Nautilus
Plataforma de Monitoreo de Precios Competitivos

Hoja de Ruta MVP de 3-4 Meses

"Explorando las profundidades del comercio electrónico"

Versión:	1.0 (Propuesta Técnica Completa)
Fecha:	Noviembre 2025
Objetivo:	Construir MVP listo para producción en 3-4 meses
Alcance:	3-4 competidores, 1,000-5,000 SKUs, monitoreo diario
Métricas:	95 %+ frescura, 97 %+ precisión, 85 %+ cobertura SKU

Stack Tecnológico: Playwright + Prefect + Polars + DuckDB + FastAPI + React

Índice

Resumen Ejecutivo

Objetivo: Construir Nautilus, plataforma de monitoreo de precios competitivos lista para producción en 3-4 meses.

Proponemos un MVP usando stack tecnológico probado: Playwright para automatización de navegadores, Prefect para orquestación y estrategias de proxy optimizadas. El enfoque híbrido combina servicios de scraping administrados con coincidencia de SKU y analítica personalizadas, equilibrando velocidad, costo y capacidad.

Resultados esperados: Monitoreo diario de 3-4 competidores en 2-3 categorías de productos con 95 %+ frescura, 97 %+ precisión y 85 %+ cobertura SKU.

Métricas Clave del Proyecto

- **Cronograma:** 16 semanas hasta despliegue
- **Presupuesto MVP:** $90-170K USD
- **Costos Operacionales:** $600-900/mes
- **Equipo:** 3.75 FTE
- **Ahorro:** 50-80 % vs plataformas comerciales
- **Stack:** Playwright, Prefect, Polars, DuckDB, FastAPI, React

1 Panorama de Plataformas: Comercial vs Open Source

El mercado de monitoreo de precios competitivos ofrece tanto plataformas comerciales llave en mano como herramientas flexibles de código abierto, cada una adecuada para diferentes necesidades organizacionales y plazos.

1.1 Plataformas Comerciales

Plataformas de nivel empresarial como Competera, Intelligence Node y DataWeave proporcionan soluciones integrales con garantías de precisión de datos del 95-99 % y eliminan la complejidad técnica. La plataforma impulsada por IA de Competera logra 98 % de precisión SLA y afirma un potencial de mejora de margen del 6 %, mientras que DataWeave sobresale en coincidencia de SKU con 95 %+ de precisión usando extracción de atributos basada en LLM. Intelligence Node monitorea el conjunto de datos de precios más grande del mundo y sirve a grandes minoristas como Macy's y Lenovo. Estas plataformas típicamente cuestan $750-$5,000+ mensuales pero reducen el tiempo de desarrollo a semanas en lugar de meses.

Soluciones de mercado medio como Prisync ofrecen puntos de entrada más accesibles a $99-$799 mensuales para 100-5,000 productos, proporcionando seguimiento ilimitado de competidores con 3 actualizaciones diarias y motores de precios dinámicos. Minderest, el líder europeo, logra 99 %+ de calidad de datos y sirve a 11 de los 50 minoristas más grandes del mundo. Para empresas que priorizan velocidad al mercado y carecen de recursos técnicos profundos, estas plataformas proporcionan el camino más rápido hacia inteligencia competitiva.

1.2 Frameworks Open Source

Automatización de navegadores: Playwright es el estándar de producción, con soporte multi-navegador y capacidades anti-detección. Sus mecanismos de auto-espera e interceptación de red lo hacen ideal para sitios de e-commerce con JavaScript dinámico.

Scrapy sigue siendo la fundación para extracción de contenido estático, ofreciendo arquitectura asíncrona madura y manejando millones de páginas eficientemente. Para el MVP de Coppel dirigido a 3-4 competidores, **combinar Playwright para sitios protegidos (20 % de objetivos) con solicitudes HTTP más ligeras para sitios no protegidos (80 %) optimiza tanto costo como rendimiento**. Este enfoque híbrido reduce costos de infraestructura en 60-70 % comparado con scraping solo con navegadores.

Orquestación: Prefect es la elección recomendada para el MVP por su configuración rápida (<30 min), rendimiento superior y diseño Python-first. Apache Airflow requiere mayor complejidad de configuración.

1.3 Servicios de Scraping Administrados

Proveedores premium (Bright Data, Oxylabs) gestionan infraestructura completa: proxies residenciales, resolución de CAPTCHA y renderizado JavaScript. Costos: $500-$5,000/mes con tasas de éxito del 95-99 %.

Alternativas económicas (ScraperAPI, ScrapingBee) ofrecen acceso basado en API a $49-$599/mes, manejando rotación de proxies y anti-bot automáticamente.

Recomendación MVP: Usar ScraperAPI o ScrapingBee para sitios desafiantes y construir scrapers simples internamente. Costo estimado: $200-$800/mes con tasa de éxito del 95 %+.

2 Arquitectura Técnica del Sistema

Las plataformas exitosas de monitoreo de precios siguen una arquitectura de microservicios con separación clara entre capas de extracción, procesamiento, almacenamiento y presentación, habilitando escalamiento independiente y aislamiento de fallas.

2.1 Arquitectura de Microservicios

La **arquitectura recomendada** se organiza alrededor de cinco servicios centrales:

- **Orquestación:** Programación de flujos de trabajo
- **Extracción:** Workers de automatización de navegadores
- **Procesamiento:** Transformación de datos
- **Almacenamiento:** Capa de persistencia
- **Gestión de proxies:** Rotación de IP con verificaciones de salud

Esta separación permite que el servicio de orquestación se ejecute continuamente mientras los workers de extracción escalan de cero a docenas basados en la profundidad de la cola, optimizando costos de infraestructura.

Las instancias de Playwright en contenedores desplegadas vía Docker habilitan escalamiento horizontal, con cada contenedor manejando 5-10 contextos de navegador concurrentes. RabbitMQ distribuye trabajos de scraping a través del pool de workers, proporcionando encolado de tareas confiable con manejo de dead letter para intentos fallidos. Redis cachea datos accedidos frecuentemente y gestiona limitación de tasa, mientras que PostgreSQL almacena metadatos operacionales (estado de trabajos, catálogos de productos, reglas de alertas).

Los datos scrapeados en crudo fluyen a S3 en formato Parquet, proporcionando almacenamiento columnar que comprime archivos CSV de 123MB a 10-20MB mientras habilita consultas analíticas rápidas.

2.2 El pipeline de datos sigue principios de arquitectura lambda

El **enfoque de doble camino** separa procesamiento por lotes (scrapes diarios completos) del procesamiento de flujo (cambios de precio en tiempo real que exceden umbrales). Los datos scrapeados aterrizan en S3 particionados por fecha, competidor y categoría, habilitando consulta eficiente de períodos de tiempo específicos sin escanear el conjunto de datos completo.

DuckDB proporciona analítica SQL directamente sobre archivos Parquet sin requerir ETL, logrando rendimiento de consultas sub-segundo en millones de registros. Polars maneja transformaciones de datos 10-50x más rápido que Pandas a través de su núcleo Rust multi-hilo, haciéndolo ideal para procesar scrapes diarios de miles de productos.

Para la escala inicial de Coppel (3-4 competidores, 2-3 categorías, probablemente 1,000-5,000 SKUs), Polars elimina la complejidad de gestión de clusters Spark mientras proporciona rendimiento suficiente. Apache Spark se vuelve necesario solo al escalar más allá de 50,000 SKUs o requerir streaming en tiempo real con latencia sub-minuto.

2.3 Stack tecnológico optimizado para entrega de 3-4 meses

2.3.1 Browser automation: Playwright

Playwright proporciona la ejecución más rápida, arquitectura moderna y herramientas de depuración listas para producción. Su soporte multi-lenguaje (JavaScript, Python, Java, .NET) ofrece flexibilidad, aunque la integración Python se adapta a las probables habilidades del equipo de datos de Coppel.

Consideración de costos: Cada instancia de navegador requiere 2-4GB RAM; presupuestar 1-2 VMs más grandes (4 vCPU, 8GB RAM) para scraping basado en navegador mientras se usan solicitudes HTTP ligeras en micro instancias ($5-10 mensuales) para sitios no protegidos reduce costos de infraestructura en 70 %.

2.3.2 Job orchestration: Prefect

El diseño Python-first, UI moderna y configuración rápida de Prefect lo hacen **la elección clara de MVP** sobre la complejidad de Airflow. El nivel gratuito de Prefect Cloud maneja 20,000 ejecuciones de tareas mensuales, suficiente para la escala inicial de Coppel. El servicio de orquestación define flujos de trabajo como código, habilitando control de versiones y pruebas automatizadas de lógica de scraping.

2.3.3 Data processing: Polars + DuckDB

La combinación Polars + DuckDB elimina infraestructura pesada mientras proporciona ganancias de rendimiento de 10-50x sobre herramientas Python tradicionales. Polars maneja transformaciones en memoria (limpieza, normalización, coincidencia de SKU), mientras que DuckDB habilita analítica SQL en el data lake Parquet sin mover datos. Este enfoque difiere migraciones de bases de datos costosas hasta que sea verdaderamente necesario.

2.3.4 Storage architecture

Parquet en S3 sirve como la **fundación del data lake**, almacenando snapshots históricos completos a $0.023 por GB mensual. PostgreSQL mantiene datos operacionales (cuentas de usuario, metadatos de trabajos, catálogo de productos, reglas de alertas), mientras que Redis cachea datos calientes y gestiona características en tiempo real (actualizaciones de dashboard, alertas de umbral).

Este enfoque de tres niveles separa cargas de trabajo analíticas (DuckDB consultando Parquet) de cargas de trabajo operacionales (transacciones PostgreSQL), optimizando tanto costo como rendimiento.

2.3.5 Backend API: FastAPI

FastAPI entrega desarrollo 2x más rápido que Django para aplicaciones enfocadas en API, logrando 21,000+ solicitudes por segundo versus 3,500 de Django. Su generación automática de documentación OpenAPI acelera la integración frontend, mientras que el soporte async nativo maneja solicitudes de scraping concurrentes eficientemente. La capa API expone endpoints RESTful para disparar scrapes, consultar datos de precios, gestionar alertas y exportar reportes.

2.3.6 Frontend dashboard: React

React con Recharts proporciona la arquitectura basada en componentes necesaria para interfaces de filtrado complejas y visualizaciones interactivas de series de tiempo. Recharts logra el equilibrio óptimo para MVPs: menor complejidad que D3.js, suficiente personalización para dashboards profesionales y fuerte soporte TypeScript.

2.4 Costos estimados de infraestructura y asignación de equipo

Costos operacionales mensuales para la escala MVP de Coppel totalizan $600-$900:

- Infraestructura: $142 (VMs, Redis, base de datos)
- Proxies: $397 (mezcla optimizada datacenter/residencial)
- Servicios: $80 (resolución CAPTCHA y monitoreo)

Esto representa 50-80 % de ahorro versus externalizar a servicios de scraping administrados a $1,500-$3,000 mensuales para volumen equivalente.

Asignación de equipo a través de los 3.75 FTE:

- 1.5 FTE ingenieros backend (FastAPI, Prefect, base de datos)
- 1.0 FTE especialista en scraping (Playwright, anti-detección)
- 0.75 FTE desarrollador frontend (dashboard React)
- 0.5 FTE DevOps (infraestructura, monitoreo)

El rol PM coordina a través de estos flujos, gestionando requerimientos de stakeholders y priorizando el conjunto de características MVP.

3 Manejo de Medidas Anti-Bot y Mantenimiento de Confiabilidad

Los sitios modernos de e-commerce despliegan tecnologías sofisticadas anti-bot que requieren contramedidas multi-capa para mantener confiabilidad de scraping en tasas de éxito del 95 %+.

3.1 La detección anti-bot opera a través de múltiples capas de fingerprinting

Plataformas sofisticadas como Cloudflare Bot Management, DataDome y PerimeterX combinan:

- **Detección del lado del servidor:** Reputación IP, fingerprinting TLS, patrones HTTP/2, análisis de encabezados
- **Verificación del lado del cliente:** APIs de navegador, fingerprinting canvas, WebGL, seguimiento de eventos

Estos sistemas logran 95 %+ de precisión en distinguir navegadores automatizados de humanos, haciendo scrapers Selenium ingenuos detectables en segundos.

3.2 El enfoque de bypass en capas equilibra costo y efectividad

Comience con los métodos más baratos y escale solo cuando esté bloqueado:

1. Google Cache (gratis pero sin frescura)
2. Navegadores headless fortificados (Puppeteer stealth, undetected-chromedriver)
3. Solucionadores open-source (FlareSolverr, cloudscraper)
4. Servicios de solución premium ($1,000-$5,000/millón páginas)

La perspectiva crítica: Reserve soluciones costosas para el 15-20 % de sitios competidores con protección sofisticada, usando solicitudes HTTP ligeras o automatización de navegador básica para el 80 % restante.

3.3 La estrategia de proxy sigue la escalera de costos

El principio de escalera de proxy minimiza costos al subir solo cuando sea necesario:

Nivel	Costo
Sin proxy	Gratis
Proxies datacenter	$0.10-$0.50/GB
Proxies residenciales	$1-$8/GB
Proxies móviles	$8-$40/GB
Desbloqueadores premium	$1,000-$5,000/millón req

Para scraping diario de 100,000 productos, una mezcla optimizada de 80 % datacenter y 20 % residencial cuesta aproximadamente $111 mensuales versus $375 para todo residencial.

Proveedores de proxy recomendados basados en benchmarks 2025:

- **Decodo** (Smartproxy): Mejor valor a 99.86 % éxito y $3.50-$1.50/GB
- **Oxylabs**: Premium (99.82 % éxito, 0.41s respuesta) a $4-$2/GB
- **SOAX/Bright Data**: Pools más grandes (155M+ IPs) a $4-$5/GB

3.4 La ingeniería de confiabilidad previene fallas silenciosas

Sistemas de respaldo multi-nivel cuando una solicitud falla:

1. Reintento simple con el mismo método

2. Cambiar a proxy residencial
3. Escalar a navegador headless con stealth
4. Involucrar servicio desbloqueador premium
5. Encolar para intervención manual

Monitoreo de tasas de éxito en tiempo real: Objetivo >95 % con alertas disparando debajo de 90 %. Rastrear tasas de éxito separadamente por sitio competidor, tipo de proxy y método de scraping para identificar patrones específicos de falla.

Validación de calidad de datos implementa verificaciones multi-capa:

- Estructural (todos los campos requeridos presentes)
- Validación de tipo (precios numéricos, fechas válidas)
- Verificación de rango (precios dentro de límites razonables)
- Consistencia (comparar con scrapes previos, señalar cambios >50 %)
- Completitud (>98 % de productos esperados capturados)

4 Coincidencia de SKU: Algoritmos y Enfoques de Implementación

La coincidencia precisa de SKU forma la fundación del monitoreo de precios competitivos, ya que productos mal emparejados hacen comparaciones de precios sin sentido. Lograr 95 %+ de precisión de coincidencia requiere un enfoque por niveles combinando identificadores exactos, coincidencia difusa de cadenas y puntuación de confianza.

4.1 La coincidencia por niveles equilibra precisión con cobertura

4.1.1 Tier 1: Coincidencia exacta

Usando identificadores estandarizados (EAN-13, UPC-A, GTIN, MPN, ISBN) proporciona 100 % de precisión donde esté disponible. Las búsquedas directas de base de datos en identificadores normalizados—eliminando guiones y espacios, validando dígitos de verificación, convirtiendo entre formatos—emparejan aproximadamente 60-70 % de productos con datos limpios.

4.1.2 Tier 2: Coincidencia difusa

Aborda el 30-40 % de productos de e-commerce que carecen de identificadores estandarizados. **RapidFuzz** emerge como la librería recomendada para implementación MVP, entregando rendimiento 5-10x más rápido que FuzzyWuzzy con licencia MIT y mantenimiento activo.

Estrategia de implementación combina múltiples señales con puntuación ponderada:

- Coincidencia de marca: 25 % peso
- Número de modelo: 30 % peso
- Nombre de producto: 25 % peso
- Proximidad de precio: 10 % peso
- Alineación de categoría: 10 % peso

Productos puntuando arriba de 85 disparan aprobación automática, 70-85 señalan para revisión manual y debajo de 70 rechazan como no coincidencias.

4.2 Manejar identificadores faltantes requiere estrategias de respaldo

La cascada de prioridad intenta múltiples métodos en secuencia:

1. Coincidencia directa EAN/UPC/GTIN
2. Coincidencia combinada MPN + Marca
3. Coincidencia difusa Marca + Modelo + Atributos Clave
4. Solo Nombre de Producto con umbral alto (90+)
5. Señalar para revisión manual

Técnicas de bloqueo optimizan rendimiento reduciendo complejidad de comparación de $O(n^2)$ a niveles manejables. Solo comparar productos dentro de la misma categoría y prefijo de marca, reduciendo 1 millón $\times$ 1 millón = 1 billón de comparaciones a aproximadamente 1 mil millones (reducción 1,000x).

4.3 El cronograma de implementación práctica

Período	Actividades
Semanas 1-2	Configuración pipeline, coincidencia exacta EAN/UPC/GTIN
Semanas 3-4	Instalación RapidFuzz, coincidencia difusa básica
Semanas 5-6	Puntuación multi-factor, bloqueo, manejo identificadores faltantes
Semanas 7-8	Interfaz revisión manual, batch processing, optimización
Semanas 9-12	Pruebas, ajuste umbrales, medición precisión/recall

Métricas de éxito para producción:

- 80 %+ tasa de coincidencia (productos emparejados exitosamente)
- 95 %+ precisión (coincidencias auto-aprobadas son correctas)
- <15 % tasa de revisión manual (intervención humana necesaria)
- <100ms por coincidencia de producto (objetivo de rendimiento)

5 Optimización de Costos para Rotación de IP y Conectividad

La gestión estratégica de proxies y optimización de infraestructura reducen costos operacionales en 60-85 % mientras mantienen tasas de éxito requeridas y frescura de datos.

5.1 Los costos de infraestructura escalan con complejidad

Comparación de proveedores cloud para la arquitectura recomendada:

Proveedor	Costo Mensual
AWS EC2	$108
DigitalOcean	$96
Vultr	$90
Hetzner (bare metal)	$50-200

El premium de costo de navegador: Scraping basado en navegador consume 2MB promedio por página versus 50-150KB para solicitudes HTTP—una diferencia de 10-40x. Usar navegadores exclusivamente para el 15-20 % de sitios que requieren JavaScript ahorra 60-80 % en costos de ancho de banda de proxy.

5.2 El presupuesto optimizado de operación diaria

Para 100,000 productos scrapeados diariamente con arquitectura optimizada:

Componente	Descripción	Costo
Infraestructura	VMs, Redis, PostgreSQL	$142
Proxies	70 % datacenter, 25 % residencial, 5 % premium	$397
Servicios	CAPTCHA solving, monitoreo	$80
TOTAL		**$619/mes**
Por 1K req		**$0.21**

Esto representa **50-80 % de ahorro** versus servicios de scraping administrados.

6 Cumplimiento Legal y Prácticas de Scraping Responsable

El web scraping para inteligencia de precios competitivos opera dentro de un marco legal complejo pero navegable que ha evolucionado significativamente a través de precedentes 2025.

6.1 El precedente hiQ v. LinkedIn establece principios centrales

El caso hito abarcando 2017-2022 estableció que **scrapear datos públicamente disponibles no viola el Computer Fraud and Abuse Act (CFAA)**, el estatuto federal anti-hackeo. La Corte del 9º Circuito afirmó que acceder información pública—datos de precios visibles sin login—no constituye "acceso no autorizado" bajo ley criminal.

Sin embargo, el acuerdo final responsabilizando a hiQ por violaciones de Términos de Servicio demuestra que la responsabilidad civil permanece a través de reclamos de incumplimiento de contrato, especialmente con acuerdos clickwrap que requieren aceptación explícita.

6.2 Robots.txt representa preferencias expresadas, no requerimientos legales

Aunque **no legalmente vinculante**, el archivo robots.txt comunica las preferencias de los propietarios de sitios web para comportamiento de crawler. Seguir estas directivas demuestra buena fe y reduce riesgo legal.

Mejores prácticas:

- Verificar `https://[sitio-objetivo].com/robots.txt` antes de scrapear

- Respetar directivas disallow para todos los user-agents
- Honrar configuraciones crawl-delay
- Reverificar periódicamente conforme las políticas cambian

6.3 Las prácticas de scraping responsable minimizan exposición legal

Mejores prácticas técnicas:

- Limitación de tasa (1-3 segundos mínimo entre solicitudes)
- Respetar directivas crawl-delay
- Scrapear durante horas de bajo tráfico
- Usar cadenas user-agent honestas con identificación de empresa
- Evitar suplantación de Googlebot o usuarios legítimos

Límites de recolección de datos:

- Restringir alcance a información pública de precios y productos
- Evitar datos personales (nombres, emails, direcciones)
- Nunca bypassear páginas de login o paywalls
- Abstenerse de circunvenir CAPTCHAs o protecciones técnicas

Banderas rojas a evitar absolutamente:

- Bypassear páginas de login o crear cuentas falsas
- Ignorar cartas de cese y desistimiento sin consulta legal
- Scrapear datos personales
- Abrumar servidores con tasas agresivas de solicitudes
- Circunvenir CAPTCHAs o protecciones técnicas
- Republicar contenido con derechos de autor sin permiso
- Violar ToS clickwrap explícitos

7 Patrones de Implementación Probados de Casos de Estudio

Las implementaciones del mundo real demuestran que los MVPs exitosos de 3-4 meses priorizan alcance estrecho, herramientas probadas y desarrollo iterativo sobre soluciones comprehensivas.

7.1 Implementación de 30 días: Fabricante global

El caso de estudio del fabricante de Apify logró despliegue completo de producción en solo **un mes**:

Especificaciones:

- 20 sitios web de e-commerce
- Aproximadamente 1,000 productos cada uno (20,000 páginas diarias)
- 5 atributos por producto (nombre, ID, precio, precio original, imagen)
- Extracción automatizada diaria

Resultados:

- Cobertura aumentó de 10 % a 100 % de productos monitoreados
- Frecuencia mejoró de verificaciones manuales semanales a monitoreo automatizado diario
- Detección de problemas se redujo de semanas a dentro de 24 horas

Factores clave de éxito:

- Alcance limitado claro (5 puntos de datos vs inteligencia comprehensiva)
- Aprovechar actores pre-construidos en lugar de construir desde cero
- Externalizar mantenimiento al equipo dedicado de Apify

7.2 Minorista australiano: 300,000 actualizaciones diarias

El caso de estudio Mobius documentó un minorista omnicanal líder:

- 10,000 SKUs a través de 33 sitios web competidores
- Monitoreo diario totalizando 300,000 unidades de producto

- Seguimiento de precios, promociones, anuncios impresos y variaciones geográficas

Resultados:

- Ventas aumentadas a través de precios competitivos
- Tasas mejoradas de retención de clientes
- Automatización reemplazando procesos manuales intensivos

7.3 Minorista multinacional: 67 % aumento de ingresos

ProWebScraper (47 tiendas en Singapur, Malasia e India):

Impacto de negocio:

- 67 % aumento de ingresos a través de estrategia de precios optimizada
- Insights de inventario en tiempo real habilitando gestión eficiente
- Mejor targeting de clientes de análisis de datos mejorado
- Costos de retención reducidos mientras se evitan desabastos

7.4 Patrones comunes de implementaciones exitosas

Priorización de características MVP:

Fase	Características
Must-have (M1-2)	5-10 competidores, 500-1K KVIs, recolección diaria, matching básico, DB, API, alertas email
Should-have (M2-3)	15-20 competidores, 2-5K SKUs, detección promociones, stock, dashboard básico, notificaciones
Nice-to-have (M3-4)	30+ sitios, catálogo completo, reviews, recomendaciones automáticas, analítica predictiva

Trampas comunes:

1. **Sobre-ingeniería del MVP:** Intentar 100+ sitios desde día uno
2. **Mala coincidencia:** Matching solo por nombres (20-30 % error)
3. **Estrategia anti-bot inadecuada:** Scripts simples sin stealth
4. **Ignorar cambios web:** Falta de monitoreo de selectores
5. **Lanzar en temporada pico:** Aumenta riesgos innecesariamente

8 Hoja de Ruta de Implementación Recomendada

Combinar insights de casos de estudio y mejores prácticas técnicas rinde una hoja de ruta práctica de 16 semanas al despliegue de producción.

8.1 Fase 1: Fundación (Semanas 1-4)

Semanas 1-2: Planificación y evaluación

- Definir top 3-4 competidores (Elektra, Liverpool, Palacio de Hierro)
- Seleccionar 2-3 categorías de productos (electrónica, electrodomésticos, muebles)
- 500-1,000 SKUs inicialmente
- Pilotos con 2-3 plataformas scraping (ScraperAPI, ScrapingBee, Apify)
- Diseñar esquema de datos para precios, productos, competidores
- Configurar ambiente desarrollo (Docker, Prefect, PostgreSQL, React)

Semanas 3-4: Infraestructura central

- Construir scrapers para top 2-3 competidores
- Configurar workflows Prefect para programación diaria
- Configurar PostgreSQL y buckets S3 para Parquet
- Desarrollar lógica inicial matching (EAN/UPC con fallback marca+modelo)
- Crear validación básica calidad de datos

Entregables:

- Infraestructura monitoreando 2-3 sitios
- 500 productos emparejados y recolectando diariamente
- Reportes básicos calidad datos
- Arquitectura documentada

8.2 Fase 2: Expansión y Analítica (Semanas 5-8)

Semanas 5-6: Escalar scraping

- Agregar competidores restantes (alcanzar 3-4 totales)
- Expandir a 1,000 SKUs completos a través de 2-3 categorías
- Implementar RapidFuzz para fuzzy matching
- Configurar rotación proxy (ScraperAPI)
- Establecer manejo comprehensivo errores con fallbacks multi-nivel
- Configurar dashboards Grafana (tasas éxito, tiempos respuesta, completitud)

Semanas 7-8: Fundación analítica

- Desarrollar endpoints FastAPI (consultas precios, comparaciones, tendencias, alertas)
- Crear dashboard React con filtrado (competidor, categoría, fechas, umbrales)
- Implementar visualizaciones series de tiempo (Recharts)
- Construir funcionalidad exportación (CSV, Parquet)
- Configurar alertas umbral (notificaciones equipo precios)

Entregables:

- 3-4 competidores y 1,000 SKUs monitoreados diariamente (95 %+ éxito)
- Dashboard funcional mostrando métricas clave
- API RESTful para acceso datos
- Sistema alertas automatizado

8.3 Fase 3: Integración y Refinamiento (Semanas 9-12)

Semanas 9-10: Conectar a sistemas de negocio

- Integrar API con sistema precios Coppel
- Implementar lógica detección promociones
- Expandir cobertura matching SKU (85 %+ match, 95 %+ precisión)
- Agregar seguimiento disponibilidad stock
- Crear reportes automatizados diarios (email a equipo precios)

Semanas 11-12: Refinar basado en feedback

- Conducir UAT con equipo precios y gerentes categoría
- Mejorar algoritmo matching (análisis falsos positivos/negativos)
- Mejorar características dashboard (puntos dolor usuarios)
- Construir interfaz revisión manual (coincidencias inciertas)
- Optimizar rendimiento consultas (loading dashboard más rápido)
- Documentar especificaciones API, lógica scraping, procedimientos

Entregables:

- API integrada con sistemas precios
- Seguimiento promociones operacional
- Dashboard mejorado con feedback incorporado
- Cobertura SKU 85 %+ con precisión 95 %+

8.4 Fase 4: Lanzamiento a Producción (Semanas 13-16)

Semanas 13-14: Preparar producción

- Optimización rendimiento (API sub-segundo para consultas comunes)
- Completar documentación técnica (mantenimiento scrapers, DB, troubleshooting)
- Escribir guías usuario (dashboard, reportes, alertas)
- Implementar manejo comprehensivo errores (casos extremos)
- Conducir revisión seguridad (autenticación API, controles acceso)
- Configurar monitoreo producción (procedimientos guardia)

Semanas 15-16: Rollout escalonado

- Desplegar a producción con usuarios piloto (5-10 personas, 1 semana)
- Recoger feedback y arreglar problemas críticos
- Expandir a equipos completos precios y merchandising (30-50 usuarios)
- Conducir sesiones entrenamiento
- Establecer canales feedback para mejoras continuas
- Monitorear rendimiento, abordar bugs, planear Fase 2

Entregables:

- Sistema listo producción con usuarios entrenados
- Procedimientos documentados
- Rendimiento monitoreado
- Roadmap siguiente fase

8.5 Métricas de éxito definen preparación

Tipo	KPI	Target
Técnicos	Uptime	>99 % (<7h downtime/mes)
	Precisión	>95 % (matching productos)
	Frescura	>95 % (datos <24h)
	Cobertura	>85 % (SKUs prioritarios)
Negocio	Adopción	>80 % (equipo usando semanal)
	Insights	>50 (cambios accionables/mes)
	Impacto	3-5 % (mejora competitividad)
	Eficiencia	>50 % (reducción investigación manual)

8.6 Decisión construir vs comprar para Coppel

Enfoque híbrido recomendado:

- Usar servicios administrados (ScraperAPI/ScrapingBee) para infraestructura anti-bot
- Construir matching SKU personalizado, analítica, dashboard, integraciones
- Equilibra velocidad, costo ($200-800/mes vs $2-5K comercial), control

Presupuesto MVP estimado:

Concepto	Descripción	Monto
Personal	2-3 ingenieros + PM (4 meses)	$80-150K
Plataforma scraping	Piloto y meses iniciales	$2-10K
Infraestructura	AWS/GCS, databases, monitoring	$2-5K
Herramientas/servicios	Proxies, anti-bot, analytics	$3-5K
TOTAL MVP		**$90-170K**

Costos mensuales continuos post-MVP:

Concepto	Monto
Plataforma scraping	$2-8K
Infraestructura	$1-3K
Mantenimiento (1 ing 50 %)	$8-12K
TOTAL	**$11-23K/mes**

Resultados esperados dentro 3-4 meses:

- Monitorear 3-4 competidores clave
- Rastrear 1,000-5,000 SKUs prioritarios
- Actualizaciones precios diarias con frescura 95 %+
- Dashboard básico y alertas
- Insights simples posicionamiento precios
- Demostrar mejora 3-5 % en categorías selectas

9 Conclusión: Camino hacia Inteligencia de Precios Competitivos

Construir un MVP de monitoreo de precios competitivos en 3-4 meses demanda alcance enfocado, tecnologías probadas y desarrollo iterativo. La arquitectura recomendada—Playwright para automatización navegadores, Prefect para orquestación, Polars y DuckDB para procesamiento datos, Parquet en S3 para almacenamiento, FastAPI para backend y React para frontend—entrega capacidades listas para producción dentro restricciones cronograma y presupuesto.

Comenzando con 3-4 competidores y 1,000 SKUs, usando servicios scraping administrados para infraestructura mientras se construye coincidencia y analítica personalizadas, y siguiendo la hoja de ruta implementación por fases posiciona a Coppel para lograr las métricas requeridas de cobertura SKU del 85 %, frescura del 95 % y precisión del 97 %.

Perspectiva clave de implementaciones exitosas

Comience estrecho, entregue valor rápidamente y expanda iterativamente.

- Fabricante logrando producción en 30 días

- Minorista australiano procesando 300,000 actualizaciones diarias
- Minorista multinacional aumentando ingresos 67 %

Todos siguieron este patrón. Sobre-ingeniería del MVP, intentar cobertura comprehensiva inmediatamente y construir todo desde cero representan los modos de falla primarios.

Al aprovechar herramientas probadas, enfocarse en valor negocio central y planear para mantenimiento continuo, Coppel puede establecer inteligencia precios competitivos que conduce impacto financiero medible dentro de un solo trimestre.

Marco legal y técnico

El marco legal apoya scraping ético de datos precios públicos al seguir mejores prácticas:

- Respetar robots.txt
- Usar identificación transparente
- Implementar limitación de tasa
- Evitar recolección datos personales
- Mantener documentación cumplimiento

El enfoque técnico equilibrando costo y rendimiento—usar solicitudes HTTP ligeras para sitios no protegidos, reservar automatización navegador para 20 % objetivos desafiantes, optimizar mezcla proxy e implementar caché—logra la confiabilidad requerida a costos operacionales de $600-900 mensuales.

El resultado: Inteligencia competitiva lista para producción entregando insights precios accionables que mejoran posición mercado mientras opera dentro límites legales y éticos.

10 Glosario de Términos Técnicos

10.1 Términos de Web Scraping

Web Scraping Proceso automatizado de extracción de datos de sitios web mediante programas que simulan la navegación humana o analizan el código HTML.

Headless Browser Navegador web sin interfaz gráfica que se ejecuta en segundo plano, utilizado para automatizar la navegación y extraer contenido dinámico renderizado con JavaScript.

Proxy Servidor intermediario que actúa como puente entre el scraper y el sitio objetivo, ocultando la IP real y permitiendo rotación de direcciones para evitar bloqueos.

User-Agent Cadena de texto que identifica el navegador y sistema operativo en las solicitudes HTTP. Los scrapers deben usar user-agents realistas para evitar detección.

Rate Limiting Técnica que limita la cantidad de solicitudes por unidad de tiempo para evitar sobrecargar servidores y reducir probabilidad de ser bloqueado.

CAPTCHA Sistema de desafío-respuesta para distinguir humanos de bots, común en sitios protegidos contra scraping automatizado.

Robots.txt Archivo en la raíz del sitio web que especifica qué páginas pueden ser accedidas por crawlers automatizados.

10.2 Arquitectura y Orquestación

Microservicios Arquitectura donde la aplicación se descompone en servicios pequeños e independientes que se comunican mediante APIs.

Orquestación Coordinación y gestión automatizada de flujos de trabajo complejos, programando y monitoreando la ejecución de tareas de scraping.

Worker Proceso que ejecuta tareas específicas de forma independiente, como scrapear un sitio o procesar datos extraídos.

Queue (Cola) Sistema de almacenamiento temporal de tareas pendientes que distribuye trabajo entre múltiples workers.

ETL Extract, Transform, Load—proceso de extraer datos de fuentes, transformarlos a formato deseado y cargarlos en destino final.

API REST Interfaz de programación que permite comunicación entre sistemas mediante protocolo HTTP con operaciones estándar (GET, POST, PUT, DELETE).

10.3 Procesamiento y Almacenamiento de Datos

DataFrame Estructura de datos tabular bidimensional similar a hoja de cálculo, optimizada para análisis y transformaciones de datos.

Parquet Formato de archivo columnar comprimido que optimiza el almacenamiento y consulta de grandes volúmenes de datos.

Data Lake Repositorio centralizado que almacena datos estructurados y no estructurados en su formato original para análisis posterior.

OLAP Online Analytical Processing—sistemas optimizados para consultas analíticas complejas sobre grandes volúmenes de datos históricos.

Redis Base de datos en memoria de tipo clave-valor, utilizada para caché y gestión de datos de alta velocidad.

PostgreSQL Sistema de gestión de bases de datos relacional de código abierto, robusto y extensible.

10.4 Matching y Análisis

SKU Stock Keeping Unit—identificador único de producto en sistemas de inventario y comercio.

Fuzzy Matching Técnica de coincidencia aproximada que encuentra similitudes entre cadenas de texto incluso con errores o variaciones.

EAN/UPC/GTIN Códigos de barras estandarizados internacionalmente para identificación única de productos.

MPN Manufacturer Part Number—número de parte asignado por el fabricante para identificar productos.

Entity Resolution Proceso de identificar y vincular registros que se refieren a la misma entidad del mundo real a través de múltiples fuentes de datos.

Precision/Recall Métricas de calidad: precisión mide exactitud de coincidencias positivas, recall mide cobertura de coincidencias relevantes.

10.5 Anti-Bot y Seguridad

Fingerprinting Técnica de identificación de navegadores mediante análisis de características únicas (canvas, WebGL, fuentes, configuraciones).

Cloudflare Plataforma de seguridad web que proporciona protección contra bots, DDoS y otros ataques.

TLS Fingerprinting Identificación de clientes mediante análisis de configuración SSL/TLS en el handshake de conexión.

Residential Proxy Proxy que usa direcciones IP asignadas a dispositivos residenciales reales, más difíciles de detectar que IPs de datacenter.

Stealth Mode Técnicas para hacer que navegadores automatizados sean indistinguibles de navegadores controlados por humanos.

10.6 Métricas y Monitoreo

Uptime Porcentaje de tiempo que el sistema está operacional y disponible.

SLA Service Level Agreement—acuerdo de nivel de servicio que especifica métricas garantizadas de rendimiento.

Frescura Porcentaje de datos actualizados dentro del período objetivo (típicamente <24 horas).

Cobertura Porcentaje de productos objetivo que son exitosamente monitoreados.

Tasa de Éxito Porcentaje de intentos de scraping que completan exitosamente sin errores o bloqueos.

KVI Key Value Item—productos estratégicos más importantes para monitorear competitivamente.

11 Referencias Principales con Código Accesible

11.1 Herramientas de Web Scraping

1. **Playwright - Browser Automation** (Microsoft)
   GitHub: https://github.com/microsoft/playwright
   Docs: https://playwright.dev/python/
   Instalación: `pip install playwright`
   63.5k en GitHub

2. **Playwright Stealth Plugin** (Anti-detección)
   GitHub: https://github.com/rebrowser/rebrowser-playwright
   PyPI: https://pypi.org/project/playwright-stealth/
   Instalación: `pip install playwright-stealth`

3. **Scrapy Framework**
   GitHub: https://github.com/scrapy/scrapy
   Docs: https://docs.scrapy.org/
   Instalación: `pip install scrapy`
   52.8k en GitHub

4. **FlareSolverr** (Bypass Cloudflare)
   GitHub: https://github.com/FlareSolverr/FlareSolverr
   Docker: `docker run -p 8191:8191 flaresolverr/flaresolverr`
   7.5k en GitHub

11.2 SKU Matching y Procesamiento

5. **RapidFuzz** (Fuzzy String Matching)
   GitHub: https://github.com/maxbachmann/RapidFuzz
   Docs: https://maxbachmann.github.io/RapidFuzz/
   Instalación: `pip install rapidfuzz`
   2.6k en GitHub

6. **Dedupe** (Entity Resolution)
   GitHub: https://github.com/dedupeio/dedupe
   Docs: https://docs.dedupe.io/
   Instalación: `pip install dedupe`
   4.1k en GitHub

7. **Polars** (DataFrame Processing)
   GitHub: https://github.com/pola-rs/polars
   Docs: https://pola.rs/

Instalación: `pip install polars`
30.2k en GitHub

11.3 Orquestación y Workflows

8. **Prefect 2.0**
   GitHub: https://github.com/PrefectHQ/prefect
   Docs: https://docs.prefect.io/
   Instalación: `pip install prefect`
   16.1k en GitHub

9. **Dagster** (Alternativa)
   GitHub: https://github.com/dagster-io/dagster
   Docs: https://docs.dagster.io/
   Instalación: `pip install dagster`
   11.5k en GitHub

11.4 Bases de Datos y Storage

10. **DuckDB** (OLAP Database)
    GitHub: https://github.com/duckdb/duckdb
    Python API: https://duckdb.org/docs/api/python/overview
    Instalación: `pip install duckdb`
    24.1k en GitHub

11. **Apache Parquet Python**
    GitHub: https://github.com/apache/arrow
    Docs: https://arrow.apache.org/docs/python/parquet.html
    Instalación: `pip install pyarrow`

11.5 APIs y Proxies

12. **ScraperAPI Python SDK**
    GitHub: https://github.com/scraperapi/scraperapi-python
    Docs: https://docs.scraperapi.com/
    Instalación: `pip install scraperapi-sdk`

13. **FastAPI Framework**
    GitHub: https://github.com/tiangolo/fastapi
    Docs: https://fastapi.tiangolo.com/
    Instalación: `pip install fastapi uvicorn`
    77.5k en GitHub

11.6 Frontend y Visualización

14. **React + Recharts Template**
    GitHub: https://github.com/recharts/recharts
    Docs: https://recharts.org/
    Instalación: `npm install recharts`
    24k en GitHub

15. **Grafana + Prometheus Stack**
    GitHub: https://github.com/prometheus/prometheus
    Docker Compose: https://github.com/vegasbrianc/prometheus
    Configuración completa para monitoreo

11.7 Ejemplos y Tutoriales Completos

16. **Web Scraping Solution - Microservices Architecture**
    GitHub: https://github.com/santagar/web-scraping-solution
    Stack: Python, RabbitMQ, Docker
    Arquitectura completa de microservicios

17. **Apache Airflow Scraping Pipeline**
    Tutorial: https://github.com/kadnan/AirflowScrapyPipeline
    Ejemplo completo con Scrapy + Airflow
    Incluye configuración de DAGs

18. **Price Monitoring con Playwright**
    GitHub: https://github.com/apify/crawlee-python
    Framework moderno para scraping
    Ejemplos de e-commerce incluidos

11.8 Utilidades y Optimización

19. **Python Robots Parser**
    GitHub: https://github.com/seomoz/reppy
    Instalación: `pip install reppy`
    Para verificar robots.txt automáticamente

20. **Cloudscraper** (Bypass básico)
    GitHub: https://github.com/VeNoMouS/cloudscraper
    Instalación: `pip install cloudscraper`
    3.6k en GitHub

11.9 Código de Inicio Rápido

Listing 1: requirements.txt para el MVP

```
# requirements.txt para el MVP
playwright==1.41.0
playwright-stealth==1.0.6
prefect==2.14.0
fastapi==0.109.0
uvicorn==0.27.0
polars==0.20.0
duckdb==0.9.2
rapidfuzz==3.6.0
pyarrow==14.0.0
redis==5.0.1
scraperapi-sdk==1.2.0
pydantic==2.5.0
sqlalchemy==2.0.0
psycopg2-binary==2.9.9
prometheus-client==0.19.0
sentry-sdk==1.39.0
```

11.10 Stack Recomendado para Comenzar

Para el MVP de Coppel, sugiero comenzar con:

1. **Playwright** + **playwright-stealth** para scraping
2. **Prefect** para orquestación (más simple que Airflow)
3. **RapidFuzz** para matching de productos
4. **Polars** + **DuckDB** para procesamiento de datos
5. **FastAPI** para el backend
6. **ScraperAPI** para manejar proxies (evita complejidad inicial)

Todos estos tienen documentación excelente, comunidades activas, y son de código abierto con licencias permisivas (MIT/Apache 2.0).