Manual Maestro

Sistema Nexoryn Tech

Guía de Operación, Fallos y Resolución de Problemas

Sistema de Gestión Integral

Este manual contiene detalles críticos sobre AFIP, Stock y Cuentas.

Contenido del Manual

1. Conceptos Fundamentales

Nexoryn Tech es un sistema de gestión en tiempo real. Esto significa que cada acción (vender, cobrar, mover mercadería) impacta de forma inmediata en los saldos y las estadísticas del tablero de control.

El sistema separa las operaciones en 'Comprobantes' (documentos legales o internos) y 'Movimientos' (el flujo físico o monetario).

2. Navegación y Atajos de Eficiencia

Uso de Tablas Inteligentes

Las tablas son el núcleo del sistema. Permiten editar datos directamente (vistas rápidas) o entrar al detalle completo.

Rueda Mouse	- Scroll vertical lento/rápido.
Shift + Rueda	- Scroll horizontal (vital para tablas con muchas columnas).
Doble Clic en Celda	- Si la columna es editable, abre el editor instantáneo.
Ctrl + F	- Foco rápido en el buscador global de la tabla.
F5	- Refresca los datos de la tabla (útil en red local).

Filtros Avanzados

No use solo el buscador global. Use los 'Filtros Avanzados' para buscar por 'Stock Bajo', 'Fecha de Alta' o 'Deuda de Cliente'. Esto reduce la carga del sistema y le da resultados exactos.

Seguridad por Inactividad

Si no hay actividad durante 5 minutos, el sistema cierra la sesión automáticamente para proteger los datos.

3. Gestión de Inventario y Stock Crítico

Cada artículo tiene un 'Stock Mínimo'. Cuando el stock actual es igual o menor a este valor, el sistema marcará el producto con una alerta visual en el inventario y en el tablero de control.

ALERTA: Diferencia de Stock

Si el stock físico no coincide con el del sistema, NO edite el número directamente en la ficha del producto. Vaya a 'Movimientos' y realice un 'AJUSTE DE STOCK' con el motivo correspondiente para que quede registro de quién y por qué se cambió.

4. Entidades y Cuentas Corrientes

El saldo de un cliente se compone de: Facturas (+) y Pagos (-).

- Saldo Positivo: El cliente le debe dinero.
- Saldo Negativo: El cliente tiene saldo a favor (le pagó de más o hubo una devolución).

Importante: Para que un pago impacte en el saldo, debe estar asociado a la Entidad Comercial correcta.

5. Facturación Electrónica (AFIP/ARCA)

El sistema se comunica con AFIP en tres pasos silenciosos:
1. Solicita un 'LoginTicket' (Token) usando sus certificados digitales.
2. Envía los datos del comprobante.
3. Recibe el CAE y la fecha de vencimiento.

Condiciones para el ÉXITO de la factura:

- El comprobante debe estar confirmado y sin CAE antes de autorizar.
- Para comprobantes letra A se requiere CUIT válido (11 dígitos) y condición IVA del receptor.
- Para letra B/C se admite CUIT (11 dígitos) o DNI (hasta 8 dígitos).
- Los certificados (.crt y .key) deben estar vigentes (vencen anualmente).
- El Punto de Venta debe ser tipo 'Web Services' (RECE).
- AFIP funciona en el .exe sin Bash, pero requiere OpenSSL accesible.

ALERTA: Error de Conexión AFIP

Si AFIP no responde, el comprobante quedará sin CAE. NO intente facturar de nuevo el mismo documento sin verificar antes en 'Comprobantes' si ya obtuvo CAE. Si el error persiste, verifique su conexión a internet o si la página de AFIP está caída (es común en horarios pico).

6. Caja y Movimientos Financieros

La caja refleja el dinero disponible. Cada usuario tiene asignada una caja o puede usar una caja central.

Recuerde realizar el 'CIERRE DE CAJA' al final del día para comparar el efectivo real con lo que el sistema indica.

7. Resolución de Problemas y Errores Comunes

Problema A: El sistema no inicia o dice "Error de Base de Datos"

- Causa: El servicio de PostgreSQL está detenido o el firewall bloquea la conexión.
- Solución: Reinicie la PC o verifique que el servicio de Postgres esté 'En Ejecución' en el administrador de tareas.

Problema B: "No se encuentra el archivo .env" o credenciales

- Causa: Se borró el archivo de configuración o está en una ubicación no soportada.
- Solución: Verifique que exista un `.env` en %APPDATA%\Nexoryn_Tech\ o junto al ejecutable, y que tenga las credenciales correctas.

Problema C: "openssl no encontrado" al facturar AFIP

- Causa: OpenSSL no está instalado o no está accesible desde la app.
- Solución: Instale OpenSSL o copie `openssl.exe` junto con `libcrypto-*.dll` y `libssl-*.dll` al directorio del ejecutable.

Problema D: El PDF no se genera o no se abre

- Causa: Un antivirus bloquea el acceso a la carpeta temporal o no tiene un lector de PDF instalado.
- Solución: Verifique que puede abrir otros archivos PDF en su computadora.

ALERTA: CORRUPCIÓN DE DATOS / CORTE DE LUZ

Si hubo un corte de luz mientras el sistema estaba guardando, es posible que el último registro falle. El sistema está protegido por 'transacciones' para evitar pérdida total, pero siempre verifique el último comprobante emitido tras un reinicio forzado.

Resumen Final de Atajos

Enter	- Acepta el formulario o busca en la tabla.
Esc	- Cancela la acción o cierra el modal actual.
Ctrl + S	- En algunas pantallas, guarda el borrador.
Tab / Shift+Tab	- Navega rápidamente entre campos de texto.
Alt + [Letra]	- Navega entre las pestañas del menú lateral.

Este manual es una herramienta dinámica. Si detecta un error no documentado, reporte al equipo técnico para su inclusión.

8. Apéndice técnico

Este apéndice incluye la documentación técnica actual del proyecto. Si se actualizan los documentos en /docs, vuelva a generar el PDF para reflejar los cambios.

Guía de Requisitos: Componentes "En Mano"

Este sistema utiliza archivos y configuraciones que, por seguridad o peso, **no están incluidos en el repositorio de Git**. Siempre que reinstales el sistema o lo muevas a otra PC, debes asegurarte de tener estos elementos.

---

1. Entorno de Software

Antes de empezar, la máquina debe tener:

- **Python 3.12+** (recomendado) agregado al PATH.
- **PostgreSQL 16+** (con `psql`, `pg_dump` y `pg_restore`).
- **OpenSSL** (para firmar solicitudes AFIP).
- AFIP funciona en el `.exe` sin Bash, pero requiere `openssl.exe` accesible (instalado o junto al ejecutable).

Instalación de dependencias

Ejecutar en la carpeta raíz:

```
pip install -r requirements.txt
```

---

2. Variables de Entorno (.env)

El archivo `.env` **no se sube a GitHub**.

Ubicaciones soportadas (prioridad):

- `%APPDATA%\Nexoryn_Tech\.env` (instalación estándar)
- `.env` junto al ejecutable o en la raíz del proyecto (modo portable/dev)

Plantilla de `.env`:

```
# Base de Datos
DATABASE_URL=postgresql://postgres:password@localhost:5432/nexoryn_tech
# o componentes individuales:
```

```
DB_HOST=localhost
DB_PORT=5432
DB_NAME=nexoryn_tech
DB_USER=postgres
DB_PASSWORD=tu_password
DB_MAINTENANCE_USER_ID=1

# Pool (opcional)
DB_POOL_MIN=1
DB_POOL_MAX=4

# PostgreSQL bin (necesario si psql/pg_dump no están en PATH)
PG_BIN_PATH="C:\Program Files\PostgreSQL\16\bin"

# AFIP
AFIP_PRODUCCION=False  # también soporta AFIP_PRODUCTION
AFIP_PUNTO_VENTA=3

# Homologación (default si AFIP_PRODUCCION=False)
# También soporta *_HOMOLOGACION
AFIP_CUIT=20XXXXXXXX9
AFIP_CERT_PATH="C:/Nexoryn/Certs/empresa_homo.crt"
AFIP_KEY_PATH="C:/Nexoryn/Certs/empresa_homo.key"
# Alternativas homologación:
# AFIP_CUIT_HOMOLOGACION=20XXXXXXXX9
# AFIP_CERT_PATH_HOMOLOGACION="C:/Nexoryn/Certs/empresa_homo.crt"
# AFIP_KEY_PATH_HOMOLOGACION="C:/Nexoryn/Certs/empresa_homo.key"

# Producción (si AFIP_PRODUCCION=True)
# También soporta *_PRODUCTION
AFIP_CUIT_PRODUCCION=20YYYYYYYY9
AFIP_CERT_PATH_PRODUCCION="C:/Nexoryn/Certs/empresa_prod.crt"
AFIP_KEY_PATH_PRODUCCION="C:/Nexoryn/Certs/empresa_prod.key"
# Alternativas producción:
# AFIP_CUIT_PRODUCTION=20YYYYYYYY9
# AFIP_CERT_PATH_PRODUCTION="C:/Nexoryn/Certs/empresa_prod.crt"
# AFIP_KEY_PATH_PRODUCTION="C:/Nexoryn/Certs/empresa_prod.key"

# UI (opcional)
NEXORYN_UI=basic  # o advanced
```

Nota: `DB_MAINTENANCE_USER_ID` debe ser un `seguridad.usuario.id` existente (por defecto 1 para el admin inicial). Se usa en restores/psql para setear `app.user_id` y evitar bloqueos por RLS.

---

3. Certificados AFIP

Para que la facturación electrónica funcione, necesitas:

- **Clave Privada (`.key`)**: Generada en tu PC. **No la compartas**.
- **Certificado (`.crt`)**: Descargado de AFIP tras subir el CSR.
- **Punto de Venta**: Número configurado en AFIP (ej: `0002`).

> Consulta `docs/GUIA_AFIP_PORTAL.md` para el paso a paso en AFIP.

---

4. Datos Iniciales (CSVs)

El script `database/init_db.py` busca archivos en: `database/csvs/*.csv`

Estos archivos contienen la información histórica o bases de datos externas que el sistema importa al primer uso.

---

5. Resumen de lo que debes llevar

- [ ] Tu archivo `.env` configurado.
- [ ] Tu certificado AFIP (`.crt`).
- [ ] Tu clave privada AFIP (`.key`).
- [ ] La carpeta `database/csvs/` (si tienes datos para importar).
- [ ] La carpeta `backups/` o `backups_incrementales/` (si quieres conservar copias previas).

Guía de Generación de Ejecutable y Configuración - Nexoryn Tech

Esta guía explica cómo generar el ejecutable (`.exe`) y cómo ubicar la configuración en producción.

1. Generación del Ejecutable

La aplicación está construida con **Flet**. Para generar el ejecutable, utilizamos `flet pack`.

Requisitos previos

```
pip install flet pyinstaller
```

Comandos de Empaquetado

```
flet pack desktop_app/main.py --name "NexorynTech" --add-data "database;database"
```

*> **Importante**: incluir `--add-data "database;database"` para que el `database.sql` esté disponible en el ejecutable.*

---

2. Ubicación de Archivos (.env, Certificados, etc.)

La aplicación busca configuración en dos lugares, en este orden:

Opción A: Instalación estándar (recomendado para PC fija)

- **Ruta**: `%APPDATA%\Nexoryn_Tech\`
- **Archivos**:
- `.env`
- `certs/` con certificados AFIP

Opción B: Modo portable (pendrives o carpeta local)

- **Ruta**: mismo directorio del `NexorynTech.exe`
- **Archivos**:
- `.env`
- `certs/`

> Las rutas relativas de `AFIP_CERT_PATH`/`AFIP_KEY_PATH` se resuelven respecto al directorio de configuración.

---

3. Ejemplo de archivo `.env`

```
# Conexión a Base de Datos
DATABASE_URL=postgresql://postgres:password@localhost:5432/nexoryn_tech

# Binarios PostgreSQL (si no están en PATH)
PG_BIN_PATH="C:\Program Files\PostgreSQL\16\bin"

# AFIP (homologación por defecto)
AFIP_PRODUCCION=False  # también soporta AFIP_PRODUCTION
AFIP_PUNTO_VENTA=3
AFIP_CUIT=20XXXXXXXX9
AFIP_CERT_PATH=certs/mi_certificado.crt
AFIP_KEY_PATH=certs/mi_llave.key
# Variantes soportadas:
#     -     Homologación:     AFIP_CUIT_HOMOLOGACION     /     AFIP_CERT_PATH_HOMOLOGACION     /
AFIP_KEY_PATH_HOMOLOGACION
# - Producción: AFIP_CUIT_PRODUCCION o AFIP_CUIT_PRODUCTION (y equivalentes para CERT/KEY)

# UI (opcional)
NEXORYN_UI=basic  # o advanced
```

---

4. OpenSSL (AFIP)

No es obligatorio instalar Git, pero **OpenSSL sí** para AFIP. Opciones:

- **Instalar Git para Windows** (incluye OpenSSL).
- **Instalar OpenSSL independiente** (Win64 OpenSSL v3.x Light).
- **Modo portable**: copiar `openssl.exe`, `libcrypto-*.dll` y `libssl-*.dll` en una carpeta `bin/` junto al `NexorynTech.exe`.

La app ya maneja `MSYS_NO_PATHCONV` automáticamente para evitar problemas de rutas en Windows.

> Nota importante: AFIP funciona en el `.exe` sin Bash. Lo único necesario es que `openssl.exe` esté accesible (instalado y en PATH, o incluido junto al ejecutable).

---

5. Implementación en Red LAN (Ejecutable Compartido)

Si planeas dejar el ejecutable en una carpeta compartida (ej: `\\SERVIDOR\Sistema\NexorynTech.exe`):

Ubicación centralizada (recomendada)

Coloca `.env` y `certs/` en la misma carpeta de red donde está el ejecutable.

```
\\SERVIDOR\Nexoryn\
??? NexorynTech.exe
??? .env
??? certs/
    ??? mi_empresa.crt
    ??? mi_empresa.key
```

Requisito en cada PC

Cada computadora que ejecute la app debe tener OpenSSL (o Git para Windows) instalado.

Seguridad de Archivos en Red

Opciones (de menor a mayor seguridad):

- **Atributos de Windows** (básico): `attrib +h +s .env` / `attrib +h +s certs`
- **Permisos NTFS** (recomendado): restringir acceso a `.env` y `certs/`
- **Almacenamiento en DB** (avanzado): guardar datos sensibles en `seguridad.config_sistema`

Estructura para Modo Portable

```
Carpeta_App/
??? NexorynTech.exe
??? .env
??? certs/
    ??? mi_certificado.crt
    ??? mi_llave.key
```

Guía de Configuración en Red Local (LAN)

Esta guía explica cómo configurar el sistema para que funcione con múltiples computadoras en la misma red. Una computadora actuará como **Servidor** (donde está la base de datos) y las demás como **Clientes**.

1. Preparar el Servidor (Computadora con la Base de Datos)

A. Obtener la IP Local

- Abre una terminal (CMD o PowerShell).
- Escribe `ipconfig` y presiona Enter.
- Busca la sección de tu adaptador (Wi?Fi o Ethernet).
- Anota la **Dirección IPv4** (ejemplo: `192.168.1.15`).

B. Configurar PostgreSQL

PostgreSQL debe aceptar conexiones remotas:

- Ubica `postgresql.conf` (ej: `C:\Program Files\PostgreSQL\[VERSION]\data`).
- Cambia `listen_addresses` a:

```
   listen_addresses = '*'
```

- En `pg_hba.conf`, agrega:

```
   host    all             all             192.168.1.0/24          md5
```

Ajusta el rango si tu red es distinta (ej: `10.0.0.0/24`).

- Reinicia el servicio de PostgreSQL.

C. Habilitar el Firewall

Permitir tráfico al puerto 5432:

- Firewall de Windows Defender > Configuración avanzada.
- Regla de entrada > Puerto > TCP > `5432`.
- Permitir la conexión.

---

2. Configurar los Clientes (Demás Computadoras)

En cada PC cliente:

- Asegúrate de tener el ejecutable o el proyecto.
- Edita el `.env` usado por la app.
- Cambia `DB_HOST` por la IP del servidor.

```
DB_HOST=192.168.1.15
```

También puedes usar `DATABASE_URL`:

```
DATABASE_URL=postgresql://postgres:password@192.168.1.15:5432/nexoryn_tech
```

Ubicación del `.env`:

- `%APPDATA%\Nexoryn_Tech\.env` (instalación estándar)
- `.env` junto al ejecutable (modo portable)

Resumen de cambios

- **Servidor**: habilitar `listen_addresses`, agregar regla en `pg_hba.conf`, abrir puerto `5432`.
- **Clientes**: apuntar `DB_HOST`/`DATABASE_URL` a la IP del servidor.

Integración AFIP / ARCA (Implementada)

Este documento describe la integración técnica actual con AFIP/ARCA para facturación electrónica en Nexoryn Tech.

Estado Actual

La integración está implementada en `desktop_app/services/afip_service.py` y se usa desde la UI básica.

- WSAA: obtención de Token/Sign
- WSFEv1: autorización de comprobantes y obtención de CAE
- Firma CMS con OpenSSL
- Caché del Token en `logs/afip_ta_wsfe_{homo|prod}.xml`

Requisitos

- Clave Fiscal nivel 3
- Certificados `.crt` y `.key` válidos
- OpenSSL instalado o incluido junto al ejecutable

Configuración (.env)

La app soporta homologación y producción. El flag `AFIP_PRODUCCION` (o `AFIP_PRODUCTION`) define qué credenciales usar.

```
# Flag de entorno (acepta AFIP_PRODUCCION o AFIP_PRODUCTION)
AFIP_PRODUCCION=False

# Credenciales Homologación (default si AFIP_PRODUCCION=False)
# También soporta *_HOMOLOGACION
AFIP_CUIT=20XXXXXXXX9
AFIP_CERT_PATH=certs/empresa_homo.crt
AFIP_KEY_PATH=certs/empresa_homo.key
# Alternativas homologación:
# AFIP_CUIT_HOMOLOGACION=20XXXXXXXX9
# AFIP_CERT_PATH_HOMOLOGACION=certs/empresa_homo.crt
# AFIP_KEY_PATH_HOMOLOGACION=certs/empresa_homo.key

# Credenciales Producción (se usan si AFIP_PRODUCCION=True)
# También soporta *_PRODUCTION
AFIP_CUIT_PRODUCCION=20YYYYYYYY9
```

```
AFIP_CERT_PATH_PRODUCCION=certs/empresa_prod.crt
AFIP_KEY_PATH_PRODUCCION=certs/empresa_prod.key
# Alternativas producción:
# AFIP_CUIT_PRODUCTION=20YYYYYYYY9
# AFIP_CERT_PATH_PRODUCTION=certs/empresa_prod.crt
# AFIP_KEY_PATH_PRODUCTION=certs/empresa_prod.key

# Punto de venta
AFIP_PUNTO_VENTA=3
```

Resolución de credenciales:

- Homologación: usa `AFIP_CUIT/AFIP_CERT_PATH/AFIP_KEY_PATH` y, si existen, prioriza `*_HOMOLOGACION`.
- Producción: usa `*_PRODUCCION` y, si no existen, toma `*_PRODUCTION`.

Resolución de rutas:

- Si las rutas son relativas, se resuelven respecto al directorio de configuración.
- Directorio de configuración: `%APPDATA%\Nexoryn_Tech\` o la carpeta del ejecutable (modo portable).

Flujo en la UI

- El botón **Autorizar AFIP** se habilita cuando:
- el documento está en estado `CONFIRMADO` o `PAGADO`
- tiene `codigo_afip`
- no tiene `cae`
- Al autorizar, se actualizan:
- `cae`, `cae_vencimiento`, `punto_venta`, `tipo_comprobante_afip`, `cuit_emisor`, `qr_data`
- solo se actualizan datos AFIP del comprobante (no crea remitos automáticamente)

> La autorización es irreversible desde la UI. Verifica los datos antes de autorizar.

Troubleshooting

`openssl no encontrado`

- Instalar OpenSSL o Git for Windows
- O copiar `openssl.exe`, `libcrypto-*.dll`, `libssl-*.dll` en `bin/` junto al `.exe`

`Certificado no encontrado` / `Clave privada no encontrada`

- Revisar rutas en `.env`
- Si son relativas, confirmar que estén junto a la configuración

`WSAA/WSFE error`

- Verificar que el certificado corresponda al CUIT configurado
- Confirmar `AFIP_PRODUCCION`/`AFIP_PRODUCTION` vs homologación

Seguridad

- No subir certificados al repositorio.
- Guardar `.key` en una ubicación segura.
- Evitar loguear contenido de certificados o tokens.

Portal AFIP

Para el alta inicial de servicios y generación de certificados, ver `docs/GUIA_AFIP_PORTAL.md`.

Guía Detallada: Trámites en el Portal de AFIP/ARCA

Esta guía describe paso a paso qué botones tocar y qué opciones elegir dentro del portal de AFIP/ARCA. No es necesario saber programar para esta parte.

> Nota: los nombres de botones pueden variar con el tiempo. Si algún texto no coincide, buscá la opción con el mismo nombre funcional (ej: "Administrador de Relaciones", "Adherir Servicio", "Web Services").

---

1. Habilitar los Servicios Necesarios

Si estos servicios no te aparecen en tu pantalla principal de AFIP, debes "agregarlos" así:

- Entra a [afip.gob.ar](https://www.afip.gob.ar) con tu **CUIT** y **Clave Fiscal (Nivel 3)**.
- Busca **"Administrador de Relaciones de Clave Fiscal"** (a veces figura como **"Administrador de Relaciones"**).
- Haz clic en **"Adherir Servicio"**.
- Haz clic sobre el logo de **"AFIP"** y luego en **"Servicios Interactivos"**.
- Busca en la lista (está por orden alfabético) y haz clic en:
- **"Administración de Certificados Digitales"** (Obligatorio para subir tu llave).
- **"Gestión de Puntos de Venta y Comprobantes"** (Obligatorio para crear el punto de venta de Nexoryn).
- Haz clic en **"Confirmar"**.
- **Importante**: Cierra la sesión y vuelve a entrar para que aparezcan en el menú principal.

---

2. Crear el Punto de Venta para el Sistema

El sistema no puede usar el mismo punto de venta que usas para "Facturador en Línea" o la App del celular.

- Entra al servicio **"Gestión de Puntos de Venta y Comprobantes"**.
- Selecciona tu nombre/empresa.
- Haz clic en **"A/B/M de Puntos de Venta"** (puede figurar como **"Administrar Puntos de Venta"**).
- Haz clic en **"Agregar"**.
- Completa los campos:
- **Número**: El que siga (ejemplo: `0002` o `0005`). Anótalo.

- **Nombre Fantasía**: "Nexoryn Tech" o el nombre de tu negocio.
- **Sistema**: Selecciona **"RECE para aplicativo y web services"** (a veces figura como "Web Services" o "FactuWS"). **Este es el paso más importante**.
- **Domicilio**: Selecciona tu dirección fiscal.
- Haz clic en **"Aceptar"** y luego **"Confirmar"**.

---

3. Cargar el Certificado Digital (Alias)

Primero debes haber generado el archivo `.csr`. Una vez que lo tengas:

- Entra al servicio **"Administración de Certificados Digitales"**.
- Haz clic en **"Agregar Alias"** o **"Agregar Certificado"**.
- En **Alias**, ponle un nombre fácil, ej: `Nexoryn_Factura`.
- Donde dice **"Archivo"**, haz clic en "Examinar" y sube el archivo `.csr`.
- Haz clic en **"Agregar"** / **"Confirmar"**.
- En la siguiente pantalla, verás un link **"Ver"** o una flecha de descarga. Descarga el certificado oficial (archivo **.crt**).

> Tip: los certificados vencen (normalmente 1 año). Guardá la fecha de vencimiento.

---

4. El "Paso Final" en el Administrador de Relaciones

Ahora tienes que decirle a AFIP: *"Este certificado que subí tiene permiso para hacer Facturas"*.

- Vuelve al **"Administrador de Relaciones de Clave Fiscal"**.
- Haz clic en **"Nueva Relación"**.
- Haz clic en **"Buscar"**.
- Haz clic en el logo de **"AFIP"** y luego en **"WebServices"**.
- Busca **"Facturación Electrónica"** (WSFEv1) y selecciónalo.
- En la parte de **"Representante"**, haz clic en **"Buscar"**.
- **¡OJO AQUÍ!**: No pongas tu CUIT. En el desplegable selecciona el **Alias** que creaste en el paso 3 (ej: `Nexoryn_Factura`).
- Haz clic en **"Confirmar"**.

Resumen de lo que debes guardar:

Al terminar esto, deberías tener:

- El número del **Punto de Venta** nuevo (para AFIP).
- Tu clave privada (archivo **.key**).
- Tu certificado firmado (archivo **.crt**).

Con estos 3 datos, el sistema ya puede hablar legalmente con AFIP.

Sistema de Backups Profesionales (FULL/DIFERENCIAL/INCREMENTAL)

Resumen

El sistema profesional de backups está implementado en `BackupManager` + `BackupIncrementalService`. Genera backups concatenables FULL + DIFERENCIAL + INCREMENTAL y los registra en base de datos.

Cómo detecta cambios:

- Usa `pg_stat_user_tables` para detectar tablas con actividad desde el último backup.
- Guarda estadísticas en `backups_incrementales/stats/*.json`.
- Si PostgreSQL reinicia y las estadísticas se reinician, el sistema considera las tablas como "cambiadas" para estar del lado seguro.

*> Esto es un incremental lógico (por tablas) y **no** basado en WAL.*

Características Principales

- Backups FULL (mensuales) con `pg_dump -F c`
- Backups DIFERENCIALES (semanales) por tablas con cambios desde el último FULL
- Backups INCREMENTALES (diarios) por tablas con cambios desde el último backup (full/dif/inc)
- Cadena de restauración: FULL + DIFERENCIAL + INCREMENTALES
- Validación por existencia de archivos y checksum SHA-256
- Scheduler integrado vía APScheduler al iniciar la app
- No modifica el sistema de backups legacy (`backups/`)

Arquitectura

Directorios

```
backups_incrementales/
??? full/
??? differential/
??? incremental/
??? stats/              # estadísticas por backup para detectar cambios
```

Base de Datos

Tablas principales:

- `seguridad.backup_manifest`

- `seguridad.backup_chain`
- `seguridad.backup_validation`
- `seguridad.backup_event` (estructura disponible, no siempre poblada)

Configuración en `seguridad.config_sistema`:

- `backup_schedules` (JSON)
- `backup_retention` (JSON)
- `backup_cloud_config` (JSON)

Scheduler Integrado

Horarios por defecto (hora local):

- **FULL**: Día 1 de cada mes a las 00:00
- **DIFERENCIAL**: Domingos a las 23:30
- **INCREMENTAL**: Diariamente a las 23:00
- **Validación**: Diariamente a las 01:00

Edición desde UI:

- El panel **Respaldos** permite cambiar días y horas.
- Los minutos se mantienen según el default (FULL 00, DIF 30, INC 00).
- Los cambios se guardan en `backup_schedules`.

Backups omitidos: Al iniciar la app, se detectan backups faltantes y se ejecutan en orden `FULL ? DIFERENCIAL ? INCREMENTAL`.

Uso desde la UI

En el panel **Respaldos** puedes:

- Ejecutar backups manuales (FULL/DIF/INC).
- Ver historial y métricas.
- Validar backups existentes.
- Configurar horarios y retención.

Restauración

API Programática

```
from desktop_app.database import Database
```

```
from desktop_app.config import load_config
from desktop_app.services.backup_incremental_service import BackupIncrementalService
from desktop_app.services.restore_service import RestoreService

config = load_config()
db = Database(config.database_url)

backup_service = BackupIncrementalService(db)
restore_service = RestoreService(db, backup_service)

# Restaurar a una fecha específica
from datetime import datetime
target_date = datetime(2025, 12, 15, 12, 0, 0)
result = restore_service.restore_to_date(target_date)

if result.exitoso:
    print(f"Restauración exitosa: {result.mensaje}")
    print(f"Backups aplicados: {result.backups_aplicados}")
    print(f"Tiempo: {result.tiempo_segundos}s")

# Previsualizar restauración
preview = restore_service.preview_restore(target_date)
print(f"Cantidad de backups: {preview['cantidad_backups']}")
print(f"Tamaño total: {preview['tamano_total_mb']} MB")
print(f"Backups a aplicar: {[b['archivo'] for b in preview['backups']]}")
```

Línea de Comandos (Bootstrap rápido)

```
python -c "
from desktop_app.database import Database
from desktop_app.config import load_config
from desktop_app.services.backup_manager import BackupManager

db = Database(load_config().database_url)
manager = BackupManager(db)
result = manager.restore_from_backup_id(123, 'nexoryn_tech_restaurado')
print(result)
"
```

> Nota: la base de datos destino debe existir antes de restaurar.

Validaciones

`validate_backup_chain(backup_id)` verifica:

- existencia del archivo
- checksum SHA-256 (si está registrado)

No valida contenido lógico ni dependencias externas.

Nube / Sync

El sistema soporta **sincronización a carpeta local** mediante `CloudStorageService`:

- `provider=LOCAL` copia el archivo a `sync_dir`.
- `provider=GOOGLE_DRIVE` y `provider=S3` están **reservados**; hoy devuelven error **no implementado** y no marcan `nube_subido`.

La configuración se guarda en `backup_cloud_config`.

Retención

- La UI permite definir retención (FULL meses / DIF semanas / INC días).
- La configuración se guarda en `backup_retention`.
- **No hay purga automática** en el sistema incremental actual.
- Existe `purge_invalid_backups()` para limpiar registros huérfanos (archivos faltantes).

Configuración

Variables de Entorno

```
# Conexión a PostgreSQL
export DATABASE_URL=postgresql://postgres:password@localhost:5432/nexoryn_tech
# o DB_HOST/DB_PORT/DB_NAME/DB_USER/DB_PASSWORD

# Ruta a binarios de PostgreSQL
export PG_BIN_PATH="C:\Program Files\PostgreSQL\16\bin"
```

Solución de Problemas

`pg_dump` / `pg_restore` no encontrados

- Instalar PostgreSQL con herramientas de línea de comandos.
- Configurar `PG_BIN_PATH` o agregar `bin` al `PATH`.

Permisos / archivos en uso

- Ejecutar la app con permisos suficientes para crear archivos en el directorio de backups.

Compatibilidad con Sistema Antiguo

El sistema legacy (`backups/` con daily/weekly/monthly/manual) **no es modificado**. Si el sistema profesional falla al iniciar, la app intenta usar el servicio legacy como fallback para no detener el flujo de respaldos.

Gestión de Base de Datos - Nexoryn Tech

Este documento describe los scripts de base de datos, el flujo de sincronización automática del esquema y el mantenimiento de logs en el sistema actual.

Scripts Principales

1. `init_db.py` (Inicializador)

Script para crear el esquema y opcionalmente importar CSVs.

- **Acciones**: Crea el esquema desde `database.sql`, importa datos desde `database/csvs/` y puede resetear esquemas.
- **Dependencias**: `pandas`, `psycopg2-binary`.
- **Ejecución**:

```
# Inicialización normal (Esquema + Importación)
python database/init_db.py

# Reset completo (Borra esquemas existentes y recrea todo)
python database/init_db.py --reset

# Solo esquema (Sin importar CSVs)
python database/init_db.py --skip-csv

# Usar otra base de datos
python database/init_db.py --db-name nexoryn_tech

# Modo simulación (no ejecuta cambios)
python database/init_db.py --dry-run
```

> Nota: el script usa por defecto `--db-name nexoryn_tech` si no se especifica.

2. `kill_sessions.py` (Terminador de Sesiones)

Utilidad para cerrar conexiones activas cuando PostgreSQL bloquea operaciones de mantenimiento.

- **Dependencias**: `psycopg`.
- **Ejecución**:

```
python database/kill_sessions.py --db-name nexoryn_tech
```

3. `db_conn.py`

Módulo de utilidad que centraliza la conexión para scripts. Usa `.env` si existe y soporta `DATABASE_URL`

o variables `DB_*`.

Sincronización Automática del Esquema (SchemaSync)

La app ejecuta una sincronización **temprana** del esquema antes de abrir el pool de conexiones.

Cómo funciona:

- Lee la versión del encabezado de `database/database.sql` (`-- Version: X.X`).
- Consulta `seguridad.config_sistema` (clave `db_version`) mediante `psql`.
- Si la versión no coincide, ejecuta `psql -f database.sql` con `ON_ERROR_STOP=1`.

Requisitos:

- `psql` en el `PATH` o definido en `PG_BIN_PATH`.

Importante:

- No aplica un diff por hashes. Re-ejecuta el archivo completo.
- El SQL está diseñado para ser idempotente (`CREATE IF NOT EXISTS`, `CREATE OR REPLACE`), pero **cambios destructivos deben manejarse en un flujo separado**.
- El esquema usa `pg_advisory_lock` al inicio del archivo para evitar concurrencia entre instancias.

Migraciones en Runtime (Database._run_migrations)

Al inicializar `Database`, se aplican migraciones seguras en caliente:

- `app.pago.id_documento` se vuelve nullable (para pagos de cuenta corriente).
- `app.movimiento_articulo.stock_resultante` se agrega si falta.
- Se actualiza el trigger `app.fn_sync_stock_resumen` para persistir `stock_resultante`.

RLS y contexto de sesión

El esquema habilita RLS en tablas núcleo:

- `app.documento`
- `app.entidad_comercial`
- `app.movimiento_articulo`

Regla operativa: las operaciones de escritura requieren que la sesión tenga `app.user_id` seteado (policies con `WITH CHECK (current_setting('app.user_id', true) IS NOT NULL)`). Si no está definido, las

escrituras pueden fallar o dejar auditoría sin usuario. Las lecturas no filtran datos en este escenario (policies con `USING (true)`).

Cómo lo gestiona la app: en conexiones normales la app setea el contexto con `set_config('app.user_id', ...)` al abrir cada transacción.

Restore y mantenimiento

Para tareas manuales o herramientas de PostgreSQL (`psql`, `pg_dump`, `pg_restore`) se debe setear `app.user_id` por sesión. Ejemplos:

```
PGOPTIONS="-c app.user_id=1" psql -h localhost -U postgres -d nexoryn_tech
PGOPTIONS="-c app.user_id=1" pg_restore -h localhost -U postgres -d nexoryn_tech backup.dump
```

Dentro de una sesión `psql`:

```
SET app.user_id = 1;
```

Los servicios de backup/restore toman `DB_MAINTENANCE_USER_ID` y lo convierten en `PGOPTIONS` automáticamente para evitar bloqueos por RLS.

Logs de Auditoría (particionado y archivado)

Particionado automático

En el arranque de la UI básica se ejecuta `migrate_to_partitioned_logs(db)` para:

- Convertir `seguridad.log_actividad` en tabla particionada por semana.
- Migrar datos históricos si existía una tabla no particionada.

Archivado automático

Se inicia un `LogArchiver` en segundo plano:

- **Retención**: `log_retencion_dias` en `seguridad.config_sistema` (default 90).
- **Directorio**: `log_directorio_archivo` (default `logs_archive`).
- **Ruta final**: `<PROJECT_ROOT>/data/<log_directorio_archivo>`.
- Archiva a `.jsonl.gz` y luego elimina los registros antiguos.
- Ejecuta `seguridad.mantener_particiones_log()` si la función existe.

Actualizaciones en Tiempo Real (solo UI avanzada)

La UI avanzada usa polling cada 5 segundos sobre `seguridad.log_actividad` mediante `Database.check_recent_activity()` para refrescar la vista activa.

Variables de Entorno Relevantes

- `DATABASE_URL` o `DB_HOST`, `DB_PORT`, `DB_NAME`, `DB_USER`, `DB_PASSWORD`.
- `DB_MAINTENANCE_USER_ID` debe ser un `seguridad.usuario.id` existente; se usa para setear `app.user_id` en restores/mantenimiento.
- `PG_BIN_PATH` para localizar `psql`, `pg_dump`, `pg_restore`.
- `DB_POOL_MIN` y `DB_POOL_MAX` para el pool de conexiones.

> Nota: si se requieren cambios riesgosos, ejecutar un flujo de migración controlado con backup previo.