CLAD

MINISTÉRIO DA
GESTÃO E DA INOVAÇÃO
EM SERVIÇOS PÚBLICOS

GOVERNO FEDERAL
BRASIL
UNIÃO E RECONSTRUÇÃO

Cooperación Técnica CLAD-DGSC: Inteligencia Artificial Aplicada al Reclutamiento y Selección en el Servicio Civil de Costa Rica

Informe final

Elaborado por el consultor
Hugo Medeiros Souto

San José, Costa Rica
Marzo 2026

Tabla de Contenido

1 Ficha Técnica y Avisos Legales

1.1 Ficha Técnica

Campo	Valor
Título	Cooperación Técnica CLAD-DGSC: Inteligencia Artificial Aplicada al Reclutamiento y Selección en el Servicio Civil de Costa Rica
Institución beneficiaria	Dirección General de Servicio Civil (DGSC), Costa Rica
Ministerio vinculado	Ministerio de Planificación Nacional y Política Económica (MIDEPLAN), Costa Rica
Marco institucional	Programa de Cooperación Técnica Horizontal del CLAD
Consultor	Hugo Medeiros Souto, del MGI Ministério da Gestão e Inovação em Serviços Públicos (MGI), Brasil
Período de la consultoría	Enero — Marzo 2026
Fecha del informe	Marzo 2026
Idioma original	Portugués (Brasil)

1.2 Aviso Legal

Las opiniones expresadas en este documento técnico son exclusivamente del autor y no reflejan necesariamente el punto de vista del Centro Latinoamericano de Administración para el Desarrollo (CLAD), de su Secretaría General, ni de los países miembros que representa. Las referencias a tecnologías, productos o servicios específicos tienen finalidad estrictamente descriptiva y no constituyen aval institucional.

El presente informe fue elaborado en el marco del Programa de Cooperación Técnica Horizontal del CLAD, mecanismo que promueve el intercambio de experiencias y buenas prácticas entre las administraciones públicas iberoamericanas, en consonancia con los principios consagrados en las Cartas Iberoamericanas y en las Declaraciones del CLAD sobre reforma del Estado y modernización de la gestión pública.

1.3 Sobre el CLAD

El Centro Latinoamericano de Administración para el Desarrollo (CLAD) es un organismo público internacional de carácter intergubernamental, constituido en

1972 bajo la iniciativa de los gobiernos de México, Perú y Venezuela. En la actualidad, integra 24 países miembros de la comunidad iberoamericana. Su mandato fundamental es la promoción del análisis y del intercambio de experiencias y conocimientos en torno a la reforma del Estado y la modernización de la Administración Pública.

2 Presentación

En las últimas décadas, la transformación digital de los Estados ha sido reconocida como eje estratégico para la modernización de las administraciones públicas iberoamericanas. La Carta Iberoamericana de Gobierno Electrónico (CLAD, 2007) ya anticipaba que la incorporación de tecnologías de la información y comunicación en la gestión pública no constituye un fin en sí mismo, sino un instrumento al servicio de un Estado más eficiente, transparente y cercano a la ciudadanía. Casi dos décadas después, la emergencia de tecnologías de inteligencia artificial ofrece una nueva dimensión a ese compromiso — la posibilidad de automatizar procesos cognitivos que, hasta hace poco, dependían exclusivamente del juicio humano.

En este contexto, la gestión del talento humano en el sector público se presenta como uno de los dominios con mayor potencial de beneficio. Los procesos de reclutamiento y selección en el servicio civil, por su naturaleza regulatoria y documental, involucran la verificación repetitiva de requisitos contra normas predefinidas — tarea que combina alto volumen de trabajo con rigor en la aplicación de criterios legales. Es precisamente en esa intersección entre escala y precisión donde la inteligencia artificial puede ofrecer una contribución sustantiva, no para sustituir el juicio profesional de los servidores públicos, sino para potenciarlo mediante la organización y preprocesamiento de evidencia documental.

2.1 El Contexto Costarricense

La Dirección General de Servicio Civil (DGSC) de Costa Rica, por medio del Área de Gestión de Empleo, es la instancia responsable de comprobar la idoneidad y proveer personal idóneo a las Oficinas de Gestión Institucional de Recursos Humanos (OGEREH) de las 46 instituciones públicas que integran el Régimen de Servicio Civil (RSC). Con más de seis décadas de experiencia institucional, la DGSC administra el proceso que vincula ciudadanos al servicio público costarricense — una función que, por el volumen y la complejidad regulatoria involucrados, exige constante modernización de los instrumentos y metodologías de trabajo.

Con la entrada en vigor de la **Ley Marco de Empleo Público N°1015G** (LMEP), del 10 de marzo de 2023, las competencias y responsabilidades de la DGSC fueron significativamente ampliadas. La nueva ley orienta a la institución hacia un **rol más**

estratégico que incluye la generación de directrices, fiscalización y asistencia técnica para todo el Sistema General de Empleo Público, al mismo tiempo que contempla la delegación de tareas operativas a las oficinas de recursos humanos de las instituciones del RSC. Esta doble dinámica — más responsabilidad estratégica y más presión por eficiencia operativa — hace urgente la modernización de los procesos internos.

Paralelamente, el **Proyecto Hacienda Digital del Bicentenario** (Préstamo 9075-CR, Banco Mundial) impulsa la interoperabilidad institucional y la simplificación de trámites mediante la implementación de un Sistema Integrado de Administración Financiera y Talento Humano. El módulo de Gestión del Talento Humano, incluido en el alcance del proyecto, prevé funcionalidades que se superponen directamente al dominio de la validación de atestados — creando tanto una oportunidad de integración futura como la necesidad de una solución operativa que funcione en el ínterin. Sin embargo, **la inteligencia artificial quedó explícitamente fuera del alcance** del Hacienda Digital. Fue esta brecha estratégica — entre el mandato ampliado por la LMEP, la modernización tecnológica en curso y la ausencia de componente de IA — la que motivó a la DGSC a recurrir a la cooperación técnica horizontal del CLAD.

2.2 La Escala del Desafío

Validar la idoneidad de un candidato al servicio público costarricense exige confrontar diversos documentos presentados — títulos, certificaciones, experiencia laboral, habilitaciones profesionales — con los requisitos específicos de su clase funcional, incluyendo la verificación ante fuentes externas oficiales. Son 117 clases distribuidas en 34 series, cada una con combinaciones propias de requisitos académicos, experiencia y habilitaciones legales. Solamente durante el período de 2024, la Unidad de Administración de Concursos atendió **410 procesos de reclutamiento**, procediendo con la validación de **58.451 ofertas** de **21.447 candidatos** — cada candidato pudiendo presentar ofertas para múltiples clases simultáneamente. Este volumen es procesado por un equipo de aproximadamente 7 analistas profesionales y 1 técnica, utilizando una infraestructura de servidores sin GPU dedicada y con 16GB de RAM por máquina virtual — restricción que añade una capa de complejidad a cualquier iniciativa de automatización basada en inteligencia artificial generativa.

El proceso actual es enteramente manual: los analistas reciben documentos en formato PDF por correo electrónico, los clasifican visualmente por tipo, extraen información relevante mediante lectura atenta, consultan ocho o más sistemas y portales externos para verificación de datos, y aplican reglas de validación que varían según la clase del cargo y el tipo de concurso. Se trata, en esencia, de un conjunto de tareas cognitivas — reconocimiento visual, clasificación, extracción de datos y razonamiento basado en reglas — realizadas íntegramente por personas, y que son precisamente las capacidades que los avances recientes en modelos de lenguaje y visión computacional han hecho accesibles y operables a escala por máquinas.

2.3 La Cooperación Técnica

Es en este escenario donde se inscribe la presente cooperación técnica horizontal del CLAD. Atendiendo la solicitud de la DGSC, el CLAD designó un consultor especializado en inteligencia artificial aplicada a la gestión pública para asesorar a la institución en la automatización del procesamiento de los atestados presentados por la ciudadanía en el Reclutamiento Abierto y Permanente. El trabajo fue desarrollado a lo largo de tres meses (enero a marzo de 2026) y comprendió cinco ejes fundamentales.

En primer lugar, se procedió al **mapeo completo del proceso AS-IS**, mediante 9 sesiones de trabajo remotas con el equipo DGSC, incluyendo demostraciones del proceso con casos reales y reunión formal de validación del mapeo. Este levantamiento detallado — que identificó 7 fases, 9 gates de validación, 48 micropasos y 11 fuentes externas de verificación — constituye, hasta donde se pudo verificar, el primer mapeo integral del proceso de validación de atestados realizado en la DGSC.

En segundo lugar, se realizó una **evaluación tecnológica rigurosa**, con benchmark comparativo de modelos de inteligencia artificial para reconocimiento óptico de caracteres (OCR) y extracción estructurada de datos, probados en la infraestructura efectivamente disponible en la DGSC. Los criterios de selección priorizaron la viabilidad operativa — modelos que efectivamente funcionan en el hardware existente — y la soberanía de datos, descartando cualquier solución que requiera el envío de datos personales a la nube.

En tercer lugar, se desarrolló el **diseño de la arquitectura de solución**, concibiendo un sistema de validación inteligente con observabilidad nativa, revisión humana obligatoria (*human-in-the-loop*) y auditabilidad de cada decisión — principios que responden no solo a buenas prácticas de ingeniería, sino a los requisitos de transparencia y rendición de cuentas propios de la administración pública costarricense.

En cuarto lugar, la arquitectura fue proyectada para **dos escenarios de evolución progresiva**: una Prueba de Concepto (POC) standalone que opera en la infraestructura existente sin dependencias externas, y una trayectoria de integración con el módulo FreeBalance CSM del Hacienda Digital, asegurando que cada fase de implementación agrega valor de manera independiente.

Finalmente, se elaboraron **recomendaciones estratégicas y operativas**, incluyendo la propuesta de un formulario de presentación preestructurado que, independientemente de la adopción de IA, puede reducir significativamente el esfuerzo de clasificación manual de los analistas.

El presente informe documenta íntegramente los resultados de esta cooperación, con el doble objetivo de servir como referencia técnica para la implementación por

parte del equipo de la DGSC y como registro institucional de la experiencia para el acervo de cooperación técnica del CLAD. Se espera que el enfoque aquí documentado — basado en modelos de código abierto, infraestructura on-premise y motor de reglas configurable — pueda inspirar iniciativas similares en otros países miembros del CLAD que enfrenten desafíos análogos en la gestión del talento humano público.

---

3 Resumen Ejecutivo

Validar la idoneidad de un candidato al servicio público costarricense exige confrontar diversos documentos presentados — títulos, certificaciones, experiencia laboral, habilitaciones profesionales — con los requisitos específicos de su clase funcional, incluyendo la verificación ante fuentes externas oficiales. El área de Gestión de Empleo de la Dirección General de Servicio Civil (DGSC) de Costa Rica es la responsable de realizar esta validación para las 46 instituciones del Régimen de Servicio Civil. Son 117 clases distribuidas en 34 series, cada una con combinaciones propias de requisitos. En 2024, un equipo de aproximadamente 10 analistas procesó 58.451 ofertas de 21.447 candidatos. La Ley Marco de Empleo Público N°10159 amplió este mandato; el Proyecto Hacienda Digital, que modernizará la infraestructura tecnológica del Estado, no contempla inteligencia artificial. Fue esta brecha estratégica la que motivó la cooperación técnica con el CLAD.

El trabajo, desarrollado entre enero y marzo de 2026, comprendió tres fases: diagnóstico y mapeo del proceso actual con el equipo operativo — produciendo el primer mapeo integral del proceso de validación de atestados en la DGSC —, evaluación de tecnologías en la infraestructura efectivamente disponible, y diseño de una arquitectura de solución que concilia las necesidades operativas con los requisitos de auditabilidad de la administración pública costarricense.

3.1 Hallazgos Principales

El diagnóstico reveló que el problema central no es de automatización de flujos administrativos — como inicialmente se supuso — sino de **comprensión documental**: extraer información estructurada de PDFs heterogéneos (escaneados, digitales, en formatos variados) y validarla contra reglas complejas. Esta distinción es determinante: requiere modelos de visión computacional y procesamiento de lenguaje natural, no solamente scripts de manipulación de datos. Sin embargo, como los documentos procesados contienen **datos personales de candidatos** que desean optar por puestos vacantes dentro **del servicio público** — números de cédula, direcciones, registros de seguridad social —, el envío a APIs de modelos comerciales en la nube viola los requisitos de protección de datos y soberanía institucional. La solución debe operar **100% on-premise**, lo que impone la utilización de modelos de código abierto dimensionados para el hardware existente (CPU-only, 16GB de RAM). La sofisticación necesaria es hoy alcanzable con

estos modelos de pequeño porte, sin costos recurrentes de licenciamiento — no por limitación, sino como consecuencia directa del imperativo de soberanía.

Las sesiones de trabajo con los analistas revelaron dos hallazgos complementarios. Primero, que **la presentación de documentos es excesivamente "suelta"**: los candidatos envían documentos innecesarios, en mala calidad, sin estandarización de nomenclaturas o agrupamiento por tipo — lo que amplifica todo el esfuerzo de clasificación y fundamenta la recomendación del formulario preestructurado. Segundo, que **la mayor fuente de ineficiencia no es la lectura de PDFs**, sino la verificación repetitiva de datos en 8 o más portales externos — CONESUP, CCSS, MEP, colegios profesionales —, cada uno con su propia interfaz y método de acceso. Este segundo hallazgo reorienta las prioridades de la solución y fundamenta la recomendación de integración progresiva con la interfaz de interoperabilidad del Hacienda Digital.

Una auditoría de calidad reveló que la extracción inicial de los manuales de clases por LLM produjo **datos plausibles pero fabricados** — 0% de precisión en los códigos de la Serie Enfermería. Este incidente, aunque crítico, valida la decisión arquitectónica de observabilidad nativa y human-in-the-loop, y contiene una lección importante para cualquier iniciativa que utilice IA en contextos regulatorios: la validación humana de outputs de LLM no es precaución excesiva — es requisito de integridad.

3.2 Solución Propuesta

El sistema propuesto es un flujo de procesamiento (pipeline) de validación inteligente que opera **100% on-premise** en la infraestructura existente de la DGSC (8 VCPUs, 16GB RAM, sin GPU), compuesto por cinco etapas: sanitización de PDFs (pikepdf), reconocimiento óptico (PaddleOCR-VL-1.5), clasificación de documentos (heurística + Gemma 3 como fallback), extracción estructurada con gramática JSON (Gemma 3 4B), y validación por reglas (config-matrix.json).

Ambos modelos de IA operan en formato GGUF mediante un **runtime unificado** (Ollama/llama.cpp), simplificando drásticamente la operación y mantenimiento. La orquestación utiliza FastAPI + Celery + Redis, con interfaz web renderizada en servidor (HTMX) y notificaciones en tiempo real. El despliegue se realiza mediante Docker Compose — un único comando levanta el stack completo.

El principio fundamental es **human-in-the-loop por defecto**: el sistema genera un informe semáforo (verde/amarillo/rojo) por gate de validación, con audit trail completo de cada decisión. Presenta evidencia organizada al analista — nunca toma decisiones finales. Cada registro de auditoría es INSERT-only con retención mínima de 5 años, asegurando la reconstitución del razonamiento del sistema para cualquier decisión pasada.

3.3 Escenarios de Evolución

El **Escenario A (POC Standalone)** busca demostrar la viabilidad del flujo de procesamiento completo en la infraestructura actual, procesando candidatos reales en 5-15 minutos por candidato (CPU). El **Escenario B (Integración FreeBalance CSM)** proyecta la evolución hacia el ecosistema Hacienda Digital — posicionando el sistema de IA como preprocesador del módulo de Gestión del Talento Humano. Cada fase agrega valor de manera independiente: la DGSC no necesita esperar por el FreeBalance, la GPU o integraciones externas para comenzar a beneficiarse del sistema.

3.4 Tres Niveles de Optimización

El análisis reveló que la optimización opera en tres niveles independientes, cada uno con valor propio: **organizacional** (formulario de presentación preestructurado, estandarización de nomenclaturas — cero tecnología, ~40% de reducción en el esfuerzo de clasificación), **determinístico** (validación de firmas digitales, cálculo de puntuaciones, detección de duplicados — reglas, sin IA), y **asistido por IA** (OCR de documentos escaneados, clasificación y extracción estructurada — Vision-Language Models). El primer nivel puede implementarse de inmediato, con costo cero, utilizando Microsoft Forms ya licenciado por la DGSC.

3.5 Recomendaciones Prioritarias

Para el POC, se recomienda procesamiento en **lotes nocturnos** (compensando la limitación de CPU), inicio inmediato con el **Docker Compose standalone** (sin esperar dependencias), **reextracción de los 34 manuales** (corrigiendo la crisis de calidad), y planificación de **adquisición de GPU** (~USD 500-800, ganancia de 10-50× en rendimiento).

3.6 Replicabilidad

La arquitectura — motor de reglas genérico + configuración específica por país — es intrínsecamente replicable para otros países miembros del CLAD con procesos similares de validación de credenciales para el servicio público. El stack (modelos open-source, Docker, CPU-only) no asume hardware especializado, licencias propietarias ni conectividad cloud — restricciones comunes en las administraciones públicas iberoamericanas. La adaptación para un nuevo contexto nacional consiste, esencialmente, en la creación de un config-matrix.json con las reglas locales.

4 Descripción y Alcance del Proyecto

4.1 Objetivos

4.1.1 Objetivo General

La cooperación técnica tuvo como propósito central recibir asesoría especializada para la automatización del procesamiento de los atestados presentados por la ciudadanía en el Reclutamiento Abierto y Permanente, buscando una resolución más efectiva en dicho proceso concursal, mediante la implementación de una herramienta informática asistida por Inteligencia Artificial.

Este objetivo responde a una necesidad concreta y medible: el equipo de analistas de la AGE procesa anualmente más de 21 mil candidaturas, cada una de las cuales requiere la verificación de múltiples requisitos contra normas específicas de la clase de cargo concursada. El tiempo invertido en esta verificación — particularmente en la clasificación de documentos recibidos de forma desestructurada y en la consulta repetitiva a portales externos — representa el principal cuello de botella operativo del proceso de reclutamiento y selección.

4.1.2 Objetivos Específicos

El primer objetivo específico consistió en **identificar estrategias efectivas para la implementación de IA** en la optimización del tiempo de revisión y procesamiento de los atestados que las personas presentan en las ofertas de servicio del Reclutamiento Abierto y Permanente. Para ello, fue necesario comprender en profundidad no solo el flujo formal del proceso, sino los patrones informales de trabajo de los analistas — las consultas paralelas, los atajos desarrollados a lo largo de años de práctica, y los puntos donde el juicio humano es genuinamente insustituible.

El segundo objetivo específico fue **recibir asesoría sobre herramientas informáticas** que contribuyan a la mejora de la asignación de recursos humanos, direccionando la revisión manual únicamente a los documentos que presenten irregularidades identificadas por la herramienta. Este objetivo traduce un principio fundamental de la arquitectura propuesta: la IA no elimina al analista del proceso, sino que reorganiza su trabajo, concentrando la atención humana en los casos que efectivamente requieren juicio calificado — los "amarillos" y "rojos" del informe semáforo — en lugar de dispersarla en la verificación repetitiva de casos que la máquina puede resolver con alta confianza.

4.2 Resultados Esperados

La cooperación técnica debía producir los insumos y orientaciones para crear una herramienta informática capaz de procesar, clasificar y validar los diversos tipos de

documentos que componen una candidatura al servicio civil costarricense. Específicamente, la herramienta debía ser capaz de:

Analizar **títulos educativos** — tanto técnicos como universitarios — verificando su autenticidad y correspondencia con los requisitos de la clase concursada, incluyendo la detección de inconsistencias entre el título declarado y el documento presentado. Verificar **instituciones educativas** contra las listas oficiales mantenidas por CONESUP (universidades privadas autorizadas), INA (formación técnica) y MEP (educación media), proceso que actualmente exige la consulta manual a múltiples portales y bases de datos.

Procesar **certificaciones de experiencia laboral**, extrayendo fechas, funciones desempeñadas y tipo de supervisión ejercida — información que frecuentemente aparece en formatos narrativos heterogéneos y requiere interpretación contextual. Identificar y clasificar **declaraciones juradas** en sus tres formatos reconocidos: servicios profesionales, empresas cerradas y experiencia en el exterior. Contrastar los documentos presentados contra los **Manuales de Clases y Especialidades** de la DGSC, que codifican los requisitos específicos de cada una de las 117 clases de cargos distribuidas en 34 series.

Adicionalmente, la herramienta debía ser capaz de verificar la incorporación a **colegios profesionales** (requisito obligatorio para diversas clases), validar **firmas digitales** y, crucialmente, generar **informes estructurados** sobre hallazgos y resultados que permitan al analista tomar decisiones informadas con eficiencia.

4.2.1 Trazabilidad: Resultados Esperados → Arquitectura

La tabla siguiente demuestra cómo cada resultado esperado del Briefing Inicial (Sección 5) es abordado por la arquitectura propuesta. Esta trazabilidad asegura que ningún requisito original fue omitido, aun cuando el enfoque técnico haya evolucionado significativamente con respecto a la formulación inicial.

#	Resultado Esperado (Briefing Inicial §5)	Gate / Componente en la Arquitectura
1	Títulos educativos (autenticidad, correspondencia)	Gate E1 (Académico) — OCR + extracción + config-matrix.json
2	Verificación de instituciones educativas (CONESUP, INA, CSP, MEP)	Etapa 3 (prevalidación contra listas públicas)
3	Certificaciones de educación formal/no formal	Gate E1 — clasificación por tipo + extracción estructurada
4	Certificaciones sobre situaciones especiales (discapacidad, afrodescendiente)	Gate E8 (Legal/Especial) — clasificación + HITL
5	Certificación especial de formación (créditos universitarios)	Gate E1 — clases profesionales con requisito de créditos

#	Resultado Esperado (Briefing Inicial §5)	Gate / Componente en la Arquitectura
6	Certificaciones de experiencia laboral (períodos, supervisión)	Gates E3 (Experiencia) y E4 (Supervisión)
7	Declaración jurada (3 tipos)	Gate E3 — clasificación de los 3 subtipos + alertas HITL
8	Supervisión de personal y tipos profesionales	Gate E4 — config-matrix.json con combos de supervisión
9	Contraste con Manuales de Clases	Motor de reglas — config-matrix.json ES el manual digitalizado
10	Incorporación a colegios profesionales	Gate E5 — fechas de incorporación vs inicio de funciones
11	Antecedentes laborales	Gate E6 — resumen para revisión humana (HITL-critical)
12	Validación de firmas digitales	Gate E2 — validación determinística (Tier 2, sin IA)
13	Informes estructurados de hallazgos	Informe semáforo + audit trail por candidato

Es relevante señalar que la formulación original del Briefing Inicial proponía un "modelo supervisado de aprendizaje (*machine learning*), asistido por herramientas de Inteligencia Artificial (Ejemplo Copilot)", implementado en "R o Python". El diagnóstico profundizado reveló que el problema es de naturaleza fundamentalmente distinta: se trata de **comprensión documental** (extraer información estructurada de documentos heterogéneos) combinada con **validación por reglas** (confrontar datos extraídos contra requisitos codificados), no de clasificación por aprendizaje supervisado. Esta reorientación — de *machine learning* genérico a *document understanding* + *rule engine* — es el hallazgo intelectual central de la cooperación y fundamenta todas las decisiones técnicas subsiguientes.

4.3 Alcance

4.3.1 Dentro del Alcance

El alcance de la cooperación abarcó seis dimensiones complementarias. El **mapeo AS-IS** documentó el proceso completo de validación de atestados, desde la publicación de la vacante hasta la comunicación al candidato, documentando cada etapa operativa, sus reglas asociadas y las fuentes de verificación involucradas. La **evaluación tecnológica** probó modelos de OCR y extracción estructurada en la infraestructura efectivamente disponible — una máquina virtual con 8 VCPUs, 16GB de RAM y sin GPU — asegurando que las recomendaciones son operativamente viables y no meras proyecciones teóricas.

La **arquitectura de solución** proyectó un flujo de procesamiento de IA con observabilidad nativa y human-in-the-loop, diseñado tanto para el **Escenario A** (POC standalone, operable de inmediato) como para el **Escenario B** (evolución progresiva hacia la integración con FreeBalance CSM y el ecosistema Hacienda Digital). Los **portales de validación interactivos** — tres herramientas HTML trilingües — permitieron al equipo DGSC validar el mapeo de procesos y requisitos de forma visual y colaborativa. Finalmente, las **recomendaciones estratégicas** incluyeron propuestas que trascienden el ámbito tecnológico, como el formulario de presentación preestructurado y la hoja de ruta de evolución institucional.

4.3.2 Fuera del Alcance

Es importante delimitar con claridad lo que quedó fuera del alcance de esta cooperación. La **integración directa con APIs externas** (CCSS, Hacienda, MEP, CONESUP) requiere acuerdos de interoperabilidad que trascienden el ámbito técnico e involucran negociaciones institucionales en curso. El sistema fue diseñado para **analistas internos**, no para candidatos — el ciudadano no interactúa directamente con la herramienta. El alcance es de **POC y arquitectura**, no de implementación productiva completa, aunque los artefactos entregados (scripts, esquemas Pydantic, configuraciones) constituyen la base técnica para la implementación. Finalmente, el enfoque utiliza **modelos de IA preentrenados de código abierto**, sin entrenamiento de modelos personalizados — una decisión deliberada que reduce costos, elimina dependencia de datasets propietarios y facilita el mantenimiento por parte del equipo de TI de la DGSC.

4.4 Cronograma de Trabajo

El trabajo se desarrolló a lo largo de tres meses, en fases que se superpusieron parcialmente para maximizar el aprovechamiento del tiempo disponible.

La primera fase, de **diagnóstico y mapeo** (5 de enero al 10 de febrero de 2026), se concentró en el levantamiento del proceso AS-IS. Incluyó reuniones de estrategia inicial con la dirección de la DGSC, tres sesiones de demostración del proceso con casos reales conducidas por el equipo de analistas de la AGE, dos sesiones sobre infraestructura y Hacienda Digital con el equipo de TI, la extracción de los requisitos de los 34 manuales de clases y la construcción de los portales de validación. La fase culminó con la reunión de validación del 9 de febrero, en la cual el equipo completo de la DGSC revisó el mapeo e identificó 10 correcciones que fueron incorporadas a la versión 2.0 de los artefactos.

La segunda fase, de **evaluación tecnológica** (11 al 26 de febrero), involucró el benchmark de modelos de OCR en GPU Tesla T4 y la evaluación de frameworks de extracción estructurada, incluyendo el descubrimiento de la incompatibilidad del T4 con el formato bfloat16 y la evaluación del modelo Gemma 3. Esta fase también incluyó la auditoría de calidad de los datos extraídos y la identificación de la crisis de integridad del config-matrix.json.

La tercera fase, de **arquitectura e informe** (26 de febrero a marzo de 2026), consolidó los hallazgos en una arquitectura de solución formal, elaboró dos escenarios de evolución y redactó el presente informe.

4.5 Vinculación con Programas de Gobierno

El proyecto se encuentra directamente alineado con múltiples agendas de modernización del Estado costarricense y de la comunidad iberoamericana.

En el plano nacional, la **Ley Marco de Empleo Público (N°101596, 2023)** constituye el marco normativo que orienta la modernización de la gestión del talento humano público en Costa Rica. La ley establece principios de eficiencia, transparencia y meritocracia que la automatización de la validación de atestados contribuye directamente a concretar. El **Proyecto Hacienda Digital del Bicentenario** representa la plataforma de interoperabilidad institucional con la cual el sistema de IA propuesto está diseñado para integrarse progresivamente — el Escenario B de este informe detalla esa trayectoria.

En el plano internacional, el proyecto responde a los **compromisos de Costa Rica con la OCDE** en materia de profesionalización y meritocracia en la función pública, y contribuye a los **Objetivos de Desarrollo Sostenible**, particularmente el ODS 16 (Paz, Justicia e Instituciones Sólidas) y el ODS 9 (Industria, Innovación e Infraestructura). En el ámbito del CLAD, la experiencia se inscribe en la tradición de cooperación horizontal que ha producido referencias valiosas en dominios como gobierno electrónico, gestión por competencias y evaluación del desempeño — a los cuales la inteligencia artificial aplicada al servicio civil viene ahora a sumarse como tema emergente.

---

5 Aspectos Metodológicos

5.1 Enfoque General

La metodología adoptada en esta consultoría partió de un principio que, aunque evidente, merece explicitación: ninguna solución tecnológica puede ser adecuadamente diseñada sin una comprensión profunda del proceso que pretende apoyar. En ese sentido, se optó por un enfoque en tres fases progresivas — diagnóstico, evaluación y arquitectura — en las cuales cada fase informó y ajustó las decisiones de la siguiente.

La **primera fase, de diagnóstico y mapeo** (5 de enero al 10 de febrero de 2026), se concentró en el levantamiento completo del proceso AS-IS mediante sesiones de trabajo con el equipo DGSC y análisis documental de los 34 manuales de clases. El objetivo no era solamente documentar el flujo formal, sino capturar las prácticas

informales, los atajos, los puntos de decisión implícitos y las fuentes de frustración cotidiana de los analistas — información que rara vez aparece en manuales de procedimientos, pero que es fundamental para el diseño de una herramienta que efectivamente resuelva problemas reales.

La **segunda fase, de evaluación tecnológica** (11 al 26 de febrero), sometió las tecnologías candidatas a pruebas en la infraestructura efectivamente disponible en la DGSC. Esta decisión metodológica fue deliberada: en lugar de evaluar modelos en condiciones ideales y proyectar su viabilidad, se optó por probarlos en las condiciones reales de operación — una máquina virtual con 8 VCPUs, 16GB de RAM y sin GPU dedicada. Las recomendaciones resultantes, por lo tanto, no son teóricas: cada tecnología seleccionada fue demostrada como operativa en el hardware existente.

La **tercera fase, de arquitectura e informe** (26 de febrero a marzo), consolidó los hallazgos de las fases anteriores en una arquitectura de solución que responde simultáneamente a las necesidades operativas de la DGSC, a los requisitos de auditabilidad de la administración pública costarricense, y a la realidad de la infraestructura disponible.

5.2 Metodología de Levantamiento

5.2.1 Sesiones de Trabajo

El levantamiento del proceso AS-IS fue conducido mediante **6 sesiones de trabajo** con el equipo DGSC entre el 5 de enero y el 9 de febrero de 2026. La estrategia de levantamiento evolucionó a lo largo de las sesiones, reflejando la profundización progresiva de la comprensión del proceso.

Las dos primeras sesiones (5 y 7 de enero) tuvieron carácter estratégico, estableciendo el encuadramiento del proyecto con la dirección de la DGSC. La sesión del 12 de enero definió KPIs y métricas de éxito. Las tres sesiones siguientes (19, 20 y 23 de enero), conducidas con el equipo de analistas de la AGE, constituyeron el núcleo del levantamiento: demostraciones del proceso con casos reales, en las cuales el consultor pudo observar en tiempo real cómo los analistas navegan entre documentos, portales externos y reglas de validación. Fue durante estas sesiones que emergieron los hallazgos más significativos — particularmente, que la mayor fuente de ineficiencia no es la lectura de PDFs en sí, sino la verificación repetitiva en portales externos.

Las sesiones del 26 y 28 de enero se enfocaron en infraestructura y el Hacienda Digital, con el equipo de TI, proporcionando los parámetros técnicos que informaron la evaluación tecnológica. Finalmente, la sesión del 9 de febrero constituyó la reunión formal de validación, en la cual el equipo completo de la DGSC revisó el mapeo e identificó **10 correcciones** que fueron incorporadas a la versión 2.0 de los artefactos.

5.2.2 Análisis Documental

El trabajo de análisis documental involucró la lectura y sistematización de un corpus heterogéneo de fuentes. Los **34 Manuales de Clases y Especialidades** — documentos PDF que codifican los requisitos de cada clase de cargo — fueron procesados para la extracción de los requisitos y su codificación en el formato JSON estructurado (config-matrix.json). El **Briefing Inicial CLAD-DGSC** y la documentación del **Proyecto Hacienda Digital** proporcionaron el encuadramiento institucional y técnico. La documentación sobre **infraestructura** de la DGSC permitió dimensionar con precisión las restricciones de hardware que condicionan las decisiones tecnológicas.

Cabe señalar que la documentación del proceso de validación de atestados no existía de forma integrada antes de esta cooperación. El mapeo producido — que identifica 7 fases, 9 gates de validación, 48 micropasos y 11 fuentes externas — constituye, hasta donde se pudo verificar, la primera documentación integral de este proceso en la DGSC.

Un instrumento metodológico central fue el **Project Canvas**, elaborado por el consultor y evolucionado colaborativamente con el equipo DGSC a lo largo de las sesiones de trabajo. El Project Canvas sirvió para estructurar progresivamente la información del proyecto — objetivos, restricciones, stakeholders, riesgos y decisiones técnicas — y se benefició de un detallamiento significativo por parte de la DGSC, que contribuyó activamente con la documentación de sus procesos e infraestructura.

Previamente al análisis documental de los manuales de clases, se consultaron los **manuales y catálogos institucionales** disponibles públicamente en el portal de la DGSC: el Manual de Clases y el Manual de Especialidades (https://www.dgsc.go.cr/ts_clases/dgsc_servicios_clases.html), la Guía de Mantenimiento del Manual Descriptivo de Especialidades, y el Catálogo de Formaciones para clases específicas. Estos instrumentos normativos proporcionaron el encuadramiento para la interpretación de los requisitos codificados en los 34 manuales de series.

5.2.3 Barrera Lingüística y Apoyo Institucional

Un desafío operativo relevante fue la **barrera lingüística** entre el consultor (lusófono) y el equipo DGSC (hispanohablante). Esta barrera fue mitigada mediante tres mecanismos: el apoyo de la asesoría internacional del **Ministério da Gestão e da Inovação em Serviços Públicos (MGI)** de Brasil, que facilitó la articulación institucional; la utilización de **traducción y anotaciones automáticas de Microsoft Teams Premium**, costeado por el MGI, que permitió la conducción eficaz de las sesiones de trabajo remotas; y los **ambientes de computación en la nube Azure** utilizados para los benchmarks técnicos (GPU Tesla T4), igualmente costeados por el

MGI. Esta cooperación triangular CLAD-MGI-DGSC fue fundamental para viabilizar el trabajo técnico en las condiciones de plazo y distancia geográfica involucradas.

5.2.4 Metodología de Planificación: BMAD Method

La planificación y diseño de la solución técnica fueron acelerados por el **BMAD Method** (BMad Methodology for AI-Driven Development, disponible en: https://github.com/bmad-code-org/BMAD-METHOD) — una metodología que emplea agentes de IA especializados para conducir de forma estructurada las etapas de concepción de producto y arquitectura de software. El framework organiza el trabajo en fases progresivas, conducidas por agentes especializados que guían el proceso de decisión mediante cuestionamiento estructurado y referencia a buenas prácticas documentadas. Este enfoque AI-driven fue utilizado como herramienta metodológica central en la consultoría, acelerando la producción de los artefactos de planificación y asegurando consistencia entre las decisiones técnicas documentadas.

5.2.5 Herramientas de Validación

Una decisión metodológica que se reveló particularmente fructífera fue la construcción de **tres portales interactivos** como instrumento de validación — lo que, en la práctica, produjo un **gemelo digital del proceso de validación de atestados**. Hasta entonces, el conocimiento de este proceso existía exclusivamente como saber tácito, distribuido entre los analistas y transmitido por la práctica cotidiana. Los portales transformaron ese conocimiento en una representación interactiva, navegable y verificable — precondición para cualquier automatización basada en IA. En lugar de presentar el mapeo en formato de documento estático — enfoque que dificulta la identificación de errores y omisiones —, se optó por herramientas que permitieron al equipo "navegar" por el proceso y verificar los requisitos clase por clase.

El **Portal Macro** (versión 6, 348KB) presenta un árbol de decisión ramificado por clase, con pestaña de fuentes de validación. El **Portal Micro** (versión 3, 301KB) visualiza el pipeline de 9 gates con 48 micropasos en formato de flujo, con overlay por clase. El **Portal de Escenarios de Prueba** (79KB) ofrece 12 escenarios predefinidos con resultados gate a gate — permitiendo al equipo simular el procesamiento de candidatos hipotéticos y verificar si el sistema aplica las reglas correctamente.

Todos los portales son archivos HTML únicos (sin dependencias externas), trilingües (español, portugués e inglés), con modo oscuro/claro y funcionalidad de exportación CSV/JSON. Fueron entregados al equipo DGSC el 11 de febrero de 2026.

5.3 Metodología de Evaluación Tecnológica

5.3.1 Criterios de Selección

La evaluación tecnológica fue regida por seis criterios, de los cuales dos son eliminatorios. La **viabilidad en la infraestructura objetivo** constituye el primer

criterio eliminatorio: la DGSC dispone de una VM con 8 VCPUs, 16GB de RAM y Ollama instalado, sin GPU dedicada. Cualquier modelo que no opere en estas condiciones es automáticamente descartado, independientemente de su calidad. El benchmark también probó el desempeño en GPU Tesla T4 16GB (disponible en el ambiente Azure utilizado para pruebas) para evaluar el potencial futuro con aceleración por hardware. La **independencia de cloud y APIs externas** es el segundo criterio eliminatorio: el sistema procesa datos personales de candidatos al servicio público, lo que impone soberanía completa de los datos — procesamiento 100% on-premise, sin costos recurrentes de licenciamiento cloud.

Entre los criterios clasificatorios, la **estabilidad** (el sistema debe completar todos los documentos sin fallos), la **calidad de extracción** (especialmente para documentos en español costarricense), la **velocidad aceptable** (minutos por candidato, no horas) y la **mantenibilidad** (el equipo de TI de la DGSC debe poder operar el sistema sin el consultor) completaron el cuadro de evaluación.

5.3.2 Benchmark OCR

Se procedió a la evaluación comparativa de 4 modelos de visión-lenguaje para reconocimiento óptico de caracteres, ejecutados mediante vLLM en la GPU NVIDIA Tesla T4 16GB. La elección del T4 como ambiente de benchmark se debe a dos razones: es una GPU ampliamente disponible en ambientes cloud a bajo costo, y representa el tipo de hardware que la DGSC podría adquirir en el futuro (una unidad T4 cuesta aproximadamente USD 500-800).

Los modelos evaluados fueron PaddleOCR-VL-1.5 (0.9B parámetros), GLM-OCR (0.9B), DeepSeek-OCR-2 (3.4B MoE) y HunyuanOCR (1B, no probado por indisponibilidad). Los resultados completos se encuentran en el **Anexo B**; la síntesis de las decisiones, en la Sección 6.

5.3.3 Evaluación de Frameworks de Extracción

Tres enfoques fueron evaluados para la extracción estructurada con garantía de JSON válido — requisito fundamental para alimentar un motor de reglas determinístico aguas abajo. LangExtract (Google), vLLM con Outlines (guided decoding) y llama.cpp con gramática GBNF (GGML BNF — variante de la notación Backus-Naur Form utilizada por el ecosistema llama.cpp para restringir la salida del modelo a estructuras JSON válidas) fueron comparados en cuanto a la garantía de conformidad con esquemas Pydantic, compatibilidad con hardware local y calidad de extracción para documentos en español. Los resultados completos se encuentran en el **Anexo C**; la síntesis, en la Sección 6.

5.4 Marco Conceptual

El diseño de la solución se asienta sobre tres principios conceptuales que, aunque aplicables a cualquier sistema de IA en contexto gubernamental, asumen particular relevancia en el dominio de la gestión del talento humano público.

5.4.1 Human-in-the-Loop por Defecto (HITL-default)

El principio de revisión humana obligatoria por defecto (*human-in-the-loop* o HITL) establece que todo nodo del pipeline sin validación automatizada comprobada requiere revisión humana. El sistema presenta evidencia organizada — nunca toma decisiones finales. Esta no es una limitación provisional del sistema, sino una decisión de diseño permanente que refleja la naturaleza del dominio: la validación de atestados para el servicio público es un acto administrativo con consecuencias jurídicas, y la responsabilidad por ese acto debe permanecer con el servidor público investido para tal efecto.

La evolución hacia mayor autonomía — es decir, la transición de gates "amarillos" (requiere revisión) a "verdes" (validación automática) — depende de métricas de precisión comprobadas en producción, validadas por el equipo DGSC, y documentadas en el audit trail. El sistema está diseñado para que esta evolución sea gradual, gate a gate, y siempre reversible.

5.4.2 Observabilidad Nativa

Cada decisión del sistema — desde el modelo de OCR utilizado para procesar una página específica hasta la regla del config-matrix.json que determinó el resultado de un gate — genera un registro auditable con timestamp, fuente y justificación. Esta observabilidad no es un requisito "nice to have": es una respuesta directa a las obligaciones de transparencia y rendición de cuentas que rigen la administración pública costarricense, incluyendo los requisitos de la Contraloría General de la República y los compromisos del país con la OCDE en materia de integridad pública.

El audit trail se implementa como tabla INSERT-only — los registros nunca se modifican ni eliminan — con retención mínima de 5 años, asegurando la reconstitución completa del razonamiento del sistema para cualquier decisión pasada.

5.4.3 Motor de Reglas Configurable

La tercera decisión conceptual fundamental es la separación entre lógica de procesamiento y reglas de negocio. Los requisitos de cada clase de cargo se codifican en un archivo JSON de configuración (config-matrix.json), no en código de programación. Cuando la DGSC actualiza un manual de clases — lo que ocurre periódicamente —, la actualización del sistema consiste en editar un archivo JSON, operación que el equipo de TI puede realizar sin intervención de desarrolladores.

Este enfoque de "configuración sobre código" es particularmente adecuado al contexto institucional: la DGSC necesita autonomía para mantener el sistema actualizado, sin depender de consultores externos ni de ciclos de desarrollo de software. El mismo enfoque hace la arquitectura replicable para otros países — la adaptación para un contexto nacional diferente consiste, esencialmente, en la creación de un nuevo config-matrix.json con las reglas locales.

---

6 Diagnóstico de la Situación Actual (AS-IS)

El proceso de validación de atestados en la DGSC constituye el corazón operativo del reclutamiento para el servicio civil costarricense. Comprender este proceso en profundidad — no solo su flujo formal, sino sus tensiones internas, sus cuellos de botella cotidianos y sus soluciones informales — fue la condición necesaria para cualquier propuesta tecnológica fundamentada. La presente sección sintetiza los hallazgos del diagnóstico; el mapeo completo, incluyendo diagramas y ejemplos detallados, se encuentra en el **Anexo A**.

6.1 Visión General del Proceso

El proceso de validación de atestados se despliega en **7 fases secuenciales** que abarcan desde la publicación de la vacante hasta la comunicación formal al candidato:

A. Publicación de la Vacante → B. Registro del Candidato → C. Presentación de Documentos → D. Recepción y Asignación → E. Validación Documental → F. Decisión y Registro → G. Comunicación al Candidato

Aunque la secuencia sugiere un flujo lineal, la realidad operativa es considerablemente más compleja. La fase de validación documental (Fase E), que concentra la mayor parte del esfuerzo de los analistas, se despliega en **6 puertas de verificación**, cada uno de los cuales puede exigir la consulta a fuentes externas, la aplicación de reglas específicas de la clase concursada, y el ejercicio del juicio profesional en situaciones de ambigüedad. A esto se suman 48 micro-pasos de validación, 22 tipos de documentos a procesar y 11 fuentes externas de verificación — una complejidad que el sistema actual gestiona casi enteramente de forma manual.

6.2 Puntos de Dolor Prioritarios

6.2.1 El Dolor Principal: Verificación en Portales Externos

El hallazgo más significativo del diagnóstico — confirmando lo que el equipo de la ARSP ya identificaba como uno de los principales cuellos de botella operativos — es que **la mayor fuente de ineficiencia no es la lectura de PDFs**. Los analistas son

profesionales experimentados que, en la mayoría de los casos, logran evaluar rápidamente un documento académico o una certificación de experiencia. Lo que consume desproporcionadamente su tiempo es la **verificación de datos en 8 o más sistemas externos**, una actividad que exige alternancia constante entre ventanas (Alt-Tab), inserción manual de datos en portales con interfaces diferentes, y espera por respuestas de sistemas cuya disponibilidad no siempre es confiable.

Los portales consultados incluyen CONESUP (universidades privadas autorizadas — acceso vía PDF público), MEP/SICOBATEC (diplomas de educación media — login institucional), CCSS (contribuciones de seguridad social — acceso institucional), Ministerio de Hacienda (situación tributaria — portal público), CONARE (reconocimiento de títulos extranjeros), los diversos colegios profesionales (cada uno con su propia interfaz), el validador de firmas digitales Agente Gaudy, y el Registro Nacional para verificación de estado de empresas.

Este hallazgo tiene implicaciones directas para el diseño de la solución: la pre-validación contra listas públicas (Etapa 3 del pipeline, que verifica automáticamente datos contra CONESUP, INA y tablas de equivalencia) ofrece un quick-win inmediato, mientras que la integración futura con la interfaz de interoperabilidad del Hacienda Digital promete resolver el cuello de botella de forma definitiva.

6.2.2 Entrada Desestructurada y Calidad de los Documentos

El segundo punto de dolor, aunque menos dramático en términos de tiempo consumido, es particularmente relevante para la viabilidad de la IA: aproximadamente **40% de las entregas no cumplen las reglas de presentación** publicadas por la DGSC (estimación de los propios analistas durante las sesiones de trabajo). Los candidatos envían documentos en órdenes variados, mezclan tipos de comprobantes en un único PDF, incluyen páginas en blanco o documentos irrelevantes, y frecuentemente no siguen las nomenclaturas esperadas. El resultado es que el analista destina una porción significativa de su tiempo clasificando y ordenando documentos antes de poder iniciar la validación propiamente dicha.

El análisis de documentos reales durante la cooperación reveló la dimensión concreta de este desafío: la **mayoría de los documentos recibidos son imágenes digitalizadas** — fotografías de diplomas, certificaciones escaneadas con sellos institucionales superpuestos al texto, declaraciones en orientaciones variadas. Un expediente puede contener decenas de páginas combinando diferentes tipos documentales en un único archivo PDF. En contraste, los documentos generados digitalmente por sistemas institucionales presentan datos limpios y estructurados. Esta asimetría tiene implicaciones directas para la elección tecnológica: el sistema no puede limitarse al reconocimiento óptico de caracteres tradicional (OCR); necesita **modelos de comprensión documental** (*Vision-Language Models*) capaces de interpretar disposición, contexto y estructura — no solo reconocer caracteres individuales. La misma asimetría fundamenta la recomendación de un formulario de presentación pre-estructurado: si los candidatos enviaran documentos pre-

categorizados, con metadatos recolectados en el punto de presentación, el problema de clasificación se reduciría sustancialmente antes incluso de cualquier intervención tecnológica.

Este hallazgo fundamentó dos decisiones de diseño importantes: la inclusión de una etapa de clasificación automática de documentos en el flujo de procesamiento de IA (Etapa 2 de la arquitectura), y la recomendación estratégica del formulario pre-estructurado que, independientemente de la adopción de IA, reduciría drásticamente ese esfuerzo de clasificación.

6.2.3 Escala versus Capacidad

La aritmética es implacable: 21.447 candidatos por año divididos entre aproximadamente 10 analistas resultan en cerca de 2.145 candidatos por analista por año — más de 8 candidatos por día hábil, cada uno con múltiples documentos a verificar y múltiples portales a consultar. Las herramientas actuales de gestión de concursos presentan limitaciones de búsqueda y trazabilidad que llevan al equipo a complementarlas con controles auxiliares — escenario común en procesos de alto volumen que preceden la digitalización completa. Esta realidad refuerza la necesidad de instrumentos que consoliden la información y ofrezcan trazabilidad nativa, como el sistema propuesto en este informe.

6.3 Infraestructura Disponible

La infraestructura tecnológica de la DGSC fue mapeada con precisión, pues constituye el parámetro más restrictivo para las decisiones de arquitectura. La DGSC dispone de 4 servidores HPE DL360 Gen11, cada uno con 2 procesadores Intel Xeon-G 5416S y 512GB de RAM, virtualizados con Windows Server Datacenter y Hyper-V. Para el proyecto de IA, se asignó una máquina virtual con las siguientes especificaciones:

Componente	Especificación
CPU	8 VCPUs (Intel Xeon-G 5416S)
RAM	16 GB
SO	Ubuntu 22.04 LTS
Runtime	Ollama (en instalación)
GPU	Ninguna disponible — operación CPU-only

La ausencia de GPU dedicada es el factor más determinante para las elecciones tecnológicas. Aunque los servidores HPE Gen11 poseen capacidad de RAM abundante, la virtualización con Hyper-V no soporta passthrough de GPU, y no hay tarjetas gráficas instaladas en el clúster. Toda la evaluación tecnológica y las recomendaciones de este informe parten de esta realidad: el sistema debe funcionar en CPU-only con 16GB de RAM.

La DGSC también dispone de licencias Microsoft 365 Business Standard, Azure AD Premium y Power BI Pro — componentes que, aunque no directamente utilizados por la solución propuesta, indican un ecosistema de TI funcional y un equipo con capacidad de administrar infraestructura de mediano porte.

6.4 Datos Cuantitativos

El mapeo cuantificó la complejidad del dominio con precisión:

- **34 series** de puestos, cubriendo desde funciones técnicas hasta profesionales especializados
- **117 clases** de puestos, cada una con requisitos específicos codificados en manuales oficiales
- **6 puertas** de validación en el pipeline de verificación
- **48 micro-pasos** distribuidos en los 9 gates
- **11 fuentes externas** de verificación (portales, bases de datos, aplicaciones de escritorio)
- **22 tipos de documentos** que un candidato puede presentar
- **4 series transicionales** (Operador de Computador, Profesional Bachiller Jefe, Programador de Computador, Artes Gráficas) con reglas de transición específicas
- **34 clases** (31,5% del total) que poseen 2 o más combinaciones alternativas de requisitos — es decir, un candidato puede cumplir los requisitos por caminos diferentes, y el sistema debe verificar todos los caminos aplicables

Esta complejidad combinatoria — 117 clases × múltiples combinaciones de requisitos × 9 gates × 22 tipos de documentos — es lo que hace al proceso intratable para soluciones simplistas y, al mismo tiempo, particularmente adecuado para un enfoque basado en motor de reglas configurable. El mapeo completo, incluyendo diagrama Mermaid y ejemplos detallados de combinaciones, se encuentra en el **Anexo A**.

---

7 Evaluación Tecnológica

La evaluación tecnológica constituye el eje empírico de esta cooperación — el punto donde las necesidades operativas identificadas en el diagnóstico encuentran las posibilidades y limitaciones concretas del hardware disponible. A diferencia de evaluaciones teóricas o basadas en benchmarks de terceros, todas las tecnologías consideradas en este informe fueron probadas en la infraestructura objetivo o en condiciones equivalentes, y los resultados reflejan el desempeño real, no el proyectado.

La estrategia de evaluación priorizó la **viabilidad en la infraestructura existente de la DGSC** (CPU-only, 16GB RAM) como criterio eliminatorio. Esta es una decisión que merece justificación explícita: en el contexto de una administración pública, una tecnología que requiere hardware inexistente no es una recomendación — es una aspiración. Las recomendaciones de este informe son operativas: el sistema propuesto funciona en lo que la DGSC tiene hoy. El benchmark en GPU Tesla T4 16GB (Azure) evaluó el potencial futuro con aceleración por hardware, pero las decisiones de arquitectura no dependen de esa aceleración.

7.1 OCR — Reconocimiento Óptico de Caracteres

El reconocimiento óptico de caracteres es la primera etapa computacionalmente intensiva del pipeline: convertir páginas PDF escaneadas — imágenes, en la práctica — en texto legible por máquina. La calidad de esta etapa condiciona todas las etapas subsiguientes: un OCR que falla en reconocer una fecha, un código o un nombre de institución compromete irreversiblemente la extracción y la validación.

Conforme se documentó en la Sección 5.2, la realidad de los documentos recibidos por la DGSC hace que esta etapa sea particularmente exigente: la mayoría son imágenes digitalizadas con calidad variable, sellos superpuestos y orientaciones diversas. Las herramientas de OCR tradicionales — que operan por reconocimiento de caracteres aislados — son insuficientes para este contexto. Los modelos evaluados son **Vision-Language Models** (VLMs): modelos que comprenden la disposición, la estructura y el contexto semántico de una página, no solamente sus caracteres. Esta distinción — entre reconocimiento de caracteres y comprensión documental — es lo que hace que la etapa de OCR sea genuinamente desafiante, especialmente bajo la restricción de infraestructura CPU-only con 16GB de RAM.

Se evaluaron tres modelos de visión-lenguaje, seleccionados por ser open-source, de pequeño porte (<=3.4B parámetros) y compatibles con la arquitectura del pipeline:

PaddleOCR-VL-1.5 (0.9B parámetros, PaddlePaddle) demostró ser el único modelo viable en las condiciones probadas. Con ~4GB de VRAM, opera a ~3-5 segundos por página en GPU T4 y ~30-60 segundos por página en CPU vía Ollama — velocidad aceptable para procesamiento en lotes. Su calidad de extracción para documentos en español resultó adecuada en las pruebas con documentos reales de la DGSC.

GLM-OCR (0.9B parámetros, THUDM), a pesar de tener el mismo tamaño que PaddleOCR, demostró un desempeño dramáticamente inferior en el T4: ~376 segundos por página — un factor de **100x más lento**. La causa raíz fue identificada en el flag `--enforce-eager` del vLLM, necesario cuando el modelo excede ~80% de la VRAM disponible. Este flag desactiva el mecanismo de paginación del vLLM y fuerza la asignación eager de memoria, resultando en una degradación catastrófica del throughput.

DeepSeek-OCR-2 (3.4B parámetros, arquitectura Mixture-of-Experts) causó error de memoria (OOM) después de procesar aproximadamente 15 páginas, indicando una necesidad de VRAM superior a los 16GB disponibles en el T4. Este resultado era previsible dado el tamaño del modelo, pero la evaluación confirmó empíricamente la inviabilidad.

La decisión por PaddleOCR-VL-1.5 es, por lo tanto, no solo preferencial sino necesaria: es el único modelo que opera de forma estable en el hardware disponible, tanto en GPU como en CPU. Los detalles completos del benchmark — incluyendo metodología, métricas por modelo y ejemplos de salida — se encuentran en el **Anexo B**.

7.2 Extracción Estructurada

Si el OCR convierte imágenes en texto, la extracción estructurada convierte texto libre en datos computables — JSON con campos tipados que un motor de reglas puede procesar de forma determinística. Esta etapa presenta un desafío técnico específico: el modelo de lenguaje debe no solo extraer información relevante, sino hacerlo en un formato **rigurosamente válido**. Un JSON malformado, un campo ausente o un tipo de dato incorrecto interrumpen el pipeline.

Se evaluaron tres enfoques para garantizar la conformidad de la salida:

LangExtract (Google) ofrece extracción estructurada con el Gemini — sin embargo, requiere envío de datos a la nube de Google. Aunque soporta Ollama como backend local, en ese modo pierde la garantía de schema constraints, operando únicamente con prompt engineering. Para un sistema que procesa datos personales de candidatos al servicio público, la dependencia de la nube fue considerada eliminatoria.

vLLM con Outlines (guided decoding) fue inicialmente seleccionado por su elegancia técnica: el guided decoding garantiza estructuralmente que la salida del modelo conforme a un esquema JSON especificado, restringiendo el vocabulario token a token durante la generación. Sin embargo, durante las pruebas, se descubrió una incompatibilidad fundamental: el Tesla T4 (compute capability 7.5) **no soporta bfloat16**, el formato numérico nativo del Gemma 3. El vLLM produce salidas vacías cuando intenta convertir de bfloat16 a float16 — una falla silenciosa particularmente peligrosa en producción.

llama.cpp con gramática GBNF (la solución seleccionada) elude la incompatibilidad utilizando formatos GGUF cuantizados que operan en float32/int4, independientemente del formato nativo del modelo. La gramática GBNF (una extensión de BNF usada por llama.cpp) garantiza que la salida conforme rigurosamente al esquema JSON esperado — con la misma garantía estructural del guided decoding, pero sin la dependencia del vLLM y sin la incompatibilidad bfloat16.

El modelo seleccionado para extracción es el **Gemma 3 4B QAT GGUF** (Google), en la cuantización q4_0. Con ~2.6GB de VRAM en la cuantización Int4, opera tanto en CPU como en GPU. En relación con el Gemma 2 2B (considerado anteriormente), ofrece ventajas decisivas: soporte a 140+ idiomas (incluyendo español con alta calidad), contexto de 128K tokens (vs. 8K), y mejora significativa en extracción JSON estructurado conforme benchmarks independientes (LLMStructBench). Los detalles completos se encuentran en los **Anexos C y D**.

7.3 Decisión de Orquestación

La evaluación inicial consideró **n8n** — plataforma open-source de automatización de workflows — como orquestador del pipeline. n8n ofrece una interfaz visual intuitiva para composición de workflows, lo que simplificaría la operación por parte del equipo de TI de la DGSC. Sin embargo, el análisis de seguridad reveló **3 vulnerabilidades críticas** (CVE con CVSS >=9.9) publicadas entre 2024-2025, involucrando ejecución remota de código y escalación de privilegios.

Para un sistema que procesa datos personales de candidatos — incluyendo números de cédula, direcciones, registros de seguridad social y situación jurídica — la adopción de una plataforma con historial reciente de vulnerabilidades de severidad máxima representaría un riesgo inaceptable. La decisión fue sustituir n8n por un stack Python nativo con superficie de ataque significativamente menor:

FastAPI sirve como framework de API REST, ofreciendo soporte nativo a SSE (Server-Sent Events) para notificaciones en tiempo real, async I/O para operaciones concurrentes, y generación automática de documentación OpenAPI. **Celery con Redis** implementa la cola de tareas asíncronas, procesando candidatos en background sin bloquear la interfaz del analista. **HTMX con Jinja2** implementa el frontend como templates server-rendered — sin framework JavaScript, sin build step, sin dependencias del ecosistema npm —, manteniendo la simplicidad operativa que era una de las ventajas originales del n8n.

7.4 Runtime Unificado

Un descubrimiento tardío — pero de impacto significativo en la simplicidad de la arquitectura — fue la disponibilización de PaddleOCR-VL-1.5 en **formato GGUF** (marzo de 2026). Esta novedad permite servir tanto el modelo de OCR como el modelo de lenguaje por el **mismo runtime**: Ollama, basado en llama.cpp.

Antes de este descubrimiento, la arquitectura requería dos runtimes distintos — PaddlePaddle nativo para el OCR y Ollama/llama.cpp para el LLM —, con sus respectivas dependencias, configuraciones y modos de operación. La unificación en un único runtime elimina esa complejidad: el equipo de TI opera un único servicio (Ollama) con un único mecanismo de actualización de modelos. La carga es secuencial — los dos modelos no caben simultáneamente en 16GB de RAM —, pero esta es una limitación transparente gestionada por el orquestador Celery.

Para la Fase 2, con GPU disponible, el vLLM soporta ambos modelos para throughput elevado, manteniendo la misma interfaz de API.

7.5 Stack Tecnológico Final

El pipeline completo, consolidando todas las decisiones tecnológicas, opera de la siguiente forma:

```
Entrada: PDF del candidato (vía e-mail IMAP o upload HTTP)
    ↓
pikepdf (sanitización)          → <1s | Remueve JavaScript/macros, extrae t
exto digital
    ↓
PaddleOCR-VL-1.5 GGUF (OCR) → CPU: ~30-60s/pág | GPU: ~3-5s/pág
    ↓
Gemma 3 4B QAT GGUF           → CPU: ~10-30s | GPU: ~2-5s
  + gramática GBNF             → Clasificación + Extracción con JSON garan
tizado
    ↓
Motor de Reglas                → <1s (config-matrix.json)
    ↓
Reporte Semáforo               → Verde/Amarillo/Rojo por gate
  + Audit Trail                → Log INSERT-only de cada decisión
```

Orquestación: FastAPI + Celery + Redis (pipeline asíncrono) **Frontend:** HTMX + Jinja2 (server-rendered, SSE para notificaciones) **Runtime IA:** Ollama (sirve PaddleOCR y Gemma 3 vía llama.cpp) **Seguridad:** pikepdf (sanitización PDF) + nginx (TLS/reverse proxy) **Database:** PostgreSQL (audit trail INSERT-only, JSONB para metadata) **Deployment:** Docker Compose, on-premise, un único comando **Infraestructura:** VM existente (8 VCPUs, 16GB RAM, Ubuntu 22.04)

Cada componente de este stack fue seleccionado por ser open-source, operable en CPU-only, documentado y mantenible por un equipo de sysadmins sin expertise en machine learning.

---

8 Arquitectura de la Solución (TO-BE)

La arquitectura propuesta en este capítulo traduce los hallazgos del diagnóstico y las decisiones tecnológicas en un sistema concreto, diseñado para operar en las condiciones reales de la DGSC. No se trata de una arquitectura aspiracional — cada componente fue seleccionado porque funciona en el hardware existente, puede ser mantenido por el equipo de TI actual, y respeta los principios de auditabilidad y soberanía de datos que rigen la administración pública costarricense.

La arquitectura detallada — incluyendo diagramas de componentes, árbol de directorios completo y especificaciones de API — se encuentra en el **Architecture Decision Document** (artefacto BMAD, anexo). La presente sección ofrece una visión de alto nivel orientada a la comprensión del sistema por parte de gestores y tomadores de decisión.

8.1 Principios de Diseño

Cinco principios orientaron las decisiones arquitectónicas, cada uno de los cuales responde a una necesidad concreta identificada en el diagnóstico.

El primer principio — **human-in-the-loop por defecto** — establece que el sistema no toma decisiones finales. Presenta evidencia organizada al analista, quien mantiene la responsabilidad por la decisión administrativa. Este no es un recurso provisional: es una decisión de diseño que refleja la naturaleza jurídica del acto de validación en el servicio civil.

El segundo principio — **observabilidad nativa** — exige que cada decisión sea rastreable: cuál modelo procesó cuál documento, qué regla fue aplicada, cuál fue el resultado y con qué nivel de confianza. Esta trazabilidad no es un "nice to have" técnico: es la respuesta a los requisitos de transparencia de la Contraloría General de la República y a los compromisos del país con la OCDE.

El tercer principio — **configuración sobre código** — codifica los requisitos de cada clase de puesto en JSON (config-matrix.json), no en lógica de programación. Cuando la DGSC actualiza un manual de clases, la actualización del sistema consiste en editar un archivo de configuración — operación que el equipo de TI realiza sin desarrolladores.

El cuarto principio — **evolución progresiva** — asegura que el sistema agrega valor en cada fase: CPU-only → GPU → integración FreeBalance CSM. Ninguna fase bloquea otra; cada una funciona autónomamente.

El quinto principio — **soberanía de datos** — garantiza procesamiento 100% on-premise, sin envío de datos personales a la nube. Esta es simultáneamente una decisión técnica y una decisión de conformidad regulatoria.

8.2 Pipeline de Procesamiento

El sistema procesa cada candidato en **5 etapas secuenciales**, orquestadas por FastAPI + Celery + Redis. La secuencialidad no es una limitación: es una consecuencia directa de la restricción de memoria (16GB RAM), que impone la carga de un modelo a la vez. Cada etapa genera registros de audit trail que alimentan el reporte final.

8.2.1 Etapa 0: Ingesta y Sanitización

El procesamiento inicia con la recepción del PDF del candidato, sea vía monitoreo de buzón de correo electrónico IMAP (flujo actual) o vía upload HTTP por la interfaz web. La herramienta **pikepdf** sanitiza el documento — removiendo JavaScript embebido, macros y otros vectores de ataque —, calcula un hash SHA-256 para detección de duplicados, e intenta extraer texto digital cuando el PDF no es escaneado. Esta última funcionalidad es particularmente relevante: documentos generados digitalmente (un diploma universitario moderno, por ejemplo) contienen texto nativo que puede ser extraído sin OCR, ahorrando tiempo de procesamiento.

8.2.2 Etapa 1: Reconocimiento Óptico de Caracteres

Para páginas identificadas como escaneadas en la Etapa 0, el **PaddleOCR-VL-1.5** (vía Ollama) convierte las imágenes en texto. Páginas con texto digital extraído en la etapa anterior no pasan por el OCR — un bypass que, para documentos parcialmente digitales, puede reducir significativamente el tiempo total de procesamiento. El sistema registra, para cada página, si fue procesada por OCR o bypass, el tiempo consumido y la confianza estimada.

8.2.3 Etapa 2: Clasificación de Documentos

Antes de extraer datos estructurados, el sistema necesita identificar **qué es** cada documento — un título académico, una certificación de experiencia, una declaración jurada, etc. Esta etapa utiliza un **enfoque híbrido** diseñado para minimizar el costo computacional sin sacrificar la precisión.

Una heurística basada en regex y palabras clave clasifica correctamente cerca del 70% de los documentos sin costo de inferencia — documentos que contienen patrones textuales inequívocos ("TÍTULO DE LICENCIATURA", "CONSTANCIA DE TRABAJO", etc.). Documentos con clasificación de baja confianza son encaminados al **Gemma 3 4B** para clasificación por LLM. Documentos que ningún método logra clasificar son señalizados para revisión humana (HITL). Este último caso — el "zero match" — es raro, pero crítico: garantiza que ningún documento es silenciosamente descartado o mal clasificado.

El Gemma 3 utilizado en esta etapa es el **mismo modelo** cargado para la Etapa 3 — no hay reload entre clasificación y extracción, lo que optimiza el uso de memoria y tiempo.

8.2.4 Etapa 3: Extracción Estructurada

Con el tipo de documento identificado, el **Gemma 3 4B QAT GGUF** extrae campos estructurados utilizando esquemas Pydantic específicos por tipo de documento. La gramática GBNF del llama.cpp garantiza que la salida es JSON válido conforme al esquema — eliminando la necesidad de parsing tolerante a errores o de reintentos por malformación.

En esta misma etapa, el sistema realiza una **pre-validación automática** contra listas públicas disponibles: universidades autorizadas por CONESUP, instituciones reconocidas por el INA, y tablas de equivalencia de títulos. Documentos cuyas instituciones emisoras son encontradas en estas listas reciben un flag "pre-validado" que reduce el volumen de verificaciones manuales en la etapa siguiente.

8.2.5 Etapa 4: Validación por Reglas

El motor de reglas determinístico aplica el **config-matrix.json** (117 clases, 34 series, 9 gates) al JSON estructurado de la Etapa 3, produciendo el **informe semáforo**:

- **Verde:** Requisito cumplido con evidencia documental — el analista puede avanzar.
- **Amarillo:** Evidencia insuficiente o ambigua — requiere revisión humana (HITL).
- **Rojo:** Requisito claramente no cumplido — el sistema presenta la justificación.

Cada decisión del motor registra la regla aplicada, su fuente en el manual de clases, los datos de entrada y el resultado — permitiendo la reconstrucción completa del razonamiento para cualquier decisión, en cualquier momento futuro.

8.3 Dos Capas de Lógica de Negocio

La arquitectura separa la lógica de negocio en dos capas complementarias que evolucionan independientemente:

La **capa Macro** (config-matrix.json) codifica **lo que** cada clase exige — requisitos académicos, años de experiencia, tipo de supervisión, registro profesional, etc. Cada clase es una "línea de configuración" que alimenta el motor de reglas. Esta capa es parametrizada: agregar una nueva clase o actualizar requisitos existentes consiste en editar el JSON.

La **capa Micro** (validation-sources.json) codifica **cómo** cada requisito es validado — el pipeline genérico de 9 gates, 48 micro-pasos, las fuentes de verificación y los tipos de documentos aceptados para cada gate. Esta capa es el "manual de procedimientos" del sistema: describe el proceso de validación independientemente de la clase específica.

La separación es fundamental para la mantenibilidad: cuando la DGSC crea una nueva clase de puesto, edita únicamente el config-matrix.json. Cuando cambia el procedimiento de validación (por ejemplo, una nueva fuente de verificación), edita únicamente el validation-sources.json. El código del sistema no necesita ser alterado en ninguno de los casos.

8.4 Interfaz del Analista

El analista accede al sistema vía **interfaz web server-rendered** (HTMX + Jinja2), servida por FastAPI detrás de nginx con TLS. La opción por HTMX — en lugar de un

framework SPA como React o Vue — es deliberada: elimina la necesidad de build tools, transpilación y gestión de dependencias JavaScript, manteniendo la simplicidad operativa que el equipo de TI de la DGSC puede mantener autónomamente.

La interfaz utiliza **SSE (Server-Sent Events)** para notificaciones en tiempo real. Cuando un candidato termina el procesamiento en background (Celery), el informe semáforo aparece automáticamente en la pantalla del analista, sin necesidad de refresh manual. El analista puede revisar candidatos que ya completaron el procesamiento mientras otros aún están en cola.

El formato central de presentación es el **informe semáforo por candidato**, organizado por gate:

```
Candidato: María Fernández (cédula: 1-1234-5678)
Clase: Profesional Jefe de Servicio Civil 3 (0405047)

Gate E1 — Académico:          🟢 Licenciatura en Administración (UCR) — CONESUP: verificado
Gate E2 — Firma Digital:      🟡 Documento con firma digital — requiere validación manual
Gate E3 — Experiencia:        🟢 8 años (requisito: 7) — fechas verificadas
Gate E4 — Supervisión:        🟡 2.5 años declarados — 18 meses sector público: requiere verificación
Gate E5 — Colegio Profesional: 🟢 CPCECR activo desde 2015 — fecha anterior al inicio de funciones
Gate E6 — Legal:              🟢 Declaración de bienes presentada

RESULTADO GENERAL: 🟡 REVISIÓN PARCIAL — 2 gates requieren verificación manual
```

Este formato concentra la atención del analista en los puntos que efectivamente requieren su juicio (los amarillos y rojos), en lugar de dispersarla en la verificación de todos los requisitos indiscriminadamente.

8.5 Mantenibilidad

La mantenibilidad del sistema fue diseñada como requisito de primera clase, no como consideración posterior. El equipo de TI de la DGSC debe ser capaz de operar y evolucionar el sistema sin el consultor — de lo contrario, la cooperación técnica habría creado una dependencia, no una capacidad.

Operación	Quién	Cómo
Agregar nueva clase de puesto	Equipo TI DGSC	Editar config-matrix.json

Operación	Quién	Cómo
Actualizar requisitos existentes	Equipo TI DGSC	Editar config-matrix.json
Agregar nueva fuente de validación	Equipo TI DGSC	Editar validation-sources.json
Actualizar modelos de IA	Equipo TI DGSC	`ollama pull <nuevo-modelo>`
Actualizar el sistema	Equipo TI DGSC	`docker compose pull && docker compose up -d`

8.6 Infraestructura y Seguridad

Componente	Tecnología	Rol
API + Frontend	FastAPI + HTMX/Jinja2	REST API, SSE, templates server-rendered
Task Queue	Celery + Redis	Procesamiento asíncrono en background
Runtime IA	Ollama (llama.cpp)	Sirve PaddleOCR y Gemma 3 (GGUF unificado)
Database	PostgreSQL	Audit trail INSERT-only, metadata JSONB
Reverse Proxy	nginx	TLS (self-signed para intranet), rate limiting
Sanitización	pikepdf	Remueve código malicioso de PDFs
Deployment	Docker Compose	Un comando levanta el stack completo

9 Escenario A: POC Standalone

El Escenario A materializa la arquitectura propuesta en el capítulo anterior en un sistema autónomo y funcional, proyectado para demostrar la viabilidad de la validación automatizada de atestados en la infraestructura existente de la DGSC, sin dependencias externas de ninguna naturaleza — sin GPU, sin nube, sin APIs de

terceros, sin licencias adicionales. El modelo de OCR seleccionado (PaddleOCR-VL-1.5) fue validado en ambiente CPU-only, confirmando que la arquitectura opera en el hardware real de la DGSC — no solo en ambiente de benchmark con GPU.

La importancia estratégica de este escenario trasciende la demostración técnica. En muchos contextos de modernización gubernamental, la espera por condiciones ideales — el hardware perfecto, la integración con todos los sistemas, el presupuesto completo — paraliza la acción. El POC está diseñado para romper esa inercia: demostrar valor inmediato con lo que existe hoy, creando evidencia empírica que fundamenta decisiones futuras de inversión.

9.1 Objetivo

El objetivo del POC es procesar un lote de candidatos reales y generar reportes semáforo gate a gate, validando la precisión del pipeline OCR → Clasificación → Extracción → Reglas contra el juicio de los analistas de la DGSC. No se trata de una simulación: el sistema procesa documentos reales, aplica reglas reales y produce resultados que pueden ser comparados directamente con las decisiones que los analistas tomarían — o ya tomaron — para los mismos candidatos.

9.2 Infraestructura

El POC opera en la máquina virtual ya asignada por el equipo de TI de la DGSC, dentro del clúster de servidores HPE Gen11 existente:

Componente	Especificación
Servidor	VM existente en el clúster HPE Gen11
CPU	8 VCPUs (Intel Xeon-G 5416S)
RAM	16 GB
SO	Ubuntu 22.04 LTS
Runtime	Ollama (ya en instalación) + Docker
GPU	Ninguna (CPU-only)

Es importante notar que no hay adquisición de hardware involucrada. La VM ya existe, el Ubuntu ya está instalado, y Ollama está en proceso de instalación por parte del equipo de TI. El POC será entregado como Docker Compose — literalmente, un único comando (`docker compose up`) levantará el stack completo con todos los componentes configurados.

9.3 Stack del POC

Componente	Tecnología	Rol
API + Frontend	FastAPI + HTMX/Jinja2	Interfaz web, SSE para

Componente	Tecnología	Rol
		notificaciones
Task Queue	Celery + Redis	Procesamiento asíncrono en background
Runtime IA	Ollama (llama.cpp)	Sirve ambos modelos (GGUF unificado)
OCR	PaddleOCR-VL-1.5 GGUF (Ollama)	PDF → texto
Extracción	Gemma 3 4B QAT GGUF (Ollama)	Clasificación + Texto → JSON estructurado
Sanitización	pikepdf	Remueve código malicioso, extrae texto digital
Reglas	Motor Python + config-matrix.json	JSON → semáforo
Reverse Proxy	nginx	TLS (self-signed), rate limiting
Database	PostgreSQL	Audit trail INSERT-only

9.4 Flujo Operativo

El día a día de operación del POC está proyectado para integrarse naturalmente en la rutina de los analistas, minimizando el cambio de hábitos.

El proceso inicia cuando el analista **reenvía el correo electrónico de un candidato** al endpoint IMAP monitoreado por el sistema, o cuando realiza upload directo de los documentos por la interfaz web. Ambos canales convergen al mismo pipeline. El sistema **sanitiza los PDFs** recibidos — removiendo código malicioso, detectando duplicados por hash, y extrayendo texto digital cuando está disponible —, y a continuación encola al candidato para procesamiento.

Celery procesa los candidatos en la cola secuencialmente: para cada uno, carga el PaddleOCR para OCR, lo descarga, carga el Gemma 3 para clasificación y extracción, lo descarga, y finalmente aplica el motor de reglas. Esta carga secuencial es transparente para el analista, que ve únicamente el resultado final.

Cuando el procesamiento de un candidato termina, el sistema **notifica al analista vía SSE** — la interfaz web se actualiza automáticamente, presentando el informe semáforo. El analista puede entonces **revisar los gates amarillos y rojos**,

registrando sus decisiones HITL directamente en la interfaz, con justificación incorporada al audit trail.

Tiempo estimado por candidato (CPU): 5-15 minutos, dependiendo del número de páginas. Tiempo estimado por candidato (GPU T4, futuro): 20-50 segundos.

La diferencia de rendimiento justifica la recomendación de procesamiento en lotes nocturnos en la Fase 1 (CPU-only) y la adquisición futura de GPU para procesamiento en tiempo real.

9.5 Limitaciones y Mitigaciones

Toda prueba de concepto opera bajo limitaciones que importa explicitar con transparencia.

La operación **CPU-only** resulta en procesamiento más lento — minutos por candidato en vez de segundos —, aunque la validación del modelo ganador en ambiente CPU-only confirmó la viabilidad operativa de esa configuración. Esta limitación de velocidad se mitiga mediante la estrategia de procesamiento en lotes nocturnos: el analista selecciona los candidatos prioritarios al final del día, el sistema procesa durante la noche, y por la mañana los reportes semáforo están listos para revisión.

La **ausencia de acceso a portales externos** significa que los gates de verificación que dependen de consultas a sistemas como CCSS, MEP o colegios profesionales permanecen amarillos (HITL). La pre-validación contra listas públicas disponibles offline (CONESUP, INA) reduce parcialmente este volumen. La resolución definitiva depende de la integración con la interfaz de interoperabilidad del Hacienda Digital (Escenario B, Fase 4).

La **calidad de los datos del config-matrix.json** requiere atención. La crisis de integridad identificada el 16 de febrero (detallada en la Sección 10.2.1) demostró que la extracción inicial por LLM produjo datos plausibles pero incorrectos. La re-extracción con metodología mejorada es prerrequisito para validación confiable.

Finalmente, la **ausencia de integración con el sistema oficial de vacantes** significa que los resultados no se registran automáticamente. La exportación CSV/JSON permite importación manual, y la integración vía API está prevista para fases futuras.

9.6 Criterios de Éxito

La evaluación del POC debe basarse en métricas objetivas, a ser acordadas con el equipo de la DGSC antes de la ejecución del piloto:

Métrica	Meta
Precisión de extracción (campos correctos)	>85%
Tasa de gates verdes correctos (true positives)	>90%

Métrica	Meta
Tasa de gates rojos correctos (true negatives)	>85%
Tiempo promedio por candidato (CPU)	<15 min
Estabilidad (candidatos procesados sin crash)	100%

Estas métricas serán calculadas comparando la salida del sistema con el juicio de los analistas en una muestra representativa de candidatos. El audit trail permite la reconstrucción completa del razonamiento para cada decisión, facilitando el análisis de discrepancias y el refinamiento progresivo del pipeline.

---

10 Escenario B: Evolución FreeBalance CSM

Si el Escenario A demuestra lo que es posible hoy, el Escenario B proyecta la trayectoria de evolución del sistema en el contexto más amplio de la modernización digital del Estado costarricense. Esta trayectoria no es hipotética: se asienta sobre decisiones de inversión ya tomadas (el Proyecto Hacienda Digital del Bicentenario) y especificaciones técnicas ya documentadas en el documento de especificaciones técnicas del proyecto, que detalla los requisitos del módulo FreeBalance CSM para Gestión del Talento Humano.

El principio orientador es claro: **cada fase agrega valor independientemente**. La DGSC no necesita esperar por el FreeBalance para comenzar a usar el sistema de IA, y el FreeBalance no necesita del sistema de IA para avanzar en sus otras funcionalidades. Cuando ambos estén operativos, la integración entre ellos potencia el valor de cada uno.

10.1 Contexto: Hacienda Digital del Bicentenario

El Proyecto Hacienda Digital del Bicentenario (Préstamo 9075-CR, Banco Mundial) representa el esfuerzo más ambicioso de modernización de la gestión financiera y de recursos humanos del Estado costarricense en las últimas décadas. El proyecto prevé la implementación de un **Sistema Integrado de Administración Financiera y Talento Humano** basado en software COTS (Commercial Off-The-Shelf), operando a través de una interfaz de interoperabilidad (bus) que conectará múltiples instituciones públicas.

El módulo de **Gestión del Talento Humano (FIN-GTH)** del documento de especificaciones técnicas especifica requisitos que se superponen directamente al alcance del sistema de IA propuesto en este informe. El requisito FIN-GTH-2 (Reclutamiento) prevé el proceso completo de reclutamiento y selección; las especificaciones incluyen validación de títulos contra CONARE, CONESUP y SINAES; el expediente digital prevé almacenamiento de "imágenes, fotos, archivos PDF"; y la

trazabilidad exige "llevar la trazabilidad entre el oferente y los procesos" — funcionalidad que el audit trail nativo del sistema de IA implementa desde la Fase 1.

Esta superposición no es accidental: ambos sistemas responden a la misma necesidad institucional, por caminos complementarios. El sistema de IA resuelve el procesamiento cognitivo (comprender documentos, extraer datos, aplicar reglas); el FreeBalance CSM resuelve la gestión transaccional (registrar resultados, mantener expedientes, generar reportes institucionales). La integración entre ambos es no solo posible, sino natural.

10.2 Estrategia de Integración

El sistema de IA está diseñado como **pre-procesador** que alimenta al FreeBalance CSM. El candidato presenta documentos; el sistema de IA los procesa, clasifica, extrae datos y aplica reglas; el resultado — JSON estructurado con informe semáforo y audit trail — se pone a disposición vía API REST para consumo por parte del CSM.

```
[Correo electrónico/Portal de presentación]
      ↓
[Sistema de IA DGSC]  ← Escenario A (standalone)
  Sanitización → OCR → Clasificación → Extracción → Reglas
      ↓
  JSON estructurado + Reporte semáforo + Audit trail
      ↓
[API REST / Bus de Interoperabilidad]  ← Escenario B (integración)
      ↓
[FreeBalance CSM - FIN-GTH]
  Expediente digital | Registro de elegibles | Trazabilidad
```

La interfaz de integración sigue el template MH-UPHD-PRO01-INS-002 (especificación de Web Services del Hacienda Digital): REST API con JSON, autenticación conforme a las políticas del bus de interoperabilidad, y endpoints de consulta de resultados por candidato. El sistema de IA no escribe directamente en el FreeBalance — expone datos validados para consumo, preservando la separación de responsabilidades.

10.3 Fases de Evolución

La evolución del sistema standalone hacia el ecosistema integrado se despliega en cuatro fases, cada una con prerrequisitos y entregas independientes:

Fase 1 (POC) corresponde al POC standalone descrito en el Capítulo 8: pipeline completo operando en CPU-only, sin integraciones externas. El valor inmediato es la organización y pre-procesamiento de los documentos para el analista, con informe semáforo y audit trail. No requiere ningún prerrequisito más allá de la infraestructura ya existente.

Fase 2 introduce GPU dedicada y validación en portales externos. La adquisición de una GPU (T4 o superior, ~USD 500-800) transforma el tiempo de procesamiento de minutos a segundos, viabilizando el procesamiento en tiempo real. La integración con portales de verificación pública (CONESUP, INA) convierte gates actualmente amarillos en verdes, reduciendo el volumen de revisión manual. Prerrequisito: adquisición de hardware y configuración de acceso a los portales.

Fase 3 implementa la integración con FreeBalance CSM cuando el módulo FIN-GTH esté disponible. El sistema de IA pasa a alimentar automáticamente el expediente digital del candidato en el CSM, eliminando la transferencia manual de datos. Prerrequisito: API del módulo FIN-GTH documentada y accesible.

Fase 4 extiende la interoperabilidad a sistemas externos — CCSS (contribuciones de seguridad social), Ministerio de Hacienda (situación tributaria), MEP (diplomas), CONESUP (autorización de universidades). En esta fase, la gran mayoría de los gates que hoy son amarillos (HITL) se convierten en verdes (automáticos), y el rol del analista evoluciona de verificador a supervisor. Prerrequisito: acuerdos de interoperabilidad y acceso a APIs de los respectivos sistemas.

10.4 Riesgos de Integración

La trayectoria de integración presenta riesgos que deben ser gestionados proactivamente, pero que no deben paralizar el inicio de la implementación.

El riesgo más significativo es el **retraso en la implementación del FreeBalance CSM**, escenario plausible dado el historial de proyectos de esta envergadura. La mitigación es intrínseca al diseño: el sistema standalone continúa operativo independientemente del CSM, generando valor desde la Fase 1. La integración, cuando ocurra, será incremental — no es un evento de "todo o nada".

La **incompatibilidad de formatos de datos** es un riesgo de severidad media, mitigado por la adopción de JSON como formato nativo en todo el pipeline — el mismo formato especificado por el ecosistema Hacienda Digital. Eventuales adaptaciones de esquema son implementables como adaptadores simples, sin alteración del core del sistema.

La **falta de documentación de API del CSM** en el momento presente es un riesgo real que justifica la decisión de diseñar interfaces genéricas (REST/JSON) fácilmente adaptables cuando la documentación esté disponible. El sistema de IA no asume nada sobre la API del CSM más allá de lo que ya está especificado en el documento de especificaciones técnicas.

Finalmente, un eventual **cambio de alcance del módulo FIN-GTH** se mitiga mediante la arquitectura modular: el sistema de IA es un pre-procesador desacoplado, no un módulo integrado al CSM. Cambios en el CSM afectan únicamente la capa de integración, no el pipeline de procesamiento.

10.5 La Temporalidad como Principio de Diseño

El Escenario B no es una promesa de integración futura — es la demostración de que la arquitectura fue concebida para operar en **dos temporalidades simultáneas**. La temporalidad inmediata (Escenario A) resuelve el problema operativo hoy, con la infraestructura disponible. La temporalidad estratégica (Escenario B) posiciona al sistema para integrarse al ecosistema Hacienda Digital cuando este esté disponible. La DGSC no necesita elegir entre actuar ahora y planificar para el futuro — la arquitectura permite ambas cosas. Esta es una decisión de diseño, no de compromiso: el sistema es modular por concepción, no por limitación.

---

11 Resultados y Evidencias

La cooperación técnica produjo un conjunto de artefactos y descubrimientos que, tomados en conjunto, constituyen la base técnica e institucional para la implementación del sistema de validación inteligente. Esta sección documenta no solo lo que fue producido, sino lo que fue descubierto — incluyendo hallazgos inesperados que alteraron la dirección del proyecto y que contienen lecciones valiosas para iniciativas similares.

11.1 Artefactos Entregados

11.1.1 Mapeo de Procesos

El artefacto más fundamental producido por la cooperación es el **mapeo completo del proceso AS-IS de validación de atestados** — un gemelo digital de un proceso que, hasta entonces, existía exclusivamente como conocimiento tácito distribuido entre los analistas y transmitido por la práctica cotidiana. Este mapeo identifica 7 fases secuenciales, 9 gates de validación, 48 micro-pasos, 11 fuentes externas de verificación y 22 tipos de documentos, y constituye, hasta donde se pudo determinar, la primera documentación integral de este proceso en la historia de la DGSC.

El mapeo fue codificado en dos artefactos estructurados complementarios: el **config-matrix.json** (versión 2.0, 172KB), que codifica los requisitos de 117 clases en 34 series en formato JSON — la capa que define **lo que** cada clase exige —, y el **validation-sources.json** (versión 2.0, 67KB), que codifica el pipeline genérico de validación con sus 9 gates, 48 pasos y 11 fuentes — la capa que define **cómo** cada requisito es verificado. Ambos fueron validados por el equipo DGSC en la reunión del 9 de febrero. Juntos, abordan cada uno de los 13 resultados esperados definidos en el Briefing Inicial (ver tabla de trazabilidad en la Sección 3.2).

11.1.2 Portales de Validación Interactivos

La decisión de construir herramientas visuales interactivas para la validación del mapeo — en lugar de documentos estáticos — resultó particularmente eficaz y produjo un resultado que trasciende la cooperación específica: los portales constituyen una **representación interactiva y navegable** de todo el proceso de validación, accesible a cualquier persona con un navegador web, sin instalación de software. Se entregaron tres portales, todos como archivos HTML únicos sin dependencias, trilingües (español, portugués e inglés), con modo oscuro/claro y exportación CSV/JSON:

El **Portal Macro** (versión 6, 348KB) presenta un árbol de decisión ramificado por clase, con pestaña integrada de fuentes de validación. El **Portal Micro** (versión 3, 301KB) visualiza el pipeline de 9 gates con 48 micro-pasos en formato de flujo, con overlay por clase. El **Portal de Escenarios de Prueba** (79KB) ofrece 12 escenarios predefinidos con resultados gate a gate.

Estos portales fueron entregados al equipo DGSC el 11 de febrero de 2026 y sirvieron como base para la reunión de validación y para las correcciones subsiguientes.

11.1.3 Evaluación Tecnológica

La evaluación tecnológica produjo tres informes técnicos detallados: el **benchmark de OCR** (Anexo B), que evaluó 4 modelos concluyendo que PaddleOCR-VL-1.5 es el único viable; el **informe de frameworks de extracción** (Anexo C), que seleccionó llama.cpp+GBNF por compatibilidad con CPU y GPU; y la **evaluación de Gemma 3** (Anexo D), que fundamentó la elección del modelo 4B QAT GGUF documentando la incompatibilidad bfloat16 del T4. La experimentación del modelo seleccionado en ambiente CPU-only — la infraestructura real de la DGSC — validó que la decisión de arquitectura no depende de aceleración por GPU, aunque el benchmark en GPU T4 evaluó el potencial futuro con hardware acelerado.

11.1.4 Scripts y POC

Se desarrollaron tres artefactos de código reutilizables: el **benchmark-ocr.py** (49KB), herramienta de benchmark con 3 backends; el **outlines_extraction_poc.py** (38KB), prueba de concepto de extracción estructurada; y el **extraction_schemas.py** (21KB), esquemas Pydantic para 11 tipos de documentos que serán reutilizados en el pipeline de producción.

11.2 Descubrimientos Críticos

11.2.1 Crisis de Calidad de los Datos (16 de febrero de 2026)

El descubrimiento más impactante de la cooperación ocurrió durante la auditoría de calidad de los datos extraídos de los manuales de clases. Una verificación detallada de la Serie Enfermería reveló resultados alarmantes: **0% de precisión** en los códigos

de clase (todos incorrectos), **3 clases ausentes** (Enfermera 7, 7-A y 8), y **1 clase fabricada** (Auxiliar de Enfermería 2, que no existe en el manual original).

La causa raíz fue identificada con claridad: durante la extracción inicial de los manuales de clases por LLM, el modelo **fabricó códigos plausibles** en lugar de extraerlos del documento. Los códigos generados seguían el patrón correcto de formato, haciendo la fabricación difícil de detectar por inspección superficial. Solo la verificación manual contra el documento original reveló que ninguno correspondía a lo que está impreso en el manual.

Este incidente tiene implicaciones que trascienden el proyecto específico. Demuestra que los modelos de lenguaje pueden producir datos estructurados plausibles pero completamente incorrectos — fenómeno documentado como "hallucination", cuya manifestación en datos tabulares es particularmente peligrosa porque se confunde con datos reales. La respuesta fue triple: detener las validaciones basadas en los datos existentes, documentar el incidente (CRITICAL- DATA-QUALITY-REPORT.md), y planificar la reextracción con metodología mejorada (extracción guiada por el índice del documento, extracción de códigos de los encabezados de sección, validación cruzada contra el sumario original).

La lección más importante es que este episodio **valida retroactivamente** la decisión arquitectónica de observabilidad nativa y audit trail. El sistema de producción, con su trazabilidad completa, habría identificado — o, mejor aún, prevenido — el problema.

11.2.2 Incompatibilidad T4 con bfloat16

La GPU Tesla T4 (compute capability 7.5) no soporta bfloat16, formato numérico nativo del Gemma 3. El vLLM produce salidas vacías al intentar la conversión — una falla silenciosa detectada solo en la inspección de los resultados. La solución fue utilizar GGUF cuantizados vía llama.cpp. El hallazgo es relevante para la comunidad: el T4, por ser la GPU más accesible en deployments gubernamentales, es probablemente donde muchas instituciones intentarán ejecutar modelos modernos.

11.2.3 El Cuello de Botella –enforce-eager

Modelos que exceden ~80% de la VRAM del T4 fuerzan el flag `--enforce-eager` del vLLM, causando degradación de **100x** en el rendimiento. Este efecto transformó al GLM-OCR — modelo de apenas 0.9B parámetros — en un modelo más lento que otros 4x mayores, eliminándolo como candidato a pesar de su tamaño compacto.

11.3 Reunión de Validación con DGSC (6 de febrero de 2026)

La reunión de validación constituyó un hito importante de la cooperación. El equipo completo de la DGSC revisó el mapeo utilizando los portales interactivos e identificó **10 correcciones**, incorporadas en la versión 2.0:

Las correcciones reflejan matices que solo los profesionales que ejecutan el proceso cotidianamente podrían identificar: reestructuración de la capacitación (string → objeto con opciones y lógica), adición de la regla de "Principio de Sustitución Jerárquica", incorporación del sistema de puntuación por experiencia estatal, creación de dos nuevos gates (E7 y E8), adición de la fuente MIDEPLAN, normalización de inconsistencias textuales, estructuración de requisitos alternativos, y codificación formal del principio HITL-default.

Estas correcciones no representan fallas del levantamiento inicial — representan el refinamiento inevitable y deseable que ocurre cuando un mapeo es validado por las personas que ejecutan el proceso. El formato visual de los portales interactivos facilitó significativamente esta validación, permitiendo identificar discrepancias que difícilmente serían detectadas en un documento de texto.

---

12 Conclusiones

Las conclusiones de esta cooperación técnica se organizan en torno a ocho ejes que, tomados en conjunto, fundamentan la viabilidad y la relevancia del sistema propuesto — y que, se espera, pueden informar iniciativas similares en otros contextos iberoamericanos.

12.1 Viabilidad Técnica Demostrada

La consultoría demostró que es **técnicamente viable** construir un sistema de validación inteligente de atestados que opera en la infraestructura existente de la DGSC, sin GPU dedicada y sin dependencia de servicios en la nube. El stack compuesto por PaddleOCR-VL-1.5 y Gemma 3 4B — ambos en formato GGUF vía runtime unificado Ollama — más FastAPI y Celery para orquestación, proporciona las capacidades de OCR, extracción estructurada con garantía de JSON válido, y procesamiento asíncrono necesarias para el pipeline completo. Todo esto ejecutable en CPU con 16GB de RAM, en el hardware que la DGSC ya posee.

Esta viabilidad no es teórica: cada componente fue probado en las condiciones reales de operación, y los resultados de los benchmarks reflejan el desempeño efectivo, no proyecciones. El descubrimiento tardío del formato GGUF para PaddleOCR (marzo de 2026) simplificó adicionalmente la arquitectura, permitiendo servir OCR y LLM por el mismo runtime — una unificación que reduce significativamente la complejidad operacional.

12.2 El Problema es de Comprensión de Documentos, No de Automatización de Flujos

La expectativa inicial de la DGSC era de una herramienta basada en "R o Python + Copilot" — una visión comprensible dado el enfoque típico de proyectos de automatización administrativa. La consultoría identificó que el problema real es de naturaleza fundamentalmente diferente: extraer información estructurada de PDFs heterogéneos (escaneados, digitales, en formatos variados) y validarla contra reglas complejas distribuidas por 34 series y 117 clases de puestos. Se trata de un problema de **comprensión documental asistida por máquina**, no de automatización de workflows.

Esta distinción es determinante para el diseño de la solución: el sistema requiere modelos de visión-lenguaje (OCR) y de procesamiento de lenguaje natural (extracción estructurada), no solo scripts de manipulación de datos. Sin embargo — y este es el punto central — como los documentos contienen datos personales de candidatos, la solución debe operar 100% on-premise, lo cual impone modelos de código abierto dimensionados para el hardware existente. Esa sofisticación es hoy alcanzable con modelos de pequeño porte, operables en hardware modesto, sin costos recurrentes de licenciamiento o cloud. La democratización de los modelos de IA alteró fundamentalmente la ecuación costo-beneficio para este tipo de aplicación gubernamental.

12.3 El Mayor Dolor Está en los Portales, No en los PDFs

El hallazgo más significativo de las sesiones de trabajo con los analistas es que la **mayor parte del tiempo** no se gasta leyendo PDFs — los analistas son profesionales experimentados que evalúan documentos con relativa rapidez. El tiempo se consume en la verificación repetitiva de datos en **8 o más portales externos** (CONESUP, CCSS, MEP, colegios profesionales, etc.), cada uno con su propia interfaz, método de acceso y disponibilidad.

Este descubrimiento reorienta las prioridades de la solución. La prevalidación contra listas públicas (disponible desde la Fase 1) ofrece un quick-win inmediato. Pero la ganancia transformacional vendrá con la integración a la interfaz de interoperabilidad del Hacienda Digital (Fase 4 del Escenario B), que promete sustituir las consultas manuales por verificaciones automatizadas. El sistema de IA, con su API REST y formatos JSON, está diseñado para esa integración desde el primer día.

12.4 Tres Niveles de Optimización — No Todo Requiere IA

Una de las conclusiones más relevantes de esta cooperación — y posiblemente la más contraintuitiva en un informe sobre inteligencia artificial — es que **la mayor parte de las mejoras identificadas no requiere IA**. El análisis del proceso de

validación reveló tres niveles de optimización independientes, cada uno de los cuales agrega valor por sí solo:

Nivel 1 — Organizacional (cero tecnología). La implementación de un formulario de envío preestructurado (vía Microsoft Forms, ya licenciado por la DGSC), combinada con la estandarización de nomenclaturas de archivos y reglas de agrupación, puede eliminar aproximadamente 40% del esfuerzo actualmente invertido en la clasificación manual de documentos. El costo de implementación es nulo; el impacto, inmediato. Estas son recomendaciones de **proceso de negocio**, fundamentadas en el análisis detallado de los portales de validación (Macro v6 y Micro v3), no suposiciones genéricas de consultoría.

Nivel 2 — Automatización determinística (reglas, sin IA). La validación de firmas digitales, el cálculo de puntuaciones por experiencia, la detección de duplicados por hash y la verificación de formatos documentales son operaciones que se resuelven con lógica de programación convencional. Representan aproximadamente 30% de los gates de validación y no requieren modelos de lenguaje — solo código bien escrito y reglas bien codificadas.

Nivel 3 — Inteligencia artificial (modelos de visión-lenguaje). El reconocimiento óptico de documentos escaneados, la clasificación automática de tipos documentales y la extracción estructurada de datos son las tareas que efectivamente requieren IA — y que justifican la inversión en modelos de comprensión documental. Representan el ~35% de automatización que solo es alcanzable con las tecnologías evaluadas en este informe.

Esta estratificación tiene implicaciones prácticas directas: el **Nivel 1 puede ser implementado el próximo lunes**, sin esperar por ninguna de las decisiones tecnológicas. El Nivel 2 puede avanzar en paralelo con el desarrollo del POC. Y el Nivel 3 es lo que el pipeline de IA resuelve. Cada nivel amplifica el impacto de los siguientes — un formulario preestructurado (Nivel 1) mejora dramáticamente la precisión del OCR (Nivel 3), porque los documentos llegan precategorizados y en mejor calidad.

12.5 Configuración sobre Código

El enfoque de codificar los 34 manuales de clases en JSON configurable (config-matrix.json) en lugar de lógica de programación es fundamental para la sostenibilidad del sistema. La DGSC actualiza manuales periódicamente — es una actividad regular de la institución. Si cada actualización exigiera intervención de desarrolladores, el sistema se volvería rápidamente obsoleto o dependiente de apoyo externo permanente.

Con el enfoque adoptado, actualizar requisitos significa editar un archivo JSON — operación que el equipo TI de la DGSC puede realizar con un editor de texto. El mismo enfoque hace la arquitectura **replicable**: adaptar el sistema para otro país

consiste, esencialmente, en crear un nuevo config-matrix.json con las reglas locales. La lógica de procesamiento permanece la misma.

12.6 Calidad de los Datos es Crítica

La crisis de calidad descubierta en la Serie Enfermería — 0% de precisión en los códigos de clase extraídos — evidencia que los modelos de lenguaje pueden fabricar datos plausibles pero incorrectos, especialmente en tareas de extracción de datos tabulares. Esta fabricación es más insidiosa que un error obvio: los códigos generados seguían el formato correcto y parecían auténticos, exigiendo verificación manual contra el documento original para su detección.

La arquitectura propuesta mitiga ese riesgo en tres niveles: gramáticas GBNF (garantizan estructura JSON válida, aunque no garantizan contenido correcto), prevalidación cruzada contra listas oficiales (detecta inconsistencias de contenido), y observabilidad completa (cada extracción es auditable, con referencia al documento fuente y a la página procesada). La lección para la comunidad es clara: la validación humana de outputs de LLM no es una precaución excesiva — es un requisito de integridad.

12.7 Evolución hacia FreeBalance CSM

La documentación técnica del Proyecto Hacienda Digital ya especifica requisitos de validación automática en el módulo de Gestión del Talento Humano, incluyendo verificación contra CONARE, CONESUP y SINAES, expediente digital y trazabilidad completa. El sistema de IA propuesto llena la brecha operacional hasta que ese módulo esté implementado, y evoluciona naturalmente hacia componente de preprocesamiento del CSM cuando las APIs estén disponibles.

La compatibilidad no es coincidencia: ambos sistemas fueron diseñados para resolver el mismo problema institucional, por caminos complementarios. La integración entre ellos multiplica el valor de cada uno — el sistema de IA proporciona la capacidad de comprensión documental que el CSM no posee, y el CSM proporciona la infraestructura transaccional y de gestión que el sistema de IA no necesita replicar.

12.8 Replicabilidad CLAD

La arquitectura basada en motor de reglas genérico (pipeline fijo) + configuración específica por país (config-matrix.json) es intrínsecamente replicable. La verificación de requisitos para el ingreso a la función pública es una necesidad universal de las administraciones iberoamericanas, y varios países miembros del CLAD enfrentan desafíos análogos: el SERVIR en Perú administra procesos de evaluación por competencias para el servicio civil peruano; la DGAEP y el INA en Portugal y Brasil trabajan en la profesionalización y evaluación de desempeño; el INAPP en Paraguay desarrolla modelos de microcredenciales; el MEFP en Bolivia moderniza la gestión de

la función pública. En todos estos contextos, la validación de credenciales documentales es un componente operacional recurrente.

La adaptación del sistema para cualquiera de estos contextos consiste, esencialmente, en dos operaciones: crear un config-matrix.json con las reglas y clases del país objetivo, y ajustar el validation-sources.json para las fuentes de verificación locales. El pipeline de procesamiento — OCR, clasificación, extracción, motor de reglas — permanece inalterado.

El stack tecnológico (modelos open-source, deployment vía Docker, infraestructura CPU-only) fue deliberadamente seleccionado para maximizar esta replicabilidad: no asume hardware especializado, no requiere licencias propietarias, y no depende de conectividad cloud — restricciones que son comunes en administraciones públicas de la región. La inversión de Costa Rica, documentada en este informe, se convierte en multiplicador cuando se comparte a través de los mecanismos de cooperación horizontal del CLAD.

---

13 Recomendaciones

Las recomendaciones que se presentan a continuación derivan directamente de los hallazgos documentados en los capítulos anteriores y reflejan la conclusión central de la Sección 11.4: la optimización del proceso opera en tres niveles independientes — organizacional, determinístico y asistido por IA —, cada uno de los cuales agrega valor por sí solo. Se organizan en tres bloques por nivel de decisión: la dirección institucional de la DGSC, el equipo operacional y de TI, y el CLAD como organismo articulador de cooperación iberoamericana.

13.1 Recomendaciones Estratégicas

13.1.1 R1. Implementar Formulario de Envío Preestructurado

Prioridad: Máxima — independiente de IA

De todas las recomendaciones de este informe, esta es posiblemente la de mayor impacto inmediato y menor costo de implementación. La creación de un **segundo canal de envío** vía formulario en línea — utilizando Microsoft Forms, ya licenciado por la DGSC — donde el candidato hace upload de documentos **por categoría** (académico, experiencia, supervisión, legal), con metadatos recopilados en el punto de envío, transformaría la calidad de la entrada antes incluso de cualquier procesamiento por IA.

El impacto estimado es significativo: eliminación de aproximadamente 40% del tiempo actualmente invertido en la clasificación manual de documentos, y mejora drástica en la precisión de la IA al recibir documentos precategorizados. El equipo

técnico de la DGSC confirmó durante las sesiones de trabajo que no existen impedimentos institucionales para proponer mejoras en el proceso de envío — se trata de un cambio administrativo, no regulatorio.

Esta recomendación es independiente de todas las demás: incluso si ninguna de las otras se implementa, el formulario preestructurado genera valor por sí solo.

13.1.2 R2. Implementar Pretriaje y Procesamiento por Lotes

Prioridad: Máxima — aborda la limitación de CPU

En infraestructura CPU-only, el procesamiento toma 5-15 minutos por candidato — un tiempo que inviabiliza el uso en tiempo real, pero que es perfectamente aceptable para procesamiento por lotes. La estrategia recomendada combina tres elementos:

Pretriaje por el analista: El analista selecciona cuáles candidatos enviar para análisis, priorizando los casos más urgentes o complejos. No todos los 21.447 candidatos/año necesitan pasar por el pipeline simultáneamente — el triaje humano dirige el esfuerzo computacional hacia donde genera más valor.

Procesamiento nocturno: Los candidatos seleccionados entran en cola y son procesados fuera del horario de trabajo. Por la mañana, los informes semáforo están listos para revisión — el analista llega al trabajo con el procesamiento ya concluido.

Resiliencia por candidato: El sistema implementa retry automático, checkpoint para reanudación en caso de interrupción, y aislamiento de fallas — un PDF corrupto o un documento ilegible no interrumpe el procesamiento del lote completo.

13.1.3 R3. Iniciar con el POC Standalone Contenerizado (Escenario A)

Prioridad: Alta — demuestra valor inmediato

La recomendación es iniciar por la validación del flujo de procesamiento completo con un lote de candidatos reales, utilizando el POC descrito en el Capítulo 8. El sistema será entregado como Docker Compose — un único comando levantará el stack completo, incluyendo todos los componentes, modelos y configuraciones. No hay necesidad de esperar por FreeBalance CSM, por la adquisición de GPU, ni por la resolución de cualquier dependencia externa.

El POC debe incluir una guía de deployment paso a paso como anexo operacional, permitiendo al equipo TI de la DGSC instalar, configurar y operar el sistema de forma autónoma.

13.1.4 R4. Reextraer los 34 Manuales de Clases

Prioridad: Alta — bloqueo para validación confiable

La crisis de calidad del config-matrix.json documentada en la Sección 10.2.1 es un bloqueo para cualquier validación basada en reglas. La reextracción debe seguir una metodología mejorada que prevenga la recurrencia del problema:

Primero, extraer el índice (sumario) de cada manual, estableciendo la lista completa de clases y sus códigos a partir del índice del documento. Segundo, extraer códigos directamente de los encabezados de sección — no del cuerpo del texto, donde el LLM puede confabular. Tercero, realizar validación cruzada sistemática: el conteo de clases extraídas debe coincidir con el índice, los formatos de códigos deben seguir el patrón documentado, y una muestra de 5 series debe ser verificada manualmente contra los PDFs originales.

13.1.5 R5. Expandir Capacidad del Parque CPU Actual

Prioridad: Alta — maximiza la inversión existente

Antes de considerar la adquisición de GPU, la prioridad es expandir las capacidades del parque actual de servidores CPU-only. La asignación de más VCPUs, más RAM por máquina virtual, o la creación de VMs adicionales para procesamiento paralelo puede multiplicar la capacidad del sistema sin inversión en hardware especializado. Los servidores HPE Gen11 disponibles poseen 512GB de RAM cada uno — margen significativo para ampliar la asignación de la VM actual de 16GB. Esta expansión debe ser la primera respuesta a las necesidades de rendimiento identificadas en el POC.

13.1.6 R6. Planificar Adquisición de GPU (Fase Futura)

Prioridad: Media — multiplica la productividad

El stack seleccionado opera en CPU, pero la productividad aumenta **10-50x** con GPU. Un NVIDIA T4 16GB (~USD 500-800) o L4 24GB (~USD 1.200) transformaría el tiempo de procesamiento por candidato de minutos a segundos, viabilizando el procesamiento en tiempo real y multiplicando la capacidad del sistema. La decisión de adquisición puede fundamentarse en los resultados del POC en CPU — los mismos benchmarks que validan la precisión demuestran la ganancia potencial con aceleración por hardware.

13.2 Recomendaciones Operacionales

13.2.1 R7. Ejecutar el POC en Lotes Nocturnos

Dado que el procesamiento en CPU toma 5-15 minutos por candidato, la estrategia operacional ideal para la Fase 1 es el procesamiento fuera del horario laboral. El analista selecciona los candidatos al final del día, el sistema procesa durante la noche, y por la mañana los informes semáforo están listos para revisión. El Celery gestiona la cola automáticamente, con notificación de conclusión e informe de eventuales fallas.

13.2.2 R8. Mantener Equipo TI Capacitado y Desarrollar Competencias en IA

El sistema fue diseñado para mantenimiento autónomo por el equipo TI de la DGSC — pero autonomía requiere capacitación. Se recomienda documentación de procedimientos de actualización (cómo convertir manuales de clases en config-matrix.json), sesión de transferencia de conocimiento antes de la finalización de la cooperación, y procedimiento documentado de actualización de modelos (`ollama pull <nuevo-modelo>`) y del sistema (`docker compose pull && docker compose up -d`).

Más allá del mantenimiento operacional, se recomienda que la DGSC considere la **adquisición de talento con competencia en desarrollo de soluciones basadas en inteligencia artificial**, o, alternativamente, busque apoyo institucional de otra área del gobierno costarricense que posea esa capacidad. La evolución de un sistema de IA — ajuste de prompts, adaptación de esquemas de extracción, entrenamiento fino de modelos, interpretación de métricas de calidad — exige competencias que van más allá de la administración de infraestructura tradicional.

13.2.3 R9. Disponibilidad de Manuales de Clases en Formato Estructurado

Prioridad: Alta — impacto estratégico de largo plazo

Una recomendación de alto impacto estratégico es **involucrar al equipo responsable de la producción de los manuales de clases y especialidades** para que elabore estos documentos en formato estructurado (hoja de cálculo, base de datos o API), y no solo en PDF. Este cambio eliminaría la necesidad de OCR y extracción por IA para los documentos normativos — precisamente la etapa donde ocurrió la crisis de calidad de los datos —, y reduciría drásticamente el riesgo de errores en la codificación del config-matrix.json. Si los datos de entrada ya nacen estructurados, toda la cadena de procesamiento se simplifica. Esta recomendación se enmarca en una perspectiva más amplia de **mentalidad AI-ready en la institución**: preparar procesos y datos para ser consumidos por sistemas inteligentes es tan importante como desarrollar los propios sistemas.

13.2.4 R10. Establecer Métricas de Precisión

Tras la implementación del POC, es fundamental comparar los resultados del sistema con el juicio de los analistas en una muestra de al menos 50 candidatos. Las métricas deben incluir precision y recall por gate, puntaje subjetivo de calidad (0-5) asignado por los analistas, y tiempo ahorrado por candidato. Estas métricas son el fundamento empírico para decisiones subsiguientes: cuáles gates pueden evolucionar de amarillo (HITL) a verde (automático), y cuáles requieren refinamiento del pipeline.

13.3 Recomendaciones para el CLAD

13.3.1 R11. Publicar como Referencia de Cooperación Técnica en IA

Este informe documenta, posiblemente por primera vez en el ámbito del CLAD, la aplicación de inteligencia artificial a un proceso concreto de gestión del talento humano público. La experiencia — incluyendo los hallazgos positivos y negativos, las decisiones tecnológicas fundamentadas y la crisis de calidad de datos — constituye una referencia valiosa para la comunidad iberoamericana.

Se recomienda la publicación en el formato estándar de cooperación técnica, anexando los artefactos BMAD (Product Brief, PRD, Architecture Decision Document) como modelo de documentación replicable. El destaque al enfoque on-premise y a la soberanía de datos debe ser enfatizado como diferencial relevante para gobiernos de la región que, como Costa Rica, priorizan el control sobre datos personales de ciudadanos.

13.3.2 R12. Crear Grupo de Trabajo CLAD sobre IA en el Servicio Público

Países como Perú (SERVIR), Brasil (DGAEP/INA), Paraguay (INAPP) y Bolivia (MEFP) enfrentan desafíos similares en la gestión del talento humano público. La validación de credenciales para el ingreso a la función pública es una necesidad universal de las administraciones iberoamericanas — y la arquitectura propuesta en este informe, con su separación entre pipeline genérico y configuración específica por país, fue deliberadamente diseñada para ser adaptable a esos contextos.

Un grupo de trabajo dedicado podría compartir aprendizajes, adaptar el config-matrix.json para marcos regulatorios nacionales, y construir progresivamente un repositorio de buenas prácticas en IA aplicada al servicio civil — contribuyendo a la agenda de modernización que el CLAD promueve desde hace más de cinco décadas.

La experiencia documentada en este informe demuestra que la inteligencia artificial aplicada a la gestión del talento humano público no es una aspiración futura — es una posibilidad presente, alcanzable con infraestructura modesta, modelos de código abierto y una decisión institucional de comenzar. La DGSC tomó esa decisión al solicitar la cooperación técnica. La arquitectura está diseñada. Los artefactos están entregados. El próximo paso es procesar el primer lote de candidatos reales y medir los resultados contra el juicio de los analistas — transformando la viabilidad demostrada en valor operacional concreto.

---

14 Referencias Bibliográficas

14.1 Legislación y Normativa

- Ley Marco de Empleo Público N°10159 (10 de marzo de 2023). https://pgrweb.go.cr/scij/Busqueda/Normativa/Normas/nrm_texto_completo.aspx?param1=NRTCCnValor1=1CnValor2=96521
- Reglamento a la Ley Marco de Empleo Público. http://www.pgrweb.go.cr/scij/Busqueda/Normativa/Normas/nrm_texto_completo.aspx?param1=NRTCCnValor1=1CnValor2=99014
- Estatuto de Servicio Civil. http://www.pgrweb.go.cr/scij/Busqueda/Normativa/Normas/nrm_texto_completo.aspx?param1=NRTCCnValor1=1CnValor2=32708
- Reglamento al Estatuto de Servicio Civil. https://www.pgrweb.go.cr/scij/Busqueda/Normativa/Normas/nrm_texto_completo.aspx?nValor1=1CnValor2=8975
- Ley de Protección al Ciudadano del Exceso de Requisitos y Trámites Administrativos N°8220 (11 de marzo de 2002).
- Circular AOTC-CIR-7-2024 (condiciones de títulos de técnico medio).
- Circular DG-CIR-4-2025 (formato de declaraciones juradas).
- Resolución DG-RES-106-2024 (Técnico Medio de MEP en contabilidad).

14.2 Documentos Institucionales

- DGSC (2025). Briefing Inicial — CLAD-DGSC: Proyecto IA para Reclutamiento. San José, Costa Rica.
- DGSC (2025). Presentación de Atestados para la Verificación de Requisitos. San José, Costa Rica.
- DGSC (2026). Infraestructure — Licencias y Equipos DGSC. San José, Costa Rica.
- Ministerio de Hacienda (2022). TDR SDO_CR-MOF-267124-NC-RFB: Solicitud de Ofertas — Sistema Integrado de Administración Financiera y Talento Humano.
- Ministerio de Hacienda (s/f). MH-UPHD-PRO01-INS-002: Especificaciones Técnicas COTs.
- Manual de Organización y Funciones de la DGSC. https://www.dgsc.go.cr/documentos/transparencia/recursosh/MANUAL_DE_ORGANIZACI%C3%93N_Y_FUNCIONES_DE_LA_DGSC.pdf

14.3 Tecnología

- PaddlePaddle (2026). PaddleOCR-VL-1.5. https://huggingface.co/PaddlePaddle/PaddleOCR-VL-1.5

- Google (2025). Gemma 3. https://ai.google.dev/gemma
- llama.cpp. https://github.com/ggml-org/llama.cpp
- Ollama. https://ollama.ai
- FastAPI. https://fastapi.tiangolo.com
- Celery Project. https://docs.celeryq.dev
- Redis. https://redis.io
- HTMX. https://htmx.org
- pikepdf. https://pikepdf.readthedocs.io
- PostgreSQL. https://www.postgresql.org
- nginx. https://nginx.org
- vLLM (evaluado para Fase 2+). https://vllm.ai

14.4 Publicaciones CLAD de Referencia

- Boyer Carrera, J. (2024). Diagnóstico de Equidad de Género en las Compras Públicas Estratégicas del Estado de Costa Rica. CLAD/DGCP-Ministerio de Hacienda.
- SERVIR Perú (2024). Informe de Consultoría: Evaluación de Desempeño por Competencias. CLAD.
- DGAEP/INA (2025). Relatório PGD: Melhoria das Políticas e Práticas de Teletrabalho. CLAD/Portugal-Brasil.
- INAPP Paraguay (2024). Propuesta de Modelo de Microcredenciales. CLAD.
- MEFP Bolivia (2024). Informe de Consultoría: Taller sobre Descentralización, Autonomías, Pacto Fiscal y Federalismo. CLAD.

---

15 Anexo A: Mapa de Procesos AS-IS Completo

Versión: 1.0 **Fecha:** 8 de febrero de 2026 **Fuente:** 8 sesiones de trabajo con el equipo DGSC (5–28 ene. 2026)

15.1 A.1 Visión General del Proceso

La DGSC gestiona el proceso de reclutamiento y validación para el Régimen de Servicio Civil (RSC) de Costa Rica, abarcando **46 instituciones públicas** y aproximadamente **60% de los servidores públicos del país**.

Indicador	Valor (2024)
Procesos de reclutamiento	410
Líneas de oferta preliminar validadas	58.451

Indicador	Valor (2024)
Candidatos revisados en detalle	21.447
Tamaño del equipo analista	~10 personas

Función central: Recibir y validar **documentos comprobatorios** (atestados) enviados por candidatos que se inscriben para puestos del Servicio Civil, verificando conformidad con los requisitos definidos en los *Manuales de Clases y Especialidades*.

15.2 A.2 Flujo de Extremo a Extremo (7 Fases)

15.2.1 Fase A: Publicación de Vacante y Requisitos

- Requisitos de cada clase de puesto son publicados en el sitio público de la DGSC
- Cada clase pertenece a una **Serie** (ej.: Serie Técnica, Serie Profesional Jefe)
- Requisitos incluyen: calificaciones académicas, experiencia, supervisión, registro profesional/legal
- Algunas clases permiten **múltiples combinaciones** (combos) de educación + experiencia

Documentos fuente: 34 manuales PDF en `clases-y-especialidade-de-carrera-administrativa/`

15.2.2 Fase B: Registro e Inscripción del Candidato

- Candidatos se registran en la plataforma **Oferta de Servicios**
- El sistema **cruza datos declarados** con requisitos de la clase
- Sistema **sugiere** clases para las cuales el candidato puede calificar
- El sistema genera la lista de documentos exigidos con base en la clase y combinación declarada

Hallazgo crítico (reunión Ene 20): La recomendación del sistema funciona bien con datos declarados, pero la **validación real depende de la revisión documental** — las discrepancias entre declaraciones y calificaciones reales son frecuentes.

15.2.3 Fase C: Envío de Documentos

Método actual: Correo electrónico con adjuntos

Formato exigido: - Documentos no firmados digitalmente: Todos consolidados en un único archivo PDF - Documentos firmados digitalmente: Enviados como archivos separados para validación individual

Problemas identificados (reuniones Ene 7, Ene 19):

Estos problemas son el detonante central que generó toda la demanda por automatización con IA.

Problema	Impacto	Frecuencia
Múltiples archivos en vez de PDF único	Atrasa revisión, complica rastreo	~40% de los envíos
Documentos innecesarios incluidos	Analista busca en páginas irrelevantes	Muy frecuente
Calidad deficiente de digitalización	Puede ser ilegible	Ocasional
Documentos fuera de orden	Clasificación manual necesaria	Frecuente
Documentos incorrectos para el puesto	Desperdicia tiempo de revisión	Ocasional

Recomendación Crítica — Formulario de Envío Preestructurado: Creación de un flujo secundario de entrada vía formularios en línea, permitiendo upload por categoría (académico, experiencia, supervisión, etc.), recopilando metadatos en el punto de envío.

15.2.4 Fase D: Recepción y Asignación de Documentos

- Recepción manual — analista abre correo electrónico y vincula al registro en la plataforma
- **Ninguna vinculación automatizada** entre correo electrónico y plataforma digital
- Trazabilidad depende de esfuerzo manual
- Cuello de botella significativo para automatización

15.2.5 Fase E: Validación Documental (El Proceso Central)

La fase más compleja y demorada. El analista valida cada documento contra los requisitos del Manual de Clases para la posición seleccionada por el candidato.

15.2.5.1 Gate E1: Validación Académica

1. **Correspondencia de nombre:** Nombre en el documento = candidato en el sistema
2. **Correspondencia de ID:** Número de cédula coincide
3. **Tipo de grado:** Cumple o excede el requisito de la clase
4. Autorización institucional:
    - Universidades privadas → **CONESUP** (lista PDF pública)
    - Instituciones públicas → **MEP** (requiere acceso al sistema)
    - Programas parauniversitarios → Regulados por el CONESUP

- Grados extranjeros → Reconocimiento por **CONARE**

5. **Detección de fraude:** Verificación visual de alteraciones

Reglas especiales: - Grados superiores satisfacen automáticamente requisitos inferiores - Créditos universitarios pueden sustituir experiencia en algunas clases - Excepción contable: Registro en el Colegio de Contadores obligatorio incluso para nivel técnico

15.2.5.2 Gate E2: Validación de Firma Digital

- Dos herramientas: **Agente Gaudy** y **Validador**
- Agente Gaudy exige documentos **individualmente** (uno por vez)
- Fricción de *context switching*: Alt-Tab entre sistema de revisión y herramienta de validación
- Ingreso de PIN puede ser necesario por documento

15.2.5.3 Gate E3: Validación de Experiencia

Campos obligatorios en las certificaciones: 1. Nombre completo del candidato 2. Número de cédula 3. Nombre de la institución/empresa 4. Fechas exactas de inicio y finalización 5. Puesto ocupado 6. Funciones desempeñadas (relevancia a la especialidad) 7. Jornada de trabajo 8. Detalles de supervisión (si aplica)

Casos especiales:

Caso	Método de Validación
Freelance/Servicios profesionales	Declaración jurada + verificación en Min. de Hacienda
Experiencia en el exterior	Regulación especial; documentación adicional
Empresas cerradas/inactivas	Verificación vía Registro Nacional
Representante legal	Min. de Hacienda + Registro Nacional

Verificaciones externas: CCSS (contribuciones), Ministerio de Hacienda (estado fiscal), Registro Nacional (estado empresarial)

15.2.5.4 Gate E4: Experiencia en Supervisión (Clases Superiores)

- Aplica a: Profesional Jefe y superiores
- Ej.: Profesional Jefe 3 exige 3 años supervisando profesionales, de los cuales 18 meses en el sector público

15.2.5.5 Gate E5: Registro en Colegio Profesional

- Verificación de miembro activo
- Fecha de registro debe ser **anterior o igual** a la fecha de empleo relevante

15.2.5.c Gate Ec: Requisitos Legales

- Declaración de Bienes (anticorrupción)
- Póliza de Fidelidad (seguro-garantía)
- Autorizaciones y licencias especiales por clase
- Códigos **SICOBATEC** para diplomas de educación media (MEP)

15.2.6 Fase F: Decisión y Registro

Dos sistemas paralelos de registro:

Sistema	Finalidad	Nivel de Detalle
Sistema Oficial	Comunicación con candidatos, registro oficial	Genérico (aprobado/no aprobado)
Matriz Excel	Seguimiento interno, consultas rápidas	Detallado — razones específicas por documento

Hallazgo crítico: La matriz Excel es la **verdadera fuente de verdad** para el equipo, no el sistema oficial.

15.2.7 Fase G: Comunicación al Candidato

- Sistema envía notificación **genérica** — no especifica cuál requisito no fue cumplido
- Candidatos que desean detalles deben contactar a la DGSC
- Analistas consultan la **matriz Excel** para responder

15.3 A.3 Combinaciones de Validación (“Combos”)

Cada clase define una o más combinaciones válidas de educación + experiencia + otros requisitos. El sistema debe verificar **TODAS** las combinaciones válidas antes de rechazar.

Ejemplo: Técnico de Servicio Civil 3

Combo	Requisito Académico	Experiencia Requerida
A	Diplomado o 3er año universidad/parauniversidad en la especialidad	0 años
B	2do año universidad/parauniversidad en la especialidad	4 años en funciones relacionadas
C	Bachiller en Educación Media +	4 años en funciones relacionadas

Combo	Requisito Académico	Experiencia Requerida
	título Técnico/Técnico Medio	

Ejemplo: Profesional Jefe de Servicio Civil 3

Requisito	Detalle
Académico	Licenciatura o Posgrado en la especialidad
Experiencia funcional	7 años de experiencia profesional (posgrado)
Experiencia en supervisión	3 años supervisando profesionales
Supervisión sector público	18 meses de los 3 años deben ser en el sector público
Colegio Profesional	Registro activo, si exigido por la especialidad

15.4 A.4 Tipos de Documentos (11 Categorías)

#	Tipo de Documento	Método de Validación	Sistemas Externos
1	Títulos educacionales (diplomas)	Nombre/ID, tipo de grado, autorización institucional	CONESUP, MEP, CONARE
2	Certificaciones educacionales (formación no formal)	Horas, institución, relevancia	INA, CSP
3	Certificaciones de situación especial	Discapacidad, ascendencia afro	CONAPDIS
4	Certificaciones educacionales especiales (créditos)	Conteo de créditos, institución	Universidades
5	Certificaciones de experiencia laboral	Campos obligatorios, cálculo de fechas	CCSS, Hacienda
6	Declaraciones juradas de trabajo (3 tipos)	Servicios profesionales / exterior / empresas cerradas	Hacienda, Reg. Nacional
7	Certificaciones de supervisión de personal	Profesionales supervisados, período, sector público	Verificación institucional
8	Manuales de clases (referencia cruzada)	Docs. del candidato vs. requisitos oficiales	Base de la DGSC

#	Tipo de Documento	Método de Validación	Sistemas Externos
9	Registro en Colegio Profesional	Miembro activo, fecha de registro	Colegios Profesionales
10	Historial/antecedentes laborales	Procesos administrativos, despidos	Registros institucionales
11	Firmas digitales	Validez del certificado, identidad del firmante	Banco Central (SINPE/FVA)

15.5 A.5 Sistemas Externos e Integraciones

15.5.1 Actualmente Utilizados (Consultas Manuales)

Sistema	Finalidad	Método de Acceso
CONESUP	Verificar autorización de universidad privada	Lista PDF pública
MEP	Verificar diplomas de educación pública (SICOBATEC)	Login oficial necesario
Agente Gaudy / Validador	Validación de firma digital	Aplicación de escritorio
CCSS	Reportes de contribución de seguridad social	Acceso institucional
Min. de Hacienda	Estado fiscal, registro de servicios profesionales	Portal público
Registro Nacional	Estado de empresas, representantes legales	Acceso limitado
Colegios Profesionales	Verificación de afiliación	Varía por colegio
TSE	Verificación de identidad para discrepancias	Acceso institucional

15.5.2 Futuro (Hacienda Digital / Agenda Digital)

Sistema	Módulo	Estado
FreeBalance CSM	Civil Service Management	Módulo atrasado, sin docs. de API
Certeza	Gestión del ciclo de vida del empleo	En desarrollo

Sistema	Módulo	Estado
APIs de Interoperabilidad	MEP, CONAPDIS, Judicial, Migración	Convenios en curso

15.6 A.6 Resumen de Puntos de Dolor (18 Identificados)

15.6.1 Puntos de Dolor del Proceso

1. **Sobrecarga de volumen:** 21.447 candidatos revisados manualmente con ~10 analistas
2. **Entrada no estructurada:** Blob PDF único con todos los documentos mezclados
3. **Envío por correo electrónico:** Ninguna vinculación automática con registros
4. **40% de incumplimiento:** Candidatos no siguen reglas de envío
5. **Documentos innecesarios:** Candidatos envían atestados irrelevantes

15.6.2 Puntos de Dolor de la Validación

6. **Context switching:** Alt-Tab entre sistema de revisión y herramientas de validación
7. **Cuello de botella de firma digital:** Validación por documento con ingreso de PIN
8. **Consultas externas manuales:** Verificación en 8+ sistemas externos
9. **Detección de fraude visual:** Documentos alterados detectados por inspección humana
10. **Discrepancias de ID:** Naturalización, cambios de nombre causan inconsistencias

15.6.3 Puntos de Dolor Sistémicos

11. **Registro doble:** Mismos datos en el sistema oficial Y en el Excel
12. **Búsqueda sin cédula:** Sistema oficial no busca por ID del candidato
13. **Notificaciones genéricas:** Candidatos no saben POR QUÉ fueron rechazados
14. Sin pista de auditoría en el sistema: Razones reales solo en el Excel

15.6.4 Puntos de Dolor Institucionales

15. **Variación de experiencia entre analistas:** Diferentes analistas = diferente velocidad/precisión
16. **Sin docs. de API del Hacienda Digital:** Imposibilidad de integrar con plataforma futura
17. **Sin infraestructura GPU:** Servidores on-premise (HP Gen11) solo CPU
18. **Burocracia de adquisición cloud:** Acceso Azure requiere proceso completo de compra

15.7 A.7 Herramientas e Infraestructura Actual

Componente	Tecnología
Servidores	4x HP Gen11 (2x CPU Intel, 512 GB RAM cada uno)
Virtualización	Windows Server Datacenter + Hyper-V
Almacenamiento	Array SSD dedicado
Respaldo	HPE StoreOnce + biblioteca de cintas
Licenciamiento	Microsoft 365 Business Standard
Directorio	Azure Active Directory Premium
Analítica	Power BI Pro
Desarrollo	Visual Studio Professional
Experimento IA	Microsoft Copilot Studio (free tier)
Modelos IA probados	Dipsy (múltiples versiones, CPU-only, lento)

15.8 A.8 Métricas y KPIs Acordados

1. **Matriz de Confusión** (Precisión/Recall) — válida solo con supervisión humana
2. **Puntaje de Calidad Subjetivo** (0-5) — analista evalúa calidad del output de la IA
3. **Explicabilidad** — nativa al diseño del sistema, no métrica separada
4. **Pista de auditoría** — cada decisión debe tener justificación exacta

15.8.1 Fases de Implementación

1. **100% supervisado:** Todas las decisiones de la IA revisadas por humano
2. **Autonomía gradual:** Reducir revisión humana conforme la IA se demuestre confiable
3. **Muestreo aleatorio:** Mantener revisión por muestra incluso en autonomía total
4. **Calibración continua:** Analistas deben acordar rúbrica de puntuación

---

15.6 A.6 Diagrama de Proceso (Mermaid)

```
flowchart TD
    A[DGSC Publica Vacante] --> B[Requisitos del Manual de Clases]
    B --> C[Candidato se Registra en la Plataforma]
    C --> D[Ingresa Datos Personales/Académicos/Experiencia]
    D --> E[Sistema Recomienda Clases Elegibles]
    E --> F[Candidato Selecciona Posición]
    F --> G[Sistema Lista Documentos Requeridos]
    G --> H[Candidato Digitaliza y Prepara Documentos]
    H --> I[Envía por Correo Electrónico como PDF Único]
```

```
I --> J[DGSC Recibe Correo Electrónico]
J --> K[Analista Vincula Correo al Registro]
K --> L[Abre PDF — Inicia Revisión]

L --> M{Gate E1: Validación Académica}
M -->|Nombre/ID| M1[Verificar Identidad]
M -->|Tipo de Grado| M2[Verificar vs Requisitos]
M -->|Institución| M3[Verificar CONESUP/MEP]
M -->|Fraude| M4[Inspección Visual]

M1 & M2 & M3 & M4 --> N{Gate E2: Firma Digital}
N --> N1[Descargar PDF]
N1 --> N2[Abrir Agente Gaudy/Validador]
N2 --> N3[Verificar Firma y Firmante]

N3 --> O{Gate E3: Experiencia}
O --> O1[Verificar Campos Obligatorios]
O1 --> O2[Calcular Total de Años]
O2 --> O3[Verificar Relevancia a la Especialidad]
O3 --> O4[Cruzar con Sistemas Externos]

O4 --> P{Gate E4: Supervisión}
P -->|Si Exigido| P1[Verificar Personal Profesional]
P1 --> P2[Calcular Período]
P2 --> P3[Verificar Porción Sector Público]
P -->|No Exigido| Q

P3 --> Q{Gate E5: Colegio Profesional}
Q -->|Si Exigido| Q1[Verificar Registro Activo]
Q1 --> Q2[Verificar Fecha de Registro]
Q -->|No Exigido| R

Q2 --> R{Gate E6: Requisitos Legales}
R --> R1[Verificar Declaraciones/Pólizas]
R1 --> R2[Autorizaciones Especiales]

R2 --> S{¿Todos los Gates Aprobados?}
S -->|Sí| T[APROBADO — Registrar en Sistema]
S -->|No| U[RECHAZADO — Registrar en Sistema]

T --> V[Registrar Detalles en Matriz Excel]
U --> V
V --> W[Enviar Correo Genérico al Candidato]

style A fill:#e1f5fe
style S fill:#fff3e0
style T fill:#e8f5e9
```

```
    style U fill:#ffebee
    style V fill:#fff9c4
```

Fuente: *artifacts/process-mapping/process-map-as-is.md* — Mapeo realizado en 8 sesiones con el equipo DGSC (enero 2026)

---

16 Anexo B: Resultados del Benchmark OCR

Fecha: 20 de febrero de 2026 **Ambiente de prueba:** Microsoft Azure NC-series, NVIDIA Tesla T4 (16 GB GDDR6) **Framework de serving:** vLLM (última versión)

16.1 B.1 Resumen Ejecutivo

Evaluación sistemática de cuatro modelos VLM de código abierto para extracción de texto de documentos del servicio civil costarricense. El objetivo: identificar el modelo ideal para el pipeline de validación de la DGSC, ejecutándose on-premise con aceleración GPU.

Resultado: PaddleOCR-VL-1.5 (0,9B parámetros) es el modelo recomendado para producción.

Modelo	Parámetros	Resultado	Velocidad/Página	Estado
PaddleOCR-VL-1.5	0,9B	SELECCIONADO	~3-5s	Rápido, estable, eficiente en memoria
GLM-OCR	0,9B	RECHAZADO	~376s	Requiere –enforce-eager; 100x lento
DeepSeek-OCR-2	3,4B MoE	RECHAZADO	N/A (crash)	Out of Memory tras ~15 páginas
Hunyuan OCR	1B	NO PROBADO	Desconocido	Restricción de tiempo;

Modelo	Parámetros	Resultado	Velocidad/Página	Estado
				probable similar al GLM

16.2 B.2 Ambiente de Prueba

Componente	Especificación
Cloud Provider	Microsoft Azure
VM	Standard NC-series con GPU
GPU	NVIDIA Tesla T4 (arquitectura Turing)
Compute Capability	7.5
VRAM	16 GB GDDR6 (15,56 GiB utilizables)
Framework	vLLM (pip install)
OS	Ubuntu 24.04 LTS
Python	3.12
CUDA	12.x
Bibliotecas	xformers, FlashInfer, Triton

16.2.1 Disponibilidad de Backends de Atención en la T4

Backend	Disponible	Notas
Flash Attention 2	NO	Requiere compute capability >= 8.0; T4 es 7.5
FlashInfer	Sí	Usado para atención del decoder
XFORMERS	Sí	Atención eficiente en memoria para GPUs antiguas
Triton Attention	Sí	Usado para atención del encoder ViT
Eager/Naive	Sí (fallback)	Extremadamente lento; último recurso

16.3 B.3 Metodología

16.3.1 Documento de Prueba

- **Archivo:** SERIE ENFERMERÍA.pdf (36 páginas)
- **Contenido:** Requisitos de clases del servicio civil para la serie Enfermería
- **Clases conocidas:** 10 clases con códigos únicos
- **Idioma:** Español

16.3.2 Ground Truth

Establecida vía **Claude Opus 4.6** (Azure AI Foundry): - Extracción completa de texto (79.481 caracteres) - Extracción estructurada de campos (10 clases con requisitos) - 36 páginas renderizadas a 300 DPI PNG

16.3.3 Métricas de Evaluación

Métrica	Descripción
Velocidad (s/página)	Tiempo real vía API vLLM
Similitud textual	SequenceMatcher ratio vs ground truth
Word F1	Precisión/recall a nivel de palabra
Detección de clases	Conteo de 10 nombres de clases encontrados
Detección de códigos	Conteo de 10 códigos de clases encontrados
Uso de memoria	Pico de consumo VRAM
Estabilidad	Capacidad de completar 36 páginas sin crash

16.4 B.4 Resultados Detallados por Modelo

16.4.1 B.4.1 GLM-OCR (0,9B) — RECHAZADO: Velocidad Inaceptable

```
VLLM_ATTENTION_BACKEND=XFORMERS vllm serve zai-org/GLM-OCR \
    --port 8000 --max-model-len 8192 --gpu-memory-utilization 0.80 \
    --enforce-eager --dtype half
```

Métrica	Valor
Velocidad por página	**376,4 segundos** (~6,3 minutos)
Proyección total (36 páginas)	3,7 horas
Peso del modelo	2,2 GiB
VRAM total usada	~12-14 GiB
Estabilidad	Estable (sin crashes con –enforce-eager)

Causa raíz: Cadena de restricciones: 1. Flash Attention 2 no disponible (T4 compute 7.5 < 8.0 requerido) 2. `--enforce-eager` necesario para evitar OOM → deshabilita CUDA graphs y torch.compile 3. Sin `--enforce-eager`: crash por OOM (378 MiB necesarios, 336 MiB libres) 4. **Efecto neto:** ~376s/página en vez de ~3-10s/página — **degradación de 100x**

16.4.2 B.4.2 DeepSeek-OCR-2 (3,4B MoE) — RECHAZADO: Out of Memory

```
vllm serve deepseek-ai/DeepSeek-OCR-2 \
    --port 8000 --trust-remote-code --max-model-len 4096 \
    --gpu-memory-utilization 0.92 --enforce-eager --dtype half
```

Métrica	Valor
Velocidad	N/A (crash)
Páginas completadas	~15 de 36 antes del OOM
Pico de VRAM	**14,G4 GiB** / 15,56 GiB disponibles
Estabilidad	**Inestable** — crashes tras ~15 páginas

Descomposición de memoria:

Componente	Memoria
PyTorch asignado	14,34 GiB
PyTorch reservado (no asignado)	466,52 MiB
Total del proceso	14,94 GiB
Disponible	15,56 GiB
Déficit	~730 MiB necesarios, 630 MiB libres

Conclusión: Excede capacidad de la T4 16 GB. **Requiere mínimo 24 GB VRAM** (L4, A10 o RTX 4090).

16.4.3 B.4.3 PaddleOCR-VL-1.5 (0,9B) — SELECCIONADO para Producción

```
vllm serve PaddlePaddle/PaddleOCR-VL-1.5 \
    --port 8000 --trust-remote-code --max-model-len 8192 \
    --max-num-batched-tokens 16384 --gpu-memory-utilization 0.90 \
    --no-enable-prefix-caching --mm-processor-cache-gb 0 --dtype half
```

Métrica	Valor
Velocidad por página	~3-5 segundos
Proyección total (36 páginas)	~2-3 minutos
VRAM estimada	~4 GiB
Estabilidad	**Estable** — completa todas las páginas
–enforce-eager necesario	No

Ventajas clave: 1. Sin necesidad de –enforce-eager → optimizaciones CUDA completas 2. 100x más rápido que GLM-OCR 3. Margen confortable: usa ~4 GB de los 16 GB, dejando espacio para modelos adicionales

16.4.4 B.4.4 HunyuanOCR (1B) — NO PROBADO

Configuración preparada pero no ejecutada por restricciones de tiempo. Expectativa: comportamiento similar al GLM-OCR (1B con –enforce-eager).

16.5 B.5 Comparación de Rendimiento

16.5.1 Tabla Resumen

Modelo	Parám.	VRAM	Vel./Pág.	Total (36pg)	enforce-eager	Viable
PaddleOCR-VL-1.5	0,9B	~4 GB	~3-5s	~2-3 min	No	SÍ
GLM-OCR	0,9B	~12 GB	~376s	~3,7 hrs	Sí (requerido)	No
DeepSeek-OCR-2	3,4B MoE	>15 GB	N/A	N/A	Sí (crash)	No
HunyuanOCR	1B	~6 GB est.	Desconocido	Desconocido	Sí (probable)	?

16.5.2 Análisis de Throughput

Modelo	Páginas/Hora	Docs/Día (8h)	Candidatos/Día*
PaddleOCR-VL-1.5	~900	~7.200	~1.440
GLM-OCR	~10	~80	~16

*Asumiendo ~5 páginas por candidato.

Contexto DGSC: Con 58.451 líneas de oferta y 21.447 candidatos (base 2024), PaddleOCR-VL-1.5 podría procesar toda la carga anual en aproximadamente **15 días hábiles** con una única GPU T4.

16.6 B.6 Descubrimientos Técnicos Críticos

16.6.1 El Cuello de Botella –enforce-eager

El flag `--enforce-eager` es el diferenciador crítico:

- **Qué hace:** Deshabilita CUDA graphs y optimizaciones torch.compile
- **Por qué es necesario:** CUDA graphs asignan memoria GPU adicional

- **Impacto:** ~100x degradación de rendimiento (3s → 376s por página)
- **Quién lo necesita:** Modelos que consumen >80% de la VRAM
- **Quién no lo necesita:** PaddleOCR-VL-1.5 (usa solo ~25% de la VRAM)

16.6.2 Indisponibilidad de Flash Attention 2

La T4 (compute capability 7.5) no puede usar Flash Attention 2 (requiere >= 8.0). Esto elimina el camino más rápido de atención y fuerza fallback a FlashInfer o XFORMERS.

16.6.3 Trade-off Memoria vs. Velocidad

```
Presupuesto VRAM:  15,56 GiB total (T4)
                   ├── Pesos del modelo: 2-15 GiB (varía)
                   ├── KV cache: 1-4 GiB (varía por contexto)
                   ├── CUDA graphs: 0,5-2 GiB (si habilitados)
                   └── Buffer de operaciones: 0,5-1 GiB

PaddleOCR-VL: ~4 GiB total → 11+ GiB margen → CUDA graphs CABEN → RÁPIDO
GLM-OCR:      ~12 GiB total → ~3 GiB margen → CUDA graphs NO CABEN → LENT
O
DeepSeek:     ~15 GiB total → ~0 GiB margen → CRASH
```

16.7 B.7 Recomendaciones de Hardware para DGSC

GPU	VRAM	Costo Est. (USD)	Modelos Soportados	Adecuación DGSC
T4	16 GB	$500-800	Solo PaddleOCR-VL (rápido)	Mínimo viable
L4	24 GB	$1.200	Todos los 4 modelos	Recomendado
A10	24 GB	$1.500	Todos + procesamiento por lotes	Producción
RTX 4090	24 GB	$1.600	Todos + inferencia más rápida	Mejor costo/beneficio
CPU only	N/A	$0	PaddleOCR-VL (~30-60s/pág.)	Fallback presupuestario

Para el POC de la DGSC: T4 16 GB es suficiente con PaddleOCR-VL-1.5. **Para producción:** Considerar L4 24 GB para flexibilidad futura. **Recomendación de deployment:** On-premise, preferido por soberanía de datos (datos personales de candidatos).

16.8 B.8 Arquitectura de Producción Recomendada

```
flowchart TD
    A["Documento del Candidato (PDF/Imagen)"] --> B

    subgraph Stage1["Etapa 1: Extracción OCR"]
        B["PaddleOCR-VL-1.5 (0,9B)<br/>~4 GB VRAM | ~3-5s/página"]
    end

    B --> C["Texto bruto extraído"]
    C --> D

    subgraph Stage2["Etapa 2: Extracción Estructurada"]
        D["Gemma 3 4B QAT GGUF<br/>vía llama.cpp + gramática GBNF<br/>~2,
6 GB | ~2-5s"]
    end

    D --> E["JSON validado (schema Pydantic)"]
    E --> F

    subgraph Stage3["Etapa 3: Motor de Reglas"]
        F["Validación basada en reglas<br/>config-matrix.json<br/>Sin GPU
 | <1s"]
    end

    F --> G["DECISIÓN + PISTA DE AUDITORÍA"]

    style Stage1 fill:#e3f2fd
    style Stage2 fill:#f3e5f5
    style Stage3 fill:#e8f5e9
```

Total por candidato: ~20-50 segundos **Total VRAM:** ~6 GB (PaddleOCR 4 GB + Gemma 2 GB) **Hardware:** T4 16 GB (10 GB de margen)

Fuente: *_bmad/clad/technical-reports/ocr-benchmark-results.md* Benchmarks GPU realizados en VM Azure, 20 de febrero de 2026

---

17 Anexo C: Evaluación de Frameworks de Extracción Estructurada

Fecha: 20 de febrero de 2026 **Estado:** Final

17.1 C.1 Resumen Ejecutivo

Evaluación de frameworks para convertir texto bruto de OCR en JSON validado para el pipeline de validación de la DGSC. El candidato original era **LangExtract** (Google), pero la investigación reveló que **no es adecuado** para el caso de uso.

Recomendación: Usar llama.cpp con gramáticas GBNF (JSON schema → grammar) y esquemas Pydantic.

17.2 C.2 Evaluación de LangExtract

17.2.1 Qué Es

LangExtract (Apache 2.0, ~33,4k stars en GitHub) es una biblioteca Python de Google que usa LLMs para extraer información estructurada de texto con **source grounding** (mapeo de offsets a nivel de carácter).

- Lanzamiento: Julio 2025
- **Versión actual:** v1.1.1 (Noviembre 2025)
- **Backend primario:** Google Gemini (2.5 Flash/Pro)

17.2.2 Por Qué No Es Adecuado

Criterio	Evaluación	Impacto
Definición de schema	Ejemplos few-shot, NO modelos Pydantic	Tenemos 117 esquemas de clases; crear ExampleData para cada uno es impracticable
Soporte a modelos locales	Sin restricciones de schema con Ollama	Pierde garantías de salida estructurada — inaceptable
Integración vLLM	No soportada	Nuestro stack estaba basado en vLLM
Dependencia de cloud	Optimizado para APIs Gemini/OpenAI	DGSC requiere on-premise, sin llamadas API externas
Madurez	5 meses	Demasiado joven para infraestructura gubernamental crítica

17.2.3 Lo Que Hace Bien (para referencia)

- **Source grounding:** Mapea entidades extraídas a offsets exactos en el texto fuente
- **Visualización HTML:** Visualizaciones de auditoría autocontenidas
- **Chunking de documentos:** Soporte nativo para documentos largos
- **Extracción multi-pass:** Múltiples intentos secuenciales para calidad

Estos conceptos son valiosos para los requisitos de observabilidad de la DGSC y pueden ser implementados por separado.

17.3 C.3 Alternativa Recomendada: llama.cpp con Gramáticas GBNF

17.3.1 Cómo Funciona

llama.cpp soporta conversión nativa de JSON schema a gramáticas **GBNF** (GGML BNF). En cada posición de token, la gramática restringe los tokens posibles para que el modelo **solo pueda producir JSON válido** correspondiente al schema objetivo.

Diferencia fundamental de prompt engineering o enfoques por retry: - **Restricción a nivel de token:** JSON inválido es físicamente imposible - **Garantía de schema:** La salida siempre corresponde al modelo Pydantic - **Sin retries:** La primera generación es siempre válida - **Integrado en llama.cpp:** Backend nativo, cero infraestructura adicional

17.3.2 Arquitectura

```
flowchart LR
    A["Texto OCR<br/>(PaddleOCR-VL)"] --> B["Ollama/llama.cpp<br/>Gemma 3
 4B QAT GGUF<br/>+ gramática GBNF"]
    B --> C["JSON Garantizadamente Válido<br/>(schema Pydantic)"]
    C --> D["Motor de Validación<br/>Basado en Reglas"]
```

17.3.3 Ejemplo de Uso

```python
from pydantic import BaseModel, Field
from typing import List, Optional

# Esquemas de extracción
class TituloAcademico(BaseModel):
    nivel_grado: str = Field(
        description="bachillerato, tecnico, licenciatura, maestria, docto
rado"
    )
    area_estudio: str
    institucion: str
```

```
    ano_graduacion: Optional[int] = None

class ExperienciaLaboral(BaseModel):
    cargo: str
    empleador: str
    fecha_inicio: str
    fecha_fin: str
    anos_calculados: float
    involucra_supervision: bool
    sector_publico: bool

class DocumentoCandidato(BaseModel):
    nombre_candidato: str
    cedula: str = Field(description="Número de cédula")
    codigo_clase: str
    titulos_academicos: List[TituloAcademico]
    experiencia_laboral: List[ExperienciaLaboral]
    colegio_profesional: Optional[str] = None

# Llamada vía Ollama (llama.cpp backend)
import requests, json

response = requests.post("http://localhost:11434/api/chat", json={
    "model": "gemma3:4b-it-qat-q4_0",
    "messages": [
        {"role": "system", "content": "Extraiga datos de calificación del
candidato del texto OCR a continuación. Sea preciso con fechas, niveles
de grado y nombres de instituciones."},
        {"role": "user", "content": ocr_text}
    ],
    "format": DocumentoCandidato.model_json_schema(),
    "stream": False
})

# Resultado GARANTIZADAMENTE válido como DocumentoCandidato JSON
candidato = DocumentoCandidato.model_validate_json(
    response.json()["message"]["content"]
)
```

17.3.4 Presupuesto de Memoria

Componente	VRAM	Notas
PaddleOCR-VL-1.5	~4-8 GB	Extracción OCR
Gemma 3 4B QAT GGUF	~2,6 GB	Extracción estructurada
Compilación de gramática	~0,1 GB	Una vez por schema
Margen KV cache	~2 GB	Para documentos largos
Total	~8-12 GB	Cabe en la T4 16 GB

17.3.5 Estimación de Rendimiento

Etapa	Tiempo	Notas
OCR (5 páginas promedio)	~15-25s	PaddleOCR-VL @ 3-5s/página
Extracción estructurada	~2-5s	Gemma 3 4B con gramática GBNF
Motor de validación	<1s	Matching de reglas en Python
Total por candidato	~20-30s	

17.4 C.4 Frameworks Alternativos Evaluados

17.4.1 Matriz Comparativa

Característica	LangExtract	Outlines (vLLM)	llama.cpp GBNF	Instructor
Schema vía Pydantic	No (few-shot)	Sí (nativo)	Sí (conversión automática)	Sí
Funciona en T4	No (Gemini cloud)	No (Gemma 3 broken)	Sí	Sí (con llama.cpp)
Garantía de output	Solo modelos cloud	Token-level	Token-level	Retry-based
Modelos locales	Degradado	Soporte total	Soporte total	Sí
Source grounding	Excelente	No	No	No
Madurez	5 meses	2+ años	3+ años	3+ años
Infra adicional	API key Gemini	Ninguna	Ninguna	Ninguna
Ajuste con nuestro stack	Pobre	Bloqueado (T4)	Excelente	Bueno (fallback)

17.4.2 Instructor (Opción de Fallback)

Instructor (3M+ descargas mensuales) envuelve llamadas LLM con validación Pydantic y retries automáticos. Útil para: - Llamadas a APIs externas (Claude, Gemini) para casos extremos - Interfaz multi-proveedor con los mismos modelos Pydantic

Limitación: Funciona a nivel de API (validación post-generación + retry), no a nivel de token.

17.5 C.5 Catálogo de Esquemas (Por Tipo de Documento)

Tipo de Documento	Schema Pydantic	Campos Clave
Título Educacional	`TituloAcademico`	nivel_grado, area, institucion, ano
Experiencia Laboral	`ExperienciaLaboral`	cargo, empleador, fechas, anos, supervision
Colegio Profesional	`ColegioProfesional`	colegio, registro, fecha, estado
Declaración Jurada	`DeclaracionJurada`	tipo, periodo, empleador, notarizada
Firma Digital	`FirmaDigital`	firmante, certificado_id, validez
Candidatura Completa	`DocumentoCandidato`	Todos los anteriores combinados

17.6 C.6 Observabilidad (Inspirada por LangExtract)

Aunque no utilizamos LangExtract, adoptamos su concepto de **source grounding**: - Registrar offsets de carácter para cada campo extraído - Generar visualizaciones HTML de auditoría (resaltar texto extraído en la fuente) - Almacenar puntajes de confianza por campo - Permitir revisión por el analista con vista lado a lado fuente/extracción

---

17.7 C.7 Conclusiones

1. **LangExtract no es adecuado** — esquemas few-shot, dependiente de Gemini, sin soporte vLLM
2. **llama.cpp GBNF es la opción ideal** — garantías a nivel de token, cero infra adicional, Pydantic nativo, funciona en la T4
3. **Instructor como fallback** — para casos extremos que requieran APIs cloud
4. **Presupuesto de memoria cabe en la T4** — PaddleOCR (~4-8 GB) + Gemma 3 4B (~2,6 GB)
5. **Concepto de source grounding adoptado** — implementar visualización de auditoría de LangExtract independientemente

---

Fuente: *`_bmad/clad/technical-reports/structured-extraction-evaluation.md`*
Evaluación de frameworks realizada en febrero de 2026

18 Anexo D: Evaluación de Gemma 3 para el Pipeline de Extracción DGSC

Fecha: 26 de febrero de 2026 **Estado:** Final

18.1 D.1 Familia de Modelos Gemma 3

Gemma 3, lanzado por Google en marzo de 2025, está disponible en **5 tamaños de parámetros**:

Modelo	Parámetros	Modalidad	Ventana de Contexto	Tokens de Entrenamiento
Gemma 3 270M	270M	Texto	32K	N/A
Gemma 3 1B	1B	Texto	32K	2T
Gemma 3 4B	4B	Texto + Visión	128K	4T
Gemma 3 12B	12B	Texto + Visión	128K	12T
Gemma 3 27B	27B	Texto + Visión	128K	14T

Los modelos 4B, 12B y 27B incluyen un encoder de visión SigLIP (~400M parámetros). Los modelos 1B y 270M son solo texto.

18.2 D.2 Requisitos de VRAM

18.2.1 Pesos del Modelo

Modelo	BF16	Int4 (QAT)
Gemma 3 1B	~2 GB	~0,5 GB
Gemma 3 4B	~8 GB	~2,6 GB
Gemma 3 12B	~24 GB	~6,6 GB
Gemma 3 27B	~54 GB	~14,1 GB

18.2.2 VRAM Total (Pesos + KV Cache)

Modelo (Int4)	Contexto 2K	Contexto 8K	Contexto 32K
Gemma 3 1B	~0,7 GB	~1,2 GB	~2,5 GB
Gemma 3 4B	~2,8 GB	~3,6 GB	~5,6 GB
Gemma 3 12B	~7,0 GB	~8,5 GB	~13 GB

Google proporciona checkpoints oficiales **Quantization-Aware Trained (QAT)** en los formatos `int4-unquantized` (PyTorch/Transformers) y `q4_0-gguf` (llama.cpp/Ollama). El QAT reduce la caída de perplejidad en 54% comparado con cuantización estándar post-entrenamiento.

18.3 D.3 Descubrimiento Crítico: Incompatibilidad T4 + vLLM

18.3.1 El Problema

La Tesla T4 (compute capability 7.5) **NO soporta bfloat16** (requiere >= 8.0). Gemma 3 fue entrenado en BF16, y forzar `dtype=float16` en vLLM produce **salidas vacías** (todos ceros). Reproducido entre vLLM v0.8.3 y v0.8.5.post1.

18.3.2 Problemas con Salida Estructurada

Múltiples bugs abiertos en vLLM: - **Issue #15766:** Salida estructurada de Gemma 3 causa errores de assertion con backend `xgrammar` (default) - **Issue #21148:** Gemma 3 27B + JSON structured decoding hace que el servidor **se congele** tras 1-2 tokens - **Issue #20341:** Salida repetida/ausente con Gemma 3 en vLLM - Backend **Outlines** en vLLM sufre regresiones de rendimiento

18.3.3 Compatibilidad por Runtime

Runtime	¿Funciona con Gemma 3 en la T4?	Notas
vLLM	NO	Salidas vacías (float16 fallback)
HuggingFace Transformers	NO	NaN/Inf por overflow
Unsloth	SÍ	Patches especiales de compatibilidad float16
llama.cpp /	SÍ	Formatos

Runtime	¿Funciona con Gemma 3 en la T4?	Notas
Ollama (GGUF)		cuantizados evitan el problema

Veredicto: vLLM + Gemma 3 en la T4 está broken para inferencia básica y salida estructurada.

18.4 D.4 Capacidades Multilingüe (Español)

Gemma 3 es una **evolución importante** para multilingüe:

- Entrenado en **140+ idiomas** (vs. solo inglés en Gemma 1 y 2)
- **Doble de datos multilingües** tanto en el preentrenamiento como en el post-entrenamiento
- 35+ idiomas soportados con instruction tuning
- Vocabulario de 256K tokens optimizado para idiomas diversos
- Español explícitamente listado como idioma soportado
- Gemma 3 27B lidera todos los modelos open-weight en español en LMSys Chatbot Arena

Para el caso de uso de la DGSC (extraer datos estructurados de documentos gubernamentales en español), Gemma 3 representa una mejora sustancial sobre Gemma 2.

18.5 D.5 Comparación con Gemma 2 2B

Característica	Gemma 2 2B	Gemma 3 1B	Gemma 3 4B
Parámetros	2B	1B	4B
Ventana de Contexto	8K	32K	128K
Multilingüe	Limitado	140+ idiomas	140+ idiomas
Soporte a Español	Débil	Fuerte	Fuerte
VRAM (BF16)	~4-5 GB	~2 GB	~8 GB
VRAM (Int4)	~1,2 GB	~0,5 GB	~2,6 GB
Modalidad	Texto	Texto	Texto + Visión
MATH benchmark	27,2%	48,0%	N/A (superior)
Arquitectura	Soft-capping attention	QK-norm (más rápido)	QK-norm (más rápido)
JSON Estructurado	N/A	Débil (10%	Significativament

Característica	Gemma 2 2B	Gemma 3 1B	Gemma 3 4B
(LLMStructBench)		flawless)	e mejor

Mejoras clave: - Gemma 3 1B equivale al Gemma 2 2B con **la mitad de los parámetros** - Ventana de contexto **4x mayor** (32K vs 8K) — crítico para texto OCR de documentos completos - **Soporte multilingüe nativo** vs esencialmente solo inglés - Eficiencia arquitectónica: ratio 5:1 local-to-global attention reduce KV cache en ~60%

Para **extracción JSON estructurada**, el paper LLMStructBench encontró que el Gemma 3 1B alcanza solo ~10% de outputs JSON perfectos. Hay "mejora pronunciada" del 1B al 4B. **El Gemma 3 4B es donde la extracción estructurada se vuelve viable.**

18.6 D.6 Análisis de Ajuste en la T4 16 GB

Presupuesto: 16 GB total. PaddleOCR-VL-1.5 necesita compartir la GPU.

18.6.1 Escenarios de Ajuste (T4 = 16 GB)

Si PaddleOCR-VL-1.5 usa ~8 GB (quedan ~8 GB para LLM):

Modelo	Pesos Int4	+ KV Cache (2K)	¿Cabe junto con PaddleOCR?
Gemma 3 1B	0,5 GB	~0,7 GB	SÍ (fácilmente)
Gemma 3 4B	2,6 GB	~2,8 GB	SÍ (confortablemente)
Gemma 3 12B	6,6 GB	~7,0 GB	MARGINAL
Gemma 3 27B	14,1 GB	~15 GB	NO

Si se ejecuta secuencialmente (descarga PaddleOCR, carga LLM):

Modelo	Int4 + KV (2K)	¿Cabe solo en la T4?
Gemma 3 1B	~0,7 GB	SÍ
Gemma 3 4B	~2,8 GB	SÍ
Gemma 3 12B	~7,0 GB	SÍ
Gemma 3 27B	~15 GB	MARGINAL

18.7 D.7 Recomendación: Gemma 3 4B QAT GGUF vía llama.cpp

18.7.1 Por Qué 4B, No 1B

- LLMStructBench muestra que el 1B alcanza solo ~10% de extracciones JSON perfectas — inaceptable para producción
- El 4B muestra "mejora pronunciada" sobre el 1B en extracción estructurada

- Cabe confortablemente en la T4 junto con PaddleOCR-VL-1.5 (~2,6 GB de pesos)
- Ventana de contexto 128K (vs 32K del 1B) — importante para texto OCR completo
- Fuerte soporte a español (140+ idiomas)

18.7.2 Por Qué GGUF vía llama.cpp, No vLLM

- T4 no soporta bfloat16 — vLLM produce salidas vacías con Gemma 3
- llama.cpp GGUF evita el problema de bfloat16 por completo
- llama.cpp soporta nativamente conversión de JSON schema a gramática GBNF
- Comprobado funcionando en hardware T4

18.7.3 Stack de Inferencia Recomendado

Componente	Tecnología	VRAM Est.
OCR	PaddleOCR-VL-1.5 (vía Ollama)	~6-8 GB
LLM de Extracción	`gemma-3-4b-it-qat-q4_0-gguf` vía Ollama	~2,6-5,6 GB
Salida Estructurada	Gramática GBNF de llama.cpp (JSON schema → grammar)	incluido
Validación de Schema	Pydantic (CPU, post-generación)	0 GB
Motor de Reglas	Python (CPU)	0 GB

18.7.4 Modelo de Deploy Recomendado

Carga secuencial (vía Ollama, que gestiona automáticamente):

1. Cargar PaddleOCR-VL-1.5, procesar todas las páginas, extraer texto, **descargar**
2. Cargar Gemma 3 4B GGUF, procesar texto OCR con gramática JSON schema, **descargar**
3. Ejecutar validación basada en reglas en CPU

Evita contención de VRAM y da a cada modelo los 16 GB completos.

18.8 D.8 Resumen de Riesgos

Riesgo	Severidad	Mitigación

Riesgo	Severidad	Mitigación
Incompatibilidad bfloat16 en la T4	ALTO	Usar GGUF vía llama.cpp (confirmado funcionando)
Bugs vLLM + Gemma 3 salida estructurada	ALTO	Usar gramática llama.cpp en vez de vLLM
VRAM de PaddleOCR-VL-1.5 mayor que estimado	MEDIO	Carga secuencial; benchmarkear uso real
Calidad de extracción de Gemma 3 4B	MEDIO	Probar con documentos reales DGSC; fallback a 12B
Limitaciones de subconjunto JSON schema de llama.cpp	BAJO	Probar todos los esquemas Pydantic contra conversor GBNF

18.6 D.6 Fuentes

- Gemma 3 Model Overview - Google AI
- Welcome Gemma 3 - Hugging Face Blog
- Gemma 3 QAT Models - Google Developers Blog
- Gemma 3 Technical Report - arXiv
- LLMStructBench - arXiv
- Gemma 3 4B on Tesla T4 - No Output - Hugging Face
- vLLM Issues #15766, #21148, #20341
- LangExtract - GitHub
- Outlines - GitHub
- llama.cpp Grammar and Structured Output

Fuente: *_bmad/clad/technical-reports/gemma3-evaluation.md* Evaluación realizada en febrero de 2026

---

16 Anexo E: Glosario

Término (ES)	Término (PT)	Definición

Término (ES)	Término (PT)	Definición
Atestado	Atestado	Documento comprobatorio presentado por el candidato
Cédula	Cédula	Número de identificación nacional
Clase	Classe	Categoría de clasificación del puesto
Colegio Profesional	Colégio Profissional	Asociación profesional obligatoria
CONESUP	CONESUP	Consejo Nacional de Enseñanza Superior — autoriza universidades privadas
CONARE	CONARE	Consejo Nacional de Rectores — reconoce títulos extranjeros
Declaración Jurada	Declaração Juramentada	Declaración notariada (experiencia freelance/exterior)
DGSC	DGSC	Dirección General de Servicio Civil
Especialidad	Especialidade	Subcategoría dentro de una clase de puesto
FreeBalance CSM	FreeBalance CSM	Civil Service Management — módulo del Hacienda Digital
GGUF	GGUF	GPT-Generated Unified Format — formato de modelos cuantizados para llama.cpp
HITL	HITL	Human-in-the-Loop — revisión humana obligatoria
Licenciatura	Licenciatura	Grado universitario de 5 años (superior al Bachillerato)
LMEP	LMEP	Ley Marco de Empleo Público N°10159 (2023)
Manual de Clases	Manual de Classes	Documento oficial que

Término (ES)	Término (PT)	Definición
		define requisitos por clase
MEP	MEP	Ministerio de Educación Pública — valida diplomas de educación pública
MIDEPLAN	MIDEPLAN	Ministerio de Planificación Nacional y Política Económica
FastAPI	FastAPI	Framework Python async para APIs REST
Celery	Celery	Sistema de colas de tareas distribuidas para Python
HTMX	HTMX	Biblioteca JavaScript para interactividad server-rendered
OCR	OCR	Optical Character Recognition — reconocimiento óptico de caracteres
Oferta de Servicios	Oferta de Serviços	Plataforma digital de registro de candidatos
Ollama	Ollama	Runtime para modelos de lenguaje locales
Póliza de Fidelidad	Apólice de Fidelidade	Requisito de seguro para algunos puestos
PRD	PRD	Product Requirements Document
Serie	Série	Grupo de clases de puestos relacionadas (ej: Serie Técnica)
SICOBATEC	SICOBATEC	Sistema de códigos del MEP para diplomas de educación media
TSE	TSE	Tribunal Supremo de Elecciones — verificación de identidad
VLM	VLM	Vision-Language Model — modelo que combina visión computacional y lenguaje natural para

Término (ES)	Término (PT)	Definición
		comprensión documental
vLLM	vLLM	Framework de serving de modelos de lenguaje con alto rendimiento
GBNF	GBNF	GGML BNF — extensión de BNF usada por llama.cpp para gramáticas de salida estructurada
SSE	SSE	Server-Sent Events — protocolo de notificaciones en tiempo real del servidor hacia el cliente
ARSP	ARSP	Área de Gestión de Empleo de la DGSC
RSC	RSC	Régimen de Servicio Civil — régimen que abarca 46 instituciones públicas costarricenses
OGEREH	OGEREH	Oficina de Gestión Institucional de Recursos Humanos — oficina de RH en cada institución del RSC
CCSS	CCSS	Caja Costarricense de Seguro Social — seguridad social, verificación de contribuciones

20 Anexo F: Product Requirements Document (PRD)

Autor: Hugo **Fecha:** 8 de marzo de 2026 **Método:** BMAD (Business-Minded AI Development)

20.1 Resumen Ejecutivo

Sistema de validación asistida por IA para la Dirección General de Servicio Civil (DGSC) de Costa Rica. Automatiza el análisis de documentos de candidatos a puestos públicos — actualmente un proceso 100% manual que consume ~40 minutos por oferente.

Diferenciador: Pipeline completa de IA (OCR + extracción estructurada + motor de reglas config-driven) operando on-premise sin GPU, con HITL asíncrono interactivo y audit trail completo para compliance con Contraloría/OCDE. Primer sistema documentado por el CLAD como modelo replicable para servicio civil.

Usuarios objetivo: Equipo de analistas de la DGSC (~10 personas), coordinadores de proceso, equipo TI (sysadmins) y auditores externos (Contraloría) vía informes exportados.

Stack técnico: PaddleOCR-VL-1.5 (OCR) → Gemma 3 4B GGUF vía llama.cpp (extracción estructurada) → Motor de reglas (config-matrix.json) → API REST/FastAPI → Frontend glassbox interactivo.

Infraestructura: VM HyperV sobre cluster HPE DL360 Gen11 (8 VCPUs, 16GB RAM, Ubuntu 22.04, sin GPU). 100% on-premise por requisito de soberanía de datos.

20.2 Criterios de Éxito

20.2.1 Éxito del Usuario

- **Tiempo por oferente:** de ~40min → <=15min (timestamp inicio/fin en el sistema)
- **Context-switches:** de 8+ Alt-Tabs → <=2 (observación)
- **Momento "aha!":** Analista ve brecha de experiencia detectada correctamente que tomaría 20min de cálculo manual
- **HITL fluido:** Cuando un documento pausa por baja confianza, la analista resuelve en <=2 clics (ver extracción, ver PDF, aprobar/rechazar) — sin interrumpir el lote
- **Tarjeta "aguardando decisión":** Dashboard muestra claramente cuántos procesos están pausados y por qué — cero sorpresas

20.2.2 Éxito de Negocio

- **3 meses:** +50% oferentes/analista/día; 100% decisiones con audit trail (elimina Excel paralelo)
- **6 meses:** Equipo TI opera sistema sin consultor
- **12 meses:** Nueva clase de puesto = editar JSON (<1 hora)
- **Adopción:** 100% de los analistas usándolo como primer paso en 3 meses

20.2.3 Éxito Técnico

- **Precisión OCR:** >=95% campos correctamente extraídos
- **Semáforo verde correcto:** >=90% (falsos positivos <10%)
- **Semáforo rojo correcto:** >=95% (falsos negativos <5%)
- **Throughput:** >=10 oferentes/hora en batch CPU-only (PDFs de 30-50 páginas)
- **AI Accountability:** 100% de las decisiones rastreables (modelo usado, regla aplicada, página fuente, resultado) — requisito Contraloría/OCDE

20.2.4 Resultados Medibles

Métrica	Baseline	Meta Fase 1	Cómo medir
Tiempo/oferente	~40 min	<=15 min	Timestamps sistema
Precisión extracción	N/A	>=95%	Ground truth manual
Semáforo correcto	N/A	>=90%	Muestreo aleatorio
Auditabilidad	0% (Excel)	100% (sistema)	Audit trail automático
Procesos pausados visibles	N/A	100%	Dashboard HITL

20.3 Alcance del Producto

El desarrollo sigue tres fases progresivas:

- **Fase 1: Implantación Asistida** — Pipeline completa con HITL-by-default, ingesta dual (email + upload), glassbox interactivo, audit trail, exportación para sistema de vacantes
- **Fase 2: Automatización Avanzada** — Integraciones con portales externos, firma digital, formulario preestructurado, pipeline de config-matrix, aceleración de infraestructura (GPU/VCPUs)
- **Fase 3: Integración Estratégica** — FreeBalance CSM / Hacienda Digital, cloud híbrido, replicabilidad CLAD, código fuente público

20.4 Jornadas de Usuario

20.4.1 Jornada 1: Analista — Camino Feliz (Validación de Lote)

Escena inicial: La analista abre el correo electrónico y encuentra 35 PDFs de oferentes para la clase Técnico de Servicio Civil 3. Cada PDF tiene 20-80 páginas con documentos heterogéneos. En el proceso actual, esto significa una semana entera de trabajo manual.

Acción ascendente: Accede al sistema vía navegador → selecciona la clase "Técnico 3" → hace upload del lote de 35 PDFs → el pipeline comienza a procesar secuencialmente. En el glassbox, ve cada oferente pasar por OCR → Extracción → Validación, gate por gate.

Clímax: Abre el dashboard y ve: 22 oferentes totalmente verdes, 8 amarillos (baja confianza en algún gate), 5 rojos. Hace clic en un amarillo — ve que el gate "Experiencia" pausó porque el sistema extrajo "4 años" con confianza de 72%. Hace clic en la referencia → ve la página exacta del PDF lado a lado con la extracción → confirma que son 4 años → aprueba con un clic.

Resolución: En 2 horas, completó lo que antes tomaba 5 días. Exporta los resultados en el formato de la planilla estándar que retroalimenta el sistema de oferta de vacantes — sin transcripción manual. El Excel paralelo no fue necesario.

20.4.2 Jornada 2: Analista — Caso Extremo (Documento Problemático)

Escena inicial: Oferente para Profesional Jefe 3 envía un PDF de 180 páginas con documentos fuera de orden, declaraciones juradas manuscritas y un título de universidad extranjera.

Acción ascendente: El OCR extrae texto de las páginas manuscritas con confianza baja (~45%). La extracción estructurada no logra mapear campos de la declaración jurada. Tres gates pausan: Académico (título extranjero → verificación CONARE pendiente), Experiencia (declaración jurada ilegible), Supervisión (documento no localizado).

Clímax: El sistema **no se bloquea** — marca los 3 gates como amarillo, continúa procesando los próximos oferentes del lote, y agrega este caso a la tarjeta "aguardando decisión". La analista ve: "3 gates pendientes — motivo: baja confianza OCR (p.47), campo no mapeado (p.112), documento no localizado (supervisión)."

Resolución: La analista resuelve manualmente solo los 3 gates problemáticos, usando la referencia a la página exacta. Los demás gates fueron validados automáticamente. Incluso en el peor caso, el sistema ahorró ~60% del trabajo.

20.4.3 Jornada 3: Coordinador — Dashboard de Supervisión

Escena inicial: El coordinador necesita dar seguimiento al progreso semanal de 3 procesos de reclutamiento simultáneos.

Acción ascendente: Accede al dashboard agregado → ve el estado de todos los lotes: cantidad de oferentes por clase, distribución de semáforos (verde/amarillo/rojo), tarjeta "aguardando decisión" con total de procesos pausados.

Clímax: Identifica que el lote de Profesional Jefe 3 tiene 40% de amarillos — visualiza la carga real de trabajo pendiente. El lote de Técnico 3 está 95% verde — dictámenes pueden ser emitidos inmediatamente.

Resolución: Acompaña la evolución de los procesos con datos reales. Exporta informe de progreso con métricas de uso de la IA para justificación institucional.

20.4.4 Jornada 4: Administrador TI — Deploy y Mantenimiento

Escena inicial: La DGSC publicó actualización en los requisitos de la Serie Técnica — 2 clases cambiaron el nivel académico mínimo.

Acción ascendente: Abre el config-matrix.json → localiza las 2 clases por código → actualiza los campos relevantes → guarda.

Clímax: Ejecuta el script de validación automatizado → JSON válido y consistente. La actualización está en producción.

Resolución: En <30 minutos, sin tocar código, sin consultor externo.

20.4.5 Jornada 5: Auditor (Contraloría) — Verificación de Cumplimiento

Escena inicial: La Contraloría solicita auditoría del proceso de reclutamiento del último trimestre.

Acción ascendente: El coordinador exporta el informe de auditoría (PDF/CSV). Para cada oferente: resultado por gate, justificación granular, modelo de IA utilizado, regla aplicada con referencia al manual de clases, decisiones HITL de la analista.

Clímax: El auditor selecciona casos para verificación. Para cada caso, la cadena completa está documentada: PDF original → texto OCR → campos extraídos → regla aplicada → resultado → decisión humana.

Resolución: Auditoría concluida sin acceso directo al sistema. Transparencia total sobre el papel de la IA.

20.4.6 Resumen de Requisitos por Jornada

Jornada	Capacidades Reveladas
Analista (feliz)	Upload en lote, procesamiento secuencial, informe semáforo, referencia a la página, aprobación en 1 clic, exportación en formato de planilla estándar para sistema de vacantes
Analista (extremo)	Pausa asíncrona, tarjeta "aguardando decisión", OCR baja confianza, fallback HITL, continuidad del lote
Coordinador	Dashboard agregado, distribución de semáforos por lote, métricas de uso de la IA, exportación de informes
TI/Sysadmin	Edición de config-matrix.json, validación automatizada, actualización de modelos, Docker Compose
Auditor	Exportación de audit trail, trazabilidad completa, transparencia de IA, informe por período

20.5 Requisitos Específicos del Dominio

20.5.1 Cumplimiento y Regulatorio

- **Ley Marco de Empleo Público (N°1015G, 2023):** El sistema opera dentro del mandato ampliado de la DGSC. Toda decisión de validación debe ser rastreable a la normativa vigente.
- **Contraloría General de la República:** Audit trail exportable que satisfaga inspecciones sin acceso directo al sistema.
- **Compromisos OCDE:** Transparencia algorítmica — documentar modelos de IA, etapas de uso y grado de autonomía. HITL-by-default cumple este requisito.
- **Manuales de Clases y Especialidades:** config-matrix.json debe reflejar fielmente los 34 manuales, con referencia a la página fuente (`source_page`).

20.5.2 Restricciones Técnicas

- **Soberanía de datos:** Datos personales de oferentes no pueden salir de la infraestructura de la DGSC. Procesamiento 100% on-premise.
- **Infraestructura:** VM HyperV sobre cluster HPE DL360 Gen11 (8 VCPUs, 16GB RAM, Ubuntu 22.04, Ollama). Sin GPU.
- **Equipo TI = sysadmins:** Sistema mantenido con competencias de infraestructura (Docker, config files, reinicio de servicios). Sin expertise en ML.
- **Sin APIs externas disponibles:** Portales de verificación no ofrecen APIs públicas. Integración futura depende de acuerdos institucionales.

20.5.3 Mitigación de Riesgos

Riesgo	Impacto	Mitigación
Falso positivo (aprobar quien no cumple)	Responsabilidad legal para DGSC	HITL-by-default + muestreo aleatorio de verdes
Hallucination en la extracción	Decisiones basadas en datos fabricados	Referencia a la página fuente + puntaje de confianza
Desactualización del config-matrix.json	Validación contra requisitos obsoletos	Módulo de actualización con flujos formales + versionamiento
Dependencia del consultor	Sistema muere tras fin de la consultoría	Docker Compose + documentación operacional + 3 niveles de autonomía TI
Confianza excesiva en el sistema	Analistas aceptan sin cuestionar	Muestreo aleatorio de verificación obligatoria

20.6 Innovación y Patrones Novedosos

20.6.1 Áreas de Innovación

1. **IA On-Premise para GovTech en País en Desarrollo — Sin GPU:** Pipeline completa de IA operando en hardware commodity (8 VCPUs, 16GB RAM, sin GPU dedicada).
2. **Motor de Reglas Config-Driven Extraído de Manuales Regulatorios vía IA:** El config-matrix.json es generado a partir de los manuales oficiales usando pipeline de OCR + extracción estructurada.
3. **Glassbox Interactivo con HITL Asíncrono:** Modelo de pausa sin bloqueo: cuando un gate tiene baja confianza, el ítem pausa mientras el pipeline continúa.
4. **Primera Metodología CLAD Documentada para Servicio Civil:** Sistema concebido como caso replicable para otros países miembros.

20.7 Especificación de Endpoints API

20.7.1 Ingesta y Procesamiento

Endpoint	Método	Descripción
`/v1/batches`	POST	Upload directo de lote de PDFs + selección de clase
`/v1/batches/{id}/status`	GET	Estado del procesamiento con semáforos por oferente
`/v1/batches/{id}/events`	GET (SSE)	Server-Sent Events — notificaciones en tiempo real
`/v1/batches/{id}/export`	GET	Exportación en formato planilla estándar (CSV/XLSX)

20.7.2 Oferentes y HITL

Endpoint	Método	Descripción
`/v1/applicants/{id}/gates`	GET	Detalle gate por gate con justificación + referencia a la página PDF
`/v1/applicants/{id}/gates/{gate}/decision`	POST	Decisión HITL (aprobar/rechazar) con motivo
`/v1/applicants/{id}/document`	GET	Sirve PDF con parámetro `?page=N` para viewer inline

20.7.3 Dashboard y Auditoría

Endpoint	Método	Descripción
`/v1/dashboard/summary`	GET	Dashboard agregado — distribución semáforos, pendencias HITL
`/v1/audit/trail`	GET	Exportación de audit trail por período (CSV/PDF)
`/v1/health`	GET	Estado de los modelos, espacio en disco, cola de procesamiento

20.7.4 Config-Matrix

Endpoint	Método	Descripción
`/v1/config/matrix`	GET	Consultar configuración actual (solo lectura)
`/v1/config/matrix/history`	GET	Historial de versiones
`/v1/config/ingest`	POST	Enviar PDF de manual → inicia pipeline de extracción
`/v1/config/ingest/{id}/preview`	GET	Vista previa del JSON candidato para revisión humana
`/v1/config/ingest/{id}/approve`	POST	Admin aprueba → commit versionado

20.8 Alcance del Proyecto y Desarrollo por Fases

20.8.1 Fase 1: Implantación Asistida

Filosofía: Entregar valor inmediato a las analistas con IA como herramienta de apoyo — HITL-by-default, toda decisión supervisada.

Capacidades Core: - Pipeline OCR → Extracción Estructurada → Motor de Reglas (CPU-only, secuencial) - Ingesta dual: email forwarding + upload directo - Informe semáforo gate por gate con justificación granular + referencia a la página PDF - Frontend glassbox interactivo con HITL asíncrono - Config-matrix.json (117 clases, 34 series) con módulo de actualización formal - API REST/JSON versionada (`/v1/`) - Deploy contenerizado (Docker Compose + health check + alerta) - Audit trail completo (AI accountability — Contraloría/OCDE) - Exportación en formato planilla estándar para sistema de vacantes

20.8.2 Fase 2: Automatización Avanzada

Fase 2a — Automatización Funcional: - Prevalidación contra listas públicas (CONESUP, INA) - Integración con portales externos (CONESUP, CCSS, MEP) - Validación de firma digital (Agente Gaudy) - Formulario preestructurado (Microsoft Forms) - Reprocesamiento de oferentes sin re-upload

Fase 2b — Aceleración de Infraestructura: - GPU on-premise o asignación adicional de VCPUs

20.8.3 Fase 3: Integración Estratégica

- Integración FreeBalance CSM / Hacienda Digital (módulo de Gestión del Talento Humano)
- Modelo híbrido cloud (PII masking para GPU en nube)
- Replicabilidad CLAD para otros países miembros
- Código fuente en repositorio público

20.8.4 Criterios de Graduación entre Fases

Transición	Criterios	Plazo Estimado
Fase 1 → Fase 2	>=95% precisión OCR en producción, 3-4 meses de uso, equipo TI autónomo	3-4 meses tras implantación
Fase 2 → Fase 3	Docs. técnicos FreeBalance CSM disponibles + acuerdo institucional	Condicionada a factores externos

20.6 Requisitos Funcionales — Fase 1 (FR1-FR56)

20.9.1 Ingesta de Documentos (FR1-FR8)

- FR1: Analista puede reenviar correo electrónico con PDFs a buzón dedicado → sistema crea batch automáticamente
- FR2: Analista puede hacer upload directo de lote de PDFs seleccionando la clase
- FR3: Sistema valida compatibilidad entre clase y contenido del PDF
- FR4: Analista puede buscar clase por nombre, código o serie (autocompletar)
- FR5: Sistema notifica sobre correos no procesables
- FR6: Sistema extrae adjuntos PDF y los asocia a un batch
- FR7: Sistema detecta PDFs duplicados (por hash) y alerta al analista
- FR8: Sistema identifica fronteras entre oferentes en PDFs multi-candidato

20.9.2 Preprocesamiento (FR9-FR10)

- FR9: Sistema extrae texto digital nativo sin OCR (bypass digital)
- FR10: Sistema detecta y descarta páginas en blanco o irrelevantes

20.9.3 Procesamiento y Validación (FR11-FR26)

- FR11: Pipeline OCR sobre páginas escaneadas
- FR12: Clasificación de tipos de documentos presentes en el PDF
- FR13: Segmentación de PDF multi-documento
- FR14: Documentos no reconocidos → clasificación manual (HITL)
- FR15: Documentos que exceden límites → notificación a la analista
- FR16: Extracción de campos estructurados (datos académicos, experiencia, supervisión)
- FR17: Validación contra reglas del config-matrix.json, gate por gate
- FR18: Evaluación de todos los caminos académicos alternativos (combos)
- FR19: Resultado consolidado (apto/no apto) con justificación gate por gate
- FR20: Error cuando clase no existe en el config-matrix
- FR21: Clasificación semáforo (verde/amarillo/rojo) por gate
- FR22: Puntaje de confianza y referencia a la página fuente
- FR23: Puntaje de confianza por campo extraído
- FR24: Detalles del cálculo de validación (ej: años computados vs. exigidos)
- FR25: Procesamiento secuencial sin bloquear por gates pausados
- FR26: Tiempo estimado de conclusión del batch

20.9.4 HITL — Decisión Humana (FR27-FR33)

- FR27: Visualización de gates pausados con justificación y referencia a la página
- FR28: Página exacta del PDF lado a lado con la extracción estructurada
- FR29: Aprobar o rechazar gate pausado con registro de motivo
- FR30: Anotaciones/comentarios en oferente o gate específico
- FR31: Reversión de decisión HITL antes de la exportación
- FR32: Bloqueo contra decisiones HITL simultáneas en el mismo gate
- FR33: Continuidad del lote mientras gates aguardan decisión

20.9.5 Dashboard y Monitoreo (FR34-FR40)

- FR34: Lista consolidada de todos los gates pendientes (cross-batch)
- FR35: Estado de todos los oferentes de un batch (distribución de semáforos)
- FR36: Progreso de revisión de la analista por batch
- FR37: Dashboard agregado de todos los batches (coordinador)
- FR38: Métricas de eficiencia (tiempo, tasa de automatización)

- FR39: Tarjeta "procesos aguardando decisión" con conteo y motivos
- FR40: Notificaciones en tiempo real vía SSE

20.9.6 Exportación e Informes (FR41-FR44)

- FR41: Exportación en formato planilla estándar para sistema de vacantes
- FR42: Filtros por estado y por gate
- FR43: Informe de progreso con métricas de IA (coordinador)
- FR44: Audit trail exportable (CSV/PDF) por período

20.9.7 Audit Trail (FR45-FR49)

- FR45: Para cada decisión: modelo de IA, regla aplicada, página fuente, resultado
- FR46: Distinción explícita: automático vs. decisión HITL
- FR47: Registro de decisiones HITL (gate, acción, motivo, timestamp)
- FR48: Cadena completa de trazabilidad exportable (auditor externo)
- FR49: Diffs de config-matrix cuando reglas son alteradas

20.9.8 Gestión y Operación (FR50-FR59)

- FR50-FR52: Config-matrix consulta, historial, alerta de desactualización
- FR53-FR56: Health check, respaldo/restauración, alertas de recursos, historial de errores
- FR57-FR59: Gestión de usuarios/roles, RBAC, login/logout

20.10 Requisitos Funcionales — Fase 2 (FR60-FR74)

- FR60-FR62: Prevalidación contra listas públicas, portales externos, firma digital
- FR63: Formulario preestructurado
- FR64-FR65: Reprocesamiento, webhooks
- FR66-FR68: Sustitución de PDF, evolución temporal, comparación entre batches
- FR69-FR72: Gestión avanzada de config-matrix (ingesta de PDF, vista previa, aprobación)
- FR73: Preprocesamiento de imagen (deskew, binarización)
- FR74: Aceleración con GPU

20.11 Requisitos Funcionales — Fase 3 (FR75-FR78)

- FR75: Sincronización con FreeBalance CSM / Hacienda Digital
- FR76: Cloud híbrido con PII masking
- FR77: Log de correcciones HITL para métricas de drift
- FR78: Código fuente en repositorio público

20.12 Requisitos No Funcionales (NFR1-NFR22)

20.12.1 Rendimiento

- NFR1: >=10 oferentes/hora en batch CPU-only (8 VCPUs, 16GB RAM)

20.12.2 Seguridad y Privacidad

- NFR2: Procesamiento 100% on-premise (soberanía de datos)
- NFR3: HTTPS con TLS
- NFR4: Hash criptográficamente seguro para contraseñas
- NFR5: Sesiones con expiración configurable
- NFR6: Logs de acceso para autenticación
- NFR7: Sanitización de PDFs (remueve JS, macros)
- NFR8: Control de acceso validado en el backend
- NFR9: Lista blanca de remitentes de correo electrónico

20.12.3 Disponibilidad

- NFR10: Recuperación automática post-reinicio
- NFR11: Falla por oferente no interrumpe batch

20.12.4 Mantenibilidad

- NFR12: Procedimientos operacionales ejecutables sin conocimiento ML
- NFR13: Actualización de modelo = sustitución de archivo + reinicio
- NFR14: Runbook con procedimientos paso a paso
- NFR15: Contenedores con dependencias fijadas

20.12.5 Observabilidad

- NFR16: 100% decisiones rastreables
- NFR17: Audit trail retenido >=5 años
- NFR18: Audit trail protegido contra alteración
- NFR19: Validación de integridad del config-matrix al iniciar

20.12.6 Operación

- NFR20: Política de retención con alertas de capacidad

20.12.7 Compatibilidad

- NFR21: Navegadores modernos (Chrome, Edge, Firefox)
- NFR22: API documentada vía OpenAPI/Swagger

Documento completo: *_bmad-output/planning-artifacts/prd.md* (38 KB)

21 Anexo G: Architecture Decision Document

Autor: Hugo **Fecha:** 9 de marzo de 2026 **Método:** BMAD (Business-Minded AI Development) **Estado:** LISTO PARA IMPLEMENTACIÓN (8 etapas concluidas)

21.1 Análisis de Contexto del Proyecto

21.1.1 Visión General de Requisitos

Requisitos Funcionales: 78 requisitos organizados en 3 fases progresivas:

- **Fase 1 — Implantación Asistida (FR1-FR5G):** Pipeline completa de validación documental con HITL-by-default. Incluye ingesta dual (email IMAP + upload HTTP), pipeline OCR → clasificación → extracción estructurada → motor de reglas, informe semáforo gate por gate con justificación granular, HITL asíncrono no bloqueante con SSE, dashboard de supervisión, exportación para sistema de vacantes, audit trail completo y RBAC con 3 roles.
- **Fase 2 — Automatización Avanzada (FR60-FR74):** Integraciones con portales externos, firma digital, formulario preestructurado, pipeline de actualización de config-matrix, aceleración de infraestructura.
- **Fase 3 — Integración Estratégica (FR75-FR78):** FreeBalance CSM, cloud híbrido, replicabilidad CLAD, código fuente público.

Requisitos No Funcionales: 22 NFRs que direccionan decisiones arquitectónicas:

- **Rendimiento (NFR1):** >=10 oferentes/hora en batch CPU-only con PDFs de 30-50 páginas
- **Seguridad (NFR2-G):** Soberanía de datos 100% on-premise, HTTPS/TLS, sanitización de PDFs, RBAC
- **Disponibilidad (NFR10-11):** Recuperación automática post-reinicio, falla aislada por oferente
- **Mantenibilidad (NFR12-15):** Operación por sysadmins sin expertise ML
- **Observabilidad (NFR16-1G):** 100% decisiones rastreables, retención mínima 5 años, audit trail inmutable

21.1.2 Restricciones Técnicas y Dependencias

Restricción	Impacto Arquitectónico
CPU-only (8 VCPUs, 16GB RAM, sin GPU)	Sequential model loading obligatorio; throughput es cuello de botella principal
PaddleOCR-VL-1.5 (~4-	No caben simultáneamente — pipeline debe

Restricción	Impacto Arquitectónico
6GB) + Gemma 3 4B Q4 (~2.5-4GB)	cargar/descargar modelos secuencialmente
Incompatibilidad T4 bfloat16	vLLM + Gemma 3 broken — usar llama.cpp GGUF con grammar GBNF
Equipo TI = sysadmins	Docker Compose, config files, runbook — sin expertise ML/Python
Sin APIs externas disponibles	Fase 1 100% standalone
Soberanía de datos (PII de candidatos)	Cloud descartada; procesamiento 100% on-premise
n8n descartado (3 CVEs CVSS >=9.9)	FastAPI + Celery + Redis — superficie de ataque menor para sistema con PII
Cumplimiento: Ley Marco N°10159, Contraloría, OCDE, CLAD	Audit trail como requisito de primera clase

21.1.3 Preocupaciones Transversales

1. **Observabilidad/Audit trail** — Permea TODOS los componentes: cada etapa debe registrar modelo usado, regla aplicada, página fuente, resultado, timestamp y decisión humana.
2. **Sequential model loading** — Un worker procesa todas las etapas de un candidato secuencialmente.
3. **Clasificación de documentos** — Heurística Python clasifica ~70% de los documentos sin costo de inferencia. Baja confianza → LLM. Cero coincidencia → HITL.
4. **Gestión de memoria** — cgroups para protección contra OOM, timeout por página.
5. **Seguridad de PDFs** — Sanitización antes del procesamiento, detección de duplicados por hash.
6. **Versionamiento de config-matrix** — Diffs entre versiones, reproducibilidad, rollback.

21.2 Stack Técnico Consolidado

Capa	Tecnología	Versión*	Justificación
Lenguaje	Python 3.12+	3.12	Ecosistema ML/IA, FastAPI nativo async
API Framework	FastAPI	latest	SSE nativo, async I/O, OpenAPI auto-generado
Task Queue	Celery + Redis	latest	Procesamiento asíncrono

Capa	Tecnología	Versión*	Justificación
			de candidatos en background
Database	PostgreSQL	16+	Audit trail, JSONB para metadata, Alembic para migraciones
Migraciones	Alembic	latest	Schema migrations versionadas
OCR	PaddleOCR-VL-1.5 GGUF	q4_0	0.9B params, vía Ollama/llama.cpp
LLM	Gemma 3 4B QAT GGUF	q4_0	Vía Ollama/llama.cpp, grammar GBNF para JSON
Structured Output	llama.cpp grammar (GBNF)	—	100% well-formed JSON
Runtime Unificado	Ollama (llama.cpp)	latest	Sirve OCR y LLM vía misma API
Frontend	HTMX + Jinja2	latest	Server-rendered, zero build step, SSE nativo
Linting	ruff	latest	Sustituye flake8+black+isort
Testing	pytest	latest	Fixtures de ground truth, mocks para OCR/LLM
Deploy	Docker Compose	latest	Standalone, health check + alerta
Infra objetivo	VM HyperV (8 VCPUs, 16GB RAM, Ubuntu 22.04)	—	CPU-only, on-premise DGSC

21.2.1 Decisión: Frontend HTMX + Jinja2

- **Zero build step** — sin Node.js, npm, webpack
- **SSE nativo** — HTMX soporta `hx-sse` para dashboard en tiempo real
- **Un solo lenguaje** — toda la lógica en Python/FastAPI
- **PDF viewer inline** — `<iframe>` con endpoint FastAPI
- **RBAC trivial** — control de visibilidad directamente en el template
- **Mantenibilidad máxima** — alteraciones de UI son alteraciones de HTML

21.3 Decisiones Arquitectónicas Core

21.3.1 Arquitectura de Datos

Decisión	Elección			Racional
ORM	SQLAlchemy	2.0	+	Async nativo para FastAPI;

Decisión	Elección	Racional
	`asyncpg`	Alembic integrado
Audit trail	Tabla `audit_events` append-only, permisos `INSERT`-only	Inmutable por diseño (NFR18)
Almacenamiento PDF	Filesystem + referencia en DB	Respaldo con rsync
Versionamiento config-matrix	JSON completo + hash SHA-256 + timestamp	Cada decisión registra cuál versión de reglas fue usada
Renderizado de página PDF	Endpoint convierte página a imagen vía `pikepdf`	Menor latencia en el viewer

21.3.2 Autenticación y Seguridad

Decisión	Elección	Racional
Sesión	Sesiones server-side en Redis	Single-server, sesiones con expiración
Hashing	bcrypt vía `passlib`	NFR4
Sanitización PDF	`pikepdf` (remueve JS/macros + bypass digital)	Un paquete resuelve sanitización y bypass OCR
Autenticación email	Lista blanca de remitentes en config	NFR9

21.3.3 API y Comunicación

Decisión	Elección	Racional
SSE	`sse-starlette`	Nativo FastAPI, HTMX consume directamente
Logging	`structlog` (JSON structured)	Consultable, futuro Prometheus/Grafana
Manejo de errores	Middleware global + errores estructurados	Contexto por gate, página, confianza

21.3.4 Infraestructura y Deploy

Decisión	Elección	Racional
Health check	`/v1/health` — PostgreSQL, Redis, Ollama	

	a, disco	Cron + alerta email
Respaldo	Script bash: pg_dump + rsync + cp config	Sysadmin-friendly

Decisión	Elección	Racional
Monitoreo Fase 1	structlog + health check + alertas email	Prometheus/Grafana diferido
Orquestación de contenedores	Docker Compose (sin Kubernetes)	Single-server

21.3.5 Estrategia de Pruebas

Decisión	Elección	Racional
Pruebas unitarias	pytest + fixtures JSON (mock OCR/LLM)	Rápido, determinístico
Pruebas de integración	pytest + contenedores Docker reales	Pipeline de extremo a extremo
Pruebas de UI	pytest + httpx (fragmentos HTMX server-side)	Sin Playwright en la Fase 1

21.4 Patrones de Implementación

21.4.1 Convenciones de Nomenclatura

Contexto	Patrón	Ejemplo
DB tablas	`snake_case` plural	`batches`, `audit_events`
API endpoints	`/v1/` + plural	`/v1/batches`, `/v1/applicants/{id}/gates`
Campos JSON	`snake_case`	`{"batch_id": "...", "gate_result": "green"}`
Funciones Python	`snake_case`	`process_applicant()`, `evaluate_gate()`
Clases Python	`PascalCase`	`ApplicantGate`, `OCRResult`
Constantes	`UPPER_SNAKE`	`MAX_PAGE_TIMEOUT`

21.4.2 Formato de Evento de Auditoría

```
{
  "event_type": "gate_evaluated|hitl_decision|ocr_completed|extraction_co
mpleted",
  "applicant_id": "uuid",
  "batch_id": "uuid",
  "gate": "academic|experience|supervision|college|legal",
  "result": "green|yellow|red",
  "confidence": 0.72,
  "model": "paddleocr-vl-1.5|gemma-3-4b-q4",
  "config_version_hash": "sha256:abc123...",
  "source_page": 47,
```

```
  "analyst_id": "uuid|null",
  "analyst_action": "approve|reject|null",
  "timestamp": "2026-03-09T14:30:00Z"
}
```

21.5 Estructura del Proyecto

```
clad-dgsc-validator/
├── docker-compose.yml              # PostgreSQL, Redis, Ollama, app
├── Dockerfile                      # Multi-stage: base + models
├── .env.example
├── pyproject.toml
├── alembic.ini
├── alembic/
│   └── versions/
│       ├── 001_initial_schema.py   # batches, applicants, documents, gat
es
│       └── 002_audit_events.py     # audit_events (INSERT-only)
├── config/
│   ├── config-matrix.json          # Reglas de validación (versionado)
│   ├── classification_rules.yaml   # Heurísticas regex/keywords
│   └── schemas/                    # Esquemas Pydantic por tipo de docum
ento
├── src/
│   ├── main.py                     # FastAPI app factory + lifespan
│   ├── config.py                   # Settings vía pydantic-settings
│   ├── api/
│   │   ├── deps.py                 # Dependencias inyectables
│   │   ├── middleware.py           # Error handler global, CORS
│   │   ├── v1/
│   │   │   ├── batches.py          # FR1-FR6: Upload, IMAP, batch CRUD
│   │   │   ├── applicants.py       # FR15-FR20: Resultados, semáforo
│   │   │   ├── gates.py            # FR21-FR30: Gates, HITL actions
│   │   │   ├── documents.py        # FR7-FR14: PDF viewer, clasificación
│   │   │   ├── admin.py            # FR50-FR59: Config, usuarios
│   │   │   ├── export.py           # FR45-FR49: CSV/PDF
│   │   │   ├── health.py           # NFR10: Health check
│   │   │   └── sse.py              # FR40-FR44: SSE endpoints
│   │   └── auth.py
│   ├── pipeline/
│   │   ├── orchestrator.py         # Celery task: coordina stages
│   │   ├── stage_1_ingest.py       # Email/upload → PDF → dedup
│   │   ├── stage_2_ocr.py          # PaddleOCR-VL-1.5 vía Ollama
│   │   ├── stage_3_classify.py     # Heurística → LLM → HITL
│   │   ├── stage_4_extract.py      # Gemma 3 4B + GBNF → schemas
│   │   ├── stage_5_rules.py        # Motor de reglas config-matrix
│   │   └── model_manager.py        # Sequential load/unload (16GB)
│   ├── models/
│   │   ├── base.py                 # DeclarativeBase + mixins
│   │   ├── batch.py, applicant.py, gate.py
```

```
│   │   ├── audit.py                # AuditEvent (INSERT-only)
│   │   ├── config_version.py       # JSON + SHA-256 hash
│   │   └── user.py                 # User, Role
│   ├── services/
│   │   ├── hitl.py, email_ingestion.py, pdf_service.py
│   │   ├── export_service.py, audit_service.py
│   │   └── config_service.py
│   └── templates/
│       ├── base.html, login.html
│       ├── dashboard/              # Visión general + progreso SSE
│       ├── applicant/              # Gates + PDF viewer
│       ├── admin/                  # Usuarios, config, audit log
│       └── components/             # Semáforo, gate card, SSE badge
├── tests/
│   ├── unit/, integration/, fixtures/
│   └── conftest.py
├── scripts/
│   ├── backup.sh, health_check.sh
│   ├── seed_users.py, load_config_matrix.py
└── docs/
    └── runbook.md
```

21.6 Flujo de Datos

```
flowchart TD
    A["Email IMAP / Upload HTTP"] --> B["Stage 1: Ingest<br/>Dedup hash,
sanitize, split PDFs<br/>CPU, <1s/doc"]
    B --> C["Stage 2: OCR<br/>PaddleOCR-VL-1.5 GGUF vía Ollama<br/>~4GB,
~7s/pág CPU"]
    C --> D{"Stage 3: Classify<br/>Heurística + LLM fallback"}
    D -->|"unknown docs"| H1["HITL Queue"]
    D --> E["Stage 4: Extract<br/>Gemma 3 4B + GBNF grammar<br/>~4GB, ~2-
5s/doc"]
    E --> F{"Stage 5: Rules<br/>config-matrix.json"}
    F -->|"yellow gates"| H2["HITL Queue"]
    F --> G["PostgreSQL<br/>Results + Audit"]
    G --> SSE["SSE Events<br/>Real-time"]
    SSE --> DASH["Dashboard<br/>HTMX"]

    style A fill:#e3f2fd
    style C fill:#e8eaf6
    style E fill:#f3e5f5
    style F fill:#e8f5e9
    style DASH fill:#fff9c4
```

Nota: Los modelos cargan SECUENCIALMENTE vía Ollama. Los Stages 3 y 4 comparten Gemma 3 4B — sin reload entre ellos. Cada candidato es procesado por 1 Celery worker de extremo a extremo.

21.7 Límites Arquitectónicos

Frontera	Protocolo	Patrón
API ↔ Pipeline	Celery task dispatch (async)	`orchestrator.delay(batch_id, applicant_id)`
Pipeline ↔ Models	`model_manager.py` load/unload	Un modelo por vez, timeout por página
API ↔ Frontend	HTTP + SSE	Fragmentos HTMX, SSE events tipados
API ↔ Database	Sesiones SQLAlchemy async	Repository pattern vía services
Pipeline ↔ HITL	Gate pausa, SSE notifica, API reanuda	Pipeline continúa otros gates
Todos ↔ Audit	`audit_service.log_event()`	INSERT-only, schema fijo

21.8 Validación de la Arquitectura

21.8.1 Resultados de Pruebas de Estrés (5 Métodos)

- **Pre-mortem:** 2 brechas (checkpoint/resume + cleanup automático) — incorporadas
- **Red Team vs Blue Team:** 1 brecha (TLS/HTTPS) — nginx reverse proxy incorporado
- **Failure Mode Analysis:** 2 brechas (timeout global + Ollama cold start) — incorporadas
- **Self-Consistency:** Documento internamente consistente
- **Chaos Monkey:** Kill de componentes mid-pipeline — checkpoint/resume resuelve

21.8.2 Brechas Identificadas y Resueltas

#	Brecha	Resolución
1	Checkpoint/resume por oferente	Guarda `current_stage` en el DB tras cada stage
2	Cleanup automático de batches	Script cron + política de retención
3	TLS/HTTPS no especificado	nginx reverse proxy con TLS en docker-compose
4	Timeout global por candidato	`MAX_CANDIDATE_TIMEOUT` vía pydantic-settings
5	Ollama cold start	Health check + pre-warm en el arranque

#	Brecha	Resolución
	(~30-60s)	
6	Runtime OCR fragmentado	Unificado: PaddleOCR-VL-1.5 GGUF vía Ollama

21.8.3 Cobertura de Requisitos

- Fase 1 (FR1-FR59): 100% cubiertos
- **NFRs (1-20):** 100% cubiertos
- **Fases 2-3 (FR60-FR78):** Documentados como decisiones diferidas con puntos de extensión

21.6 Hoja de Ruta Arquitectónica

Fase	Infraestructura	OCR	LLM	Orquestación
Fase 1	CPU-only (8 VCPUs, 16GB)	PaddleOCR-VL-1.5 GGUF vía Ollama	Gemma 3 4B GGUF vía Ollama	FastAPI + Celery + Redis
Fase 2	+GPU (T4/A10) opcional	PaddleOCR-VL-1.5 vía vLLM	Gemma 3 4B vía vLLM	+integraciones externas
Fase 3	Cloud híbrido	Mismo	Mismo	+FreeBalance CSM, APIs gov

21.10 Evaluación de Preparación Arquitectónica

Estado General: LISTO PARA IMPLEMENTACIÓN

Fortalezas Clave: - Runtime unificado Ollama/llama.cpp — una API para OCR y LLM - Stack Python-only (sin Node.js/build tooling) - Sequential model loading determinístico para 16GB - HITL async no bloquea pipeline - Audit trail como primera clase en todos los componentes - Operable por sysadmins sin expertise ML - Ruta de evolución clara: GGUF CPU → vLLM GPU

Secuencia de Desarrollo: 1. Scaffold (docker-compose + FastAPI + Celery + PostgreSQL) 2. Models SQLAlchemy + migraciones Alembic 3. Pipeline stages (ingest → OCR → classify → extract → rules) 4. Dashboard HTMX + SSE 5. HITL async + audit trail 6. Pruebas de integración con ground truth

Documento completo: _*bmad-output/planning-artifacts/architecture.md* (35 KB)

CLAD

MINISTÉRIO DA
GESTÃO E DA INOVAÇÃO
EM SERVIÇOS PÚBLICOS

GOVERNO FEDERAL
BRASIL
UNIÃO E RECONSTRUÇÃO