



## **Maestría en Inteligencia Artificial Aplicada**

### **Proyecto Integrador**

#### **Avance 7: Resumen Ejecutivo**

Profesores:

Dra. Grettel Barceló Alonso

Dr. Luis Eduardo Falcón

Dr. Guillermo Mota Medina

Equipo 59

A01795457 Renzo Antonio Zagni Vestrini

A01362405 Roger Alexei Urrutia Parke

A01795501 Héctor Raúl Solorio Meneses

## Contenido

<b>Introducción</b>	<b>3</b>
<b>1. Síntesis del Problema</b>	<b>3</b>
<b>2. Hallazgos del Análisis Exploratorio de Datos (EDA)</b>	<b>4</b>
<b>3. Modelos Generados y Elección del Modelo Final</b>	<b>4</b>
<b>4. Recomendaciones Clave para la Implementación</b>	<b>5</b>
<b>5. Análisis Costo–Beneficio</b>	<b>6</b>
a) Costos incurridos en la fase de desarrollo	6
b) Modelo financiero para la implementación en producción	6
c) Interpretación ejecutiva (enfoque hipotético)	8
<b>6. Riesgos y Desafíos</b>	<b>9</b>
<b>Conclusión</b>	<b>10</b>

## **Introducción**

El presente resumen ejecutivo consolida los resultados, hallazgos y proyecciones del Proyecto Integrador del Equipo 59, desarrollado como parte de la Maestría en Inteligencia Artificial Aplicada. El trabajo propone una solución innovadora que combina inteligencia artificial generativa (Gen AI) y arquitecturas multiagente para la creación automatizada de aplicaciones completas a partir de descripciones en lenguaje natural.

El proyecto surge del reto que enfrentan las organizaciones modernas de reducir la dependencia de equipos técnicos especializados para el desarrollo de aplicaciones internas. La solución diseñada busca que usuarios de negocio puedan generar aplicaciones funcionales con backend, frontend y base de datos; simplemente mediante prompts descriptivos. El sistema, basado en técnicas de Retrieval-Augmented Generation (RAG) y orquestación de agentes, se integrará en la plataforma Digital Process Automation (DPA) de Intelenz, permitiendo un modelo no-code/low-code impulsado por IA.

Este desarrollo tiene como propósito democratizar la creación de software, reducir los tiempos de implementación de semanas a horas y fortalecer la competitividad empresarial mediante automatización inteligente.

El documento presenta la justificación teórica, los hallazgos del análisis exploratorio, la selección del modelo óptimo y las recomendaciones para su implementación en un entorno productivo. Los resultados demuestran la viabilidad técnica del sistema, con una tasa de éxito superior al 90 %, tiempos de inferencia menores a 8 segundos y costos de operación competitivos.

Además, se incluyen un análisis costos y una evaluación de riesgos que respaldan su implementación controlada en la nube, validando su potencial como modelo escalable y sostenible para entornos empresariales.

### **1. Síntesis del Problema**

Las organizaciones dependen de sus departamentos de TI para construir soluciones internas, generando cuellos de botella, altos costos y retrasos. En el caso de *Intelenz*, la plataforma DPA requiere actualmente de desarrolladores para configurar procesos complejos, lo que limita su escalabilidad. Se identificó la necesidad de incorporar IA generativa para que usuarios no técnicos pudieran describir en lenguaje natural sus requerimientos y obtener aplicaciones listas para desplegar.

El problema se enmarca en la tendencia global hacia herramientas *no-code/low-code*, que según Gartner (2023) representarán el 65 % del desarrollo empresarial para 2025. La propuesta busca

resolver esta brecha tecnológica a través de un sistema capaz de traducir intenciones en soluciones digitales completas, reduciendo dependencia técnica y costos operativos.

## 2. Hallazgos del Análisis Exploratorio de Datos (EDA)

Durante la fase exploratoria se diseñó un sistema compuesto por múltiples agentes inteligentes que colaboran entre sí para construir una aplicación funcional.

- **Parser Agent:** interpreta los prompts y genera un esquema estructurado de entidades y relaciones.
- **Backend Agent:** produce automáticamente el código SQL, las APIs y la lógica de negocio.
- **Frontend Agent:** construye los formularios, vistas y componentes de interfaz en React/Next.js.
- **Integrator Agent:** valida consistencia, ensambla módulos y genera una aplicación ejecutable.

La integración con módulos LLM permitió que el sistema consultara bases de conocimiento y esquemas previos, mejorando la precisión semántica.

Los resultados de esta etapa evidenciaron que el pipeline agéntico logró una tasa de éxito del 90% en la generación de artefactos funcionales y tiempos promedio de ejecución menores a 10 seg. por prompt.

De la misma forma, se identificaron y se resolvieron los principales retos técnicos:

- Ambigüedad de los prompts.
- Seguridad ante inyección de código.
- Consistencia entre backend y frontend.

Aun con estos desafíos, la fase exploratoria confirmó la viabilidad técnica y escalabilidad del sistema, sentando las bases para el modelado avanzado y despliegue productivo.

## 3. Modelos Generados y Elección del Modelo Final

Se evaluaron distintos modelos de lenguaje (LLM) incluyendo GPT-4o-mini, GPT-4-turbo GPT-5, GPT-5-mini, Claude-Opus y DeepSeek-Code. Cada modelo fue analizado en términos de costo por inferencia, latencia, coherencia estructural y tasa de éxito de generación.

Modelo	Tokens	Costo	Runtime (sec)	Lineas de Código	$\sigma$ (Runtime)
GPT-5	31,823	\$0.11504	475	1,104	139.99
GPT-5-mini	21,502	\$0.00694	258.31	916	60.19
Claude-opus-4-1	15,872	\$0.00612	200.65	875	31.25
Claude-sonnet-4-5	12,649	\$0.00212	100.22	875	7.39
DeepSeek-coder	5,533	\$0.00245	171.90	430	60.58
GPT-4o-mini	3,931	\$0.00061	41.76	410	14.47
GPT-4-turbo	2,718	\$0.00465	53.54	334	32.16

Tabla 1: Resultados promedio de evaluación de costos y desempeño de modelos LLM

Los resultados mostraron que GPT-5-mini ofreció el mejor equilibrio entre rendimiento, costo y precisión, generando resultados equivalentes a los de modelos más grandes con un 80 % menos de costo operativo y latencias promedio de 475 segundos.

El modelo final fue implementado en una arquitectura modular, donde un agente principal coordina la comunicación entre los módulos de análisis, generación y validación de artefactos.

Este enfoque de orquestación asegura la consistencia entre los componentes generados (base de datos, API y frontend) y facilita la escalabilidad del sistema en entornos de nube.

Paso seguido, se aplicó una entonación fina de prompts, que redujo significativamente las alucinaciones, estandarizó la estructura de salida y mejoró la coherencia general del código.

La decisión final priorizó la reproducibilidad técnica, eficiencia económica y sostenibilidad del modelo, elementos clave para su adopción empresarial.

## 4. Recomendaciones Clave para la Implementación

El análisis de viabilidad (Avance 6) confirmó que el sistema multiagente cumple con los estándares requeridos para una implementación productiva, alcanzando una tasa de éxito del 90 %, latencia  $\leq 8$  s y tasa de error  $< 2$  %.

Recomendaciones principales:

- **Infraestructura:** desplegar el sistema en Google Cloud Platform (GCP) por su compatibilidad con servicios de IA y escalabilidad automática.
- **Pipeline MLOps:** automatizar validaciones, pruebas unitarias y control de versiones de prompts.
- **Seguridad:** implementar guardrails, autenticación robusta, cifrado y auditorías continuas.
- **Gobernanza de IA:** incorporar revisión Human-in-the-Loop en generación de artefactos críticos.

- **Despliegue gradual:** iniciar con pilotos (canary releases) antes del escalado total.

Estas acciones garantizan una transición controlada, segura y escalable hacia la operación productiva.

## 5. Análisis Costo–Beneficio

### a) Costos incurridos en la fase de desarrollo

Durante la etapa de desarrollo del proyecto se implementó una estrategia de optimización de recursos orientada a maximizar el aprendizaje y la experimentación sin incurrir en gastos elevados.

Se aprovechó el uso de infraestructura y herramientas personales de los integrantes del equipo, evitando la necesidad de adquirir hardware especializado o servicios empresariales. En consecuencia, los costos asociados se limitaron principalmente al uso de plataformas en la nube y APIs comerciales bajo modalidad de pago por uso.

Entorno de desarrollo: Se utilizó Google Collaboratory Pro, que permitió la ejecución de notebooks en GPU de alta disponibilidad, facilitando la experimentación con modelos de lenguaje y agentes. El costo total estimado fue de aproximadamente USD 10 mensuales durante el periodo activo de pruebas.

Modelos de lenguaje (LLM): Se realizaron pruebas controladas con las APIs de OpenAI (GPT-5 y GPT-5-mini), Anthropic Claude y DeepSeek, todas bajo esquemas pay-per-token. Los costos fueron estrictamente monitoreados, alcanzando un promedio de USD 0.12 por mil tokens procesados, con un gasto acumulado de alrededor de USD 70 durante todo el proceso experimental.

Almacenamiento y gestión de datos: Se utilizó Google Drive y postgres database evitando costos adicionales de infraestructura.

Herramientas de soporte: Para el control de versiones y colaboración se emplearon plataformas gratuitas como GitHub, sin costos de licencia.

En conjunto, la inversión total estimada no superó los USD 100, lo que demuestra una ejecución altamente eficiente en términos financieros.

Este enfoque permitió al equipo concentrarse en la validez técnica y conceptual del modelo, manteniendo la investigación dentro de un presupuesto académico sostenible.

### b) Modelo financiero para la implementación en producción

Con el fin de estimar la viabilidad económica futura del sistema propuesto, se desarrolló un modelo financiero preliminar basado en supuestos realistas sobre el comportamiento de costos durante el primer año de operación.

El objetivo de este modelo no es proyectar ingresos ni resultados financieros definitivos, sino ofrecer una visión estructurada de los costos operativos esperados si la solución fuera implementada en un entorno productivo.

Los valores que se presentan a continuación se basan en supuestos hipotéticos de adopción y uso del sistema, considerando un escenario de crecimiento progresivo de usuarios y generación de aplicaciones a lo largo de doce meses.

<b>Open Ai</b>	\$0.12
<b>Hosting de Core App por mes</b>	\$570.00
<b>Numero de Aplicaciones generadas por usuario por mes</b>	2
<b>Numero de usuarios Mes 0</b>	100
<b>Crecimiento mensual de Usuarios</b>	50%
<b>Hosting Container por aplicación por mes</b>	\$12.00
<b>Costo Por Usuario</b>	\$24.00

Tabla 2: Costos Unitarios y Estimaciones de Adopción

	<b>Costo Open Ai</b>	<b>N. de Aplicaciones</b>	<b>N. de Usuarios</b>	<b>Hosting Container</b>	<b>Hosting Core App</b>
<b>Mes 1</b>	\$24	200	100	\$2,400	\$570
<b>Mes 2</b>	\$36	300	150	\$3,600	\$570
<b>Mes 3</b>	\$54	450	225	\$5,400	\$570
<b>Mes 4</b>	\$81	675	338	\$8,100	\$570
<b>Mes 5</b>	\$122	1,013	506	\$12,150	\$570
<b>Mes 6</b>	\$182	1,519	759	\$18,225	\$570
<b>Mes 7</b>	\$273	2,278	1,139	\$27,338	\$570
<b>Mes 8</b>	\$410	3,417	1,709	\$41,006	\$570
<b>Mes 9</b>	\$615	5,126	2,563	\$61,509	\$570
<b>Mes 10</b>	\$923	7,689	3,844	\$92,264	\$570
<b>Mes 11</b>	\$1,384	11,533	5,767	\$138,396	\$570
<b>Mes 12</b>	\$2,076	17,300	8,650	\$207,594	\$570
				<b>Est. Costos Primer Año:</b>	<b>\$631,002</b>

Tabla 3: Proyección de Costos de Primeros 12 Meses

### Interpretación general del modelo

El modelo permite visualizar la evolución esperada de los costos operativos en función del crecimiento hipotético de usuarios y del consumo de recursos en la nube.

El principal componente del gasto está asociado al hosting de contenedores (infraestructura elástica por aplicación generada), seguido del uso de APIs de modelos de lenguaje y el costo fijo del servidor central.

Los resultados obtenidos evidencian que la escalabilidad del sistema implica un incremento casi exponencial de costos, lo que subraya la importancia de optimizar el uso de tokens, consolidar contenedores y gestionar eficientemente los recursos de cómputo.

En fases futuras del proyecto, la incorporación de estrategias de autoscaling, almacenamiento compartido y modelos de lenguaje más eficientes podría reducir significativamente los costos unitarios, mejorando la sostenibilidad operativa y la viabilidad de la implementación a gran escala.

### **c) Interpretación ejecutiva (enfoque hipotético)**

El crecimiento proyectado del 50 % mensual se plantea únicamente como un escenario de adopción hipotético, útil para dimensionar el impacto que tendría el aumento de usuarios en los costos de infraestructura y consumo de modelos LLM durante el primer año.

En esta etapa del proyecto no se ha definido todavía un modelo comercial ni un precio de venta por usuario, por lo que no es posible estimar de manera responsable el punto de equilibrio ni el ROI real.

Lo que nos aporta el modelo financiero es una visión clara de la estructura de costos: el gasto anual estaría dominado por el hosting de contenedores y el uso de las APIs de IA, mientras que el costo de la aplicación core se mantiene prácticamente constante.

A partir de estos costos, en fases posteriores, una vez definidos los posibles esquemas de monetización como licenciamiento, suscripciones por usuario, uso por aplicación, etc.; se podrán construir distintos escenarios de ingreso y, entonces sí, calcular el punto de equilibrio y el ROI con mayor precisión.

En términos cualitativos, los beneficios potenciales del sistema son:

- Reducción significativa del esfuerzo y tiempo de desarrollo de aplicaciones internas.
- Mayor productividad de los equipos de negocio y disminución del *time-to-market* de nuevas soluciones.
- Posibilidad de explotar en el futuro diversas modalidades de monetización (licenciamiento a clientes, modelo SaaS, integración como módulo premium de la plataforma DPA).

Adicionalmente, se identifican beneficios intangibles relevantes: mayor autonomía de las áreas de negocio, fortalecimiento de una cultura de innovación continua y posicionamiento competitivo de Intelenz como referente en soluciones de automatización impulsadas por IA generativa.

## 6. Riesgos y Desafíos

Como en toda solución basada en inteligencia artificial, el proyecto presenta una serie de riesgos técnicos, operativos y de cumplimiento que deben ser evaluados cuidadosamente antes de su implementación en un entorno productivo.

La naturaleza generativa del sistema que combina agentes autónomos, procesamiento de lenguaje natural y acceso a modelos externos introduce nuevos desafíos en materia de seguridad, confiabilidad y gobernanza de los datos.

Durante el análisis de riesgos, se estableció un marco de clasificación basado en cuatro dimensiones: datos, ataques, prueba y confianza, y cumplimiento normativo.

Cada categoría fue analizada considerando la probabilidad de ocurrencia y el impacto potencial sobre la operación o la reputación de la organización.

A continuación, se resumen los principales riesgos identificados y las estrategias propuestas para mitigarlos:

Categoría	Riesgo	Estrategia de mitigación
<b>Datos</b>	Prompts ambiguos, inconsistentes o con sesgo que afecten la calidad de los artefactos generados.	Implementar validación automática de entradas, curación del dataset de entrenamiento y filtros semánticos previos al procesamiento.
<b>Ataques</b>	Inyección de código malicioso, uso indebido de prompts o accesos no autorizados a los endpoints del sistema.	Aplicar cifrado en tránsito y en reposo, autenticación de múltiples factores, firewalls de aplicaciones web (WAF) y auditorías de seguridad periódicas.
<b>Prueba y confianza</b>	Generación de código con errores funcionales o estructuras no verificadas que comprometan la integridad de la aplicación.	Incorporar pruebas automatizadas (unitarias y de integración), validaciones sintácticas, revisiones Human-in-the-Loop y versionado de artefactos.
<b>Cumplimiento</b>	Falta de trazabilidad en el proceso de generación, uso indebido de datos sensibles o incumplimiento de normas regulatorias.	Alinear la arquitectura con marcos de cumplimiento como GDPR y HIPAA, mantener registros auditables y asegurar la trazabilidad completa del ciclo de vida del modelo.

Tabla 4: Riesgos y Desafíos

En conjunto, estos riesgos se consideran moderados y manejables siempre que se apliquen mecanismos de control continuo, monitoreo automatizado y gobernanza responsable de la IA.

La implementación de buenas prácticas de MLOps y la revisión periódica del comportamiento de los agentes garantizarán la confiabilidad, transparencia y seguridad de la solución en entornos reales.

## **Conclusión**

El proyecto “Generador de Aplicaciones con IA Generativa y Agentes” representa una solución integral, escalable y sostenible para transformar el desarrollo de software empresarial.

Los resultados técnicos confirman que la combinación de IA Generativa, RAG y orquestación multiagente permite automatizar la generación de aplicaciones completas manteniendo estándares de calidad, seguridad y trazabilidad.

La implementación en GCP, junto con el uso eficiente de GPT-5-mini, asegura una infraestructura rentable y de alto rendimiento.

El análisis financiero y de riesgos demuestra que el modelo es viable y estratégicamente beneficioso para Intelenz, con un retorno rápido de la inversión y un potencial de expansión comercial.

Este proyecto refleja el impacto real que la inteligencia artificial aplicada puede generar en la innovación empresarial, consolidando un modelo replicable que combina eficiencia técnica, valor económico y visión estratégica para la era digital.

En perspectiva, este proyecto establece un modelo replicable de aplicación de la inteligencia artificial generativa para la automatización de procesos complejos, fortaleciendo la eficiencia operativa y la capacidad de innovación de las organizaciones.