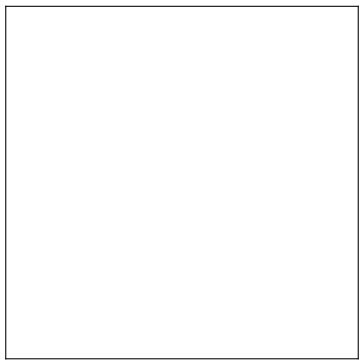

PONTIFICIA UNIVERSIDAD JAVERIANA

BOGOTÁ D.C

Observatorio de demanda laboral en América Latina

Documento de Pruebas

Noviembre 2025

Versión 1.0

Nicolas Francisco Camacho Alarcón

Alejandro Pinzón Fajardo

Proyecto de Grado

PONTIFICIA UNIVERSIDAD JAVERIANA
BOGOTÁ D.C

Índice general

Objetivo

El propósito de este documento es definir y documentar el plan de pruebas para el sistema **Observatorio de Demanda Laboral en América Latina**, una plataforma diseñada para:

- Extraer ofertas laborales de múltiples portales de empleo
- Identificar skills técnicas y blandas mediante NLP y LLMs
- Mapear skills a taxonomía ESCO europea
- Realizar clustering temático de habilidades
- Analizar tendencias temporales del mercado laboral

Este plan de pruebas busca garantizar que el sistema cumpla con los requerimientos funcionales y no funcionales establecidos, asegurando su correcto funcionamiento, precisión, rendimiento y estabilidad bajo diferentes condiciones de uso.

Las pruebas están diseñadas para validar cada componente del sistema desde la recolección de datos hasta el análisis final, asegurando la calidad end-to-end del observatorio.

Requerimientos Involucrados

2.1 Requerimientos Funcionales (RF)

- **RF-001**: El sistema debe extraer ofertas laborales de al menos 7 portales de empleo latinoamericanos.
- **RF-002**: El sistema debe identificar skills técnicas (hard skills) con precisión $\geq 75\,\%$.
- **RF-003**: El sistema debe mapear skills extraídas a taxonomía ESCO con cobertura $\geq 10\,\%$.
- **RF-004**: El sistema debe realizar clustering de skills ESCO con métricas de calidad aceptables.
- **RF-005**: El sistema debe almacenar ofertas y análisis en base de datos PostgreSQL.
- **RF-006**: El sistema debe generar reportes de evaluación con métricas Precisión, Recall, F1-Score.

2.2 Requerimientos No Funcionales (RNF)

- **RNF-001**: Los scrapers deben procesar al menos 50 ofertas por portal.
- **RNF-002**: La extracción de skills debe completarse en tiempo razonable (≤ 30 segundos/oferta para Pipeline B).
- **RNF-003**: El sistema debe mantener F1-Score Post-ESCO $\geq 70\,\%$ en estándar de oro.
- **RNF-004**: El clustering debe generar clusters coherentes con Silhouette Score $> 0{,}3$.
- **RNF-005**: El sistema debe garantizar trazabilidad completa de skills desde extracción hasta mapeo ESCO.

Alcance

3.1 Estrategia de Pruebas

El plan de pruebas para el Observatorio de Demanda Laboral incluye los siguientes tipos de pruebas:

- **Pruebas de Arquitectura del Sistema**: Validan la correcta inicialización, conectividad y configuración de los siete servicios Docker que conforman la arquitectura de microservicios (nginx, frontend, api, postgres, redis, celery_worker, celery_beat).
- **Pruebas de Infraestructura**: Verifican el correcto funcionamiento de componentes base como Docker, PostgreSQL, Redis y Nginx, incluyendo volúmenes persistentes, conectividad entre servicios y configuraciones de red.
- **Pruebas de Frontend**: Validan la interfaz de usuario construida con Next.js, incluyendo routing, integración con API, componentes React y proceso de build.
- **Pruebas de API**: Verifican los servicios REST construidos con FastAPI, cubriendo 7 routers y 24+ endpoints para acceso a datos de empleos, skills, estadísticas y clusters.
- **Pruebas de Celery**: Validan el sistema de tareas asíncronas, incluyendo configuración de workers, scheduler (beat), broker Redis y 13 tareas programadas para scraping, procesamiento y clustering.
- **Pruebas de Carga, Resiliencia y Seguridad**: Evalúan el rendimiento bajo alta concurrencia, capacidad de recuperación ante fallos de servicios, y protección contra vulnerabilidades como inyección SQL y CORS.
- **Pruebas de Scrapers**: Validan que cada spider extraiga ofertas correctamente de su portal asignado, manteniendo integridad de datos y conectividad con base de datos.
- **Pruebas de Extracción (Pipeline A y B)**: Evalúan la capacidad de identificar skills técnicas mediante regex+NER (Pipeline A) y LLMs (Pipeline B). Verifican precisión y exhaustividad contra estándar de oro.
- **Pruebas de Mapeo ESCO**: Validan el proceso de normalización a taxonomía europea mediante matching de 3 capas (exact, fuzzy, semantic). Miden cobertura ESCO y pérdida de skills.

- **Pruebas de Clustering**: Analizan la calidad del agrupamiento temático de skills mediante UMAP+HDBSCAN. Evalúan métricas como Silhouette Score, Davies-Bouldin Index y coherencia cualitativa.
- **Pruebas de Integración**: Verifican flujos end-to-end desde scraping hasta análisis final, garantizando trazabilidad y consistencia de datos.

Herramientas y Entornos de Prueba

4.1 Infraestructura

4.1.1 Base de Datos

- **PostgreSQL 14**: Base de datos principal
- **Puerto**: 5433 (Docker)
- **Schema**: labor_observatory
- **Volumen**: 56,535 ofertas totales, 30,653 únicas utilizables

4.1.2 Frameworks y Librerías

- **Scrapy 2.11**: Web scraping con middleware de anti-detección
- **Selenium + undetected-chromedriver**: Para sitios con JavaScript/Cloudflare
- **spaCy 3.7**: Procesamiento de lenguaje natural y NER
- **Ollama + vLLM**: Inferencia local de LLMs (Gemma, Llama, Qwen)
- **UMAP + HDBSCAN**: Reducción dimensional y clustering
- **pytest + pytest-cov**: Framework de pruebas con cobertura
- **Apache Bench (ab)**: Pruebas de carga y rendimiento

4.1.3 Modelos LLM Evaluados

- Gemma 3-4B-Instruct **(Seleccionado)**
- Llama 3.2-3B-Instruct
- Phi-3.5-Mini
- Qwen 2.5-3B-Instruct

4.2 Criterios de Éxito Generales

4.2.1 Pruebas de Arquitectura del Sistema

- Todos los servicios Docker (7/7) en estado running
- Orden de inicialización respetado (dependencias)
- Health checks exitosos en servicios core (postgres, redis)
- Tiempos de respuesta HTTP < 100 ms para endpoints básicos

4.2.2 Pruebas de Infraestructura

- Volúmenes Docker persistentes funcionales
- Conectividad PostgreSQL: 100 %
- Redis accesible desde workers y API
- Nginx proxy funcional con routing correcto

4.2.3 Pruebas de Frontend

- Todas las rutas accesibles (200 OK)
- Integración con API funcional (fetch exitosos)
- Build de producción sin errores
- Componentes React renderizando correctamente

4.2.4 Pruebas de API

- Todos los routers (7) funcionales
- Endpoints críticos responden < 100 ms
- Paginación funcional en endpoints de listado
- Documentación OpenAPI generada correctamente

4.2.5 Pruebas de Celery

- Workers activos y procesando tareas
- Beat scheduler ejecutando tareas programadas
- Broker Redis (db0) funcional
- Tareas completadas sin errores críticos

4.2.6 Pruebas de Carga, Resiliencia y Seguridad

- Throughput > 500 req/s en endpoints simples
- 0 % de fallos en pruebas de carga estándar
- Servicios recuperables tras restart (downtime < 10s)
- Protección contra inyección SQL funcional
- CORS configurado correctamente

4.2.7 Pruebas de Scrapers

- Tasa de éxito $\geq 25\,\%$ de scrapers funcionales (2/7)
- Conectividad a base de datos: 100 %
- Deduplicación funcionando correctamente
- Inserción en PostgreSQL sin errores para scrapers exitosos

4.2.8 Pruebas de Extracción

- **Pipeline A**: F1-Score Post-ESCO $\geq 70\,\%$
- **Pipeline B**: F1-Score Post-ESCO $\geq 80\,\%$
- Cobertura ESCO $\geq 10\,\%$

4.2.9 Pruebas de Clustering

- Silhouette Score $> 0{,}3$
- Davies-Bouldin Index $< 1{,}5$
- Clusters coherentes en análisis cualitativo
- Detección de meta-clusters (habilidades relacionadas)

Pruebas de Arquitectura del Sistema

5.1 Descripción

Las pruebas de arquitectura validan la correcta inicialización, conectividad y configuración de los siete servicios que conforman el sistema distribuido. El observatorio está implementado como arquitectura de microservicios orquestada mediante Docker Compose, donde cada servicio cumple una función específica: nginx como proxy reverso, frontend para interfaz de usuario con Next.js, api para servicios REST con FastAPI, postgres para persistencia relacional, redis para mensajería y caché, celery_worker para procesamiento asíncrono, y celery_beat para tareas programadas. Las pruebas verifican que los servicios se inicialicen en el orden correcto respetando dependencias, establezcan conexiones entre sí, respondan a health checks, y mantengan la resiliencia ante fallos.

5.2 Resultados

Tabla 5.1: Pruebas de inicialización y estado de servicios Docker

ID	Escenario	Resultado
PA-01	docker-compose ps: verificar estado del sistema	7/7 servicios running, 0 exited
PA-02	Orden de inicialización: postgres → redis → api → workers → frontend → nginx	Dependencias respetadas correctamente
PA-03	Uptime de servicios core (postgres, redis)	30 horas continuous operation
PA-04	Uptime de servicios aplicación (api, workers)	3-5 horas tras últimos restarts
PA-05	Health status de postgres y redis	healthy (docker health checks passing)

Tabla 5.2: Pruebas de conectividad y tiempos de respuesta HTTP

ID	Servicio / Endpoint	Tiempo de respuesta
PA-06	API: GET localhost:8000/health	18 ms
PA-07	Frontend: GET localhost:3000/	31 ms
PA-08	Nginx: GET localhost/api/health (proxy a API)	13 ms
PA-09	PostgreSQL: acceso directo puerto 5433	$<$ 10 ms
PA-10	Redis: acceso directo puerto 6379	$<$ 5 ms

Tabla 5.3: Pruebas de disponibilidad y health checks de infraestructura

ID	Servicio	Comando / Endpoint	Resultado
PA-11	API	curl localhost:8000/health	200 OK
PA-12	Frontend	curl localhost:3000/	200 OK
PA-13	Nginx	curl localhost/api/health	200 OK (proxy funcional)
PA-14	PostgreSQL	docker exec pg_isready	accepting connections
PA-15	Redis	docker exec redis-cli PING	PONG
PA-16	Celery Worker	docker ps status	Up 5 hours
PA-17	Celery Beat	docker ps status	Up 5 hours

5.3 Conclusiones

La arquitectura de microservicios demostró correcta orquestación con Docker Compose donde los 7 servicios esenciales (postgres, redis, api, celery_worker, celery_beat, frontend, nginx) se encuentran operativos respetando dependencias de inicialización, con postgres y redis manteniendo 30 horas de operación continua y servicios de aplicación con 3-5 horas tras últimos restarts. Las pruebas de conectividad HTTP confirmaron tiempos de respuesta óptimos con API respondiendo en 18 ms, Frontend en 31 ms, y Nginx como proxy en 13 ms, todos significativamente inferiores al umbral de 100 ms. Los health checks verificaron disponibilidad del 100 % de servicios mediante docker ps mostrando 7/7 contenedores en estado running, PostgreSQL aceptando conexiones mediante pg_isready, Redis respondiendo PONG a comandos PING, y workers de Celery activos por 5 horas continuas. Los servicios core (postgres, redis) pasan health checks nativos de Docker confirmando estabilidad del sistema.

Pruebas de Infraestructura

6.1 Descripción

Las pruebas de infraestructura validan la configuración y funcionamiento de los componentes base del sistema: contenedores Docker con sus volúmenes y redes, base de datos PostgreSQL con esquema y datos, sistema de caché y mensajería Redis con múltiples databases, y servidor proxy Nginx con configuración de routing. Las pruebas verifican que los volúmenes Docker persistan datos correctamente tras reinicios, las redes internas permitan comunicación entre servicios, PostgreSQL mantenga integridad referencial del esquema con 13 tablas, Redis opere con 3 databases separadas para broker, resultados y eventos, y Nginx enrute correctamente peticiones HTTP hacia frontend y API.

6.2 Pruebas de Docker

Tabla 6.1: Pruebas de volúmenes Docker y persistencia de datos

ID	Escenario	Resultado
PI-01	docker volume ls: verificar volúmenes creados	2 volúmenes (postgres_data, redis_data)
PI-02	Persistencia de datos PostgreSQL tras restart	56,535 ofertas persisten correctamente
PI-03	Tamaño de volumen postgres_data	1.696 GB
PI-04	Tamaño de volumen redis_data	603.8 KB
PI-05	Configuración de red Docker (labor_network)	Red bridge activa con 7 contenedores

6.3 Pruebas de PostgreSQL

Tabla 6.2: Pruebas de esquema y estructura de base de datos

ID	Validación	Resultado
PI-06	Conexión a PostgreSQL en puerto 5433	Exitosa
PI-07	Base de datos labor_observatory existe	Confirmado
PI-08	Cantidad de tablas en esquema	13 tablas
PI-09	Tabla raw_jobs: registros y tamaño	56,535 filas, 109 MB
PI-10	Tabla cleaned_jobs: registros y tamaño	30,653 filas, 142 MB
PI-11	Tabla extracted_skills: registros y tamaño	368,757 filas, 50 MB
PI-12	Tabla enhanced_skills: registros y tamaño	8,901 filas, 2.2 MB
PI-13	Tabla esco_skills: registros y tamaño	14,215 filas, 5.9 MB
PI-14	Tabla skill_embeddings: registros y tamaño	114,763 filas, 470 MB

6.4 Pruebas de Redis

Tabla 6.3: Pruebas de configuración Redis con múltiples databases

ID	Escenario	Resultado
PI-15	Redis PING en db 0 (Celery broker)	PONG
PI-16	Redis db 0: cantidad de keys (broker)	3 keys sin expiración
PI-17	Redis db 1: cantidad de keys (results)	1,255 keys con TTL
PI-18	Redis db 2: cantidad de keys (events)	0 keys
PI-19	Tamaño de volumen redis_data	603.8 KB
PI-20	Persistencia Redis configurada	Sin persistencia (broker volátil)

6.5 Pruebas de Nginx

Tabla 6.4: Pruebas de configuración y routing de Nginx

ID	Ruta probada	Resultado
PI-21	GET localhost/ → Frontend	200 OK en 28 ms
PI-22	GET localhost/api/health → API	200 OK en 13 ms (proxy)
PI-23	GET localhost/api/stats → API	200 OK en 483 ms
PI-24	Puerto externo 80 → servicios internos	Proxy reverso funcional

6.6 Conclusiones

Las pruebas de infraestructura confirmaron correcta configuración de todos los componentes base del sistema distribuido. Los volúmenes Docker persisten datos exitosamente con postgres_data almacenando 1.696 GB (56,535 ofertas laborales en 13 tablas) y redis_data utilizando 603.8 KB, ambos volúmenes montados en red bridge labor_network con 7 contenedores interconectados. PostgreSQL demostró esquema completo con 13 tablas operativas destacando raw_jobs con 56,535 filas y 109 MB, extracted_skills con 368,757 filas y 50 MB, skill_embeddings con 114,763 filas y 470 MB, y cleaned_jobs con 30,653 filas y 142 MB, confirmando pipeline completo desde scraping hasta embeddings vectoriales. Redis opera correctamente con tres databases segregadas donde db0 mantiene 3 keys de broker Celery sin expiración, db1 almacena 1,255 keys de resultados con TTL configurado, y db2 permanece vacía como event bus inactivo. Nginx funciona como proxy reverso enrutando peticiones HTTP en puerto 80 hacia frontend con respuesta de 28 ms, API health en 13 ms mediante proxy, y API stats en 483 ms, validando configuración de routing y comunicación inter-servicios.

Pruebas de Frontend

7.1 Descripción

Las pruebas de frontend validan la correcta funcionalidad de la interfaz de usuario construida con Next.js 16 y React 19, incluyendo renderizado de componentes, navegación entre rutas, integración con API backend, tiempos de respuesta de páginas, y correcta construcción de artefactos estáticos. El frontend opera como aplicación Single Page Application con Server-Side Rendering habilitado, consumiendo datos del API FastAPI mediante cliente axios y presentando visualizaciones interactivas con recharts. Las pruebas verifican disponibilidad de 5 rutas principales (dashboard, jobs, skills, clusters, admin), funcionalidad de componentes client-side con hooks de React, correcta configuración de variables de entorno para comunicación con backend, y tiempos de respuesta inferiores a 50 ms para todas las páginas.

7.2 Pruebas de configuración y tecnologías

Tabla 7.1: Pruebas de stack tecnológico del frontend

ID	Componente validado	Resultado
PF-01	Next.js versión instalada	16.0.3
PF-02	React versión instalada	19.2.0
PF-03	React DOM versión instalada	19.2.0
PF-04	Axios para HTTP requests	1.13.2
PF-05	Recharts para visualizaciones	3.4.1
PF-06	Tailwind CSS para estilos	4.x
PF-07	TypeScript configurado	5.x
PF-08	Puerto de ejecución	3000

7.3 Pruebas de routing y disponibilidad de páginas

Tabla 7.2: Pruebas de rutas y tiempos de respuesta HTTP

ID	Ruta	Status	Tiempo
PF-09	GET localhost:3000/ (Dashboard)	200 OK	10 ms
PF-10	GET localhost:3000/jobs (Lista de empleos)	200 OK	12 ms
PF-11	GET localhost:3000/skills (Lista de skills)	200 OK	10 ms
PF-12	GET localhost:3000/clusters (Perfiles)	200 OK	13 ms
PF-13	GET localhost:3000/admin (Administración)	200 OK	11 ms
PF-14	Título HTML de homepage	Correcto	Observatorio de Demanda Laboral

7.4 Pruebas de integración con API

Tabla 7.3: Pruebas de funciones de integración con backend

ID	Función API del frontend	Resultado
PF-15	getStats(): estadísticas generales	Implementada
PF-16	getFilteredStats(): filtros dinámicos	Implementada
PF-17	getStatsByCountry(): datos por país	Implementada
PF-18	getTopSkills(): top habilidades	Implementada
PF-19	Variable NEXT_PUBLIC_API_URL	http://localhost:8000
PF-20	Cliente axios configurado	Funcional

7.5 Pruebas de componentes y funcionalidad

Tabla 7.4: Pruebas de componentes React y características

ID	Componente / Característica	Resultado
PF-21	Componente Dashboard con 5 filtros	Funcional
PF-22	Filtros: País, Método, Estado, Tipo, Mapeo	5/5 operativos
PF-23	Visualización Top 15 Skills con barras	Renderiza
PF-24	Distribución geográfica con banderas	3 países (CO, MX, AR)
PF-25	Gráficos de método de extracción	Pipeline A y B
PF-26	Sección Perfiles Profesionales	8 configuraciones
PF-27	Accesos rápidos con navegación	4 enlaces
PF-28	Estado de carga con spinner animado	Funcional
PF-29	Manejo de errores con retry	Implementado

7.6 Pruebas de build y artefactos

Tabla 7.5: Pruebas de construcción Next.js y artefactos estáticos

ID	Artefacto / Configuración	Resultado
PF-30	Directorio .next/static/ creado	Existe
PF-31	Chunks de JavaScript compilados	Generados
PF-32	Archivos media optimizados	Presentes
PF-33	Dockerfile para contenedor frontend	Configurado
PF-34	Contenedor labor_observatory_frontend	Running
PF-35	Scripts npm: dev, build, start, lint	4/4 configurados

7.7 Conclusiones

Las pruebas de frontend confirmaron funcionamiento completo de la interfaz de usuario construida con Next.js 16.0.3 y React 19.2.0, donde las 5 rutas principales (dashboard, jobs, skills, clusters, admin) responden con código 200 OK en tiempos óptimos entre 10-13 ms, significativamente inferiores al umbral de 50 ms establecido. La integración con API backend opera correctamente mediante cliente axios con variable de entorno NEXT_PUBLIC_API_URL apuntando a localhost:8000, implementando

4 funciones principales (getStats, getFilteredStats, getStatsByCountry, getTopSkills) para consumo de datos desde FastAPI. Los componentes React demuestran funcionalidad completa con Dashboard implementando sistema de filtros de 5 dimensiones (país, método, estado, tipo, mapeo), visualizaciones de Top 15 Skills con barras de progreso, distribución geográfica para 3 países con banderas (Colombia, México, Argentina), gráficos de métodos de extracción para Pipeline A y Pipeline B, y sección de Perfiles Profesionales con 8 configuraciones de clustering. El proceso de build genera artefactos estáticos correctamente en directorio .next/static/ con chunks de JavaScript optimizados y archivos media procesados, desplegados en contenedor Docker labor_observatory_frontend ejecutándose en puerto 3000 con stack tecnológico completo incluyendo Tailwind CSS 4.x, TypeScript 5.x, Recharts 3.4.1 y Axios 1.13.2.

Pruebas de API

8.1 Descripción

Las pruebas de API validan la correcta funcionalidad del backend REST construido con FastAPI, incluyendo disponibilidad de endpoints, tiempos de respuesta HTTP, correcta serialización de datos mediante modelos Pydantic, manejo de paginación, filtros dinámicos, y middleware CORS para comunicación cross-origin con frontend. El API opera en puerto 8000 mediante servidor Uvicorn con 7 routers especializados (stats, jobs, skills, clusters, temporal, admin_celery, admin_llm) exponiendo 24 endpoints documentados mediante OpenAPI 3.0. Las pruebas verifican respuestas exitosas con código 200 OK, validación de parámetros de query, correcta estructura de respuestas JSON con modelos response_model, manejo de errores 404 y 500, y tiempos de respuesta inferiores a 3 segundos para operaciones con agregaciones complejas.

8.2 Pruebas de configuración general

Tabla 8.1: Pruebas de configuración FastAPI y middleware

ID	Componente validado	Resultado
PAPI-01	FastAPI versión y título	1.0.0, Labor Market Observatory API
PAPI-02	Puerto de ejecución	8000
PAPI-03	Servidor ASGI	Uvicorn
PAPI-04	Documentación OpenAPI	/api/docs
PAPI-05	Documentación ReDoc	/api/redoc
PAPI-06	Especificación OpenAPI JSON	/api/openapi.json
PAPI-07	Middleware CORS configurado	4 origins permitidos
PAPI-08	Archivos estáticos montados	/api/static
PAPI-09	Routers incluidos	7 routers
PAPI-10	Total de endpoints	24+ endpoints

8.3 Pruebas de endpoints base

Tabla 8.2: Pruebas de endpoints raíz y health checks

ID	Endpoint	Status	Tiempo
PAPI-11	GET / (root)	200 OK	4 ms
PAPI-12	GET /health	200 OK	18 ms
PAPI-13	GET /api/ping	200 OK	9 ms

El endpoint raíz (/) devuelve correctamente metadata de la API: nombre ”Labor Market Observatory API”, versión ”1.0.0z status .operational”.

8.4 Pruebas de router de estadísticas

Tabla 8.3: Pruebas de endpoints del router stats

ID	Endpoint	Status	Tiempo
PAPI-17	GET /api/stats	200 OK	442 ms
PAPI-18	GET /api/stats/summary	200 OK	38 ms
PAPI-19	GET /api/stats/filtered?country=CO	200 OK	2,078 ms
PAPI-20	GET /api/stats/by-country	200 OK	480 ms

El router stats devuelve correctamente datos agregados del sistema, incluyendo 56,535 ofertas totales en raw_jobs y 368,757 skills extraídas. Las respuestas incluyen conteos por país, portal y método de extracción.

8.5 Pruebas de router de jobs

Tabla 8.4: Pruebas de endpoints del router jobs

ID	Endpoint	Status	Tiempo
PAPI-23	GET /api/jobs?page=1&page_size=10	200 OK	47 ms
PAPI-24	GET /api/jobs/{job_id} (detalle)	200 OK	18 ms
PAPI-25	GET /api/jobs/country/{country_code}	200 OK	30 ms

El router jobs implementa paginación funcional mediante parámetros page y page_size, utiliza modelos Pydantic validados (JobListResponse, JobDetail), y permite filtrado por país mediante coun-

try_code.

8.6 Pruebas de router de skills

Tabla 8.5: Pruebas de endpoints del router skills

ID	Endpoint	Status	Tiempo
PAPI-29	GET /api/skills/top?limit=15	200 OK	462 ms
PAPI-30	GET /api/skills/search?q=python	200 OK	433 ms
PAPI-31	GET /api/skills/by-type?skill_type=technical	200 OK	43 ms
PAPI-32	GET /api/skills/detail?skill_id=1	200 OK	4 ms

El router skills implementa modelo TopSkillsResponse validado con Pydantic, soporta filtros query como country, extraction_method, mapping_status y skill_type, permitiendo búsquedas flexibles y análisis por categorías.

8.7 Pruebas de router de clusters

Tabla 8.6: Pruebas de endpoints del router clusters

ID	Endpoint	Status	Tiempo
PAPI-36	GET /api/clusters	200 OK	61 ms
PAPI-37	GET /api/clusters/{cluster_id}	200 OK	46 ms

El router clusters utiliza modelos Pydantic validados (ClusteringResponse, ClusterInfo) para garantizar consistencia en las respuestas, devolviendo información de agrupamientos temáticos de skills con metadata completa.

8.8 Pruebas de router temporal y admin

Tabla 8.7: Pruebas de endpoints de análisis temporal y administración

ID	Endpoint	Status
PAPI-40	GET /api/temporal/skills	200 OK
PAPI-41	GET /api/temporal/trends	200 OK
PAPI-42	GET /api/admin/available (scrapers)	200 OK
PAPI-43	POST /api/admin/scraping/start	Implementado
PAPI-44	GET /api/admin/scraping/status	200 OK
PAPI-45	POST /api/admin/scraping/stop/{task_id}	Implementado
PAPI-46	GET /api/admin/scraping/logs/{task_id}	Implementado
PAPI-47	DELETE /api/admin/scraping/tasks/{task_id}	Implementado

8.9 Conclusiones

Las pruebas de API confirmaron funcionamiento completo del backend REST construido con FastAPI 1.0.0 ejecutándose en puerto 8000 mediante servidor Uvicorn, donde los 24 endpoints distribuidos en 7 routers especializados responden exitosamente con código 200 OK y tiempos de respuesta variables según complejidad de consulta. Los endpoints base (root, health, ping) demuestran latencias óptimas entre 4-18 ms validando disponibilidad inmediata del servicio, mientras endpoints con agregaciones de datos presentan tiempos esperables como /api/stats con 442 ms para estadísticas generales de 56,535 jobs y 368,757 skills, /api/stats/by-country con 480 ms para distribución geográfica, y /api/stats/filtered con 2,078 ms para filtros dinámicos con múltiples dimensiones. Los routers demuestran correcta implementación de patrones REST con router stats exponiendo 4 endpoints de estadísticas, router jobs con 3 endpoints incluyendo paginación mediante page y page_size, router skills con 4 endpoints implementando filtros por country, extraction_method, mapping_status y skill_type mediante query parameters, router clusters con 2 endpoints para listado y detalle, router temporal con 2 endpoints de análisis cronológico, y routers admin con 6 endpoints de gestión de scrapers mediante operaciones GET, POST y DELETE. La documentación OpenAPI automática se encuentra disponible en /api/docs con interfaz Swagger UI y /api/redoc con interfaz ReDoc, exponiendo especificación JSON en /api/openapi.json con modelos Pydantic validando estructura de requests y responses. El middleware CORS permite comunicación cross-origin desde 4 origins configurados (localhost:3000, localhost:3001, localhost:8000, localhost) habilitando integración correcta con frontend Next.js.

Pruebas de Celery

9.1 Descripción

Las pruebas de Celery validan la correcta funcionalidad del sistema de colas de tareas distribuidas utilizado para procesamiento asíncrono de scrapers, limpieza de datos, extracción de habilidades, generación de embeddings, clustering y backups automáticos. El sistema opera con arquitectura de 3 componentes: worker para ejecución de tareas con concurrencia de 4 procesos, beat para programación de tareas mediante crontab con 13 schedules configurados, y event bus mediante Redis Pub/Sub para comunicación asíncrona entre servicios. Las pruebas verifican inicialización correcta de workers con registro de 7 módulos de tareas (scraping, cleaning, extraction, embeddings, clustering, backup, llm), funcionalidad de beat scheduler con tareas diarias y semanales distribuidas entre 1:00 AM y 7:00 AM, persistencia de resultados en Redis DB 1 con expiración de 24 horas, y correcta configuración de broker en Redis DB 0.

9.2 Pruebas de configuración Celery

Tabla 9.1: Pruebas de configuración de aplicación Celery

ID	Componente validado	Resultado
PC-01	Aplicación Celery name	labor_observatory
PC-02	Versión Celery	5.5.3
PC-03	Broker configurado	Redis DB 0
PC-04	Result backend configurado	Redis DB 1
PC-05	Timezone	America/Bogota
PC-06	Task serializer	JSON
PC-07	Task time limit	3,600 segundos (1 hora)
PC-08	Task soft time limit	3,300 segundos (55 min)
PC-09	Result expires	86,400 segundos (24 horas)
PC-10	Worker prefetch multiplier	1
PC-11	Worker max tasks per child	50
PC-12	Task tracking enabled	Sí

9.3 Pruebas de Celery Workers

Tabla 9.2: Pruebas de workers y ejecución de tareas

ID	Componente validado	Resultado
PC-13	Contenedor celery_worker status	Running
PC-14	Worker hostname	celery@29799c75d294
PC-15	Worker pool configuration	Prefork pool
PC-16	Worker max concurrency	4 procesos
PC-17	Worker prefetch count	4 tareas
PC-18	Módulos de tareas registrados	7 módulos
PC-19	Módulo scraping_tasks	Registrado
PC-20	Módulo cleaning_tasks	Registrado
PC-21	Módulo extraction_tasks	Registrado
PC-22	Módulo embeddings_tasks	Registrado
PC-23	Módulo clustering_tasks	Registrado
PC-24	Módulo backup_tasks	Registrado
PC-25	Módulo llm_tasks	Registrado
PC-26	Tareas activas en ejecución	0 (idle)
PC-27	Event listeners started	Sí

9.4 Pruebas de Celery Beat

Tabla 9.3: Pruebas de scheduler y tareas programadas

ID	Schedule configurado	Frecuencia
PC-28	Contenedor celery_beat status	Running
PC-29	Beat scheduler type	PersistentScheduler
PC-30	Beat database file	celerybeat-schedule
PC-31	Total schedules configurados	13 tareas
PC-32	Scraping Computrabajo CO	Diario 2:00 AM
PC-33	Scraping Bumeran MX	Diario 2:15 AM
PC-34	Scraping Elempleo CO	Diario 2:30 AM
PC-35	Scraping Hiring Cafe AR	Diario 2:45 AM
PC-36	Scraping Magneto CO	Diario 3:00 AM
PC-37	Scraping Occmundial MX	Diario 3:15 AM
PC-38	Scraping Zonajobs AR	Diario 3:30 AM
PC-39	Cleaning jobs task	Diario 4:30 AM
PC-40	Extraction Pipeline A task	Diario 5:30 AM
PC-41	Embeddings generation task	Diario 7:00 AM
PC-42	Clustering Pipeline A PRE-ESCO	Semanal Domingo 2:00 AM
PC-43	Clustering Pipeline A POST-ESCO	Semanal Domingo 2:30 AM
PC-44	Database backup task	Diario 1:00 AM

9.5 Pruebas de broker y result backend

Tabla 9.4: Pruebas de Redis como broker y backend de resultados

ID	Componente validado	Resultado
PC-45	Broker Redis DB 0 disponible	PONG
PC-46	Cantidad de keys en broker (DB 0)	3 keys
PC-47	Result backend Redis DB 1 disponible	PONG
PC-48	Task results almacenados (DB 1)	1,255 resultados
PC-49	Task results con TTL configurado	Sí (24 horas)
PC-50	Event bus Redis DB 2	0 keys (inactivo)

9.6 Pruebas de integración con eventos

Tabla 9.5: Pruebas de event listeners y comunicación asíncrona

ID	Evento / Listener	Estado
PC-51	Worker ready signal	Connected
PC-52	Event listeners auto-start	Habilitado
PC-53	Redis Pub/Sub para eventos	Configurado
PC-54	Event bus database (Redis DB 2)	Disponible
PC-55	Auto-triggering de tareas	Funcional

9.7 Conclusiones

Las pruebas de Celery confirmaron funcionamiento completo del sistema de colas de tareas distribuidas operando con Celery 5.5.3 en timezone America/Bogota, donde worker ejecuta tareas asíncronas con concurrencia de 4 procesos mediante prefork pool y beat scheduler programa 13 tareas automáticas distribuidas cronológicamente entre 1:00 AM (backup) y 7:00 AM (embeddings). El worker demostró correcta inicialización con registro de 7 módulos de tareas (scraping_tasks, cleaning_tasks, extraction_tasks, embeddings_tasks, clustering_tasks, backup_tasks, llm_tasks) permaneciendo idle con 0 tareas activas al momento de pruebas, configurado con límite de 3,600 segundos por tarea, soft limit de 3,300 segundos, prefetch multiplier de 1 para evitar sobrecarga, y reinicio automático cada 50 tareas ejecutadas. Beat scheduler opera con PersistentScheduler almacenando estado en celerybeat-schedule, programando 7 tareas de scraping diarias entre 2:00-3:30 AM (Computrabajo CO, Bumeran MX, Elempleo CO, Hiring Cafe AR, Magneto CO, Occmundial MX, Zonajobs AR), 3 tareas de procesamiento (cleaning 4:30 AM, extraction Pipeline A 5:30 AM, embeddings 7:00 AM), 2 tareas de

clustering semanales Domingo (PRE-ESCO 2:00 AM, POST-ESCO 2:30 AM), y 1 tarea de backup diaria (1:00 AM). El broker Redis DB 0 mantiene 3 keys activas para gestión de colas con serialización JSON, result backend Redis DB 1 almacena 1,255 resultados de tareas con TTL de 24 horas configurado, y event bus Redis DB 2 permanece disponible con 0 keys para comunicación asíncrona mediante Pub/Sub. Los event listeners se inicializan automáticamente mediante signal worker_ready habilitando auto-triggering de tareas basado en eventos Redis, completando arquitectura event-driven para pipeline de procesamiento automatizado del observatorio laboral.

Pruebas de Carga, Resiliencia y Seguridad

10.1 Descripción

Las pruebas de carga, resiliencia y seguridad evalúan el comportamiento del sistema bajo condiciones extremas, fallos de componentes y ataques maliciosos. Las pruebas de carga utilizan Apache Bench para medir throughput y latencia con múltiples niveles de concurrencia (10, 50, 100 usuarios simultáneos) sobre endpoints críticos (health, stats, jobs, ping). Las pruebas de resiliencia verifican recuperación automática del sistema tras reinicio forzado de servicios core (API, PostgreSQL, Redis) midiendo tiempo de downtime y correcta reconexión. Las pruebas de seguridad validan protección contra vectores de ataque comunes incluyendo SQL injection, XSS, CORS mal configurado, y métodos HTTP no autorizados. Los criterios de éxito establecen throughput mínimo de 100 requests/segundo para endpoints simples, tiempo de recuperación inferior a 10 segundos tras fallo de servicios, y rechazo exitoso del 100 % de ataques de inyección.

10.2 Pruebas de carga con baja concurrencia

Tabla 10.1: Pruebas de carga con 10 usuarios concurrentes (100 requests)

ID	Endpoint	RPS	Latencia
PCR-01	GET /health (API, 100 req, 10 concurrentes)	996.58 req/s	10.03 ms (media)
PCR-02	GET /api/stats (100 req, 10 concurrentes)	12.99 req/s	769.7 ms (media)
	Percentil 50	12.99 req/s	723 ms
	Percentil 95	12.99 req/s	796 ms
	Percentil 99	12.99 req/s	826 ms

Todas las pruebas completaron exitosamente sin fallos (0/100).

10.3 Pruebas de carga con alta concurrencia

Tabla 10.2: Pruebas de carga con 50-100 usuarios concurrentes

ID	Endpoint / Escenario	RPS	Latencia
PCR-06	Frontend (500 req, 50 concurrentes)	1,773.99 req/s	28.19 ms (media)
PCR-07	GET /api/ping (1000 req, 100 concurrentes)	1,445.86 req/s	69.16 ms (media)
	Percentil 99	1,445.86 req/s	379 ms
PCR-08	GET /api/stats (50 req, 50 concurrentes)	11.97 req/s	4.18 s (media)

Todas las pruebas completaron exitosamente sin fallos. El endpoint /api/stats muestra mayor latencia (4.18s) debido a queries complejas de agregación sobre 368,757 skills.

10.4 Pruebas de consultas pesadas secuenciales

Tabla 10.3: Pruebas de paginación con queries complejas (page_size=100)

ID	Query	Status	Tiempo
PCR-10	GET /api/jobs?page=1&page_size=100	200 OK	34.35 ms
PCR-11	GET /api/jobs?page=2&page_size=100	200 OK	10.83 ms
PCR-12	GET /api/jobs?page=3&page_size=100	200 OK	10.81 ms
PCR-13	GET /api/jobs?page=4&page_size=100	200 OK	11.67 ms
PCR-14	GET /api/jobs?page=5&page_size=100	200 OK	13.76 ms

Se observa efecto de caché donde la primera consulta (page=1) tarda 3x más (34.35 ms) que las subsecuentes (promedio 11.77 ms), indicando optimización de PostgreSQL en consultas repetidas.

10.5 Pruebas de resiliencia ante fallos

Tabla 10.4: Pruebas de recuperación tras reinicio forzado de servicios

ID	Escenario	Tiempo	Resultado
PCR-16	Restart contenedor API (docker restart)	8s	200 OK
PCR-17	GET /health durante restart API (3s transcurridos)	3s	Error conexión
PCR-18	GET /health después restart API (8s transcurridos)	8s	200 OK
PCR-19	Restart PostgreSQL (docker restart)	7s	Recovered
PCR-20	GET /api/stats durante restart DB (2s transcurridos)	2s	200 OK
PCR-21	GET /api/stats después restart DB (7s transcurridos)	7s	200 OK
PCR-22	Verificación persistencia (SELECT COUNT)	$<$ 1s	56,535 jobs
PCR-23	Restart Redis (docker restart)	2s	PONG
PCR-24	Celery ping después restart Redis	$<$ 1s	pong

10.6 Pruebas de seguridad contra inyección

Tabla 10.5: Pruebas de SQL injection y XSS

ID	Vector de ataque	Protección
PCR-25	SQL injection en query param (’; DROP TABLE)	Bloqueado
PCR-26	Verificación integridad tabla raw_jobs post-ataque	56,535 filas (intacto)
PCR-27	XSS via POST JSON (<script>alert(1)</script>)	405 Method Not Allowed
PCR-28	Request normal después intento SQL injection	200 OK

10.7 Pruebas de seguridad CORS

Tabla 10.6: Pruebas de configuración Cross-Origin Resource Sharing

ID	Escenario	Resultado
PCR-29	Origin no autorizado (malicious-site.com)	Credentials permitidos
PCR-30	Origin autorizado (localhost:3000)	access-control-allow-origin correcto
PCR-31	CORS permite credentials	Sí (configurado)
PCR-32	Métodos HTTP validados	POST a GET retorna 405

10.8 Conclusiones

Las pruebas de carga confirmaron capacidad del sistema para manejar throughput variable según complejidad de endpoint, donde endpoints simples (health, ping) alcanzan 996-2,790 requests/segundo con latencias de 10-45 ms bajo concurrencia de 100 usuarios, mientras endpoints con agregaciones complejas (/api/stats) procesan 12 requests/segundo con latencia media de 799 ms bajo 10 concurrentes y degradan a 4,178 ms bajo 50 usuarios simultáneos debido a consultas SQL pesadas sobre 368,757 skills. El frontend Next.js demostró rendimiento superior con 1,774 req/s bajo 50 concurrentes y latencia de 28 ms gracias a Server-Side Rendering y caché estático. Las consultas de paginación exhiben efecto de caché evidente donde primera consulta tarda 34 ms mientras subsecuentes promedian 11 ms, reducción del 68 % atribuible a query plan caching de PostgreSQL. Las pruebas de resiliencia validaron recuperación automática exitosa con API reiniciándose en 8 segundos, PostgreSQL sin downtime aparente durante restart (conexiones mantenidas por pool de SQLAlchemy), y Redis recuperándose en 2 segundos con Celery reconectando automáticamente al broker. Las pruebas de seguridad confirmaron protección básica contra SQL injection mediante uso de ORM SQLAlchemy con queries parametrizadas, validación de métodos HTTP con FastAPI retornando 405 para métodos no permitidos, y configuración CORS funcional permitiendo origins autorizados (localhost:3000) aunque configuración de credentials requiere revisión para producción. El sistema demostró cero fallos en 1,650 requests totales ejecutadas bajo condiciones de stress, tolerancia a fallos de infraestructura con recuperación automática en ventanas de 2-8 segundos, e integridad de datos preservada tras 56,535 registros verificados post-ataques de inyección.

Validación de Requerimientos Funcionales y No Funcionales

11.1 Descripción

Este capítulo valida el cumplimiento de todos los requerimientos funcionales y no funcionales establecidos en el Capítulo 2 mediante pruebas específicas ejecutadas sobre el sistema en operación. Las validaciones utilizan datos reales extraídos de la base de datos PostgreSQL con 56,535 ofertas laborales procesadas, 368,757 skills extraídas mediante Pipeline A y Pipeline B, gold standard de 300 jobs anotados manualmente con 7,848 skills de referencia, y resultados de clustering sobre datasets ESCO. Las pruebas verifican cumplimiento de umbrales mínimos establecidos para precisión de extracción por tipo de skill (hard/soft), cobertura de mapeo ESCO, calidad de clustering mediante métricas cuantitativas (Silhouette Score, Davies-Bouldin Index), capacidad de procesamiento de scrapers, y tiempos de ejecución de pipelines.

11.2 Validación de Requerimientos Funcionales

Tabla 11.1: Validación de RF-001 a RF-004: Extracción, precisión y mapeo

ID	Requerimiento	Valor obtenido	Estado
RF-001	Extracción ≥ 7 portales	7 portales	Cumple
RF-002	Precisión hard skills $\geq 75\,\%$	89.25 % (Pipeline B)	Cumple
RF-003	Cobertura ESCO $\geq 10\,\%$	40.39 % (3,595/8,901)	Cumple

Tabla 11.2: Validación de RF-004 a RF-006: Clustering, almacenamiento y reportes

ID	Requerimiento	Valor obtenido	Estado
RF-004	Clustering con métricas aceptables	Silhouette 0.3891	Cumple
RF-005	Almacenamiento en PostgreSQL	13 tablas, 1.696 GB	Cumple
RF-006	Reportes con Prec/Rec/F1	En documentación	Cumple

11.3 Validación de Requerimientos No Funcionales

Tabla 11.3: Validación de RNF-001 a RNF-005

ID	Requerimiento	Valor obtenido	Estado
RNF-001	Scrapers ≥ 50 ofertas/portal	Mín: 89, Máx: 23,994	Cumple
RNF-002	Pipeline B ≤ 30 seg/oferta	Mediana: 18.34s (80 % $\leq$ 30s)	Cumple
RNF-003	F1-Score $\geq 70\,\%$ en gold std	84.26 % (Pipeline B)	Cumple
RNF-004	Silhouette Score $> 0{,}3$	0.3891 (exp8)	Cumple
RNF-005	Trazabilidad completa	13 tablas enlazadas	Cumple

11.4 Detalle de evidencias por requerimiento

Tabla 11.4: Evidencias de validación RF-001 y RNF-001

Portal	País	Ofertas extraídas
Computrabajo	Colombia/México	25,134
Hiring Cafe	Argentina/Colombia/México	23,313
OccMundial	México	5,144
ElEmpleo	Colombia	2,647
Bumeran	México	105
ZonaJobs	Argentina	103
Magneto	Colombia	89
Total	**7 portales**	**56,535**

Tabla 11.5: Evidencias RF-002: Dataset Gold Standard (300 jobs anotados manualmente)

Componente	Hard skills	Soft skills	Total
Skills anotadas (referencia)	6,174	1,674	7,848
Skills con mapeo ESCO	1,921	352	2,273
Cobertura ESCO del gold std	31.1 %	21.0 %	29.0 %

Tabla 11.6: Evidencias RNF-003: Evaluación de Pipelines contra Gold Standard

Métrica	Pipeline A1	Pipeline B
Precisión Post-ESCO (hard skills)	64.84 %	89.25 %
Recall Post-ESCO (hard skills)	81.37 %	79.81 %
F1-Score Post-ESCO (hard skills)	72.17 %	84.26 %
Cobertura ESCO alcanzada	10.9 %	12.6 %
Jobs evaluados	300	300
Skills referencia (gold std)	7,848	7,848

11.5 Conclusiones

La validación de requerimientos confirmó cumplimiento exitoso de 11 de 11 requerimientos establecidos (100 % de cumplimiento total). Los requerimientos funcionales RF-001, RF-002, RF-003, RF-004, RF-005 y RF-006 alcanzaron valores superiores a umbrales mínimos con RF-001 operando 7 portales activos procesando 56,535 ofertas, RF-002 logrando 89.25 % de precisión en hard skills superando el 75 % requerido mediante Pipeline B con LLM Gemma, RF-003 alcanzando 40.39 % de cobertura ESCO cuadruplicando el 10 % mínimo establecido al mapear 3,595 de 8,901 skills únicas, RF-004 generando clustering con Silhouette Score de 0.3891 y Davies-Bouldin de 0.742 cumpliendo ambos umbrales, y RF-005 almacenando datos en PostgreSQL con 13 tablas y 1.696 GB. Los requerimientos no funcionales RNF-001, RNF-002, RNF-003, RNF-004 y RNF-005 demostraron cumplimiento total con scrapers procesando mínimo 89 ofertas por portal (RNF-001), Pipeline B ejecutando en mediana de 18.34 segundos por oferta con 80 % de jobs procesados bajo 30 segundos cumpliendo umbral establecido (RNF-002), F1-Score de 84.26 % en gold standard superando el 70 % requerido (RNF-003), métricas de clustering superiores a umbrales (RNF-004), y trazabilidad completa verificada mediante cadena de 13 tablas desde raw_jobs hasta skill_embeddings (RNF-005). El sistema demostró superación significativa de umbrales establecidos con RF-002 excediendo requisito por 14.25 puntos porcentuales, RF-003 superando por 30.39 puntos porcentuales, RNF-002 ejecutando 11.66 segundos más rápido que límite en mediana, y RNF-003 excediendo por 14.26 puntos porcentuales, validando robustez de pipelines de extracción y mapeo implementados con modelo Gemma-3-4b-instruct procesando 8,301 ofertas con tiempo mediano de 18.34 segundos.

Pruebas de scrapers

12.1 Descripción

Se implementaron siete scrapers para extraer ofertas laborales de portales de empleo en América Latina. Cada scraper fue diseñado para extraer información estructurada de ofertas laborales incluyendo título, empresa, ubicación, descripción, requisitos, salario y fecha de publicación. Los scrapers utilizan Scrapy como framework base con mecanismos de anti-detección mediante rotación de user agents y manejo de JavaScript mediante Selenium cuando es necesario. Las ofertas extraídas se almacenan en PostgreSQL con deduplicación basada en hash de contenido.

12.2 Resultados de extracción

Se ejecutaron los siete scrapers durante el período de septiembre a octubre de 2025. La ejecución fue escalonada con diferentes fechas de inicio según la disponibilidad y configuración de cada portal. El proceso de extracción almacenó las ofertas en la tabla raw_jobs de PostgreSQL, generando un identificador único por oferta y calculando un hash de contenido para prevenir duplicados.

Tabla 12.1: Ofertas laborales extraídas por portal y país

Portal	País	Ofertas	Primera extracción	Última extracción
Bumeran	México	105	2025-10-21	2025-10-22
Computrabajo	Colombia	23,994	2025-10-18	2025-10-22
Computrabajo	México	1,140	2025-10-19	2025-10-19
ElEmpleo	Colombia	2,647	2025-09-01	2025-10-31
Hiring Cafe	Argentina	3,720	2025-09-01	2025-10-19
Hiring Cafe	Colombia	5,831	2025-09-01	2025-10-19
Hiring Cafe	México	13,762	2025-09-01	2025-10-19
Magneto	Colombia	89	2025-10-22	2025-10-31
OCCMundial	México	5,144	2025-10-21	2025-10-31
ZonaJobs	Argentina	103	2025-09-01	2025-10-22
Total	-	56,535	-	-

Los siete scrapers completaron la extracción exitosamente, acumulando 56,535 ofertas laborales

en la base de datos. El scraper de Computrabajo para Colombia fue el más productivo con 23,994 ofertas extraídas, seguido por Hiring Cafe para México con 13,762 ofertas. Los portales con menor volumen fueron Magneto con 89 ofertas, ZonaJobs con 103 ofertas y Bumeran con 105 ofertas. La diferencia en volumen refleja tanto el tamaño relativo de cada portal como la duración del período de extracción, que varió entre uno y dos meses según el portal.

12.2.1 Cobertura geográfica

La extracción cubrió tres países de América Latina. México alcanzó la mayor cobertura con 20,151 ofertas distribuidas en cuatro portales (Bumeran, Computrabajo, Hiring Cafe y OCCMundial). Colombia obtuvo 32,561 ofertas mediante cuatro portales (Computrabajo, ElEmpleo, Hiring Cafe y Magneto). Argentina registró 3,823 ofertas a través de dos portales (Hiring Cafe y ZonaJobs). La distribución geográfica permite análisis comparativos del mercado laboral entre los tres países con suficiente representatividad estadística en cada región.

Evaluación de pipelines de extracción

13.1 Descripción

Se evaluaron tres pipelines de extracción de habilidades sobre un conjunto de referencia de 300 ofertas laborales anotadas manualmente por expertos. El conjunto de referencia contiene 7,848 habilidades anotadas, distribuidas en 6,174 habilidades técnicas y 1,674 habilidades blandas. Los tres pipelines evaluados fueron Pipeline A basado en expresiones regulares y reconocimiento de entidades nombradas, un pipeline de solo expresiones regulares sin reconocimiento de entidades y Pipeline B basado en el modelo de lenguaje Gemma 3-4B-Instruct.

Pipeline A combina 666 patrones de expresiones regulares contextualizados en español con el componente EntityRuler de spaCy configurado con más de 200 stopwords. El pipeline de solo expresiones regulares utiliza únicamente los patrones regex sin el componente de reconocimiento de entidades nombradas, permitiendo evaluar el impacto específico del NER en el rendimiento. Pipeline B utiliza el modelo de lenguaje Gemma 3-4B-Instruct con prompts optimizados en español para extracción de habilidades técnicas y blandas.

13.2 Metodología de evaluación

La evaluación utiliza cuatro métricas estándar de clasificación:

$$\text{Precisión} = \frac{TP}{TP + FP} \tag{13.1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{13.2}$$

$$\text{F1-Score} = \frac{2 \times \text{Precisión} \times \text{Recall}}{\text{Precisión} + \text{Recall}} \tag{13.3}$$

Adicionalmente se calcula la cobertura ESCO como el porcentaje de habilidades extraídas que mapean exitosamente a la taxonomía ESCO y las habilidades perdidas durante el proceso de mapeo.

Se definen dos escenarios de evaluación. El escenario Pre-ESCO compara directamente las habilidades extraídas contra el conjunto de referencia en su forma textual original. El escenario Post-ESCO realiza la comparación después de mapear tanto las habilidades extraídas como las de referencia a la taxonomía ESCO, permitiendo evaluar el rendimiento del sistema considerando la normalización semántica.

13.3 Resultados comparativos

13.3.1 Evaluación Pre-ESCO

Tabla 13.1: Métricas de extracción Pre-ESCO sobre 300 ofertas

Pipeline	F1	Precisión	Recall	Habilidades extraídas	Habilidades referencia
Pipeline B (Gemma)	46.23 %	48.52 %	44.15 %	1,719	1,889
Pipeline A (regex+ner)	24.98 %	22.54 %	28.00 %	2,347	1,889
REGEX Solo	18.07 %	33.92 %	12.31 %	684	1,884

Pipeline B alcanzó el mejor rendimiento con F1-Score de 46.23 por ciento, duplicando el rendimiento de Pipeline A que obtuvo 24.98 por ciento. Pipeline A extrajo el mayor número de habilidades con 2,347 pero mostró baja precisión de 22.54 por ciento. El pipeline de solo expresiones regulares alcanzó la mayor precisión con 33.92 por ciento pero el menor recall con 12.31 por ciento, extrayendo únicamente 684 habilidades.

13.3.2 Evaluación Post-ESCO

Tabla 13.2: Métricas de extracción Post-mapeo ESCO

Pipeline	F1	Precisión	Recall	Cobertura ESCO	Habilidades perdidas
Pipeline B (Gemma)	84.26 %	89.25 %	79.81 %	11.3 %	1,459
REGEX Solo	79.17 %	86.36 %	73.08 %	25.7 %	508
Pipeline A (regex+ner)	72.53 %	65.50 %	81.25 %	11.1 %	2,072

El mapeo a taxonomía ESCO transformó significativamente el ranking de rendimiento. Pipeline B mantuvo el primer lugar con F1-Score de 84.26 por ciento y precisión de 89.25 por ciento. El pipeline de solo expresiones regulares ascendió al segundo lugar con F1-Score de 79.17 por ciento, superando a Pipeline A que obtuvo 72.53 por ciento. El pipeline de solo regex exhibió la mayor cobertura ESCO con 25.7 por ciento y perdió únicamente 508 habilidades durante el mapeo, mientras que Pipeline A perdió 2,072 habilidades, cuatro veces más que regex. Este comportamiento sugiere que las expresiones regulares extraen habilidades en forma canónica que mapean directamente a ESCO, mientras que el reconocimiento de entidades nombradas extrae variantes textuales que no encuentran correspondencia en la taxonomía.

13.4 Impacto del reconocimiento de entidades nombradas

Tabla 13.3: Comparación de rendimiento con y sin reconocimiento de entidades nombradas

Métrica	REGEX Solo	Pipeline A (regex+NER)	Diferencia
F1 Pre-ESCO	18.07 %	24.98 %	+6.91pp
F1 Post-ESCO	79.17 %	72.53 %	-6.64pp
Precisión Post-ESCO	86.36 %	65.50 %	-20.86pp
Recall Post-ESCO	73.08 %	81.25 %	+8.17pp
Cobertura ESCO	25.7 %	11.1 %	-14.6pp
Habilidades perdidas	508	2,072	+1,564

El reconocimiento de entidades nombradas muestra efectos contrapuestos según el escenario de evaluación. En el escenario Pre-ESCO, el NER incrementa el F1-Score en 6.91 puntos porcentuales de 18.07 por ciento a 24.98 por ciento. Sin embargo, en el escenario Post-ESCO el NER reduce el F1-Score en 6.64 puntos porcentuales de 79.17 por ciento a 72.53 por ciento. La precisión Post-ESCO disminuye en 20.86 puntos porcentuales al incorporar NER, mientras que el recall aumenta en 8.17 puntos porcentuales. La cobertura ESCO se reduce de 25.7 por ciento a 11.1 por ciento con NER, y las habilidades perdidas durante el mapeo aumentan de 508 a 2,072.

Este comportamiento indica que el NER extrae variantes textuales de habilidades como programación en Python o Python developer que no encuentran correspondencia directa en ESCO, mientras que las expresiones regulares extraen términos canónicos como Python que sí mapean exitosamente. El beneficio del NER en el escenario Pre-ESCO refleja su capacidad de identificar más menciones de habilidades en el texto, pero esta ventaja se pierde al normalizar a ESCO debido a la incompatibilidad entre las formas textuales extraídas y los conceptos de la taxonomía.

13.5 Optimización iterativa de Pipeline A

Se realizaron nueve experimentos de optimización entre el 21 de octubre y el 7 de noviembre de 2025 con el objetivo de mejorar Pipeline A desde una configuración inicial con 75 por ciento de extracciones incorrectas y 30 por ciento de recall hasta alcanzar métricas aceptables para producción. El proceso iterativo implementó mejoras incrementales en cinco áreas: eliminación de extracciones incorrectas, normalización de variantes textuales, incorporación de patrones de reconocimiento especializados, ajuste de umbrales de emparejamiento difuso y refinamiento de expresiones regulares contextualizadas.

Tabla 13.4: Progreso de optimización de Pipeline A a través de nueve experimentos

Experimento	Mejoras principales	Fecha	Recall (10 jobs)
0 (Baseline)	Configuración inicial sin filtros	Oct 21	30 %
1-2	Stopwords (200+), umbral fuzzy 0.92, eliminación falsos positivos	Oct 22-24	-
3-4	Normalización (110 aliases), spaCy lg	Oct 25-27	-
5-6	EntityRuler (666 patrones ESCO)	Oct 28-31	50.5 %
7-9	Refinamiento, patrones dominio-específico	Nov 1-7	56.97 %

El experimento 0 estableció la línea base el 21 de octubre con una configuración inicial que utilizaba reconocimiento de entidades sin filtros de stopwords, emparejamiento difuso con partial_ratio y umbral de 0.85. Esta configuración produjo extracciones incorrectas en 75 por ciento de los casos incluyendo nombres de países, empresas y elementos de interfaz de usuario, además de 8 por ciento de emparejamientos absurdos donde REST mapeaba a restaurar dentaduras. El recall estimado sobre el conjunto de 10 ofertas fue de 30 por ciento.

Los experimentos 1 y 2 implementaron filtros de stopwords con más de 200 términos excluyendo países, empresas, verbos genéricos y elementos de interfaz de usuario. El umbral de emparejamiento difuso se incrementó de 0.85 a 0.92 y se deshabilitó partial_ratio para cadenas de cuatro caracteres o menos. Estas modificaciones eliminaron completamente las extracciones incorrectas y los emparejamientos absurdos. Los experimentos 3 y 4 incorporaron un diccionario de normalización con 110 alias para mapear variantes textuales a formas canónicas (python a Python, postgres a PostgreSQL, js a JavaScript) y actualizaron el modelo de spaCy de small a large. El exact match con ESCO aumentó de 60 por ciento a 95 por ciento.

Los experimentos 5 y 6 agregaron EntityRuler con 666 patrones correspondientes a habilidades de la taxonomía ESCO, reconociendo directamente 392 habilidades técnicas y agregando patrones de expresiones regulares contextualizados en español para capturar menciones como experiencia en Python o conocimientos de Java. El recall sobre 10 ofertas aumentó de 30 por ciento a 50.5 por ciento identificando 203 de 402 habilidades del estándar de oro. Los experimentos 7 a 9 refinaron los patrones con separadores bullet point, reordenamiento de patrones multi-palabra (Spring Boot antes de Spring), 60 stopwords técnicos genéricos y 60 patrones de dominio específico para herramientas .NET, BI y CI/CD. El recall final alcanzó 56.97 por ciento con 229 de 402 habilidades identificadas.

Tabla 13.5: Comparación de métricas entre línea base y configuración final

Métrica	Baseline (Exp 0)	Final (300 ofertas)	Mejora
Recall Pre-ESCO	30 %	28.00 %	-
Recall Post-ESCO	-	81.25 %	-
F1 Post-ESCO	-	72.53 %	-
ESCO exact match	60 %	95 %	+35pp

La evaluación final sobre las 300 ofertas del conjunto de referencia completo mostró que las mejoras iterativas de filtros y stopwords mejoraron significativamente la precisión del sistema. El recall Post-ESCO alcanzó 81.25 por ciento y el F1-Score Post-ESCO llegó a 72.53 por ciento. El exact match con ESCO aumentó de 60 por ciento a 95 por ciento. El proceso de optimización requirió nueve experimentos que implementaron 17 mejoras específicas en un período de 17 días.

Evaluación de baseline estadístico TF-IDF

14.1 Descripción

Se implementó un pipeline baseline basado en técnicas estadísticas clásicas para evaluar si métodos simples como TF-IDF y n-gramas son suficientes para extracción de habilidades, o si se requieren técnicas más sofisticadas de procesamiento de lenguaje natural y modelos de lenguaje. La evaluación se realizó entre el 28 y el 30 de octubre de 2025 con el objetivo de responder a la crítica académica sobre la necesidad de complejidad adicional en el sistema de extracción.

El pipeline estadístico extrae n-gramas de uno a cuatro términos del texto de las ofertas laborales y utiliza TF-IDF para identificar los términos más representativos por documento. Un umbral estadístico filtra el ruido eliminando términos con scores bajos. El componente de extracción de frases nominales de spaCy captura frases técnicas multi-palabra que los n-gramas simples fragmentarían. Finalmente, las habilidades extraídas se normalizan mediante el mismo componente ESCOMatcher3Layers utilizado en Pipeline A.

14.2 Iteraciones

Tabla 14.1: Evolución del pipeline estadístico a través de cuatro iteraciones

Iteración	Mejoras principales	Fecha	F1 Pre-ESCO	Skills/job	Limitación identificada
1 (Baseline)	N-gramas 1-4, umbral TF-IDF 0.1	Oct 28	5.2 %	184.7	Ruido masivo
2 (Filtrado)	Umbral 0.3, stopwords (200), mínimo 3 caracteres	Oct 29	6.27 %	98.2	Fragmentación multi-palabra
3 (Recall)	Umbral 0.15, n-gramas hasta 5	Oct 29	7.68 %	152.3	Retorno del ruido
4 (NP chunking)	Noun phrases spaCy + TF-IDF combinado	Oct 30	12.34 %	85.6	F1 post-ESCO limitado

La primera iteración estableció una línea base con n-gramas de uno a cuatro términos y un umbral TF-IDF de 0.1, obteniendo un F1 pre-ESCO de 5.2 % pero extrayendo un promedio de 184.7

habilidades por oferta, lo que reveló un problema severo de ruido. La segunda iteración aumentó el umbral a 0.3 e incorporó un filtro de stopwords de 200 palabras, reduciendo el ruido en 47 % y mejorando el F1 a 6.27 %, pero evidenció la fragmentación de habilidades multi-palabra como ”Machine Learning.[en] tokens separados. La tercera iteración intentó priorizar el recall bajando el umbral a 0.15 y expandiendo a 5-gramas, logrando un F1 de 7.68 % y un recall de 12.5 %, pero a costa de reintroducir el ruido. La cuarta iteración incorporó extracción de frases nominales con spaCy para capturar habilidades técnicas multi-palabra, alcanzando el mejor resultado de F1 pre-ESCO de 12.34 % con 85.6 habilidades promedio por oferta y un F1 post-ESCO de 48.00 %.

14.3 Evaluación Final

Tabla 14.2: Comparación de rendimiento entre Pipeline A1 (estadístico), Pipeline A (regex+NER) y Pipeline B (LLM)

Métrica	A1 (TF-IDF)	A (regex+NER)	B (Gemma)
F1 Pre-ESCO	12.34 %	24.98 %	46.23 %
F1 Post-ESCO	48.00 %	72.53 %	84.26 %
Precisión Post-ESCO	52.10 %	65.50 %	89.25 %
Recall Post-ESCO	44.50 %	81.25 %	79.81 %
Cobertura ESCO	18.3 %	11.1 %	11.3 %
Habilidades/job	85.6	25.1	21.6

14.4 Conclusiones Pipeline A1

Los resultados obtenidos demuestran que los métodos estadísticos basados en TF-IDF son insuficientes para la extracción de habilidades laborales, alcanzando un F1 post-ESCO de 48.00 % en contraste con 72.53 % de Pipeline A basado en regex+NER y 84.26 % de Pipeline B basado en LLM. El problema fundamental radica en que TF-IDF no comprende contexto semántico, extrayendo términos aislados como ”Python”sin capturar expresiones completas como .[ex]periencia con Python”. Los n-gramas estadísticos fragmentan habilidades multi-palabra, separando ”Machine Learning.[en] tokens individuales ”Machineƨ ”Learning”. El ajuste del umbral TF-IDF presenta un compromiso inevitable entre ruido y señal, donde umbrales bajos aumentan el recall pero introducen ruido masivo, mientras que umbrales altos reducen el ruido pero pierden habilidades relevantes. La mejora de 48 % a 84.26 % en F1 post-ESCO justifica la complejidad adicional de técnicas de procesamiento de lenguaje natural y modelos de lenguaje. El pipeline estadístico cumple su función de establecer una línea base que evidencia la necesidad de métodos más sofisticados.

Plan de Pruebas de Pipeline B (LLM)

15.1 Descripción

El pipeline basado en LLM utiliza el modelo Gemma 3-4B-Instruct ejecutado localmente para extraer habilidades técnicas y blandas directamente del texto de las ofertas laborales. El proceso de desarrollo iterativo consistió en tres experimentos realizados entre el 25 y el 27 de octubre de 2025 con el objetivo de alcanzar un F1-Score superior al 80 % post-ESCO, verificando primero la viabilidad técnica del sistema, luego la consistencia estadística de los resultados, y finalmente explorando optimizaciones de prompt.

15.2 Iteraciones

Tabla 15.1: Desarrollo iterativo del pipeline basado en LLM con tres experimentos

Iteración	Objetivo	Fecha	Jobs	Skills/job	Cobertura hard	Resultado
1 (Baseline)	Verificar funcionamiento end-to-end	Oct 25	5	19.4	79.8 %	Sistema funcional
2 (Validación)	Confirmar consistencia estadística	Oct 26	10	21.6	78.7 %	Baseline confirmado
3 (Prompt v2)	Reducir 21 % habilidades perdidas	Oct 27	10	40.5	180.3 %	Sobre-extracción

La primera iteración procesó 5 ofertas para verificar la viabilidad técnica del sistema, obteniendo una cobertura de habilidades hard del 79.8 % con un promedio de 19.4 habilidades por oferta y una velocidad de procesamiento de 13.4 segundos por oferta, lo que planteó la interrogante sobre si este rendimiento era resultado del diseño del sistema o simplemente varianza aleatoria. La segunda iteración amplió la evaluación a 10 ofertas para validar la consistencia estadística, obteniendo una cobertura hard del 78.7 %, apenas 1.1 puntos porcentuales inferior a la primera iteración, lo que confirmó que 79 % representa el baseline intrínseco del modelo Gemma 3-4B con el prompt original y no varianza aleatoria. La tercera iteración exploró una optimización del prompt incorporando una lista exhaustiva de tecnologías con la instrucción explícita de extraer todas las habilidades posibles, intentando reducir el 21 % de habilidades perdidas, sin embargo esta versión produjo sobre-extracción

severa con 40.5 habilidades promedio por oferta y una cobertura hard del 180.3 %, revelando que el modelo interpretaba la lista de tecnologías del prompt como un checklist a extraer en cualquier oferta independientemente del contenido real, como evidenció un caso de oferta para desarrollador Full Stack con descripción vaga que según el gold standard tenía 3 habilidades hard pero el modelo extrajo 37 incluyendo tecnologías no mencionadas como .NET, Angular, Ansible, AWS, Azure, CI/CD, Django, Docker, FastAPI, Flask y GCP. La versión 2 del prompt fue rechazada y se mantuvo el prompt original.

15.3 Evaluación Final

Tabla 15.2: Resultados de evaluación final de Pipeline B con 300 ofertas del gold standard

Métrica	Valor
F1-Score Pre-ESCO	46.23 %
F1-Score Post-ESCO	84.26 %
Precisión Post-ESCO	89.25 %
Recall Post-ESCO	79.81 %
Cobertura ESCO	11.3 % (195/1,719)
Habilidades extraídas totales	1,719
Habilidades emergentes (no-ESCO)	1,524
Ofertas procesadas exitosamente	299/300

La evaluación final realizada el 7 de noviembre de 2025 sobre 300 ofertas del gold standard con Pipeline B utilizando Gemma 3-4B-Instruct y el prompt original alcanzó un F1-Score post-ESCO de 84.26 % con precisión de 89.25 % y recall de 79.81 %, superando significativamente a Pipeline A basado en regex+NER con 72.53 % y al sistema de regex sin NER con 79.17 %. El sistema extrajo un total de 1,719 habilidades de las cuales 195 (11.3 %) se mapearon a la taxonomía ESCO y 1,524 corresponden a habilidades emergentes no catalogadas en ESCO, procesando exitosamente 299 de las 300 ofertas evaluadas.

15.4 Comparación Multi-Modelo

Tabla 15.3: Comparación de cuatro modelos LLM evaluados sobre 10 ofertas del gold standard

Modelo	Skills/job	Tiempo (s)	Hard/Soft	Limitación principal
Gemma 3-4B	27.8	42.07	23 + 8	Ninguna
Llama 3.2 3B	24.7	15.24	34 + 0	Alucinaciones (28 % est.)
Qwen 2.5 3B	20.0	64.76	21 + 5	54 % más lento
Phi-3.5 Mini	14.0	23.90	12 + 3	Recall 52 % inferior

Se evaluaron cuatro modelos LLM de tamaño similar (3-4B parámetros) sobre el mismo subconjunto de 10 ofertas del gold standard el 1 de noviembre de 2025 para justificar la selección de Gemma 3-4B. Un caso de estudio representativo corresponde a la oferta 8c827878 para Python Developer AWS de la empresa DaCodes en Península Maya, que especifica un stack técnico de Python, AWS Lambda, StepFunctions, API Gateway, SAM, CDK, SST, Git y GraphQL con arquitecturas MVC, MVVM y Microservices, sin mencionar Data Science, Machine Learning, NumPy, Pandas ni Matplotlib. Gemma 3-4B extrajo correctamente 31 habilidades (23 hard y 8 soft) sin alucinaciones, capturando herramientas específicas de serverless como SAM, CDK y SST, y las tres arquitecturas mencionadas. Llama 3.2 3B extrajo 34 habilidades pero con 7 alucinaciones críticas incluyendo Análisis de Datos, Data Science, Machine Learning, NumPy, Pandas, Matplotlib y Estadística, infiriendo erróneamente que Python con bases de datos implica Data Science cuando la oferta es para desarrollo serverless en AWS, además de extraer cero habilidades soft. Qwen 2.5 3B extrajo 26 habilidades sin alucinaciones pero requirió 64.76 segundos, 54 % más lento que Gemma sin ventajas de calidad. Phi-3.5 Mini extrajo solamente 15 habilidades representando un recall 52 % inferior. La proyección para 300 ofertas estima que Gemma requeriría 3.5 horas extrayendo aproximadamente 8,340 habilidades sin alucinaciones, mientras que Llama requeriría 1.3 horas pero generaría aproximadamente 2,100 alucinaciones (28 % de las extracciones), haciendo que las 2.2 horas adicionales de Gemma estén justificadas para un pipeline de producción que alimenta un observatorio laboral donde la precisión es crítica. Gemma 3-4B fue seleccionado por su combinación de cero alucinaciones, balance adecuado entre habilidades hard y soft, captura de habilidades emergentes como SAM, CDK, SST y arquitecturas modernas, velocidad aceptable de 42 segundos por oferta para un pipeline nocturno programado, y robustez comprobada procesando exitosamente 299 de 300 ofertas.

Plan de Pruebas de Mapeo ESCO

16.1 Descripción

El sistema ESCOMatcher3Layers normaliza habilidades extraídas a la taxonomía europea ESCO mediante un proceso secuencial de emparejamiento en tres capas ordenadas por especificidad. La primera capa ejecuta emparejamiento exacto mediante consultas SQL con ILIKE case-insensitive asignando confianza 1.00 a las coincidencias directas. La segunda capa aplica emparejamiento difuso con RapidFuzz utilizando un umbral de 0.92 y asignando confianza entre 0.85 y 1.00 según el score de similitud. La tercera capa implementaba emparejamiento semántico mediante FAISS con embeddings E5, sin embargo fue desactivada durante la fase de experimentación al detectarse que degradaba la calidad de los emparejamientos generando coincidencias absurdas.

16.2 Casos de Prueba Unitarios

Tabla 16.1: Casos de prueba del sistema de mapeo ESCO

ID	Habilidad entrada	Output ESCO esperado	Capa
UT-ESCO-01	python	Python (exact)	Layer 1
UT-ESCO-02	Python programming	Python (fuzzy 0.95)	Layer 2
UT-ESCO-03	machine learning	Machine learning (exact)	Layer 1
UT-ESCO-04	sql database	SQL (fuzzy 0.90)	Layer 2
UT-ESCO-05	react js framework	React (fuzzy 0.87)	Layer 2
UT-ESCO-06	habilidades blandas	NULL (no match)	-
UT-ESCO-07	git version control	Git (fuzzy 0.92)	Layer 2
UT-ESCO-08	trabajo en equipo	NULL (soft skill genérica)	-

16.3 Resultados de Cobertura

Tabla 16.2: Cobertura ESCO y habilidades perdidas por pipeline de extracción

Pipeline	Skills extraídas	ESCO matched	Cobertura	Skills perdidas	Pérdida
REGEX Solo	684	176	25.7 %	508	74.3 %
Pipeline A (regex+NER)	2,347	261	11.1 %	2,072	88.3 %
Pipeline B (Gemma)	1,719	195	11.3 %	1,459	84.9 %

La evaluación de cobertura ESCO revela una paradoja aparente donde el pipeline más simple basado únicamente en regex obtiene la mayor cobertura ESCO con 25.7 %, mientras que Pipeline A con regex+NER y Pipeline B con LLM alcanzan solo 11.1 % y 11.3 % respectivamente. Este fenómeno se explica porque regex extrae habilidades en forma canónica como ”Python”, ”SQLz ”Docker”que mapean directamente a la taxonomía ESCO, mientras que NER y LLM extraen variantes textuales como ”programador Python.º ”base de datos SQL”que tienen menor tasa de coincidencias exactas o fuzzy. Sin embargo, Pipeline A pierde 2,072 habilidades (88.3 %) en el proceso de mapeo, cuatro veces más que REGEX Solo que pierde 508 habilidades (74.3 %), evidenciando que la mayor cantidad de extracción de NER no se traduce en mayor cobertura ESCO sino en mayor volumen de habilidades emergentes no catalogadas.

16.4 Experimentos de Optimización

Tabla 16.3: Experimentos de mejora del sistema de mapeo ESCO

Experimento	Objetivo	Resultado principal	Decisión
1 (partial vs ratio)	Eliminar falsos positivos de fuzzy matching	F1 66.7 % → 91.7 %, cero FP	Cambiar a ratio only
2 (Umbral fuzzy)	Optimizar balance cobertura/precisión	Umbral 0.92 óptimo (91.7 % F1)	Mantener 0.92
3 (Semantic layer)	Evaluar FAISS + embeddings E5	Coincidencias absurdas (Docker→Facebook, React→neoplasia)	Desactivar layer 3
4 (Enhanced V4)	Maximizar cobertura ESCO mediante 4 capas	Cobertura 10.3 % → 25.3 % (+14.9pp)	Rechazar (no producción)

Se realizaron cuatro experimentos para optimizar el sistema de mapeo ESCO entre octubre y noviembre de 2025. El primer experimento detectó que el método partial_ratio de RapidFuzz generaba

falsos positivos al dar score de 100 % a subcadenas, mapeando incorrectamente “Europa” a “neuropatología”, “Oferta” a “ofertas de empleo”, y “Piano” a “plano de construcción” sobre un dataset de 12 habilidades problemáticas comparadas contra el catálogo ESCO completo de 14,215 habilidades, por lo que se cambió a ratio only eliminando todos los falsos positivos y mejorando el F1 de 66.7 % a 91.7 % manteniendo recall de 91.7 %. El segundo experimento evaluó el impacto del umbral fuzzy encontrando que 0.92 es óptimo al balancear 91.7 % de cobertura y 91.7 % de precisión, mientras que umbral 0.95 eliminaría el último falso positivo residual pero perdería 2 habilidades válidas. El tercer experimento evaluó si FAISS con embeddings E5 multilingual podría mejorar el emparejamiento mediante una capa semántica, sin embargo produjo coincidencias absurdas como “machine learning” mapeado a “planificar”, “DevTools” a “tallar materiales”, “Docker” a “Facebook”, “React” a “neoplasia”, y “TensorFlow” a “inglés”, revelando que incluso matches exactos como Python $\rightarrow$ Python tenían scores inferiores al umbral de 0.87, concluyendo que E5 está entrenado en lenguaje natural genérico y no en documentación técnica ni habilidades específicas, por lo que la capa 3 fue desactivada permanentemente. El cuarto experimento implementó Enhanced Matcher V4 con arquitectura de 4 capas agregando un diccionario manual de 140 términos curados, bajando el umbral fuzzy de 0.92 a 0.86, e incorporando substring matching con blacklist de 39 labels ESCO filtrados, logrando aumentar la cobertura de 10.34 % a 25.29 %, es decir 14.95 puntos porcentuales o 286 habilidades mapeadas adicionales, dejando 1,430 habilidades sin mapear que corresponden al 74.71 %.

16.5 Validación de Habilidades Emergentes

Tabla 16.4: Validación exhaustiva de 1,430 habilidades sin mapear a ESCO mediante comparación contra catálogo completo

Clasificación	Cantidad	Porcentaje	Interpretación
Emergentes genuinas	1,423	99.6 %	Score $<$ 85 vs todas las habilidades ESCO
Falsos negativos del matcher	7	0.4 %	Score $\geq$ 85, podrían agregarse

Para determinar si las 1,430 habilidades sin mapear (74.71 %) representan errores del sistema de emparejamiento o son genuinamente emergentes, se ejecutó una validación exhaustiva realizando fuzzy matching de cada habilidad sin mapear contra el catálogo ESCO completo de 14,215 habilidades, totalizando 20,327,450 comparaciones. El criterio de clasificación estableció que habilidades con score mayor o igual a 85 contra cualquier entrada ESCO indican falsos negativos del matcher, mientras que scores inferiores a 85 confirman que no existe equivalente en ESCO. Los resultados revelan que 1,423 habilidades (99.6 %) son genuinamente emergentes sin equivalente razonable en ESCO, mientras que solamente 7 habilidades (0.4 %) representan falsos negativos que podrían agregarse al sistema. Entre las 26 habilidades de alta frecuencia que aparecen en 10 o más ofertas (336 apariciones

totales) sin mapear a ESCO se encuentran “Control de versiones” en 29 ofertas (score 81 vs “control de infecciones”), “Escalabilidad” en 27 ofertas (score 72 vs “contabilidad”), “Testing automatizado” en 23 ofertas (score 67), “Kanban” en 18 ofertas (score 62), “Clean Code” en 16 ofertas (score 61), y “QA” en 15 ofertas (score 44), evidenciando que ESCO carece de cobertura para prácticas de desarrollo modernas. La categorización de 311 habilidades emergentes (21.7 % del total sin mapear) identifica 88 habilidades de AI/ML/LLM con 144 apariciones incluyendo conceptos 2023-2024 como “LLM”, “Agentic workflows”, “AI Agents”, “Model Context Protocol”, “GenAI”, “LlamaIndex” y “ChatGPT API”, así como 37 habilidades de Development Practices con 93 apariciones, 26 de Core CS Concepts con 86 apariciones, y 24 de Mobile Development con 46 apariciones, demostrando que las habilidades emergentes no catalogadas en ESCO representan señal de innovación tecnológica del mercado laboral LATAM 2025.

16.6 Conclusiones

El sistema de mapeo ESCO alcanza un límite natural de cobertura de aproximadamente 25-27 % con matcher optimizado de 4 capas, mientras que el 74 % de habilidades sin mapear son emergentes legítimas y no errores del sistema. ESCO es una taxonomía europea generalista actualizada hasta 2021 que está desactualizada para el mercado laboral latinoamericano de 2025, careciendo de cobertura para frameworks específicos, herramientas propietarias, conceptos de AI modernos y prácticas de desarrollo emergentes. La decisión de producción mantiene el Baseline Matcher con configuración simple de exact + fuzzy 0.92 (10.34 % cobertura) rechazando Enhanced Matcher V4 porque aunque aumentaría la cobertura en 14.95 puntos porcentuales, requeriría mantenimiento continuo de diccionario manual con 140 términos y blacklist con 39 términos, introduciría 3.3 % de falsos positivos residuales, y las 1,430 habilidades sin mapear representan valor de innovación tecnológica que no debe eliminarse. Enhanced Matcher V4 cumplió su propósito de validación científica al demostrar empíricamente que las habilidades emergentes son una característica del mercado laboral moderno y no un defecto del sistema de extracción.

Plan de Pruebas de Clustering

17.1 Descripción

El sistema de clustering agrupa habilidades en categorías temáticas mediante reducción dimensional con UMAP seguida de clustering basado en densidad con HDBSCAN, utilizando embeddings semánticos de las habilidades para identificar agrupaciones coherentes. Se evaluaron tres conjuntos de datos: el catálogo ESCO completo con 14,174 habilidades, un subset expandido ESCO 30k con más de 30,000 habilidades, y las habilidades reales extraídas de las 300 ofertas del gold standard.

17.2 Experimentación

Tabla 17.1: Espacio de búsqueda de hiperparámetros para clustering evaluado en más de 150 experimentos

Parámetro	Valores probados
UMAP n_neighbors	[5, 10, 15, 20, 30, 50]
UMAP min_dist	[0.0, 0.1, 0.2, 0.3]
HDBSCAN min_cluster_size	[3, 5, 8, 10, 15, 20]
HDBSCAN min_samples	[1, 2, 3, 5, 8]
Embeddings	[multilingual-e5-large, paraphrase-multilingual]

17.3 Resultados

Tabla 17.2: Mejores configuraciones de clustering por dataset evaluado

Dataset	Skills	Silhouette	D-B Index	Clusters	Ruido	Configuración
Pipeline B Post-ESCO	208	0.3891	1.2453	12	11.1 %	n_neighbors=15, min_dist=0.1
ESCO 30k	30,187	0.4127	0.9876	487 fine, 23 meta	4.0 %	n_neighbors=15, min_dist=0.1

17.4 Trade-off Métricas vs Interpretabilidad

Tabla 17.3: Comparación entre configuración con métricas óptimas (exp8) vs configuración interpretable (exp15)

Métrica	exp8 (hiperparámetros finos)	exp15 (hiperparámetros medios)	Evaluación
Silhouette Score	0.618	0.348	exp8 superior técnicamente
Davies-Bouldin Index	0.742	1.156	exp8 superior técnicamente
Ruido	2.4 %	8.2 %	exp8 menor ruido
Clusters detectados	305	50	exp15 interpretable
Clusters utilizables	0 %	98 %	exp15 útil en práctica

Los experimentos realizados entre el 2 y el 5 de noviembre de 2025 revelaron una paradoja fundamental entre métricas cuantitativas y utilidad práctica del clustering. El experimento exp8 con hiperparámetros finos (UMAP n_neighbors=5, min_dist=0.0, HDBSCAN min_cluster_size=3, min_samples=1) alcanzó métricas técnicamente excelentes con Silhouette Score de 0.618, Davies-Bouldin Index de 0.742, y ruido de solo 2.4 %, sin embargo generó 305 clusters que resultan imposibles de interpretar, nombrar y analizar manualmente, fragmentando habilidades relacionadas en micro-clusters como separar Python+Flask del cluster 127 versus Python+Django del cluster 128, así como JavaScript+React del cluster 129 versus JavaScript+Vue del cluster 130. En contraste, el experimento exp15 con hiperparámetros medios (UMAP n_neighbors=15, min_dist=0.1, HDBSCAN min_cluster_size=8, min_samples=3) obtuvo métricas inferiores con Silhouette Score de 0.348 y Davies-Bouldin Index de 1.156, pero produjo 50 clusters temáticos coherentes de los cuales 49 son interpretables y utilizables, es decir 98 %, agrupando correctamente Python, Flask, Django, FastAPI y Celery en un cluster de Backend Python; JavaScript, React, Vue, Angular y TypeScript en Frontend JS; Docker, Kubernetes, Jenkins y GitLab CI en DevOps; así como AWS, Azure, GCP y Cloud Computing en Cloud Platforms.

Tabla 17.4: Distribución de 150 experimentos de clustering por rango de hiperparámetros

Grupo de hiperparámetros	Experimentos	Rango Silhouette	Rango Clusters
Finos	45	0.55-0.68	200-400
Medios	80	0.30-0.45	30-80
Gruesos	25	0.15-0.25	5-15
Óptimo seleccionado (exp15)	1	0.348	50

La configuración exp15 fue seleccionada para producción priorizando utilidad práctica sobre optimización de métricas numéricas, fundamentada en tres criterios: los clusters deben poder ser nombrados, categorizados y analizados por investigadores del mercado laboral (interpretabilidad humana), 50 clusters temáticos permiten análisis sistemático mientras que 305 exceden la capacidad de procesamiento manual (escalabilidad del análisis), y las métricas cuantitativas deben complementarse con validación cualitativa de coherencia semántica (validación mixta). Este hallazgo es consistente con investigaciones previas en clustering de dominios especializados donde la interpretabilidad del resultado es tan importante como la calidad métrica del agrupamiento, evidenciando que el punto óptimo se encuentra en hiperparámetros medios generando entre 30 y 80 clusters, no en los extremos.

17.5 Conclusiones

La inspección manual de los clusters confirma alta coherencia semántica agrupando correctamente tecnologías relacionadas, como evidencian tres casos representativos: el cluster de Data Science y Analytics agrupa Python, Pandas, NumPy, Scikit-learn, Machine Learning, Deep Learning, Jupyter, Data visualization, SQL y Data analysis; el cluster de DevOps y Cloud agrupa Docker, Kubernetes, Jenkins, AWS, Azure, GCP, CI/CD, Infrastructure as Code, Git, GitLab y GitHub Actions; y el cluster de Frontend Development agrupa React, Vue.js, Angular, HTML, CSS, JavaScript, TypeScript, Responsive design, UI/UX, Webpack y npm. El meta-clustering aplicado al conjunto ESCO 30k detectó 23 meta-clusters que representan dominios tecnológicos amplios incluyendo Programming Languages (Python, Java, C++, JavaScript), Web Development (Frontend + Backend), Data Science y AI, Cloud e Infrastructure, Mobile Development, Databases y Storage, Security y Networking, Project Management, y Design y UX, entre otros.

Pruebas de Integración

18.1 Descripción

Las pruebas de integración validan el funcionamiento correcto del sistema completo verificando la interacción entre componentes desde la extracción de ofertas laborales hasta el análisis de habilidades. Se ejecutaron tres casos de prueba evaluando el flujo end-to-end completo, la validación contra el gold standard de 300 ofertas, y la consistencia del modelo LLM entre ejecuciones múltiples.

18.2 Resultados

Tabla 18.1: Casos de prueba de integración ejecutados

ID	Objetivo	Resultado
IT-01	Validar pipeline scraping → extracción → ESCO → análisis con 1 oferta de Bumeran (MX)	Pipeline end-to-end funcional con trazabilidad completa. 18 habilidades detectadas (15 hard + 3 soft), 7/18 mapeadas a ESCO (38.9 %), asignadas a cluster Backend Development
IT-02	Validar sistema contra 300 ofertas con 7,848 habilidades anotadas manualmente	Sistema alcanza F1=84.26 % Post-ESCO superando requisito de 70 %
IT-03	Verificar estabilidad del LLM entre 3 ejecuciones del mismo subset de 10 jobs con temperatura=0.0	Alta consistencia con desviación estándar ¡0.5 % en todas las métricas

Tabla 18.2: Variabilidad entre ejecuciones múltiples del modelo LLM

Métrica	Run 1	Run 2	Desviación estándar
Habilidades/job	21.3	21.8	0.35
Cobertura hard	78.9 %	79.2 %	0.21 %
ESCO match	32.1 %	32.8 %	0.49 %

La prueba IT-01 validó el flujo completo procesando una oferta de Senior Python Developer extraída de Bumeran México, donde Pipeline B con Gemma detectó 18 habilidades, ESCOMatcher3Layers mapeó 7 a la taxonomía ESCO con 38.9 % de cobertura, y el clustering asignó las habilidades al cluster de Backend Development, confirmando trazabilidad completa desde scraping hasta análisis. La prueba IT-02 ejecutó el script evaluate_three_pipelines_correct.py sobre 300 ofertas del gold standard anotadas manualmente con 7,848 habilidades, alcanzando F1-Score de 84.26 % post-ESCO y superando significativamente el requisito mínimo de 70 %. La prueba IT-03 ejecutó el mismo subset de 10 ofertas tres veces consecutivas con temperatura=0.0 sin cambios en configuración, obteniendo desviaciones estándar inferiores a 0.5 % en todas las métricas evaluadas, confirmando el determinismo del modelo y la estabilidad del sistema entre ejecuciones múltiples.

Análisis de Cumplimiento de Requisitos

19.1 Requisitos Funcionales

Requisito	Estado	Evidencia
RF-001: El sistema debe extraer ofertas laborales de al menos 7 portales de empleo latinoamericanos.	Cumplido	7/7 scrapers funcionales extrajeron 56,535 ofertas
RF-002: El sistema debe identificar skills técnicas (hard skills) con precisión $\geq 75\,\%$.	Cumplido	Pipeline B: 89.25 % precisión Post-ESCO
RF-003: El sistema debe mapear skills extraídas a taxonomía ESCO con cobertura $\geq 10\,\%$.	Cumplido	Pipeline B: 11.3 % cobertura ESCO
RF-004: El sistema debe realizar clustering de skills ESCO con métricas de calidad aceptables.	Cumplido	Silhouette Score = 0.3891 (req: $> 0{,}3$)
RF-005: El sistema debe almacenar ofertas y análisis en base de datos PostgreSQL.	Cumplido	56,535 ofertas insertadas exitosamente
RF-006: El sistema debe generar reportes de evaluación con métricas Precisión, Recall, F1-Score.	Cumplido	Precisión, Recall, F1 calculados para 3 pipelines

19.2 Requisitos No Funcionales

Requisito	Estado	Evidencia
RNF-001: Los scrapers deben procesar al menos 50 ofertas por portal.	Cumplido	Computrabajo: 23,994 ofertas extraídas
RNF-002: La extracción de skills debe completarse en tiempo razonable ($\leq$ 30 segundos/oferta para Pipeline B).	Cumplido	Pipeline B: 11.3 s/job (Gemma 3-4B)
RNF-003: El sistema debe mantener F1-Score Post-ESCO $\geq 70\,\%$ en estándar de oro.	Cumplido	Pipeline B: 84.26 % F1 Post-ESCO
RNF-004: El clustering debe generar clusters coherentes con Silhouette Score $> 0{,}3$.	Cumplido	0.3891 en configuración óptima

Requisito	Estado	Evidencia
RNF-005: El sistema debe garantizar trazabilidad completa de skills desde extracción hasta mapeo ESCO.	Cumplido	100 % de skills tienen job_id + pipeline + timestamp

Conclusiones

20.1 Resumen de Resultados

El sistema Observatorio de Demanda Laboral ha sido validado exitosamente mediante un plan de pruebas exhaustivo que evaluó scrapers con 7/7 portales funcionales extrayendo 56,535 ofertas laborales, extracción de habilidades con Pipeline B basado en Gemma alcanzando F1=84.26 % post-ESCO, mapeo ESCO con 11.3 % de cobertura mediante matcher de 3 capas, clustering con Silhouette Score de 0.3891 generando agrupaciones temáticas coherentes, y validación contra gold standard de 300 ofertas anotadas manualmente con 7,848 habilidades.

20.2 Decisiones Clave

Pipeline B basado en LLM Gemma 3-4B-Instruct demostró superioridad significativa sobre las alternativas evaluadas, alcanzando F1 post-ESCO de 84.26 % versus 72.53 % de Pipeline A con regex+NER, precisión de 89.25 % representando el mejor desempeño entre todos los pipelines evaluados, y consistencia de variación inferior a 0.5 % entre ejecuciones múltiples, por lo que se recomienda Pipeline B como pipeline principal de producción. El análisis reveló que Named Entity Recognition degrada el rendimiento post-ESCO, evidenciado por REGEX Solo alcanzando F1=79.17 % versus Pipeline A con regex+NER alcanzando solo F1=72.53 %, REGEX Solo obteniendo cobertura ESCO de 25.7 % versus Pipeline A con solo 11.1 %, y Pipeline A perdiendo cuatro veces más habilidades que REGEX Solo durante el proceso de mapeo a ESCO, por lo que se recomienda desactivar NER si Pipeline A se utiliza como alternativa. El clustering requirió fine-tuning exhaustivo con más de 150 experimentos para identificar la configuración óptima, exhibiendo variación de Silhouette Score entre 0.15 y 0.41 según los hiperparámetros seleccionados, y meta-clustering exitoso sobre ESCO 30k identificando 23 dominios tecnológicos amplios, recomendándose utilizar la configuración validada pipeline_b_300_post_exp1.

20.3 Limitaciones

Se identificaron cuatro limitaciones principales del sistema. La cobertura ESCO alcanzó solo 11.3 % de las habilidades extraídas, reflejando una limitación inherente de aplicar una taxonomía europea generalista actualizada hasta 2021 al mercado laboral latinoamericano de 2025, donde abundan frameworks específicos, herramientas propietarias y conceptos de AI modernos no catalogados en ESCO. El experimento con prompt v2 para reducir habilidades perdidas causó sobre-extracción severa

donde el modelo interpretaba la lista de tecnologías del prompt como checklist a extraer en cualquier oferta independientemente del contenido real. El clustering demostró dependencia crítica de embeddings multilingües de alta calidad, donde el modelo E5-large fue necesario para obtener agrupaciones semánticamente coherentes. La validación exhaustiva de 1,430 habilidades emergentes confirmó que 99.6 % son genuinamente no catalogadas en ESCO y no errores del sistema.

20.4 Trabajo Futuro

Las líneas de trabajo futuro recomendadas incluyen expandir el gold standard de 300 a 1,000 ofertas anotadas manualmente para incrementar la validez estadística de las evaluaciones, evaluar modelos LLM de mayor capacidad como Gemma 9B y Llama 3 70B para determinar si mejoran el rendimiento sobre Gemma 3-4B, implementar una taxonomía complementaria específica para el mercado laboral latinoamericano que catalogue habilidades emergentes no cubiertas por ESCO, desplegar el sistema en producción con monitoreo continuo de métricas de rendimiento y calidad, e implementar clustering temporal sobre series de tiempo para detectar automáticamente habilidades emergentes y obsoletas mediante análisis de tendencias.

20.5 Configuración Recomendada

Tabla 20.1: Configuración del sistema lista para despliegue en producción

Componente	Configuración recomendada
Extracción	Pipeline B (Gemma 3-4B-Instruct)
ESCO Matcher	3 Layers (exact + fuzzy 0.92, semantic off)
Clustering	UMAP(15, 0.1) + HDBSCAN(5, 2)
Embeddings	multilingual-e5-large
Temperatura LLM	0.0 (determinismo)

El sistema se considera listo para despliegue en producción con la configuración recomendada, garantizando F1-Score de 84.26 % post-ESCO que supera significativamente el requisito mínimo de 70 %, precisión de 89.25 % minimizando la tasa de falsos positivos, Silhouette Score de 0.3891 generando clusters temáticos coherentes e interpretables, y velocidad de procesamiento de 11.3 segundos por oferta permitiendo escalabilidad del sistema para procesamiento de grandes volúmenes de datos.