Guía de Integraciones de Terceros - INMOVA

🔗 Tabla de Contenidos

💳 Integraciones de Pago

Stripe (Recomendado)

Estado: ✅ Totalmente implementado y funcional

Funcionalidades:
- Pagos únicos
- Suscripciones recurrentes (rentas mensuales)
- Gestión de métodos de pago
- Webhooks para sincronización automática
- Recibos y facturas

Configuración

1. **Crear cuenta Stripe:**
   - Accede a https://dashboard.stripe.com/register
   - Completa el proceso de registro
   - Verifica tu cuenta (requerido para modo producción)

2. **Obtener claves API:**
   ```
   Dashboard > Developers > API keys
   ```

Para Desarrollo:
- Secret key: sk_test_...
- Publishable key: pk_test_...

Para Producción:
- Secret key: sk_live_...

- Publishable key: pk_live_...
```

1. **Configurar variables de entorno:**
   bash
   STRIPE_SECRET_KEY="sk_test_xxxxxxxxxxxxxxxxxxx"
   STRIPE_PUBLISHABLE_KEY="pk_test_xxxxxxxxxxxxxxxxxxx"
   STRIPE_WEBHOOK_SECRET="whsec_xxxxxxxxxxxxxxxxxxx"

2. **Configurar Webhook:**
   ```
   Dashboard > Developers > Webhooks > Add endpoint

URL del endpoint: https://tu-dominio.com/api/stripe/webhook

Eventos a escuchar:
- 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
```

1. **Testing:**
   - Tarjeta de prueba exitosa: `4242 4242 4242 4242`
   - Tarjeta de prueba fallida: `4000 0000 0000 0002`
   - Cualquier CVV de 3 dígitos
   - Cualquier fecha futura

## Uso en la Aplicación

**Portal del Inquilino:**

```
// Ver pagos pendientes
GET /api/portal-inquilino/payments

// Crear intención de pago
POST /api/stripe/create-payment-intent
Body: { paymentId: "payment_id" }

// El frontend usa Stripe Elements para el pago
```

Administración:

```
// Crear suscripción recurrente
POST /api/stripe/create-subscription
Body: { contractId: "contract_id" }

// Cancelar suscripción
POST /api/stripe/cancel-subscription
Body: { subscriptionId: "sub_id", cancelAtPeriodEnd: false }
```
```

Dashboard de Pagos:
- Acceso: `/pagos/components/PaymentsDashboard`
- Estadísticas: `/api/stripe/stats`
- Historial: `/api/stripe/payments`

🏦 Integraciones Bancarias

Bankinter Open Banking (PSD2)

Estado: 🟡 Preparado (Demo Mode por defecto)

Funcionalidades:
- Verificación de ingresos de inquilinos
- Conexión de cuentas bancarias
- Sincronización de transacciones
- Conciliación automática de pagos
- Iniciación de pagos (futuro)

Configuración

📄 **Consulta la guía completa:** `/DOCS/INTEGRACIONES_BANKINTER.md`

Pasos rápidos:

1. **Solicitar acceso a Bankinter:**
   - Contactar: openbanking@bankinter.com
   - Proporcionar información de la empresa
   - Esperar aprobación (1-2 semanas)

2. **Obtener credenciales:**
   `- Client ID`
   `- Client Secret`
   `- URLs de API (sandbox/producción)`

3. **Configurar `.env`:**
   `bash`
   `BANKINTER_CLIENT_ID="tu_client_id"`
   `BANKINTER_CLIENT_SECRET="tu_client_secret"`
   `BANKINTER_API_BASE_URL="https://apis.bankinter.com"`
   `BANKINTER_REDIRECT_URI="https://inmova.app/api/open-banking/bankinter/callback"`

4. **Verificar implementación:**
   - Servicio: `/lib/bankinter-integration-service.ts`
   - Wrapper: `/lib/open-banking-service.ts`
   - UI: `/app/open-banking/page.tsx`

Demo Mode

Si NO configuras las credenciales, el sistema funciona en modo demo:
- Genera números de cuenta ficticios
- Simula transacciones aleatorias
- Calcula ingresos estimados
- Permite probar el flujo completo

📈 Integraciones Contables

1. Zucchetti

Estado: 🟡 Preparado (Demo Mode)

Funcionalidades:
- Sincronización de clientes (inquilinos)
- Generación de facturas
- Registro de pagos
- Conciliación contable

Activación

📄 **Consulta:** `/DOCS/ZUCCHETTI_INTEGRATION.md` (si existe)

```
# Variables de entorno
ZUCCHETTI_CLIENT_ID="tu_client_id"
ZUCCHETTI_CLIENT_SECRET="tu_client_secret"
ZUCCHETTI_API_BASE_URL="https://api.zucchetti.com"
ZUCCHETTI_REDIRECT_URI="https://inmova.app/api/accounting/zucchetti/callback"
```

Código: `/lib/zucchetti-integration-service.ts`

2. ContaSimple

Estado: 🟡 Preparado

```
CONTASIMPLE_AUTH_KEY="tu_api_key"
CONTASIMPLE_API_URL="https://api.contasimple.com/v2"
```

UI: `/app/contabilidad/page.tsx` (tarjeta ContaSimple)

3. Sage

Estado: 🟡 Preparado

```
SAGE_CLIENT_ID=""
SAGE_CLIENT_SECRET=""
SAGE_REDIRECT_URI="https://inmova.app/api/accounting/sage/callback"
```

4. Holded

Estado: 🟡 Preparado

```
HOLDED_API_KEY="tu_api_key"
```

5. A3 Software

Estado: 🟡 Preparado

```
A3_API_KEY="tu_api_key"
A3_API_URL="https://api.a3software.com"
```

6. Alegra

Estado: 🟡 Preparado

```
ALEGRA_EMAIL="tu_email@empresa.com"
ALEGRA_TOKEN="tu_token"
```

Modo Demo para Todas

Cuando NO están configuradas:
- Simulan sincronizaciones exitosas
- Generan IDs de factura ficticios
- Registran logs en consola
- Permiten probar flujos

Activación Real:
1. Obtener credenciales del proveedor
2. Configurar `.env`
3. Reiniciar servidor
4. Probar conexión desde `/contabilidad`

✍️ Firma Digital

DocuSign

Estado: 🟡 Preparado (Demo Mode)

Funcionalidades:
- Envío de contratos para firma
- Seguimiento de estado
- Notificaciones automáticas
- Almacenamiento de documentos firmados

Configuración Completa

📄 **Guía detallada:** `/INTEGRACION_DOCUSIGN_VIDARO.md`

Resumen:

1. **Crear cuenta DocuSign:**
   - Sandbox: https://demo.docusign.net
   - Producción: https://www.docusign.com

2. **Crear aplicación OAuth:**
   - DocuSign Admin > Integrations > Apps and Keys
   - Anotar Integration Key, User ID, Account ID

3. **Generar par de claves RSA:**
   `bash`
   `openssl genrsa -out private.key 2048`
   `openssl rsa -in private.key -pubout -out public.key`

4. **Configurar `.env`:**
   `bash`
   `DOCUSIGN_INTEGRATION_KEY="tu_integration_key"`
   `DOCUSIGN_USER_ID="tu_user_id"`
   `DOCUSIGN_ACCOUNT_ID="tu_account_id"`
   `DOCUSIGN_PRIVATE_KEY="-----BEGIN RSA PRIVATE KEY-----\nMIIE...\n-----END RSA PRIVATE KEY-----"`
   `DOCUSIGN_BASE_URL="https://demo.docusign.net/restapi"`

5. **Configurar Webhook (opcional):**
   `URL: https://inmova.app/api/digital-signature/webhook`
   `Eventos: sent, delivered, signed, declined, voided`

Uso en INMOVA

Página de gestión: `/firma-digital`

```
// Crear solicitud de firma
POST /api/digital-signature
Body: {
  titulo: "Contrato de Arrendamiento",
  contractId: "contract_id",
  firmantes: [
    { email: "inquilino@email.com", nombre: "Juan Pérez", rol: "inquilino" }
  ],
  documentoBase64: "JVBERi0x..." // PDF en base64
}

// Firmar documento
POST /api/digital-signature/[id]/sign
Body: {
  firmanteId: "firmante_id",
  nombreCompleto: "Juan Pérez",
  dni: "12345678A",
  ubicacion: "Madrid, España"
}
```

Demo Mode:
- Genera tokens de firma ficticios
- Simula envío de emails
- Permite probar flujo completo

📊 Análisis y Métricas

Google Analytics 4

Estado: ✅ Funcional (requiere ID)

Configuración

1. **Crear propiedad GA4:**
   - https://analytics.google.com
   - Admin > Create Property
   - Seleccionar “Web”
   - Obtener Measurement ID (G-XXXXXXXXXX)

2. **Configurar `.env`:**
   `bash`
   `NEXT_PUBLIC_GA_MEASUREMENT_ID="G-XXXXXXXXXX"`

3. **Eventos personalizados:**
   - Tracking automático de rutas
   - Eventos de usuario (login, signup)
   - Acciones clave (crear contrato, pago completado)

Código: Script inyectado en `<head>` via `app/layout.tsx`

Sentry (Monitoreo de Errores)

Estado: ✅ Funcional (requiere DSN)

Configuración

1. **Crear proyecto Sentry:**
   - https://sentry.io/signup
   - Create Project > Next.js
   - Copiar DSN

2. **Configurar `.env`:**
   `bash`
   `NEXT_PUBLIC_SENTRY_DSN="https://xxxxx@xxxxx.ingest.sentry.io/xxxxx"`
   `SENTRY_ORG="tu-organizacion"`
   `SENTRY_PROJECT="inmova"`

3. **Funcionalidades:**
   - Captura automática de errores
   - Source maps para debugging
   - Alertas en tiempo real
   - Performance monitoring

🔐 Autenticación Social

Google SSO

Estado: ✅ Funcional (requiere Client ID)

Configuración

1. **Google Cloud Console:**
   - https://console.cloud.google.com
   - APIs & Services > Credentials
   - Create OAuth 2.0 Client ID
   - Application type: Web application
   - Authorized redirect URIs:
     - `http://localhost:3000/api/auth/callback/google`
     - `https://inmova.app/api/auth/callback/google`

2. **Configurar `.env`:**
   `bash`

```
   GOOGLE_CLIENT_ID="xxxxxx.apps.googleusercontent.com"
   GOOGLE_CLIENT_SECRET="GOCSPX-xxxxxx"
```

3. **Uso:**
   - Botón “Iniciar sesión con Google” en `/login`
   - Automáticamente crea usuario si no existe

📄 **Guía completa:** Usa `get_third_party_integration_guidelines` con `google_sso`

📧 Notificaciones y Comunicación

SendGrid (Emails Transaccionales)

Estado: ✅ Funcional (requiere API Key)

Configuración

1. **Crear cuenta SendGrid:**
   - https://signup.sendgrid.com
   - Verificar email y dominio

2. **Crear API Key:**
   - Settings > API Keys > Create API Key
   - Permisos: Full Access

3. **Configurar `.env`:**
   ```
   bash
   SENDGRID_API_KEY="SG.xxxxxxxxxxxxxxxxxxx"
   SENDGRID_FROM_EMAIL="noreply@inmova.app"
   SENDGRID_FROM_NAME="INMOVA"
   ```

4. **Verificar dominio (Producción):**
   - Settings > Sender Authentication
   - Domain Authentication
   - Añadir registros DNS (CNAME, SPF, DKIM)

Tipos de Emails

- Confirmación de registro
- Recordatorios de pago
- Notificaciones de mantenimiento
- Recibos de pago (adjunta PDF)
- Alertas de sistema

Código:
- Configuración: `/lib/email-config.ts`
- Plantillas: `/lib/email-templates.ts`
- Uso: `await sendEmail({ to, subject, html, attachments })`

SMS (Simulado)

Estado: 🟡 Demo Mode

Servicio: `/lib/sms-service.ts`

Para integración real, agregar proveedor como Twilio:

```
TWILIO_ACCOUNT_SID=""
TWILIO_AUTH_TOKEN=""
TWILIO_PHONE_NUMBER="+34XXXXXXXXX"
```

🗄️ Almacenamiento

AWS S3

Estado: ✅ Totalmente funcional

Configuración

Las credenciales AWS se configuran automáticamente en el servidor de producción. Solo necesitas:

```
AWS_BUCKET_NAME="inmova-production-files"
AWS_FOLDER_PREFIX="production/"
```

Funcionalidades

- Subida de archivos (imágenes, PDFs, documentos)
- Organización por carpetas
- URLs firmadas para acceso privado
- URLs públicas para recursos compartidos
- Eliminación segura

Uso

```
import { uploadFile, getFileUrl, deleteFile } from '@/lib/s3';

// Subir archivo
const s3Key = await uploadFile(buffer, fileName, isPublic);

// Obtener URL (firmada si es privado)
const url = await getFileUrl(s3Key, isPublic);

// Eliminar
await deleteFile(s3Key);
```

Archivos Públicos: Fotos de propiedades, logos, imágenes de galerías
Archivos Privados: Contratos, DNIs, documentos de inquilinos, recibos

---

🐛 Troubleshooting

Stripe

Problema: “No such customer”

```
// Solución: Verificar que el inquilino tenga stripeCustomerId
const stripeCustomer = await getOrCreateStripeCustomer(
  tenantId,
  email,
  name
);
```

Problema: Webhook no se recibe
- Verificar URL del webhook en Dashboard
- Comprobar que el secreto del webhook coincida
- Revisar logs en Stripe Dashboard > Developers > Webhooks

SendGrid

Problema: Emails no se envían
- Verificar API key válida
- Comprobar dominio autenticado
- Revisar Activity en SendGrid Dashboard

Problema: Emails van a spam
- Autenticar dominio con SPF/DKIM
- Calentar IP (enviar volumen gradualmente)
- Evitar contenido spammy

Google Analytics

Problema: No aparecen datos
- Verificar que Measurement ID es correcto
- Comprobar que no hay bloqueadores de anuncios
- Esperar 24-48h para datos iniciales
- Usar modo Debug: `gtag('config', 'G-XXX', { 'debug_mode': true })`

AWS S3

Problema: “Access Denied”
- Verificar credenciales AWS
- Comprobar políticas del bucket
- Revisar IAM roles/permissions

Problema: Archivos no se muestran
- Si es privado, generar URL firmada
- Verificar que el bucket y prefix son correctos
- Comprobar que el archivo existe: `aws s3 ls s3://bucket/prefix/`

Integraciones en Demo Mode

Síntoma: Dice “Demo” o “No configurado”

Solución:
1. Verificar variables de entorno configuradas
2. Reiniciar servidor Next.js: `yarn dev`
3. Comprobar logs en consola del servidor
4. Usar `isConfigured()` para validar

📞 Soporte

Para ayuda con integraciones:

- 📧 Email Técnico: dev@inmova.com
- 📧 Soporte General: soporte@inmova.com
- 🌐 Documentación: Carpeta `/DOCS/` en el proyecto

Documentos Relacionados:
- `/README.md` - Configuración general
- `/DOCS/INTEGRACIONES_BANKINTER.md` - Open Banking
- `/INTEGRACION_DOCUSIGN_VIDARO.md` - Firma Digital
- `/MEJORAS_SUPERADMIN.md` - Panel de administración
