# 🚀 Guía de Despliegue - SMS CloudMX

## 📋 Tabla de Contenidos



## 🛠️ Preparación

### Requisitos Previos

- Node.js 18+
- PostgreSQL database
- Cuenta en Vercel (recomendado)
- Dominio propio (opcional)

### Verificar Build Local

Antes de desplegar, verifica que todo funcione localmente:

```
yarn build
yarn start
```

# 🔐 Variables de Entorno

## Variables Requeridas en Producción

```
# Base de datos
DATABASE_URL="postgresql://usuario:password@host:5432/database"

# NextAuth
NEXTAUTH_URL="https://tudominio.com"
NEXTAUTH_SECRET="clave-secreta-muy-segura-de-al-menos-32-caracteres"

# Aplicación
APP_NAME="SMS CloudMX"
APP_URL="https://tudominio.com"

# Proveedor SMS
SMS_API_KEY="tu-clave-api-sms"
SMS_API_URL="https://api.proveedor-sms.com"

# Email (opcional)
EMAIL_FROM="noreply@tudominio.com"
EMAIL_SERVER_HOST="smtp.gmail.com"
EMAIL_SERVER_PORT=587
EMAIL_SERVER_USER="tu-email@gmail.com"
EMAIL_SERVER_PASSWORD="password-aplicacion"

# Analytics (opcional)
GOOGLE_ANALYTICS_ID="G-XXXXXXXXXX"

# Almacenamiento (si usas S3)
AWS_ACCESS_KEY_ID="tu-access-key"
AWS_SECRET_ACCESS_KEY="tu-secret-key"
AWS_REGION="us-east-1"
AWS_BUCKET_NAME="sms-cloudmx-uploads"
```

# 🗄️ Base de Datos

## Configuración PostgreSQL

### Opción 1: Supabase (Recomendado)

1. Crea cuenta en Supabase (https://supabase.com)
2. Crea nuevo proyecto
3. Copia la cadena de conexión
4. Añade a `DATABASE_URL`

### Opción 2: Neon

1. Crea cuenta en Neon (https://neon.tech)
2. Crea nueva base de datos
3. Copia cadena de conexión
4. Añade a `DATABASE_URL`

### Opción 3: Railway

1. Crea cuenta en Railway (https://railway.app)
2. Deploy PostgreSQL
3. Copia variables de entorno

4. Configura `DATABASE_URL`

### Migraciones

```
# Generar cliente Prisma
npx prisma generate

# Aplicar migraciones
npx prisma db push

# Verificar esquema
npx prisma studio
```

## 🌐 Despliegue en Vercel

### 1. Preparar Repositorio

```
# Si no tienes git inicializado
git init
git add .
git commit -m "Initial deployment"

# Conectar con GitHub
git remote add origin https://github.com/qhosting/sms.git
git push -u origin main
```

### 2. Conectar con Vercel

1. Ve a Vercel (https://vercel.com)
2. Importa desde GitHub
3. Selecciona el repositorio `qhosting/sms`
4. Configura las variables de entorno

### 3. Configurar Variables en Vercel

En el dashboard de Vercel:
- Settings → Environment Variables
- Añade todas las variables de `.env.example`
- Usa valores reales de producción

### 4. Build Commands (Automático)

Vercel detectará automáticamente:

```
{
  "buildCommand": "yarn build",
  "devCommand": "yarn dev",
  "installCommand": "yarn install"
}
```

# 🐳 Despliegue con Docker

## Dockerfile

```dockerfile
FROM node:18-alpine

WORKDIR /app

# Instalar dependencias
COPY package.json yarn.lock ./
RUN yarn install --frozen-lockfile

# Copiar código
COPY . .

# Generar Prisma client
RUN npx prisma generate

# Build aplicación
RUN yarn build

EXPOSE 3000

CMD ["yarn", "start"]
```

## Docker Compose

```yaml
version: '3.8'

services:
  app:
    build: .
    ports:
      - "3000:3000"
    environment:
      - DATABASE_URL=postgresql://postgres:password@db:5432/sms_cloudmx
      - NEXTAUTH_URL=http://localhost:3000
      - NEXTAUTH_SECRET=your-secret-key
    depends_on:
      - db

  db:
    image: postgres:15
    environment:
      - POSTGRES_DB=sms_cloudmx
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=password
    volumes:
      - postgres_data:/var/lib/postgresql/data
    ports:
      - "5432:5432"

volumes:
  postgres_data:
```

### Comandos Docker

```
# Build y ejecutar
docker-compose up -d

# Ver logs
docker-compose logs -f app

# Ejecutar migraciones
docker-compose exec app npx prisma db push
```

## 🌍 Configuración de Dominio

### DNS Records

```
Tipo    Nombre    Valor
A       @         ip-de-vercel
CNAME   www       tuapp.vercel.app
```

### SSL Certificate

Vercel proporciona SSL automático para dominios personalizados.

## 📊 Monitoreo

### Health Check Endpoint

Crea `/app/api/health/route.ts`:

```
import { NextResponse } from 'next/server';
import { PrismaClient } from '@prisma/client';

const prisma = new PrismaClient();

export async function GET() {
  try {
    await prisma.$queryRaw`SELECT 1`;
    return NextResponse.json({
      status: 'healthy',
      timestamp: new Date().toISOString()
    });
  } catch (error) {
    return NextResponse.json(
      { status: 'unhealthy', error: 'Database connection failed' },
      { status: 500 }
    );
  }
}
```

### Logging (Opcional)

Integra con servicios como:
- LogRocket (https://logrocket.com)
- Sentry (https://sentry.io)
- DataDog (https://datadoghq.com)

# 🚨 Troubleshooting

## Errores Comunes

### Build Failures

```
# Limpiar cache
rm -rf .next node_modules
yarn install
yarn build
```

### Database Connection Issues

```
# Verificar conexión
npx prisma studio

# Regenerar cliente
npx prisma generate
npx prisma db push
```

### Environment Variables

- Verificar que todas las variables estén configuradas
- No usar comillas en Vercel dashboard
- Verificar nombres exactos de variables

## Logs de Vercel

```
# Instalar CLI
npm i -g vercel

# Ver logs en tiempo real
vercel logs
```

# ✅ Checklist de Despliegue

- [ ] Build local exitoso
- [ ] Base de datos configurada
- [ ] Variables de entorno configuradas
- [ ] Repositorio en GitHub actualizado
- [ ] Proyecto conectado en Vercel
- [ ] Dominio personalizado (opcional)
- [ ] SSL certificate activo
- [ ] Health check funcionando
- [ ] Pruebas de funcionalidad básica
- [ ] Monitoreo configurado

# 🔄 Actualizaciones

## Deployment Automático

Cada push a `main` desplegará automáticamente en Vercel.

### Rollback

```
# Ver deployments
vercel list

# Hacer rollback a deployment específico
vercel rollback [deployment-url]
```

---

¡Tu aplicación SMS CloudMX está lista para producción! 🚀