



# CONFIGURACIÓN DE PRODUCCIÓN - INMOVA

---

## RESUMEN EJECUTIVO

---

Esta guía te ayudará a configurar todos los servicios críticos para producción en el siguiente orden:

1.  **Stripe LIVE** (15 min) - Pagos en producción
  2.  **SendGrid** (20 min) - Emails transaccionales
  3.  **Backups Automáticos** (5 min) - Respaldos diarios
  4.  **Sentry** (15 min) - Monitoreo de errores
  5.  **UptimeRobot** (10 min) - Monitoreo 24/7
  6.  **Cloudflare CDN** (20 min) - Opcional pero recomendado
  7.  **PostgreSQL Client** (2 min) - Para backups locales
- 

## 1 STRIPE - MODO LIVE (CRÍTICO)

---

### Paso 1: Obtener claves LIVE de Stripe

1. Ve a <https://dashboard.stripe.com>
2. Asegúrate de estar en modo **LIVE** (switch arriba a la derecha)
3. Ve a **Developers → API keys**
4. Copia:
  - Secret key (empieza con `sk_live_...`)
  - Publishable key (empieza con `pk_live_...`)

### Paso 2: Configurar Webhook

1. En Stripe Dashboard, ve a **Developers → Webhooks**
2. Click en **Add endpoint**
3. URL del endpoint: <https://inmova.app/api/stripe/webhook>
4. Selecciona estos eventos:
  - `payment_intent.succeeded`
  - `payment_intent.payment_failed`
  - `payment_intent.canceled`
  - `customer.subscription.created`
  - `customer.subscription.updated`
  - `customer.subscription.deleted`
  - `invoice.payment_succeeded`
  - `invoice.payment_failed`
5. Copia el **Signing secret** (empieza con `whsec_...`)

### Paso 3: Actualizar variables de entorno

Actualiza estas variables en tu archivo `.env`:

```
# Stripe LIVE - Reemplaza los valores placeholder
STRIPE_SECRET_KEY=sk_live_TU_CLAVE_SECRETA_AQUI
STRIPE_PUBLISHABLE_KEY=pk_live_TU_CLAVE_PUBLICA_AQUI
STRIPE_WEBHOOK_SECRET=whsec_TU_WEBHOOK_SECRET_AQUI
NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY=pk_live_TU_CLAVE_PUBLICA_AQUI
```

## Paso 4: Verificar configuración

```
cd /home/ubuntu/homming_vidaro/nextjs_space
yarn build
```

## 2 SENDGRID - EMAIL TRANSACCIONAL (CRÍTICO)

### Paso 1: Crear cuenta y obtener API Key

1. Ve a <https://sendgrid.com> y crea una cuenta
2. Ve a **Settings → API Keys**
3. Click en **Create API Key**
4. Nombre: INMOVA Production
5. Permisos: **Full Access**
6. Copia la API Key (solo se muestra una vez)

### Paso 2: Verificar dominio (Recomendado)

1. En SendGrid, ve a **Settings → Sender Authentication**
2. Click en **Authenticate Your Domain**
3. Sigue los pasos para añadir registros DNS a tu dominio inmova.app
4. Verifica que el dominio esté autenticado (mejora deliverability)

### Paso 3: Configurar remitente

1. Ve a **Settings → Sender Authentication**
2. Click en **Create New Sender**
3. Configura:
  - From Name: INMOVA
  - From Email: noreply@inmova.app
  - Reply To: soporte@inmova.app

### Paso 4: Actualizar variables de entorno

```
# SendGrid
SENDGRID_API_KEY=SG.TU_API_KEY_AQUI
SENDGRID_FROM_EMAIL=noreply@inmova.app
SENDGRID_FROM_NAME=INMOVA
```

### Paso 5: Probar envío

Después de actualizar `.env`, puedes probar el envío desde la app:

- Ve a cualquier sección que envíe emails (ej: crear pago)
- Los emails ahora se enviarán por SendGrid

## 3 BACKUPS AUTOMÁTICOS (CRÍTICO)

### Sistema implementado

Ya tienes endpoints API para backups:

- POST /api/backup/create - Crear backup manual
- GET /api/backup/list - Listar backups
- POST /api/backup/restore - Restaurar backup

### Opción A: Cron Jobs del Sistema (Recomendado)

Si tienes acceso SSH a tu servidor:

```
# Editar crontab
crontab -e

# Añadir estas líneas:
# Backup diario a las 3:00 AM
0 3 * * * curl -X POST https://inmova.app/api/backup/create \n -H "Authorization: Bearer ${CRON_SECRET}" \n -H "Content-Type: application/json" \n -d '{"tipo":"completo"}' >> /home/ubuntu/logs/backup-daily.log 2>&1

# Backup semanal los domingos a las 2:00 AM
0 2 * * 0 curl -X POST https://inmova.app/api/backup/create \n -H "Authorization: Bearer ${CRON_SECRET}" \n -H "Content-Type: application/json" \n -d '{"tipo":"completo"}' >> /home/ubuntu/logs/backup-weekly.log 2>&1
```

### Opción B: Servicio externo de Cron (EasyCron, cron-job.org)

1. Ve a <https://www.easycron.com> o <https://cron-job.org>
2. Crea una cuenta
3. Configura un cron job:
  - URL: `https://inmova.app/api/backup/create`
  - Método: POST
  - Headers:
 

```
Authorization: Bearer fa787a4d35969abc72ce755a816d1031445e846ca4fa52983569d97143ea2210
Content-Type: application/json
```
  - Body: `{"tipo":"completo"}`
  - Frecuencia: Diaria a las 3:00 AM

### Verificar backups

Puedes ver los backups desde:

- Admin → Backups en la app
- O consultando: GET /api/backup/list

## 4 SENTRY - ERROR TRACKING (CRÍTICO)

### Paso 1: Crear proyecto en Sentry

1. Ve a <https://sentry.io> y crea una cuenta
2. Click en **Create Project**

3. Selecciona **Next.js**
4. Nombre del proyecto: `inmova-production`
5. Copia el **DSN** (empieza con `https://...@sentry.io/...`)

## Paso 2: Actualizar variables de entorno

```
# Sentry
NEXT_PUBLIC_SENTRY_DSN=https://TU_DSN_AQUI@sentry.io/TU_PROJECT_ID
SENTRY_AUTH_TOKEN=TU_AUTH_TOKEN_AQUI
SENTRY_ORG=tu-organizacion
SENTRY_PROJECT=inmova-production
```

## Paso 3: Instalar dependencias

```
cd /home/ubuntu/homming_vidaro/nextjs_space
yarn add @sentry/nextjs
```

## Paso 4: Build y deploy

```
yarn build
```

Ahora Sentry capturará automáticamente todos los errores de:

- Frontend (errores de React)
- Backend (errores de API)
- Errores no capturados

## 5 UPTIMEROBOT - MONITORING 24/7 (CRÍTICO)

### Paso 1: Crear cuenta

1. Ve a <https://uptimerobot.com>
2. Crea una cuenta gratuita (50 monitores)

### Paso 2: Crear monitor principal

1. Click en **Add New Monitor**
2. Configuración:
  - Monitor Type: **HTTP(s)**
  - Friendly Name: `INMOVA Production`
  - URL: `https://inmova.app/api/health`
  - Monitoring Interval: **5 minutes**
  - Monitor Timeout: **30 seconds**

### Paso 3: Configurar alertas

1. En el monitor, click en **Alert Contacts**
2. Añade tu email para recibir notificaciones
3. Opcionalmente, añade:
  - Slack webhook
  - SMS (en planes de pago)
  - Telegram

## Paso 4: Monitores adicionales (Opcional)

Puedes crear monitores para:

- `https://inmova.app` (página principal)
  - `https://inmova.app/login` (login)
  - `https://inmova.app/api/dashboard` (API)
- 

## 6 CLOUDFLARE CDN (OPCIONAL - RECOMENDADO)

### Beneficios

- ⚡ Mejora velocidad de carga 40-60%
- 🛡️ Protección DDoS automática
- 📦 Cache automático de assets
- 🔒 SSL/TLS gratuito
- 🌎 CDN global

### Configuración

1. Ve a <https://cloudflare.com> y crea una cuenta
2. Click en **Add a Site**
3. Introduce tu dominio: `inmova.app`
4. Selecciona el plan **Free**
5. Cloudflare escaneará tus DNS records
6. Actualiza los nameservers en tu registrador de dominio:
  - Reemplaza los nameservers actuales por los de Cloudflare
  - Ejemplo: `ns1.cloudflare.com , ns2.cloudflare.com`
7. Espera propagación DNS (puede tardar hasta 24h)

### Configuración recomendada en Cloudflare

1. **SSL/TLS:** Full (strict)
  2. **Caching:** Standard
  3. **Speed → Optimization:**
    - Auto Minify: ON (HTML, CSS, JS)
    - Brotli: ON
  4. **Security → Settings:**
    - Security Level: Medium
    - Challenge Passage: 30 minutes
-

## 7 POSTGRESQL CLIENT (OPCIONAL)

### Para backups locales (Si tienes acceso SSH)

```
sudo apt-get update
sudo apt-get install postgresql-client

# Verificar instalación
psql --version
```

### Crear backup manual

```
# Extraer info de DATABASE_URL
export PGHOST=db-587683780.db003.hosteddb.reai.io
export PGPORT=5432
export PGDATABASE=587683780
export PGUSER=role_587683780
export PGPASSWORD=5kWw7vKJBDp9ZA2Jfkt5BdWrAjR0XDe5

# Crear backup
pg_dump > backup_$(date +%Y%m%d_%H%M%S).sql

# Comprimir
gzip backup_*.sql
```

## ✓ CHECKLIST FINAL

Antes de considerar la configuración completa:

### Stripe

- [ ] Claves LIVE configuradas en `.env`
- [ ] Webhook configurado y funcionando
- [ ] Modo LIVE activado en dashboard
- [ ] Prueba de pago realizada

### SendGrid

- [ ] API Key configurada
- [ ] Dominio verificado (opcional pero recomendado)
- [ ] Remitente configurado
- [ ] Email de prueba enviado

### Backups

- [ ] Cron job configurado (sistema o externo)
- [ ] Backup manual creado y verificado
- [ ] Logs de backup monitoreados

### Sentry

- [ ] Proyecto creado
- [ ] DSN configurado en `.env`

- [ ] Build realizado con integración
- [ ] Errores capturados correctamente

## **UptimeRobot**

- [ ] Monitor principal configurado
- [ ] Alertas de email configuradas
- [ ] Health endpoint respondiendo

## **Cloudflare (Opcional)**

- [ ] Cuenta creada
- [ ] Nameservers actualizados
- [ ] DNS propagado
- [ ] SSL funcionando

## **PostgreSQL Client (Opcional)**

- [ ] Cliente instalado
  - [ ] Backup manual probado
- 

## **SOPORTE**

Si encuentras problemas durante la configuración:

1. **Stripe**: <https://support.stripe.com>
  2. **SendGrid**: <https://support.sendgrid.com>
  3. **Sentry**: <https://sentry.io/support>
  4. **UptimeRobot**: [support@uptimerobot.com](mailto:support@uptimerobot.com)
  5. **Cloudflare**: <https://support.cloudflare.com>
- 

## **MONITOREO POST-CONFIGURACIÓN**

### **Dashboards a revisar diariamente:**

1. **Stripe Dashboard**: Pagos, disputas, saldos
2. **SendGrid Dashboard**: Deliverability, bounces, spam reports
3. **Sentry Dashboard**: Errores, performance
4. **UptimeRobot**: Uptime %, response times
5. **Cloudflare Analytics**: Traffic, threats blocked

### **Métricas clave:**

- **Uptime objetivo**: > 99.9%
  - **Response time objetivo**: < 500ms
  - **Error rate objetivo**: < 0.1%
  - **Email deliverability objetivo**: > 98%
  - **Stripe success rate objetivo**: > 95%
-

## ACTUALIZACIONES

Recuerda rebuild y redeploy después de cambiar variables de entorno:

```
cd /home/ubuntu/homming_vidaro/nextjs_space  
yarn build  
# Luego redeploy según tu método de deployment
```

---

**Última actualización:** Diciembre 2024

**Versión:** 1.0