Integración ContaSimple - Completada ✅

Estado: ACTIVA Y FUNCIONAL

La integración con ContaSimple ha sido completamente implementada y está lista para usar en producción.

📋 Resumen de Implementación

1. Configuración de Credenciales

Variables de Entorno Configuradas:

```
CONTASIMPLE_AUTH_KEY=4aed1d54316045bd8e8819310a2abb94
CONTASIMPLE_API_URL=https://api.contasimple.com/api/v2
```

2. Método de Autenticación

ContaSimple utiliza un sistema OAuth2 adaptado:

1. **POST** `/oauth/token` con los siguientes parámetros en `form-data`:
   - `grant_type`: "authentication_key"
   - `key`: La clave de autorización
2. **Respuesta**: Obtiene un `access_token` válido por 1 hora
3. **Uso**: El token se incluye en todas las llamadas posteriores:
   `Authorization: Bearer {access_token}`

🛠️ Componentes Implementados

A. Servicio de Integración

Archivo: `/lib/contasimple-integration-service.ts`

Métodos Principales:

Autenticación

- `authenticate()`: Obtiene un access_token usando la clave de autorización
- `isTokenValid()`: Verifica si el token actual es válido
- `ensureValidToken()`: Renueva el token automáticamente si es necesario

Gestión de Clientes

- `createCustomer(customer)`: Crea un cliente en ContaSimple
- `getCustomer(customerId)`: Obtiene información de un cliente
- `updateCustomer(customerId, data)`: Actualiza un cliente
- `syncTenantToCustomer(tenant)`: Sincroniza un inquilino de INMOVA como cliente

Gestión de Facturas

- `createInvoice(invoice)` : Crea una factura
- `getInvoice(invoiceId)` : Obtiene información de una factura
- `sendInvoice(invoiceId, email)` : Envía una factura por email
- `cancelInvoice(invoiceId)` : Cancela una factura
- `createInvoiceFromContract(contract, customerId)` : Crea una factura desde un contrato de INMOVA

Gestión de Pagos

- `registerPayment(payment)` : Registra un pago en ContaSimple
- `syncPaymentToContaSimple(payment, invoiceId)` : Sincroniza un pago de INMOVA

Gestión de Gastos

- `createExpense(expense)` : Registra un gasto
- `getExpenses(filters)` : Obtiene gastos filtrados
- `syncExpenseToContaSimple(expense)` : Sincroniza un gasto de INMOVA

B. Endpoints API

1. `/api/accounting/contasimple/status` (GET)

Verifica el estado de la configuración de ContaSimple

Respuesta:

```
{
  "configured": true,
  "provider": "ContaSimple",
  "status": "ready",
  "features": [
    "Sincronización de clientes",
    "Generación de facturas",
    "Registro de pagos",
    "Registro de gastos",
    "Contabilidad simplificada"
  ]
}
```

2. `/api/accounting/contasimple/test-connection` (GET)

Prueba la conexión con ContaSimple y obtiene un token de acceso

Respuesta Exitosa:

```
{
  "success": true,
  "message": "Conexión exitosa con ContaSimple",
  "data": {
    "tokenType": "Bearer",
    "expiresIn": 3600,
    "authenticated": true
  }
}
```

3. `/api/accounting/contasimple/customers` (POST)

Sincroniza un inquilino como cliente en ContaSimple

Request Body:

```
{
  "tenantId": "clxyz123"
}
```

Respuesta:

```
{
  "success": true,
  "message": "Cliente sincronizado exitosamente: Juan Pérez",
  "data": {
    "id": "customer_123",
    "name": "Juan Pérez",
    "taxId": "12345678X",
    "email": "juan@example.com"
  }
}
```

4. `/api/accounting/contasimple/invoices` (POST)

Crea una factura desde un contrato

Request Body:

```
{
  "contractId": "contract_123"
}
```

Respuesta:

```
{
  "success": true,
  "message": "Factura creada exitosamente: 2024-0123",
  "data": {
    "id": "invoice_123",
    "number": "2024-0123",
    "series": "A",
    "subtotal": 1000,
    "iva": 210,
    "total": 1210,
    "status": "draft"
  }
}
```

5. `/api/accounting/contasimple/payments` (POST)

Registra un pago en ContaSimple

Request Body:

```
{
  "paymentId": "payment_123"
}
```

6. `/api/accounting/contasimple/expenses` (POST)

Registra un gasto en ContaSimple

Request Body:

```
{
  "expenseId": "expense_123"
}
```

C. Base de Datos

Campos Agregados al Schema de Prisma:

1. **Modelo Tenant:**
   - `contasimpleCustomerId`: Almacena el ID del cliente en ContaSimple
2. **Modelo Contract:**
   - `contasimpleInvoiceId`: Almacena el ID de la factura en ContaSimple
3. **Modelo Payment:**
   - `contasimplePaymentId`: Almacena el ID del pago en ContaSimple
4. **Modelo Expense:**
   - `contasimpleExpenseId`: Almacena el ID del gasto en ContaSimple

D. Interfaz de Usuario

Ubicación: `/contabilidad` - Sección “Integraciones de Software de Contabilidad”

Funcionalidades:

1. **Estado de Configuración:**
   - Badge verde “✓ Activa” cuando está configurado
   - Badge gris “Demo” cuando no está configurado
2. **Botón “Probar Conexión”:**
   - Verifica la autenticación con ContaSimple
   - Muestra resultado en notificación
3. **Botones de Acción:**
   - “Sincronizar Clientes”: Guía al usuario a la sección de inquilinos
   - “Crear Facturas”: Guía al usuario a la sección de contratos
   - “Registrar Pagos”: Guía al usuario a la sección de pagos
   - “Registrar Gastos”: Funcionalidad disponible

📖 Guía de Uso

1. Verificar la Configuración

1. Acceder a `/contabilidad` en la aplicación
2. Buscar la sección “ContaSimple”
3. Verificar que aparezca el badge verde “✓ Activa”
4. Clic en “Probar Conexión” para verificar autenticación

2. Sincronizar un Inquilino como Cliente

Opción A: Desde la API

```
const response = await fetch('/api/accounting/contasimple/customers', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ tenantId: 'tenant_id_here' })
});
```

Opción B: Desde el Código

```
import { getContaSimpleService } from '@/lib/contasimple-integration-service';

const contaSimple = getContaSimpleService();
const customer = await contaSimple.syncTenantToCustomer(tenant);
```

3. Crear una Factura desde un Contrato

Requisito: El inquilino debe estar sincronizado primero (tener `contasimpleCustomerId`)

```
const response = await fetch('/api/accounting/contasimple/invoices', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ contractId: 'contract_id_here' })
});
```

4. Registrar un Pago

Requisito: El contrato debe tener una factura en ContaSimple (tener `contasimpleInvoiceId`)

```
const response = await fetch('/api/accounting/contasimple/payments', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ paymentId: 'payment_id_here' })
});
```

5. Registrar un Gasto

```
const response = await fetch('/api/accounting/contasimple/expenses', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ expenseId: 'expense_id_here' })
});
```

🔄 Flujo de Sincronización Completo

Ejemplo: Sincronizar un Contrato Completo

```typescript
import { getContaSimpleService } from '@/lib/contasimple-integration-service';
import { prisma } from '@/lib/db';

async function syncCompleteContract(contractId: string) {
  const contaSimple = getContaSimpleService();

  // 1. Obtener contrato con datos relacionados
  const contract = await prisma.contract.findUnique({
    where: { id: contractId },
    include: {
      tenant: true,
      unit: { include: { building: true } }
    }
  });

  // 2. Sincronizar inquilino como cliente (si no está sincronizado)
  if (!contract.tenant.contasimpleCustomerId) {
    const customer = await contaSimple.syncTenantToCustomer(contract.tenant);
    await prisma.tenant.update({
      where: { id: contract.tenant.id },
      data: { contasimpleCustomerId: customer.id }
    });
  }

  // 3. Crear factura del contrato
  const invoice = await contaSimple.createInvoiceFromContract(
    contract,
    contract.tenant.contasimpleCustomerId
  );

  await prisma.contract.update({
    where: { id: contractId },
    data: { contasimpleInvoiceId: invoice.id }
  });

  // 4. Registrar pagos existentes
  const payments = await prisma.payment.findMany({
    where: {
      contractId,
      estado: 'pagado',
      contasimplePaymentId: null
    }
  });

  for (const payment of payments) {
    const csPayment = await contaSimple.syncPaymentToContaSimple(
      payment,
      invoice.id
    );

    await prisma.payment.update({
      where: { id: payment.id },
      data: { contasimplePaymentId: csPayment.id }
    });
  }

  return {
    customer: contract.tenant.contasimpleCustomerId,
    invoice: invoice.id,
    paymentsSync: payments.length
  };
}
```

🔐 Seguridad

Gestión de Tokens

- **Duración:** Los tokens tienen una validez de 1 hora
- **Renovación Automática:** El servicio renueva automáticamente los tokens 5 minutos antes de su expiración
- **Almacenamiento:** Los tokens se almacenan en memoria durante la ejecución

Credenciales

- Las credenciales se almacenan en variables de entorno (`.env`)
- Nunca se exponen en el código del cliente
- Solo se utilizan en el lado del servidor

⚠️ Manejo de Errores

Todos los métodos del servicio incluyen manejo de errores robusto:

```
try {
  const customer = await contaSimple.createCustomer(customerData);
  console.log('✅ Cliente creado:', customer.id);
} catch (error: any) {
  console.error('❌ Error:', error.message);
  // El error incluye detalles de la respuesta de la API
}
```

Errores Comunes

1. **Error de Autenticación:**
   - **Causa:** Clave de autorización inválida o expirada
   - **Solución:** Verificar CONTASIMPLE_AUTH_KEY

2. **Token Expirado:**
   - **Causa:** El token tiene más de 1 hora
   - **Solución:** El servicio renueva automáticamente

3. **Cliente No Encontrado:**
   - **Causa:** El inquilino no tiene contasimpleCustomerId
   - **Solución:** Sincronizar el inquilino primero

4. **Factura Sin Cliente:**
   - **Causa:** Intentar crear una factura sin cliente asociado
   - **Solución:** Verificar que el tenant tenga contasimpleCustomerId

📊 Datos de Ejemplo

Cliente (Customer)

```
{
  id: 'customer_abc123',
  name: 'Juan Pérez García',
  taxId: '12345678X',
  email: 'juan.perez@example.com',
  phone: '+34 600 123 456',
  customerType: 'individual',
  address: {
    street: 'Calle Mayor 123',
    city: 'Madrid',
    postalCode: '28001',
    province: 'Madrid',
    country: 'España'
  }
}
```

Factura (Invoice)

```
{
  id: 'invoice_xyz789',
  number: '2024-0123',
  series: 'A',
  date: '2024-12-01',
  dueDate: '2024-12-31',
  customerId: 'customer_abc123',
  items: [
    {
      description: 'Alquiler - Edificio Central - Piso 3A',
      quantity: 1,
      unitPrice: 1000,
      ivaRate: 21,
      amount: 1000
    }
  ],
  subtotal: 1000,
  taxBase: 1000,
  iva: 210,
  total: 1210,
  status: 'draft'
}
```

Pago (Payment)

```
{
  id: 'payment_pqr456',
  invoiceId: 'invoice_xyz789',
  date: '2024-12-05',
  amount: 1210,
  method: 'transfer',
  reference: 'PAGO-2024-123',
  bankAccount: 'ES12 1234 5678 9012 3456 7890'
}
```

Gasto (Expense)

```
{
  id: 'expense_def321',
  date: '2024-12-01',
  description: 'Reparación fontanería piso 3A',
  amount: 150.50,
  category: 'Mantenimiento',
  taxDeductible: true,
  supplierId: 'provider_123',
  receiptUrl: 'https://s3.../factura-fontanero.pdf'
}
```

🧪 Testing

Probar la Autenticación

```
# Desde el navegador o Postman
GET /api/accounting/contasimple/test-connection
```

Verificar Estado de Configuración

```
GET /api/accounting/contasimple/status
```

📚 Documentación Adicional

API de ContaSimple

- **Documentación Swagger:** https://api.contasimple.com/swagger
- **API Base URL:** https://api.contasimple.com/api/v2

Endpoints Principales de ContaSimple

Método	Endpoint	Descripción
POST	/oauth/token	Obtener token de acceso
POST	/customers	Crear cliente
GET	/customers/{id}	Obtener cliente
PUT	/customers/{id}	Actualizar cliente
POST	/invoices	Crear factura
GET	/invoices/{id}	Obtener factura
POST	/invoices/{id}/send	Enviar factura por email
POST	/invoices/{id}/cancel	Cancelar factura
POST	/payments	Registrar pago
POST	/expenses	Registrar gasto
GET	/expenses	Obtener gastos

✅ Checklist de Implementación

- [x] Configuración de variables de entorno
- [x] Servicio de integración con ContaSimple
- [x] Métodos de autenticación OAuth2
- [x] Gestión automática de tokens
- [x] CRUD de clientes
- [x] CRUD de facturas
- [x] Registro de pagos
- [x] Registro de gastos
- [x] Endpoints API REST
- [x] Campos en base de datos (Prisma)
- [x] Interfaz de usuario
- [x] Manejo de errores
- [x] Documentación completa
- [x] Testing básico

🎉 Conclusión

La integración con **ContaSimple** está **completamente implementada y funcional**. La aplicación INMOVA ahora puede:

1. ✅ Autenticarse automáticamente con ContaSimple
2. ✅ Sincronizar inquilinos como clientes
3. ✅ Crear facturas desde contratos
4. ✅ Registrar pagos en el sistema de contabilidad
5. ✅ Registrar gastos operacionales
6. ✅ Mantener sincronizados los datos entre ambos sistemas

La integración está lista para producción y puede ser utilizada inmediatamente.

📞 Soporte

Para cualquier consulta sobre la integración con ContaSimple:

- **Documentación API:** https://api.contasimple.com/swagger
- **Soporte ContaSimple:** https://www.contasimple.com/soporte

---

Fecha de Implementación: 1 de Diciembre de 2024
Versión: 1.0.0
Estado: ✅ PRODUCCIÓN READY