📝 Guía de Integración DocuSign para Vidaro

📋 Resumen Ejecutivo

Esta guía detalla los pasos necesarios para activar la integración real de DocuSign en la plataforma INMOVA donde **Vidaro tiene cuenta activa**.

Estado Actual: ✅ Código preparado | ⏳ Credenciales pendientes | 🔧 Activación lista

---

🎯 Paso 1: Obtener Credenciales de DocuSign

1.1 Acceder al Portal de Desarrolladores

1. Ve a: **https://developers.docusign.com/**
2. Inicia sesión con las credenciales de **Vidaro**
3. Si es primera vez, acepta los términos de desarrollador

1.2 Crear Aplicación de Integración

1. En el dashboard, navega a **“Apps and Keys”**
2. Haz clic en **“Add App and Integration Key”**
3. Configuración de la app:
   - **App Name:** `INMOVA - Vidaro`
   - **Description:** `Plataforma de gestión inmobiliaria con firma digital integrada`
   - **Integration Type:** Service Integration
4. Guarda la aplicación

1.3 Obtener Integration Key (Client ID)

Después de crear la app, verás:

```
Integration Key: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

✅ **Copia este valor** - lo necesitarás para `DOCUSIGN_INTEGRATION_KEY`

1.4 Obtener User ID y Account ID

1. En la misma página, busca la sección **“Service Integration”**
2. Encontrarás:
   `API Username (User ID): xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`
   `Account ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`
3. ✅ **Copia estos valores**

1.5 Generar Par de Claves RSA

Opción A: Desde la interfaz de DocuSign (Recomendado)

1. En la configuración de tu app, busca **“RSA Keypairs”**
2. Haz clic en **“Generate RSA”**
3. DocuSign te mostrará la **clave privada** en pantalla

4. ⚠️ **IMPORTANTE:** Copia y guarda inmediatamente la clave privada
5. La clave pública se guarda automáticamente en DocuSign

Opción B: Generar localmente

```
# Generar clave privada RSA de 2048 bits
openssl genrsa -out docusign_private_key.pem 2048

# Generar clave pública correspondiente
openssl rsa -in docusign_private_key.pem -pubout -out docusign_public_key.pem

# Ver la clave privada (para copiar)
cat docusign_private_key.pem

# Ver la clave pública (para subir a DocuSign)
cat docusign_public_key.pem
```

Si usas Opción B:
- Copia el contenido de `docusign_public_key.pem`
- En DocuSign, ve a **“RSA Keypairs” > “Add RSA Keypair”**
- Pega la clave pública y guarda

1.6 Configurar Redirect URIs

En la sección **“Redirect URIs”** de tu app:

```
Producción:
https://inmova.app/api/digital-signature/callback

Desarrollo (opcional):
http://localhost:3000/api/digital-signature/callback
```

1.7 Obtener Consent de Usuario

Para que la app pueda actuar en nombre de Vidaro:

1. **Construye la URL de consent:**

```
https://account-d.docusign.com/oauth/auth?response_type=code&scope=signature%20impersonation&client_id=TU_INTEGRATION_KEY&redirect_uri=https://inmova.app/api/digital-signature/callback
```

1. **Reemplaza** `TU_INTEGRATION_KEY` con tu Integration Key real
2. **Abre esta URL en el navegador**
3. Inicia sesión con la cuenta de **Vidaro**
4. Haz clic en **“Allow Access”** para autorizar
5. Serás redirigido (puede dar error 404, es normal por ahora)

✅ **La autorización queda registrada en DocuSign**

🔧 Paso 2: Configurar Variables de Entorno

2.1 Actualizar el archivo .env

En el archivo `/home/ubuntu/homming_vidaro/nextjs_space/.env`, las variables ya están añadidas. **Reemplaza los valores placeholder:**

```
# ================================================================================
# DOCUSIGN - Configuración para Vidaro
# ================================================================================
DOCUSIGN_INTEGRATION_KEY=tu_integration_key_real_aqui
DOCUSIGN_USER_ID=tu_user_id_real_aqui
DOCUSIGN_ACCOUNT_ID=tu_account_id_real_aqui
DOCUSIGN_PRIVATE_KEY="-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA...tu_clave_privada_completa...
-----END RSA PRIVATE KEY-----"
DOCUSIGN_BASE_PATH=https://demo.docusign.net/restapi
```

2.2 Formato Correcto de la Clave Privada

⚠️ **MUY IMPORTANTE:** La clave privada debe incluir:
- Las líneas `-----BEGIN RSA PRIVATE KEY-----` y `-----END RSA PRIVATE KEY-----`
- Todo el contenido debe estar en **una sola línea** con `\n` para saltos de línea, O
- Estar entrecomillada con comillas dobles y mantener los saltos de línea reales

Ejemplo correcto (opción 1 - una línea):

```
DOCUSIGN_PRIVATE_KEY="-----BEGIN RSA PRIVATE KEY-----\nMIIEowIBAAKCAQEA...\n-----END
RSA PRIVATE KEY-----"
```

Ejemplo correcto (opción 2 - multilinea):

```
DOCUSIGN_PRIVATE_KEY="-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAxxx...
...más líneas...
-----END RSA PRIVATE KEY-----"
```

2.3 Elegir Entorno

Para pruebas (Sandbox):

```
DOCUSIGN_BASE_PATH=https://demo.docusign.net/restapi
```

Para producción:

```
DOCUSIGN_BASE_PATH=https://na1.docusign.net/restapi
```

(Verifica la región correcta de tu cuenta Vidaro: na1, na2, na3, eu, etc.)

⚙️ Paso 3: Activar la Integración Real

⚠️ NOTA IMPORTANTE

Debido a incompatibilidades del paquete oficial `docusign-esign` con Next.js/Webpack, la integración requiere implementación personalizada usando la API REST de DocuSign directamente.

Opciones disponibles:
1. **Usar API REST directamente** (Recomendado) - Sin dependencias externas problemáticas
2. **Implementación en API Route separada** - Aislar DocuSign del bundle principal
3. **Microservicio externo** - Servicio Node.js independiente que maneja DocuSign

3.1 Opción A: Usar API REST Directamente (Recomendado)

Esta opción no requiere instalar `docusign-esign` y es compatible con Next.js.

3.2 Actualizar el Servicio de Firma Digital con API REST

El servicio ya está preparado en `/home/ubuntu/homming_vidaro/nextjs_space/lib/digital-signature-service.ts`

Reemplaza la función `enviarDocuSignEnvelope` con esta implementación usando API REST directa:

```
// No se requieren imports adicionales - solo fetch nativo

// Función para obtener token JWT de DocuSign
async function getDocuSignJWT() {
  const jwtLifeSec = 3600; // 1 hora
  const scopes = ['signature', 'impersonation'];

  const oAuth = docusign.ApiClient.OAuth;
  const results = await oAuth.getAccessToken(
    DOCUSIGN_INTEGRATION_KEY!,
    DOCUSIGN_USER_ID!,
    scopes,
    DOCUSIGN_PRIVATE_KEY!,
    jwtLifeSec
  );

  return results.accessToken;
}

// Implementación real de enviarDocuSignEnvelope
async function enviarDocuSignEnvelope(params: {
  titulo: string;
  documentUrl: string;
  mensaje?: string;
  firmantes: FirmanteData[];
  diasExpiracion: number;
}) {
  try {
    // 1. Obtener token de acceso
    const accessToken = await getDocuSignJWT();

    // 2. Configurar cliente API
    const apiClient = new docusign.ApiClient();
    apiClient.setBasePath(DOCUSIGN_BASE_PATH!);
    apiClient.addDefaultHeader('Authorization', `Bearer ${accessToken}`);

    // 3. Descargar el documento del URL
    const response = await fetch(params.documentUrl);
    const documentBuffer = await response.arrayBuffer();
    const documentBase64 = Buffer.from(documentBuffer).toString('base64');

    // 4. Crear envelope definition
    const envelopeDefinition: docusign.EnvelopeDefinition = {
      emailSubject: params.titulo,
      emailBlurb: params.mensaje || 'Por favor, firme este documento',
      documents: [{
        documentBase64,
        name: params.titulo,
        fileExtension: 'pdf',
        documentId: '1'
      }],
      recipients: {
        signers: params.firmantes.map((firmante, index) => ({
          email: firmante.email,
          name: firmante.nombre,
          recipientId: String(index + 1),
          routingOrder: String(index + 1),
          tabs: {
            signHereTabs: [{
              anchorString: '/sn1/',
              anchorUnits: 'pixels',
              anchorXOffset: '20',
```

```
              anchorYOffset: '10'
            }]
          }
        }))
      },
      status: 'sent',
      notification: {
        useAccountDefaults: false,
        reminders: {
          reminderEnabled: true,
          reminderDelay: '2',
          reminderFrequency: '2'
        },
        expirations: {
          expireEnabled: true,
          expireAfter: String(params.diasExpiracion),
          expireWarn: '2'
        }
      }
    };

    // 5. Enviar envelope
    const envelopesApi = new docusign.EnvelopesApi(apiClient);
    const results = await envelopesApi.createEnvelope(
      DOCUSIGN_ACCOUNT_ID!,
      { envelopeDefinition }
    );

    logger.info('✅ [DocuSign] Envelope enviado correctamente', {
      envelopeId: results.envelopeId
    });

    return {
      envelopeId: results.envelopeId!,
      status: results.status!,
      message: 'Documento enviado via DocuSign'
    };

  } catch (error) {
    logError('❌ [DocuSign] Error al enviar envelope', error as Error);
    throw error;
  }
}
```

3.3 Reiniciar el Servidor

```
# Si usas desarrollo local
yarn dev

# Si ya está desplegado
# El servidor se reiniciará automáticamente al detectar cambios
```

✅ Paso 4: Probar la Integración

4.1 Verificar Detección Automática

El sistema detecta automáticamente si DocuSign está configurado:

```
// En digital-signature-service.ts
const isDocuSignConfigured = !!(DOCUSIGN_INTEGRATION_KEY && DOCUSIGN_USER_ID && DOC-
USIGN_ACCOUNT_ID);

export function getActiveProvider(): 'docusign' | 'signaturit' | 'demo' {
  if (isDocuSignConfigured) return 'docusign';
  if (isSignaturitConfigured) return 'signaturit';
  return 'demo';
}
```

4.2 Probar desde la Interfaz

1. Inicia sesión como administrador de Vidaro:
   - Email: `admin@vidaro.es`
   - Password: `Inmova2025!`

2. Ve a **Firma Digital** en el menú lateral

3. Crea una nueva solicitud de firma:
   - Selecciona un contrato o documento
   - Añade firmantes con sus emails
   - Envía el documento

4. Verifica:
   - ✅ El documento se envía a DocuSign (no modo demo)
   - ✅ Los firmantes reciben el email de DocuSign
   - ✅ El estado se actualiza en la plataforma

4.3 Monitorear Logs

```
# Ver logs del servidor
tail -f /home/ubuntu/homming_vidaro/nextjs_space/logs/combined.log | grep DocuSign
```

Busca mensajes como:

```
✅ [DocuSign] Envelope enviado correctamente { envelopeId: 'xxx-xxx-xxx' }
```

4.4 Verificar en DocuSign

1. Inicia sesión en DocuSign con la cuenta de Vidaro
2. Ve a **“Manage” > “Sent”**
3. Deberías ver los documentos enviados desde INMOVA

🔐 Seguridad y Mejores Prácticas

✅ Checklist de Seguridad

- [ ] La clave privada RSA está almacenada de forma segura
- [ ] Las variables de entorno NO están en el repositorio Git
- [ ] Se usa HTTPS en producción (ya configurado: `https://inmova.app`)
- [ ] Los logs no exponen credenciales sensibles

- [ ] Se configuró el consent de usuario correctamente
- [ ] Se probó la revocación de acceso

🔒 Proteger la Clave Privada

```
# Verificar que .env está en .gitignore
grep -q ".env" /home/ubuntu/homming_vidaro/nextjs_space/.gitignore && echo "✅ .env
protegido" || echo "⚠️ Añadir .env a .gitignore"

# Establecer permisos restrictivos
chmod 600 /home/ubuntu/homming_vidaro/nextjs_space/.env
```

📊 Monitoreo de Uso

DocuSign tiene límites de API. Monitorea el uso en:
- **Dashboard de DocuSign** > **Settings** > **API and Keys** > **Usage**

🐛 Troubleshooting

Problema: “Invalid JWT token”

Causa: Token JWT mal formado o expirado

Solución:
1. Verifica que la clave privada esté completa
2. Asegúrate de que no hay espacios extra o caracteres extraños
3. Regenera el consent de usuario (Paso 1.7)

Problema: “USER_AUTHENTICATION_FAILED”

Causa: El consent de usuario no se ha completado

Solución:
1. Ve a la URL de consent (Paso 1.7)
2. Autoriza la aplicación nuevamente
3. Espera 5 minutos para que se propague

Problema: “ACCOUNT_LACKS_PERMISSIONS”

Causa: La cuenta de Vidaro no tiene permisos de API

Solución:
1. Contacta con soporte de DocuSign
2. Solicita activación de permisos de API para la cuenta

Problema: Documentos no se envían

Diagnóstico:

```
# Ver logs detallados
tail -f /home/ubuntu/homming_vidaro/nextjs_space/logs/error.log | grep -A 5 "DocuSign"
```

Soluciones comunes:
- Verifica que `DOCUSIGN_BASE_PATH` sea correcta

- Confirma que el `ACCOUNT_ID` corresponde a la cuenta de Vidaro
- Revisa que los emails de firmantes sean válidos

Problema: “Cannot find module ‘docusign-esign’“

Solución:

```
cd /home/ubuntu/homming_vidaro/nextjs_space
yarn add docusign-esign jsonwebtoken
yarn prisma generate
```

📚 Recursos Adicionales

Documentación Oficial

- **DocuSign REST API:** https://developers.docusign.com/docs/esign-rest-api/
- **SDK de Node.js:** https://github.com/docusign/docusign-esign-node-client
- **JWT Authorization:** https://developers.docusign.com/platform/auth/jwt/

Ejemplos de Código

- **Código de ejemplo oficial:** https://github.com/docusign/code-examples-node
- **Quick Start Guide:** https://developers.docusign.com/docs/esign-rest-api/esign101/

Soporte

- **Stack Overflow:** Tag `docusignapi`
- **Community Forum:** https://community.docusign.com/
- **Soporte de INMOVA:** `soporte@inmova.com`

📝 Checklist Final de Activación

Pre-Activación

- [ ] Cuenta de DocuSign de Vidaro está activa
- [ ] App creada en el portal de desarrolladores
- [ ] Integration Key obtenido
- [ ] User ID y Account ID obtenidos
- [ ] Par de claves RSA generado
- [ ] Consent de usuario completado

Configuración

- [ ] Variables de entorno configuradas en `.env`
- [ ] Clave privada con formato correcto
- [ ] Dependencias instaladas (`docusign-esign` , `jsonwebtoken`)
- [ ] Función `enviarDocuSignEnvelope` actualizada con código real
- [ ] Servidor reiniciado

Pruebas

- [ ] Detección automática de DocuSign funcionando
- [ ] Envío de documento de prueba exitoso
- [ ] Email recibido por firmante de prueba
- [ ] Firma completada y estado actualizado
- [ ] Verificación en dashboard de DocuSign

Producción

- [ ] Probado en entorno de desarrollo (demo)
- [ ] `DOCUSIGN_BASE_PATH` cambiado a producción
- [ ] Monitoreo de logs configurado
- [ ] Plan de contingencia definido
- [ ] Documentación actualizada para el equipo

🎯 Próximos Pasos

1. **Completar la integración de webhooks** para recibir notificaciones de DocuSign
2. **Implementar firma en lote** para múltiples documentos
3. **Añadir plantillas de DocuSign** para contratos estándar
4. **Configurar recordatorios automáticos** para firmantes
5. **Integrar con el sistema de auditoría** de INMOVA

👥 Contacto y Soporte

Para dudas sobre esta integración:
- **Documentación técnica:** Este archivo
- **Soporte INMOVA:** `soporte@inmova.com`
- **Emergencias:** Contactar al equipo de desarrollo

Para cuentas de DocuSign:
- **Soporte DocuSign:** https://support.docusign.com/
- **Administrador de cuenta Vidaro:** (contacto interno de Vidaro)

Última actualización: Diciembre 2025
Versión del documento: 1.0
Preparado para: Vidaro / INMOVA