MANUAL TÉCNICO DEL DISEÑO Y DESARROLLO DE UNA APLICACIÓN DE ESCRITORIO PARA LA VERIFICACIÓN AUTOMATIZADA DEL DESPACHO DE MATERIALES MEDIANTE VISIÓN ARTIFICIAL EN LA FERRETERÍA DURÁN, APULO 2026

MIGUEL ANGEL GONZALEZ POSADA
SAMUEL ESTEBAN VILLALBA DURAN

UNIVERSIDAD PILOTO DE COLOMBIA – SECCIONAL ALTO DEL MAGDALENA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
GIRARDOT, CUNDINAMARCA
AÑO: 2026

TABLA DE CONTENIDO

LISTA DE TABLAS

PRESENTACIÓN

Este documento contextualiza la dimensión técnica del sistema LogiCheck. Hace referencia íntegra a los componentes, algoritmos y dependencias lógicas necesarias para la verificación de despachos logísticos mediante el análisis de cámaras RTSP e inteligencia artificial.

El presente manual va dirigido rigurosamente a personal especializado en tecnologías de la información, abarcando perfiles tales como:
- Ingenieros de Sistemas y Arquitectos de Software.
- Desarrolladores especializados en librerías gráficas (Qt/PySide6) e Inteligencia Artificial.
- Administradores de Bases de Datos SQLite.
- Personal de Soporte Técnico enfocado en despliegues de hardware (GPU) y entornos Python.

El nivel de conocimiento requerido para la manipulación e interpretación de este documento asume bases firmes en programación orientada a objetos, entendimiento avanzado sobre redes neuronales convolucionales, manejo de gestores de variables de entorno (Entornos Virtuales), y nociones de protocolos de video (RTSP/FFMPEG), así como conocimientos robustos de concurrencia e hilos.

Visión General: LogiCheck opera estructurado bajo una Arquitectura Moderna en Capas y Patrón MVC acoplado a Componentes Modulares por Hilos. La plataforma desacopla la Capa de Visualización (Cliente - PySide6) de las capas de Procesamiento Profundo (Servicios de Inferencia Local) y la Base de Datos Relacional, posibilitando máxima fluidez de operación para el usuario final sin congelamientos de interfaz.

RESUMEN EJECUTIVO

El principal propósito de este resumen es brindar a un lector técnico una comprensión expedita sobre qué es LogiCheck y cómo ha sido construido, prescindiendo inicialmente de recorrer el manual entero.

Descripción Global y Objetivo: LogiCheck es una solución fiscal y de auditoría in situ. Su objetivo principal es resolver la pérdida de inventarios debida a despachos erróneos rastreando en tiempo real cajas, bultos y tuberías cargadas a vehículos mediante inferencia óptica paralela a gran escala.

Tecnologías Utilizadas y Arquitectura: Operando bajo una arquitectura por capas modulares (Presentación, Negocio Algorítmico, y Persistencia Relacional). Todo interactúa usando el patrón de eventos asíncronos 'Signals & Slots' de Qt. Emplea Python 3 como lenguaje matriz, PySide6 (interfaz Front-end), SQLite3 (acceso relacional seguro), Ultralytics/PyTorch (núcleo YOLOv26 Tensorizado), y solicitudes HTTP Asíncronas (Notificaciones Push API).

El manual consta de 6 capítulos primarios. El primero encuadra requerimientos de funcionamiento base; el segundo, la abstracción gráfica arquitectónica; el tercero expone entrañas técnicas, interfaces, Inteligencia artificial y esquemas de Base de datos forense. El cuarto indica al operario los procesos algorítmicos para instaurar el software paso a paso; el quinto determina topes físicos empresariales; y el finaliza listando fundamentos bibliográficos.

FINALIDAD DEL MANUAL

Este manual técnico ha sido confeccionado obedeciendo cuatro preceptos fundacionales:

1. Capacitar minuciosamente al personal técnico en el despliegue del software mitigando la complejidad subyacente de la configuración de ecosistemas con tecnología CUDA de NVIDIA.

2. Proveer documentación resolutiva para facilitar tareas de mantenimiento correctivo y predictivo frente a anomalías de hardware o inestabilidad de protocolos de transmisión IP.

3. Constituir el marco de referencia formal, brindando bases sólidas en POO (Programación Orientada a Objetos) a equipos futuros que aborden necesidades de integración, escalamiento y modificaciones modulares de LogiCheck sin mermar la estabilidad preexistente del framerate en la IA.

4. Documentar con celo académico, avalado por código fidedigno (esquemas de migración y directivas procedimentales reales), la culminación funcional y estructural del aplicativo.

INTRODUCCIÓN

El presente manual guiará al ingeniero inmerso en la plataforma LogiCheck a través de una documentación dividida coherentemente para revelar su comportamiento interno frente a condiciones comerciales arduas.

En el Capítulo 1, el lector ubicará los requerimientos físicos y entornos de software necesarios. El Capítulo 2 presentará a través del modelamiento diagramado universal la interacción y el flujo operativo. En el Capítulo 3 se efectúa la deconstrucción o 'disección' técnica, inspeccionando flujos visuales, concurrencia analítica compleja y el mapa tabular de retención SQLite. En el Capítulo 4 el despliegue es dictado paso por paso de forma replicable y validado por evidencia de consola, para por último estipular las limitantes lógicas continuas en el Capítulo 5 cerrando con los aportes al conocimiento investigado en el Capítulo 6.

El lector puede utilizar este registro progresivamente para un sondeo amplio de aprendizaje o referenciar específicamente las sub-secciones de 'Diccionario de Datos' y 'Archivos de Configuración' con fines estrictamente investigativos de arquitectura subyacente.

1. REQUISITOS DEL SISTEMA

1.1 Requisitos de hardware

LogiCheck realiza un uso intensivo y determinístico en cálculos vectoriales masivos. Como tal, el hardware trasciende el plano básico y demanda un acople gráfico específico.

- Procesador Mínimo: Intel Core i5 de 9.ª Generación o AMD Ryzen 5 equivalente. (Recomendado: Intel Core i7 12.ª Generación).

- Memoria RAM Principal: Mínimo 8 GB DDR4. (Recomendado: 16 GB DDR4/DDR5 para estabilizar cachés transitorios de OpenCV).

- Almacenamiento: Capacidad Mínima de 20 GB libres idealmente en un disco en formato NVMe SSD para maximizar los flujos de lectura en tensores PT.

- Arquitectura: Exclusivo para plataformas OS de 64 bits (x86_64 o amd64). LogiCheck maneja video en alta definición e Inteligencia Artificial al mismo tiempo, lo cual consume mucha memoria y potencia. Por eso, no puede correr en computadoras viejas de 32 bits; son necesarios los 64 bits para gestionar el gran volumen de datos sin colapsar el sistema.

- GPU (Requisito Indivisible): Tarjeta gráfica NVIDIA GTX 1650 con VRAM de 4 GB o superior para delegación de cómputo en CUDA Cores.

1.2 Requisitos de software

- Sistema operativo: Windows 10/11 Professional (Compilación de 64 bits con DirectX habilitado) o Distribuciones Linux certificadas por NVIDIA Drivers (Ej: Ubuntu 22.04).

- Lenguajes de programación: Python 3.10 o versiones minor release subsiguientes (exige soporte Tipificación Robusta estricta).

- Framework visual: PySide6 (Binding transicional nativo Qt for Python).

- Motores de base de datos: SQLite3. Es un motor relacional integrado directamente en el programa que no necesita de un servidor externo o conexión a internet para funcionar (está libre de sockets locales). Al funcionar como un archivo independiente dentro de la computadora, garantiza que LogiCheck sea extremadamente rápido, fácil de instalar y que los datos de la ferretería estén siempre seguros y disponibles sin depender de servicios de terceros.

- Librerías y Dependencias Críticas: El sistema utiliza un conjunto de herramientas especializadas que actúan como sus órganos vitales: OpenCV funciona como los "ojos" para procesar el video de las cámaras; Ultralytics (YOLO) es el "cerebro" encargado de la Inteligencia Artificial; PyMuPDF es el lector especializado que extrae datos de las facturas de Siigo; y Pandas es el organizador que gestiona grandes tablas de información. Sin estas librerías,

LogiCheck no podría realizar el ciclo completo de ver, pensar y auditar los despachos.

1.3 Configuraciones previas

Variables de Entorno: Se deben pre-instalar los binarios CUDNN/CUDA Runtime a nivel del PATH del Sistema Operativo Windows, vinculados estrictamente según coincidan contra la instalación del módulo 'Torch' definido.

Permisos/Red: Garantizar que la unidad del firewall admita tráfico saliente en Puerto TCP/UDP 554 para decodificación RTSP interna de las videocámaras IP y tráfico seguro HTTPS (Puerto 443) para el posteo de Notificaciones CallMeBot y Telegram.

1.4 Portabilidad y adaptabilidad

LogiCheck exhibe una elevada adaptabilidad producto de su aislamiento en un Virtual Environment (.venv). Funciona intrínsecamente igual sin requerir refactorizado, saltando de ambiente servidor hacia laptops locales. Resulta en una facilidad de actualización notable con solo emplear rutinas de `pip install --upgrade` dado el esquema abstracto.

1.5 Contingencia

Esta investigación revisa la estabilidad operativa del sistema LogiCheck, siendo coherente con las exigencias de procesamiento que demandan las redes neuronales en entornos locales.

Teniendo en cuenta la posible saturación de hardware, el diseño integra mecanismos preventivos contra el estrangulamiento térmico de la GPU y el desbordamiento de memoria mediante la liberación controlada de recursos. Siguiendo este hilo conductor, se establecieron rutinas de optimización que monitorean los tensores de memoria, garantizando la ejecución continua de los modelos YOLOv26 sin comprometer la interfaz. Según plantea la ingeniería de software moderna, la robustez es esencial para la calidad, por lo cual la arquitectura defensiva intercepta anomalías antes de cierres inesperados. De igual manera, se implementaron protocolos de transporte de video tolerantes a las fluctuaciones de red presentes en Apulo. Teniendo en cuenta que las transmisiones RTSP son susceptibles a latencias, el sistema incorpora una auto-recuperación silenciosa que reinicializa los flujos de auditoría.

Como dice la teoría sobre sistemas resilientes, estas medidas aseguran la verificación logística frente a factores externos, fortaleciendo en consecuencia la confiabilidad de los datos en la Ferretería Durán.

2. DIAGRAMAS DE MODELAMIENTO

Para abstraer la operativa empresarial de Ferretería Durán en consonancia algorítmica, se modela el comportamiento usando arquetipos de ingeniería altamente detallados:

Diagrama de Arquitectura Multihilo (Signals y QThread)

El siguiente diagrama presenta la organización de concurrrencia y flujo modular del sistema. Evidencia tajantemente la separación de tareas (Separation of Concerns). El sistema PySide ejerce de hilo maestro escuchando pulsaciones e interfaz en la Capa de Presentación. Separado de este hilo maestro, reside el núcleo analítico de YoloV26 y OpenCV encapsulados en algoritmos paralelos que se conectan finalmente a la base de datos local SQLite.

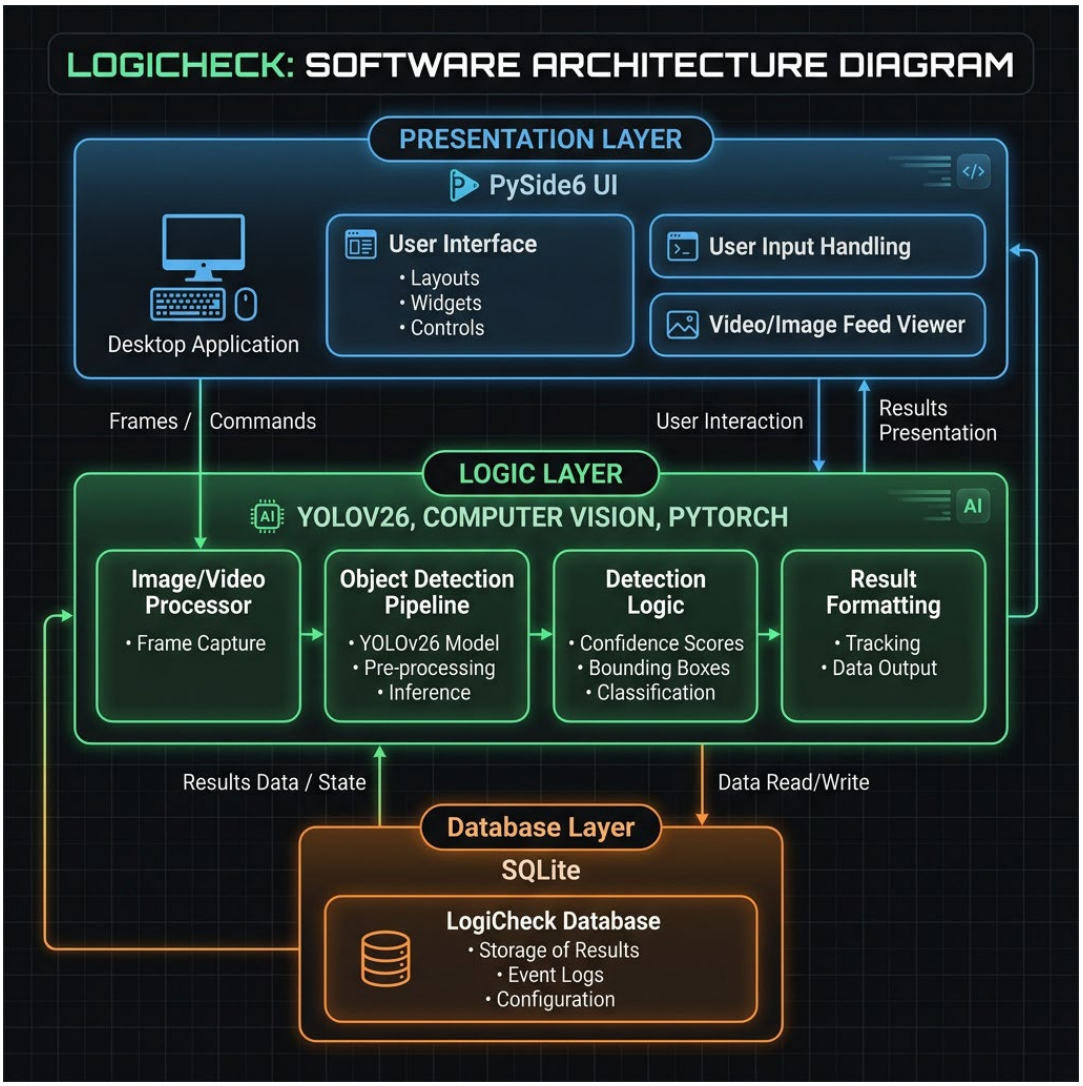

Figura 1: Diagrama de Arquitectura de Capas de LogiCheck (Presentación, Lógica y Datos). Generado con Nano Banana.

Propósito: Asegurar que el lector entienda que la aplicación no sufre bloqueos colaterales y separa el proceso UI y Negocio de manera formal y trazable.

Diagrama de Casos de Uso Sistemáticos

Relación interactiva del aplicativo: El Usuario ('Administrador'/'Operador') interactúa en el caso de uso 'Iniciar Jornada' llamando a 'ui.login_dialog'. Seguidamente, el caso 'Aprobar Carga de Factura' inicializa el 'invoice_parser.py' de modo hermético, retornando las métricas leídas directamente hacia los objetos observables en 'main_window.py'.

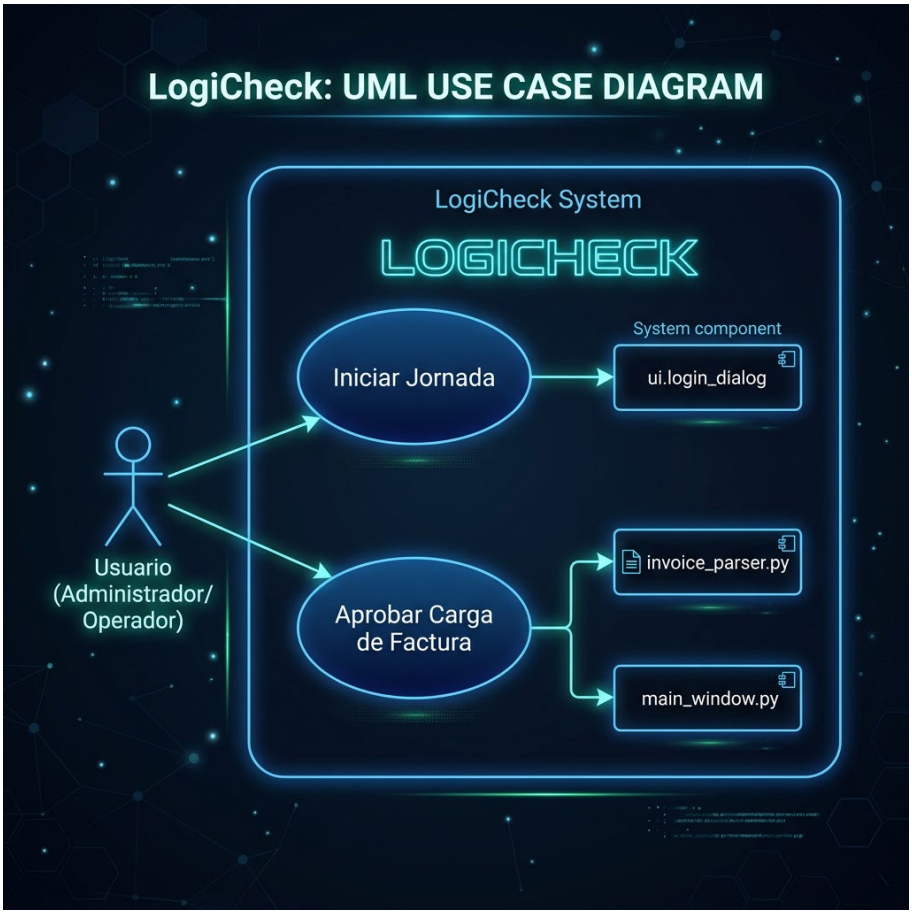

Figura 2: Diagrama de Casos de Uso - Interacción de Capas. Generado con Nano Banana.

3. ASPECTOS TÉCNICOS DEL DESARROLLO DEL SISTEMA

3.1 Arquitectura del sistema

La solución implementada posee un diseño en capas con alta cohesión y bajo acoplamiento, distinguiendo drásticamente el espacio donde interfiere el usuario del perímetro analítico IA.

a. Capa de presentación

Tecnologías utilizadas: Instanciada enteramente en entorno Python apoyada por PySide6 operando interfaces QWidgets tradicionales con estilos impulsados mediante Glassmorphism y la paleta de colores CSS/QSS Catppuccin. Esta capa gestiona las notificaciones on-screen, transiciones suaves (QPropertyAnimation) y alertas en Modal.

Forma de interacción: Orientada a eventos. El despachador invoca análisis cruzado dando un click tras confirmar un número de remisión. Al presionar el botón de disparo algorítmico, los componentes de la interfaz restringen el click transitoriamente mitigando peticiones redundantes.

i. Relación de directorios

Visualización de la distribución orientada al mantenimiento estructural de la aplicación en disco:

```
LogiCheck/
├── .venv/                [Librerías compiladas locales]
├── assets/               [Iconos de la capa de presentación]
├── core/                 [Lógica profunda del aplicativo]
│   ├── db_migrations.py  [Migraciones y scripts de SQLite]
│   ├── invoice_parser.py [Motor PDF Siigo con PyMuPDF]
│   └── yolo_manager.py   [IA con multihilo en QThread]
├── ui/                   [Vistas y Componentes GUI PySide6]
│   ├── main_window.py    [Renderizador central]
│   ├── splash_screen.py  [Vista de Carga de librerías CUDA]
│   └── login_dialog.py   [Formulario de Autenticación]
├── resources/
│   └── style.qss         [Hoja de Configuracion CSS Visual]
├── logicheck_users.db    [Archivo Físico de Base de Datos]
├── main.py               [Archivo Inicializador Maestro]
└── requirements.txt
```

ii. Mapa de navegación o estructura del sistema

1. Pantalla Inicial (ui.login_dialog): Formulario aislado y modal demandando credenciales para protección RBAC.
2. Splash Screen Dinámico (ui.splash_screen): Sirve como amortiguador para instanciar en RAM la carga pesada del modelo Pytorch sin pasmar agresivamente el Front-end.
3. Main Window / Interfaz Maestra (ui.main_window): Donde coexisten sub-ventanas (Submenús integrados en Layouts y Widgets QStackedLayout). En este lienzo se transmiten las gráficas, tableros de registro fotográfico forense y despliegue del Grid principal de cámaras IP.

Para ilustrar este lienzo interactivo, la siguiente captura real refleja la distribución del panel de control de LogiCheck ('Dashboard') durante una jornada de auditoría activa. Se exponen los Bounding Boxes de Inteligencia Artificial superpuestos en tiempo real al video y los paneles laterales con las comparativas simultáneas frente a la factura SIIGO:

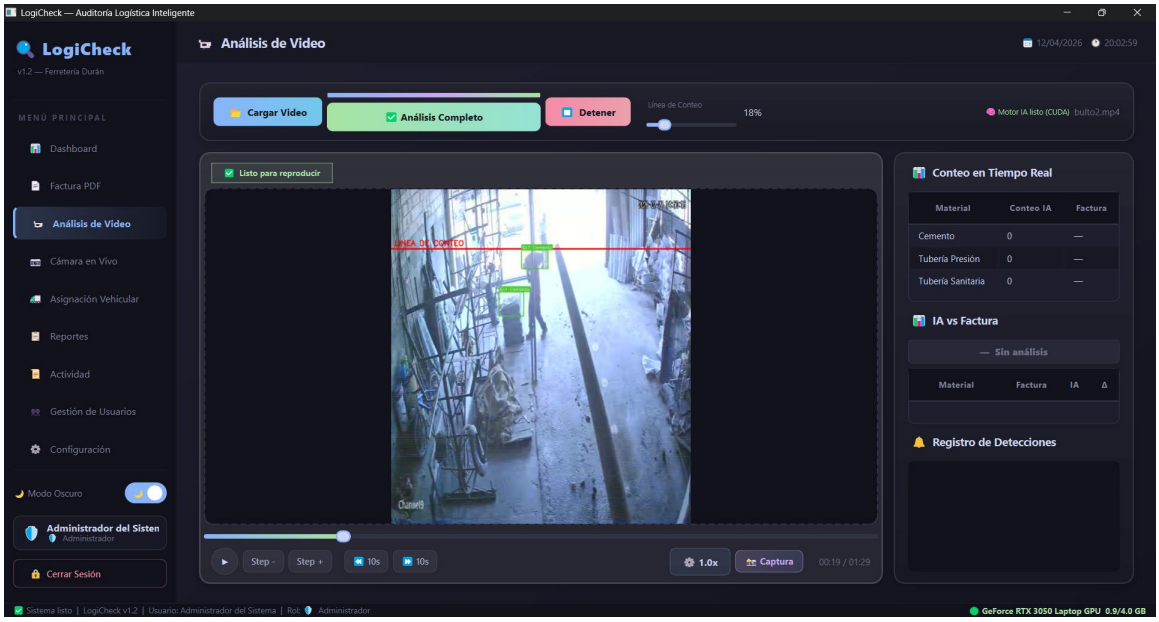

Figura 3: Dashboard Real de LogiCheck – Empleando IA Yolo en la detección y UI Catppuccin.

iii. Archivos de configuración

- Nombre de Archivo: requirements.txt. Sujeta el versionamiento absoluto dependiente de los frameworks para lograr clonaciones idénticas de Python.
- Nombre de Archivo: resources/style.qss. Retiene las variables genéricas de estilos (colores base de catppuccin, curvaturas px) asumiendo un papel análogo

a un CSS global de arquitectura web, evitando la inyección dura de hojas de estilo (hardcoded) en cada código Python UI.

b. Capa Lógica y Visión Artificial
El corazón intelectual del aplicativo. Cada píxel recuperado por OpenCV es matemáticamente normalizado de rango matricial int de 0 a 255 a notación fraccional continua [0,1], sufriendo además una mutación espacial (Letterbox Resize). Posteriormente se inyecta en el objeto Yolo de Ultralytics (yolo26). Resulta imprescindible la especificación técnica de la evaluación lógica:

- Confidence Threshold = 0.20 (Asume predicciones por encima del veinte por ciento permitiendo operar con baja luminosidad o desenfoque en la bodega).

- IoU Threshold = 0.5 (Intersect on Union, encargado de impedir sobre-conteos al ignorar cajas delimitadoras colindantes o encima de las mismas si se empalman). Con los datos ya asimilados en tensores CUDA, se envían directivas por Signal Event Loops liberando al 'Main Thread'.

c. Capa de base de datos
Empleamos SQLite3. Optimizada debido a ser relacional, transaccional por lotes (Atomicity) y altamente portátil pues no exige correr como sub-proceso demonio en el host. La información cuenta con rigurosidad impidiendo corrupciones asíncronas de escritura.

i. Diagrama entidad–relación
La entidad fuerte base 'usuarios' alberga los datos biográficos y permisos operacionales fijos. Esta distribuye una cardinalidad transversal delegándose en la tabla foránea 'auditorias', que asume las métricas de inspección en las remisiones cruzadas de SIIGO e IA. Si un usuario orquesta doce auditorías diurnas, existirá formalidad cardinal donde 1 id de empleado corresponde a N (1:N) despachos guardados por él mismo resguardado bajo clave foránea (Foreign Key referencial).

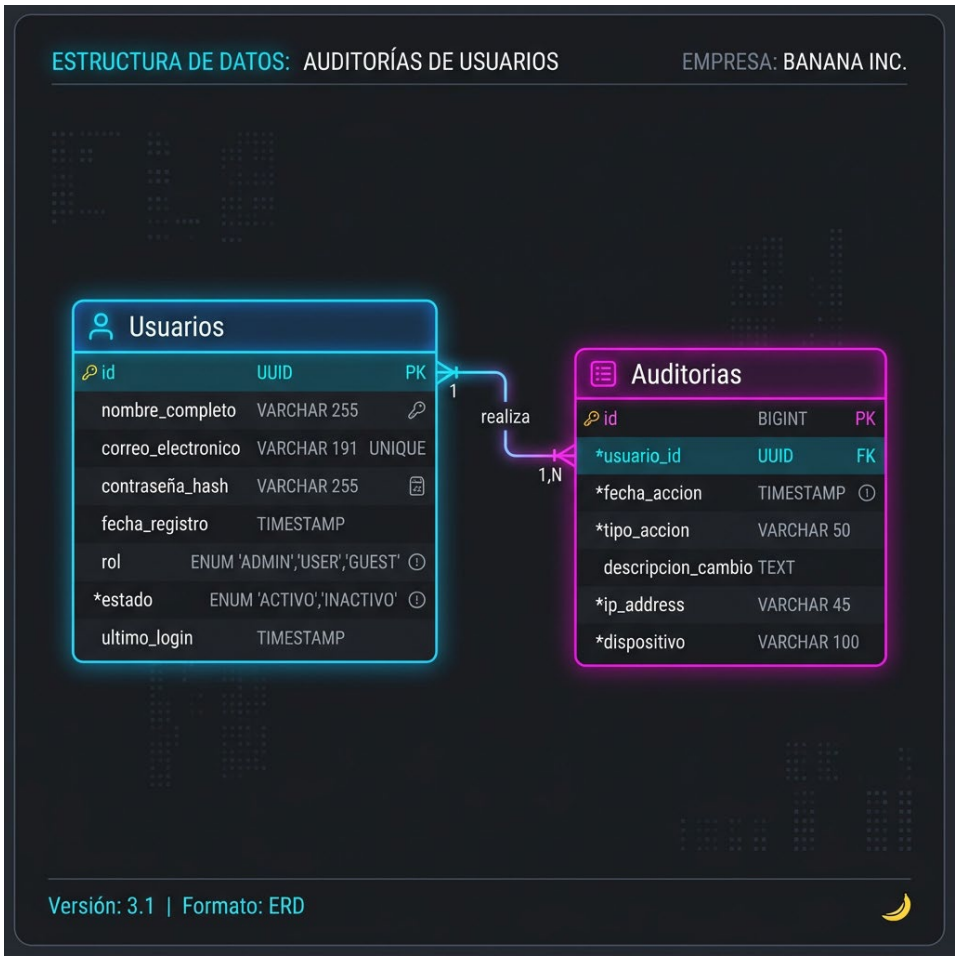

Figura 4: Diagrama Entidad-Relación entre Usuarios y las Auditorías Generadas. Generado con Nano Banana, en base a información Sqlite.

ii. Diccionario de datos
Estructura primaria forense recabada para mantenimiento de esquema:

Table 1. Estructura primaria forense

Campo de Tabla	Tipo Dato Primitivo	Restricciones / Integridad Foránea
[usuarios].id	INTEGER (PK)	PRIMARY KEY AUTOINCREMENT. Pilar de toda relación de llave ajena.
[usuarios].username	TEXT	UNIQUE NOT NULL. Reclamo de ingreso irrepetible.
[usuarios].password_hash	TEXT	NOT NULL. Cifrado unidireccional de la contraseña.
[usuarios].password_salt	TEXT	NOT NULL. Entropía aleatoria para escudar hashes de diccionarios pre-computados.
[usuarios].role	TEXT	NOT NULL. Nivel gerencial/técnico de RBAC.
[auditorias].id	INTEGER (PK)	Identificador universal de registro remisional.
[auditorias].user_id	INTEGER (FK)	FK (usuarios.id) ON DELETE SET NULL. Protege dependencias de usuario eliminado.
[auditorias].fecha	TEXT/DATETIME	DEFAULT (datetime('now')). Almacena instante del clic temporal exacto.
[auditorias].factura_no	TEXT	DEFAULT. Número correlacional extraído por Invoice Parsing.
[auditorias].resultado	TEXT	Condición Semántica dictada POST IA: Ej. CONFORME, DISCREPANCIA.

Para impedir la impunidad documental a nivel de motor (Engine), aplicamos Triggers DDL inmutables que crean logs automáticos en 'auditorias_historico' ante modificaciones:

```
CREATE TRIGGER IF NOT EXISTS trg_auditorias_update
AFTER UPDATE ON auditorias
BEGIN
   INSERT INTO auditorias_historico (auditoria_id, operacion,
usuario_modificador, datos_anteriores, datos_nuevos)
   VALUES (NEW.id, 'UPDATE', NEW.username,
          'Resultado: ' || OLD.resultado || ', Vehículo: ' || OLD.vehiculo,
          'Resultado: ' || NEW.resultado || ', Vehículo: ' || NEW.vehiculo);
END;
```

- **Tabla: VEHICULOS**

Descripción:

Almacena la información de los vehículos disponibles para el transporte de materiales en los despachos de la Ferretería Durán. Cada vehículo puede ser asociado a múltiples auditorías de carga**.**

Table 2. Vehículos

TABLA: VEHICULOS					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
id	INT	-	PK	NOT NULL	Identificador único del vehículo. Se genera automáticamente.
código	VARCHAR	50	-	UNIQUE	Código interno de identificación del vehículo dentro del sistema.
tipo	VARCHAR	50	-	-	Tipo de vehículo: motocarro, camión, camioneta, entre otros.
placa	VARCHAR	20	-	UNIQUE	Placa de identificación del vehículo. No puede repetirse.
capacidad_max_peso_kg	DECIMAL	10,2	-	-	Capacidad máxima de carga en kilogramos

TABLA: VEHICULOS					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
					que soporta el vehículo.
capacidad_max_vol_m3	DECIMAL	10,2	-	-	Capacidad máxima de carga en metros cúbicos del vehículo.
activo	BOOLEAN	-	-	NOT NULL	Indica si el vehículo está disponible (TRUE) o fuera de servicio (FALSE).
created_at	DATETIME	-	-	NOT NULL	Fecha y hora en que el vehículo fue registrado en el sistema.

Relaciones:

- Un vehículo puede estar asociado a múltiples auditorías de despacho (relación 1:N con AUDITORIAS).

- **Tabla: MATERIALES**

Descripción:
Almacena el catálogo de materiales de construcción reconocidos por el sistema para su detección y verificación durante el proceso de despacho. Las categorías registradas corresponden a las clases detectables por el modelo de visión artificial YOLOv26.

Table 3. Materiales

TABLA: MATERIALES					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
id	INT	-	PK	NOT NULL	Identificador único del material. Se

TABLA: MATERIALES					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
					genera automáticamente.
código	VARCHAR	50	-	UNIQUE	Código interno de identificación del material dentro del catálogo.
nombre	VARCHAR	150	-	NOT NULL	Nombre descriptivo del material: cemento, tubería de presión, tubería sanitaria.
categoría	VARCHAR	100	-	-	Categoría a la que pertenece el material, usada para agrupación en reportes.
peso_unitario_kg	DECIMAL	10,2	-	-	Peso unitario del material en kilogramos. Usado para cálculo de carga.
volumen_unitario_m3	DECIMAL	10,4	-	-	Volumen unitario del material en metros cúbicos. Usado para cálculo de carga.
activo	BOOLEAN	-	-	NOT NULL	Indica si el material está activo (TRUE) o desactivado (FALSE) en el catálogo.
created_at	DATETIME	-	-	NOT NULL	Fecha y hora en que el material fue registrado en el sistema.

Relaciones:

- Un material puede aparecer en múltiples auditorías (relación N:M con AUDITORIAS, resuelta mediante AUDITORIA_MATERIALES).

- **Tabla: AUDITORIAS**

Descripción:
Constituye la entidad central del sistema LogiCheck. Almacena el registro completo de cada proceso de verificación de despacho realizado, integrando la información proveniente de la factura electrónica con los resultados del análisis de video mediante visión artificial.

Table 4. Auditorias

TABLA: AUDITORIAS					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
id	INT	-	PK	NOT NULL	Identificador único de la auditoría. Se genera automáticamente.
user_id	INT	-	FK	NOT NULL	ID del usuario que realizó la auditoría. Referencia a la tabla USUARIOS.
vehiculo_id	INT	-	FK	-	ID del vehículo utilizado en el despacho. Referencia a la tabla VEHICULOS.
fecha	DATETIME	-	-	NOT NULL	Fecha y hora en que se realizó la auditoría de despacho.
factura_no	VARCHAR	100	-	-	Número de la factura electrónica asociada al despacho verificado.
cliente	VARCHAR	200	-	-	Nombre del cliente al que corresponde el

TABLA: AUDITORIAS					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
					despacho facturado.
video_nombre	VARCHAR	255	-	-	Nombre del archivo de video procesado durante la auditoría.
conteo_ia	JSON	-	-	-	Resultado del conteo de materiales detectados por el modelo de IA en formato JSON.
conteo_factura	JSON	-	-	-	Cantidades de materiales registradas en la factura en formato JSON.
discrepancias	JSON	-	-	-	Diferencias calculadas entre el conteo de IA y el conteo de factura en formato JSON.
resultado	VARCHAR	50	-	-	Resultado general de la auditoría: conforme o con_discrepancias.
notas	TEXT	-	-	-	Observaciones adicionales registradas por el usuario durante la auditoría.
capturas	JSON	-	-	-	Rutas o referencias de las capturas de pantalla generadas durante el análisis.

Relaciones:

- Una auditoría pertenece a un único usuario (relación N:1 con USUARIOS).
- Una auditoría puede estar asociada a un vehículo (relación N:1 con VEHICULOS).

- Una auditoría puede involucrar múltiples materiales (relación 1:N con AUDITORIA_MATERIALES).
- Una auditoría puede generar múltiples notificaciones (relación 1:N con NOTIFICACIONES_QUEUE).
- Una auditoría puede tener múltiples registros históricos de cambios (relación 1:N con AUDITORIAS_HISTORICO).

- **Tabla: AUDITORIA_MATERIALES**

Descripción:

Tabla intermedia que resuelve la relación muchos a muchos entre AUDITORIAS y MATERIALES. Registra el detalle de cada material verificado en una auditoría, almacenando las cantidades detectadas por IA, las registradas en la factura y la diferencia calculada entre ambas.

Table 5. Auditoria_materiales

TABLA: AUDITORIA_MATERIALES					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
id	INT	-	PK	NOT NULL	Identificador único del registro de detalle. Se genera automáticamente.
auditoria_id	INT	-	FK	NOT NULL	ID de la auditoría a la que pertenece este detalle. Referencia a AUDITORIAS.
material_id	INT	-	FK	NOT NULL	ID del material verificado en la auditoría. Referencia a MATERIALES.
cantidad_ia	INT	-	-	-	Cantidad de unidades del material detectadas por el modelo de visión artificial.
cantidad_factura	INT	-	-	-	Cantidad de unidades del material registradas en la factura electrónica.

TABLA: AUDITORIA_MATERIALES					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
diferencia	INT	-	-	-	Diferencia entre la cantidad facturada y la detectada (cantidad_ia - cantidad_factura).

Relaciones:

- Cada registro pertenece a una única auditoría (relación N:1 con AUDITORIAS).
- Cada registro referencia a un único material del catálogo (relación N:1 con MATERIALES).

- **Tabla: ACTIVITY_LOGS**

Descripción:

Almacena el registro cronológico de todas las acciones relevantes realizadas por los usuarios dentro del sistema, tales como inicios de sesión, creación de auditorías, modificación de usuarios, generación de reportes y cambios de rol. Permite la auditoría y el seguimiento del uso del sistema.

Table 6. Activity_logs

TABLA: ACTIVITY_LOGS					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
id	INT	-	PK	NOT NULL	Identificador único del registro de actividad. Se genera automáticamente.
user_id	INT	-	FK	-	ID del usuario que realizó la acción. Referencia a la tabla USUARIOS.
action	VARCHAR	100	-	-	Tipo de acción ejecutada: login, create_audit, update_user, generate_report, entre

TABLA: ACTIVITY_LOGS					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
					otras.
description	TEXT	-	-	-	Descripción detallada de la acción realizada, con contexto relevante del evento.
timestamp	DATETIME	-	-	NOT NULL	Fecha y hora exactas en que se registró la acción en el sistema.

Relaciones:

- Cada registro de actividad está asociado a un usuario (relación N:1 con USUARIOS).

- **Tabla: AUDITORIAS_HISTORICO**

Descripción:

Almacena el historial de modificaciones realizadas sobre los registros de auditoría. Cada vez que se modifica una auditoría, el sistema guarda el estado anterior y el nuevo junto con el usuario responsable del cambio, garantizando la trazabilidad e integridad de la información.

Table 7. Auditorias_historico

TABLA: AUDITORIAS_HISTORICO					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
id	INT	-	PK	NOT NULL	Identificador único del registro histórico. Se genera automáticamente.
auditoria_id	INT	-	FK	NOT NULL	ID de la auditoría que fue modificada. Referencia a la tabla

TABLA: AUDITORIAS_HISTORICO					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
					AUDITORIAS.
usuario_modificador	INT	-	FK	-	ID del usuario que realizó la modificación. Referencia a la tabla USUARIOS.
operacion	VARCHAR	50	-	-	Tipo de operación realizada sobre la auditoría: UPDATE, DELETE u otra acción.
fecha_cambio	DATETIME	-	-	NOT NULL	Fecha y hora exactas en que se realizó la modificación sobre la auditoría.
datos_anteriores	JSON	-	-	-	Estado de los campos de la auditoría antes de aplicar la modificación.
datos_nuevos	JSON	-	-	-	Estado de los campos de la auditoría después de aplicar la modificación.

Relaciones:

- Cada registro histórico corresponde a una auditoría específica (relación N:1 con AUDITORIAS).
- Cada registro histórico está asociado al usuario que realizó el cambio (relación N:1 con USUARIOS).

- **Tabla: NOTIFICACIONES_QUEUE**

Descripción:

Gestiona la cola de notificaciones generadas por el sistema ante eventos relevantes, principalmente discrepancias detectadas en el proceso de comparación entre los materiales despachados y los registrados en la factura. Almacena el estado de procesamiento de cada notificación.

Table 8. Notificaciones_queue

TABLA: NOTIFICACIONES_QUEUE					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
id	INT	-	PK	NOT NULL	Identificador único de la notificación. Se genera automáticamente.
auditoria_id	INT	-	FK	-	ID de la auditoría que generó la notificación. Referencia a AUDITORIAS.
tipo	VARCHAR	50	-	-	Tipo de notificación generada: discrepancia, alerta u otro evento relevante.
mensaje	TEXT	-	-	-	Contenido del mensaje de la notificación con el detalle del evento detectado.
destinatario	VARCHAR	150	-	-	Usuario o rol al que va dirigida la notificación dentro del sistema.
estado	VARCHAR	50	-	-	Estado actual de la notificación: pendiente, enviada o con_error.

TABLA: NOTIFICACIONES_QUEUE					
Campo	**Tipo**	**Tamaño**	**PK/FK**	**Restricción**	**Descripción**
intentos	INT	-	-	-	Número de intentos realizados para procesar y enviar la notificación.
fecha_creacion	DATETIME	-	-	NOT NULL	Fecha y hora en que fue generada la notificación por el sistema.
fecha_envio	DATETIME	-	-	-	Fecha y hora en que la notificación fue procesada y enviada exitosamente.

Relaciones:

- Cada notificación está asociada a una auditoría específica (relación N:1 con AUDITORIAS).

4. INSTALACIÓN DEL SISTEMA Y SERVICIOS

4.1 Instalación del sistema (Paso a Paso)

Para inyectar el sistema LogiCheck por primera vez de manera pulcra:

1. Virtual Environment Setup (Requisito Previo): Utilizar la CLI o Windows Console dentro de la raíz clonada del proyecto y emitir la instancia virgen de Python.

```
C:\Users\...> python -m venv .venv
C:\Users\...> .venv\Scripts\activate
(.venv) C:\Users\...>
```

2. Descarga de Archivos de Repositorios Pypi: Al consolidar la máquina aislada (denotada por paréntesis en la consola CLI), mandar la cascada analítica con PIP:

```
(.venv) C:\Users\...> pip install -r requirements.txt
Collecting PySide6==6.8.1 ...
[100%] Successfully installed opencv-python ultralytics pymupdf requests
```

4.2 Instalación de servicios adicionales

LogiCheck actúa bajo arquitectura 'Stand-alone' sin necesidad de correr contenedores adicionales a nivel de red debido a que la BD SQLite3 se ejecuta

desde sus propias bibliotecas embebidas, omitiendo puertos DDL engorrosos de PostgreSQL/MySQL en máquinas cliente.

5. REQUERIMIENTOS DE HARDWARE (OPERACIÓN)

Para gozar de plena estabilidad algorítmica ininterrumpida de Lunes a Sábado en operación logística continua, el hardware mandatorio de operación establece límites y cotas obligatorias para CUDA.

Hardware de Escalabilidad Básica: Como se listaba, la GTX 1650 con 4GB es el umbral para 2 a 4 cámaras RTSP simultáneas (Streams de 720p). En proyecciones de una Escalabilidad Robusta intentando alojar la malla completa admisible para LogiCheck de 16 Canales Simultáneos, las recomendaciones de hardware migran obligatoriamente a soluciones Enterprise GPUs (Ej. Serie NVIDIA RTX 3060 de 12GB VRAM o hardware de Grado Data Center Tesla T4) para asegurar que el pool de tensores PyTorch que mantienen un 'Thread' por canal y no sucumba a excepciones fatales del recolector de basura de la tarjeta gráfica (Memory Overflow).

6. BIBLIOGRAFÍA

PySide6 Documentation. (2026). Qt for Python Reference Handbook.

Ultralytics, LLC. (2026). YOLO Framework, Convolutional Foundations, and Tracking Paradigms.

NVIDIA Corporation. (2026). CUDA Platform, Unified Memory Model and Tensor Parallelization Compute Arch.

SQLite Consortium. (2026). SQL As Understood By SQLite DB Engine and Relational Foreign Key Actions.