```

Gerar Secrets Seguros:

```
# NEXTAUTH_SECRET
openssl rand -base64 32

# JWT_SECRET
openssl rand -base64 32

# DB_PASSWORD (senha forte)
openssl rand -base64 24
```

3.3. Verificar Configurações

```
# Verificar se todos os arquivos estão presentes
ls -la

# Verificar .env (sem expor valores)
grep -v "^#" .env | grep -v "^$" | cut -d= -f1
```

---

4. Deploy com Docker

4.1. Deploy Inicial

```
# Dar permissões aos scripts
chmod +x deploy.sh backup.sh

# Executar deploy inicial
./deploy.sh

# Selecionar opção: 1) Deploy inicial
```

Ou manualmente:

```
# Build das imagens
docker compose build --no-cache

# Subir os serviços
docker compose up -d

# Verificar status
docker compose ps

# Ver logs
docker compose logs -f
```

4.2. Executar Migrations

```
# Entrar no container da aplicação
docker compose exec web sh

# Executar migrations
npx prisma migrate deploy

# (Opcional) Popular com dados iniciais
npx prisma db seed

# Sair do container
exit
```

4.3. Verificar Saúde da Aplicação

```
# Health check
curl http://localhost:3000/api/health

# Resposta esperada:
# {"status":"ok","timestamp":"2024-11-14T...","service":"Portal NewLoc"}
```

4.4. Testar Login

```
# Testar autenticação
curl -X POST http://localhost:3000/api/auth/login \
  -H "Content-Type: application/json" \
  -d '{"email":"admin@newloc.com","password":"Admin@123"}'

# Deve retornar o token e dados do usuário
```

---

5. Configuração do NGINX

5.1. Instalar NGINX

```
# Ubuntu/Debian
sudo apt install nginx -y

# CentOS/RHEL
sudo yum install nginx -y

# Iniciar e habilitar
sudo systemctl start nginx
sudo systemctl enable nginx
```

5.2. Configurar Proxy Reverso

```
# Copiar configuração
sudo cp nginx.conf /etc/nginx/sites-available/newloc

# Criar symlink
sudo ln -s /etc/nginx/sites-available/newloc /etc/nginx/sites-enabled/

# Remover configuração default
sudo rm /etc/nginx/sites-enabled/default

# Testar configuração
sudo nginx -t

# Recarregar NGINX
sudo systemctl reload nginx
```

5.3. Ajustar nginx.conf

Edite `/etc/nginx/sites-available/newloc`:

```
# Upstream (backend)
upstream nextjs_app {
    server localhost:3000;
    keepalive 32;
}

server {
    listen 80;
    server_name app.newloc.com;

    # Temporariamente permitir HTTP para configurar SSL
    location / {
        proxy_pass http://nextjs_app;
        proxy_http_version 1.1;
        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;
    }
}
```

---

6. SSL/HTTPS com Let’s Encrypt

6.1. Instalar Certbot

```
# Ubuntu/Debian
sudo apt install certbot python3-certbot-nginx -y

# CentOS/RHEL
sudo yum install certbot python3-certbot-nginx -y
```

6.2. Obter Certificado SSL

```
# Obter certificado automaticamente
sudo certbot --nginx -d app.newloc.com

# Responder às perguntas:
# - Email: seu@email.com
# - Termos: Agree
# - Redirect HTTP → HTTPS: Yes (opção 2)
```

6.3. Renovação Automática

```
# Testar renovação
sudo certbot renew --dry-run

# Certbot adiciona automaticamente um cron job
# Verificar:
sudo systemctl status certbot.timer
```

6.4. Verificar SSL

```
# Testar HTTPS
curl -I https://app.newloc.com

# Verificar certificado
openssl s_client -connect app.newloc.com:443 -servername app.newloc.com
```

---

7. Configuração do Domínio

7.1. Registros DNS

Configure no seu provedor de domínio:

```
Tipo    Nome    Valor               TTL
A       app     IP_DO_SEU_SERVIDOR  3600
```

Exemplo:

```
A    app.newloc.com    203.0.113.100    3600
```

7.2. Verificar Propagação DNS

```
# Linux/Mac
dig app.newloc.com

# Windows
nslookup app.newloc.com

# Online
# https://dnschecker.org
```

7.3. Aguardar Propagação

A propagação DNS pode levar de alguns minutos até 48 horas.

---

8. Monitoramento e Logs

8.1. Logs dos Serviços

```
# Logs da aplicação
docker compose logs -f web

# Logs do banco de dados
docker compose logs -f db

# Logs de todos os serviços
docker compose logs -f

# Últimas 100 linhas
docker compose logs --tail=100

# Logs do NGINX
sudo tail -f /var/log/nginx/newloc_access.log
sudo tail -f /var/log/nginx/newloc_error.log
```

8.2. Monitorar Recursos

```
# Status dos containers
docker compose ps

# Uso de recursos
docker stats

# Espaço em disco
df -h
docker system df
```

8.3. Monitorar Banco de Dados

```
# Entrar no PostgreSQL
docker compose exec db psql -U postgres portal_newloc

# Verificar tabelas
\dt

# Contar registros
SELECT COUNT(*) FROM usuarios_portal;
SELECT COUNT(*) FROM documentos_operacoes;

# Sair
\q
```

8.4. Adminer (Opcional)

Para ativar o Adminer:

```
# Subir com o profile "tools"
docker compose --profile tools up -d adminer

# Acessar: http://SEU_IP:8080
# - Sistema: PostgreSQL
# - Servidor: db
# - Usuário: postgres
# - Senha: (do .env)
# - Base de dados: portal_newloc
```

⚠️ **IMPORTANTE:** Não exponha o Adminer publicamente em produção!

---

9. Backup e Recuperação

9.1. Backup Automático

```
# Executar backup manual
./backup.sh

# O backup será salvo em: ./backups/newloc_backup_YYYYMMDD_HHMMSS.sql.gz
```

9.2. Agendar Backups (Cron)

```
# Editar crontab
crontab -e

# Adicionar linha para backup diário às 3h da manhã
0 3 * * * cd /opt/portal_newloc && ./backup.sh >> /var/log/newloc_backup.log 2>&1
```

9.3. Restaurar Backup

```
# Listar backups disponíveis
ls -lh backups/

# Restaurar backup específico
gunzip -c backups/newloc_backup_20241114_030000.sql.gz | \
  docker exec -i newloc_postgres psql -U postgres portal_newloc

# Reiniciar aplicação
docker compose restart web
```

9.4. Backup Completo do Sistema

```
# Parar serviços
docker compose down

# Fazer backup do volume
sudo tar -czf newloc_full_backup_$(date +%Y%m%d).tar.gz \
  /var/lib/docker/volumes/newloc_pgdata \
  /opt/portal_newloc/.env \
  /opt/portal_newloc/nextjs_space

# Subir serviços novamente
docker compose up -d
```

---

10. Troubleshooting

10.1. Aplicação não inicia

```
# Verificar logs
docker compose logs web

# Verificar se banco está funcionando
docker compose exec db pg_isready -U postgres

# Recriar containers
docker compose down
docker compose up -d --force-recreate
```

10.2. Erro de Conexão com Banco

```
# Verificar variáveis de ambiente
docker compose exec web env | grep DATABASE

# Testar conexão manualmente
docker compose exec web npx prisma db pull
```

10.3. Migrations Falhando

```
# Reset do banco (⚠️ CUIDADO: apaga dados!)
docker compose exec web npx prisma migrate reset

# Ou aplicar migrations manualmente
docker compose exec web npx prisma migrate deploy
```

10.4. NGINX 502 Bad Gateway

```
# Verificar se aplicação está rodando
curl http://localhost:3000/api/health

# Verificar logs do NGINX
sudo tail -f /var/log/nginx/error.log

# Testar configuração do NGINX
sudo nginx -t

# Reiniciar NGINX
sudo systemctl restart nginx
```

10.5. SSL não funciona

```
# Verificar certificado
sudo certbot certificates

# Renovar certificado
sudo certbot renew --force-renewal

# Verificar configuração NGINX
sudo nginx -t
```

10.6. Alto uso de memória

```
# Ver uso de memória
docker stats

# Limpar logs antigos
docker compose logs --tail=0 web

# Limpar imagens não utilizadas
docker system prune -a
```

10.7. Sessões expirando muito rápido

Verificar configuração no código:
- `lib/auth.ts` → `createSession()` → 8 horas de validade

```
// Ajustar se necessário
const expiracao = new Date(Date.now() + 8 * 60 * 60 * 1000); // 8 horas
```

📊 Checklist de Deploy

Pré-Deploy

- [ ] Servidor configurado com requisitos mínimos
- [ ] Docker e Docker Compose instalados
- [ ] Firewall configurado (portas 80/443)
- [ ] Domínio apontando para o servidor

- [ ] Variáveis de ambiente configuradas
- [ ] Secrets gerados com openssl

Deploy

- [ ] Build das imagens Docker concluído
- [ ] Containers iniciados com sucesso
- [ ] Migrations aplicadas
- [ ] Seed executado (dados iniciais)
- [ ] Health check respondendo OK
- [ ] Login funcionando

NGINX e SSL

- [ ] NGINX instalado e rodando
- [ ] Proxy reverso configurado
- [ ] Certificado SSL obtido
- [ ] HTTPS funcionando
- [ ] Redirecionamento HTTP → HTTPS ativo

Segurança

- [ ] Senhas fortes configuradas
- [ ] Secrets únicos gerados
- [ ] Firewall ativo
- [ ] Adminer não exposto publicamente
- [ ] Rate limiting configurado no NGINX

Backup e Monitoramento

- [ ] Script de backup testado
- [ ] Cron job de backup configurado
- [ ] Logs sendo gerados corretamente
- [ ] Monitoramento de recursos configurado

Pós-Deploy

- [ ] Teste de login como admin
- [ ] Teste de criação de cliente
- [ ] Teste de criação de documento
- [ ] Teste de listagem de documentos
- [ ] Teste de visualização de imagem
- [ ] Teste de download de imagem
- [ ] Verificação de permissões (admin vs cliente)

🆘 Comandos Úteis

```
# Ver status dos serviços
docker compose ps

# Reiniciar todos os serviços
docker compose restart

# Reiniciar apenas a aplicação
docker compose restart web

# Ver logs em tempo real
docker compose logs -f

# Entrar no container
docker compose exec web sh

# Atualizar aplicação
./deploy.sh  # Opção 2

# Fazer backup
./backup.sh

# Limpar tudo e reinstalar
./deploy.sh  # Opção 6

# Verificar uso de espaço
docker system df
df -h

# Limpar logs do Docker
truncate -s 0 /var/lib/docker/containers/*/*-json.log

# Ver processos em execução
docker compose top

# Parar todos os serviços
docker compose down

# Parar e remover volumes (⚠️ CUIDADO)
docker compose down -v
```

📞 Suporte

Para mais informações, consulte:

- **Documentação da API:** API_DOCUMENTATION.md (./API_DOCUMENTATION.md)
- **Integração com n8n:** N8N_INTEGRATION.md (./N8N_INTEGRATION.md)
- **Logs do sistema:** `docker compose logs -f`
- **Status dos serviços:** `docker compose ps`

🎉 Conclusão

Seguindo este guia, você terá o Portal NewLoc rodando em produção com:

✅ Aplicação Next.js otimizada
✅ Banco de dados PostgreSQL
✅ SSL/HTTPS configurado
✅ Proxy reverso com NGINX
✅ Backups automatizados
✅ Monitoramento configurado
✅ Integração pronta para n8n

Acesse: https://app.newloc.com
Login Admin: admin@newloc.com / Admin@123

🚀 **Seu portal está pronto para uso!**