ADITOR AI

Prompt Maestro de Auditoría

Versión 1.0 — Documento técnico interno

---

01. ¿Qué es este documento?

Este documento contiene el prompt maestro que vive dentro de Aditor AI. Es el cerebro que se ejecuta cada vez que un usuario genera una auditoría, ya sea manual o automática semanal.

Está compuesto por tres partes que trabajan juntas: el System Prompt que define el rol y las reglas del LLM, el User Prompt con los datos variables de cada usuario, y el XML Schema que define exactamente el formato del output.

LLM recomendado	Claude API (claude-sonnet-4-6) o GPT-4o
Temperatura	0.3 — Respuestas consistentes, poco aleatorias
Max tokens output	4000 — Suficiente para reporte completo
Formato de respuesta	XML estricto, sin texto adicional fuera del schema
Idioma	Español neutro / rioplatense según mercado

---

02. System Prompt

Este prompt se envía como mensaje de sistema antes de cualquier interacción. Define la identidad, el rol y las reglas absolutas del LLM. No cambia entre usuarios.

```
Eres AditorAI, un auditor experto en campañas de Meta Ads especializado
en e-commerce de LATAM y España. Tu único objetivo es analizar métricas
publicitarias y devolver un diagnóstico claro, accionable y honesto.

REGLAS ABSOLUTAS:
- Responde ÚNICAMENTE en formato XML válido. Sin texto fuera del XML.
- No uses Markdown. No uses explicaciones previas. Solo XML.
- Usa lenguaje directo, simple y en español rioplatense/neutro.
- Nunca digas "podría", "quizás" o "considerar". Di exactamente qué hacer.
- Cada recomendación debe tener una acción concreta, no una sugerencia vaga.
- Si una campaña tiene ROAS < 1.0 con más de $100 gastados: es pausar, siempre.
- Prioriza hallazgos por impacto económico, de mayor a menor.
```

Por qué cada regla

- Temperatura 0.3: evita que el LLM sea creativo donde no debe serlo. La auditoría tiene que ser consistente, no variar entre ejecuciones del mismo dataset.
- 'Sin texto fuera del XML': si el LLM agrega texto como 'Aquí está tu análisis:', el parser de tu app se rompe. Esta regla lo previene.
- 'ROAS < 1.0 con más de $100 gastados = pausar siempre': elimina la ambigüedad. El LLM no puede dudar en casos claros.
- 'Prioriza por impacto económico': el hallazgo más importante aparece primero. El usuario ve lo más crítico sin scrollear.

---

03. User Prompt

Este prompt se construye dinámicamente en tu backend cada vez que un usuario dispara una auditoría. Las variables entre llaves {} se reemplazan con datos reales de Meta Ads API.

```
Analizá las siguientes campañas de Meta Ads de una tienda de e-commerce.

PERÍODO: Últimos 30 días con tendencia semanal (4 semanas)
MONEDA: {moneda}
INDUSTRIA: E-commerce
MERCADO: {pais}
```

```
DATOS DE CAMPAÑAS:
{campanias_json}

Devuelve ÚNICAMENTE el XML con el siguiente schema. Sin texto adicional.
```

Variables dinámicas

{moneda}	ARS / USD / EUR — según país del usuario
{pais}	Argentina / México / España / etc.
{campanias_json}	Array JSON con todas las campañas y métricas del período
{fecha}	Fecha actual ISO 8601 — ej: 2026-02-18
{total}	Número total de campañas analizadas

Estructura del JSON de campañas

El campo {campanias_json} debe enviarse con esta estructura por cada campaña:

```
[
  {
    "id": "23851234567890",
    "nombre": "Retargeting 7 días — Compradores",
    "estado": "activa",
    "presupuesto_diario": 50,
    "metricas_30d": {
      "roas": 4.2,
      "ctr": 1.9,
      "cpm": 9.4,
      "cpc": 0.49,
      "cpa": 18.5,
      "frecuencia": 2.1,
      "gasto_total": 1240,
      "conversiones": 67,
      "valor_conversiones": 5208,
```

```
      "pagos_iniciados": 89,
      "visitas_landing": 2530,
      "mensajes_iniciados": 12
    },
    "tendencia_semanal": {
      "semana_1": { "roas": 3.8, "gasto": 290 },
      "semana_2": { "roas": 4.1, "gasto": 310 },
      "semana_3": { "roas": 4.5, "gasto": 320 },
      "semana_4": { "roas": 4.4, "gasto": 320 }
    },
    "creativos": [
      { "id": "ad_001", "tipo": "video", "ctr": 2.1, "frecuencia": 1.8 },
      { "id": "ad_002", "tipo": "imagen", "ctr": 1.6, "frecuencia": 2.4 }
    ]
  }
]
```

04. XML Schema de Output

Este es el formato exacto que debe devolver el LLM. Tu frontend parsea este XML y lo convierte en la interfaz visual que ve el usuario. Cada tag corresponde a un elemento de la UI.

```
<?xml version="1.0" encoding="UTF-8"?>
<auditoria>

  <!-- SCORE GENERAL DE LA CUENTA -->
  <score_cuenta>
    <valor><!-- Número del 0 al 100 --></valor>
    <nivel><!-- "critico" | "regular" | "bueno" | "excelente" --></nivel>
    <resumen><!-- 1 oración directa explicando el estado general --></resumen>
  </score_cuenta>

  <!-- RESUMEN EJECUTIVO: máximo 3 líneas, lenguaje humano -->
  <resumen_ejecutivo><!-- texto --></resumen_ejecutivo>

  <!-- MÉTRICAS GLOBALES DE LA CUENTA -->
  <metricas_globales>
```

```
    <roas_promedio><!-- número --></roas_promedio>
    <ctr_promedio><!-- número % --></ctr_promedio>
    <cpm_promedio><!-- número --></cpm_promedio>
    <cpc_promedio><!-- número --></cpc_promedio>
    <cpa_promedio><!-- número --></cpa_promedio>
    <gasto_total><!-- número --></gasto_total>
    <conversiones_totales><!-- número --></conversiones_totales>
    <valor_conversiones_total><!-- número --></valor_conversiones_total>
    <pagos_iniciados_total><!-- número --></pagos_iniciados_total>
    <visitas_landing_total><!-- número --></visitas_landing_total>
    <tasa_conversion_landing><!-- % visitas que convierten --
></tasa_conversion_landing>
    <tendencia_roas><!-- "mejorando" | "estable" | "cayendo" --></tendencia_roas>
  </metricas_globales>

  <!-- HALLAZGOS: ordenados por impacto económico descendente -->
  <hallazgos>

    <hallazgo id="1">
      <tipo><!-- "pausar" | "escalar" | "atencion" | "fatiga" | "optimizar_landing"
--></tipo>
      <urgencia><!-- "alta" | "media" | "baja" --></urgencia>
      <campana_nombre><!-- nombre exacto --></campana_nombre>
      <campana_id><!-- id de Meta --></campana_id>
      <impacto_economico><!-- dinero perdido o ganado potencial en USD/ARS --
></impacto_economico>

      <metricas_relevantes>
        <roas><!-- valor --></roas>
        <ctr><!-- valor % --></ctr>
        <cpm><!-- valor --></cpm>
        <cpc><!-- valor --></cpc>
        <frecuencia><!-- valor --></frecuencia>
        <gasto_periodo><!-- valor --></gasto_periodo>
        <conversiones><!-- valor --></conversiones>
        <pagos_iniciados><!-- valor --></pagos_iniciados>
        <visitas_landing><!-- valor --></visitas_landing>
        <tasa_conversion><!-- % --></tasa_conversion>
      </metricas_relevantes>
```

```
      <tendencia_semanal>
        <semana_1><!-- roas semana 1 --></semana_1>
        <semana_2><!-- roas semana 2 --></semana_2>
        <semana_3><!-- roas semana 3 --></semana_3>
        <semana_4><!-- roas semana 4 --></semana_4>
        <direccion><!-- "mejorando" | "estable" | "cayendo" --></direccion>
      </tendencia_semanal>

      <diagnostico><!-- Explicación en 2 líneas máximo. Qué está pasando y por qué. --></diagnostico>
      <accion_concreta><!-- Una sola acción específica. Ej: "Pausar hoy antes de las 18hs" --></accion_concreta>
      <benchmark_referencia><!-- Qué benchmark se usó para detectar este problema --></benchmark_referencia>
    </hallazgo>

    <!-- Repetir <hallazgo> por cada campaña con problema o oportunidad -->

  </hallazgos>

  <!-- REDISTRIBUCIÓN DE PRESUPUESTO SUGERIDA -->
  <redistribucion_presupuesto>
    <presupuesto_liberado><!-- dinero de campañas a pausar --></presupuesto_liberado>
    <sugerencia><!-- Cómo redistribuir ese presupuesto entre campañas ganadoras --></sugerencia>
    <campanas_receptoras>
      <campana>
        <nombre><!-- nombre --></nombre>
        <presupuesto_sugerido><!-- monto --></presupuesto_sugerido>
        <motivo><!-- por qué esta campaña merece más presupuesto --></motivo>
      </campana>
    </campanas_receptoras>
  </redistribucion_presupuesto>

  <!-- HOOKS SUGERIDOS basados en creativos con mejor CTR -->
  <hooks_sugeridos>

    <hook id="1">
      <tipo><!-- "dolor" | "curiosidad" | "contraste" | "identidad" | "prueba_social" --></tipo>
```

```
      <texto><!-- El hook completo listo para usar --></texto>
      <score_estimado><!-- número del 0 al 100 --></score_estimado>
      <basado_en><!-- qué creativo o campaña inspiró este hook --></basado_en>
      <formato_recomendado><!-- "video" | "imagen_estatica" | "carrusel" --
></formato_recomendado>
    </hook>

    <!-- Generar entre 4 y 6 hooks -->

  </hooks_sugeridos>

  <!-- ANGULOS DE VENTA SUGERIDOS -->
  <angulos_sugeridos>
    <angulo id="1">
      <titulo><!-- Nombre del ángulo --></titulo>
      <descripcion><!-- En qué consiste el ángulo --></descripcion>
      <ejemplo_copy><!-- Un ejemplo de copy listo para usar --></ejemplo_copy>
    </angulo>
    <!-- Generar 3 ángulos -->
  </angulos_sugeridos>

  <!-- PRÓXIMOS PASOS: lista de acciones ordenadas por prioridad -->
  <proximos_pasos>
    <paso orden="1">
      <accion><!-- Acción específica --></accion>
      <plazo><!-- "hoy" | "esta semana" | "próximos 15 días" --></plazo>
      <impacto_esperado><!-- Qué mejora esperar si se ejecuta --
></impacto_esperado>
    </paso>
    <!-- Entre 3 y 5 pasos -->
  </proximos_pasos>

  <metadata>
    <fecha_analisis>{fecha}</fecha_analisis>
    <periodo_analizado>Últimos 30 días</periodo_analizado>
    <total_campanias_analizadas>{total}</total_campanias_analizadas>
    <version_prompt>1.0</version_prompt>
  </metadata>
```

```
</auditoria>
```

05. Benchmarks de Referencia

El LLM usa estos benchmarks internamente para clasificar cada métrica. Están calibrados para e-commerce en LATAM y España. Podés ajustarlos en el System Prompt si el mercado de un usuario es diferente.

Métrica	Bajo 🔴	Aceptable 🟡	Bueno 🟢
ROAS e-commerce	< 1.5x	1.5x – 2.5x	> 2.5x
CTR (Feed)	< 0.8%	0.8% – 1.5%	> 1.5%
CTR (Stories/Reels)	< 0.5%	0.5% – 1.0%	> 1.0%
CPM (LATAM)	> $18	$10 – $18	< $10
CPM (España)	> $28	$16 – $28	< $16
CPC	> $2.5	$0.8 – $2.5	< $0.8
Frecuencia	> 5.0	3.0 – 5.0	< 3.0
Conv. landing a compra	< 1%	1% – 3%	> 3%
Pagos iniciados / visitas	< 3%	3% – 8%	> 8%

Reglas adicionales de clasificación

- Frecuencia > 5.0 en los últimos 7 días → hallazgo tipo 'fatiga' independientemente del ROAS actual, porque el deterioro es inminente.
- CTR alto (> 1.5%) + conversiones bajas (< 1% de visitas) → hallazgo tipo 'optimizar_landing'. El anuncio funciona, el problema está en la página de destino.
- Pagos iniciados / visitas < 3% → alerta de checkout. El usuario llega al carrito pero no completa la compra.
- Mensajes iniciados sin conversiones en 7 días → posible problema de respuesta o de calificación del lead.
- Tendencia de ROAS cayendo 3 semanas consecutivas → urgencia alta aunque el ROAS actual siga siendo positivo.

06. Cómo integrarlo en Antigravity

Este prompt no se escribe directamente en Antigravity. Vive en el backend de tu app como una función que se llama cuando el usuario dispara una auditoría. El prompt para Antigravity es diferente: le dice al agente cómo construir esa función.

Flujo técnico completo

1. **Meta Ads API devuelve las campañas del usuario en formato JSON.**
2. Tu backend (Next.js en Vercel) construye el User Prompt reemplazando las variables con los datos reales.
3. Tu backend llama a Claude API o GPT-4o enviando System Prompt + User Prompt.
4. El LLM devuelve el XML completo con el reporte.
5. Tu backend parsea el XML y guarda el reporte en Supabase asociado al usuario.
6. Tu frontend lee el reporte de Supabase y lo renderiza con los colores, badges y secciones del prototipo.
7. Si es auditoría semanal automática, se envía un email al usuario con el resumen y un link al reporte completo.

Prompt para Antigravity — Función de auditoría

Cuando construyas la app en Antigravity, usá este prompt para que el agente genere la función backend:

```
Crea una función llamada generateAudit() en TypeScript que:

1. Reciba como parámetro un array de campañas con este tipo:
   Campaign[] con las métricas definidas en el JSON de ejemplo adjunto.

2. Construya el User Prompt reemplazando estas variables:
   - {moneda}: desde el perfil del usuario en Supabase
   - {pais}: desde el perfil del usuario en Supabase
   - {campanias_json}: JSON.stringify del array recibido
   - {fecha}: new Date().toISOString()
   - {total}: campaigns.length

3. Llame a la Claude API con:
   - Model: claude-sonnet-4-6
   - Temperature: 0.3
   - Max_tokens: 4000
   - System: [PEGAR SYSTEM PROMPT COMPLETO AQUÍ]
   - User: [USER PROMPT CON VARIABLES REEMPLAZADAS]
```

```
4. Parsee el XML de respuesta con un parser robusto.
   Si el XML es inválido, reintente una vez antes de fallar.

5. Guarde el resultado en la tabla "auditorias" de Supabase
   con los campos: user_id, ad_account_id, fecha, xml_raw,
   score, hallazgos_count, tipo (manual | automatica).

6. Retorne el objeto parseado para que el frontend lo renderice.

Usa el SDK oficial de Anthropic para TypeScript.
Maneja errores con try/catch y loguea en Supabase logs.
```

---

07. Email de Auditoría Semanal Automática

Cada lunes a las 8:00 AM hora del usuario, Aditor AI genera la auditoría automáticamente y envía este email. El trigger puede configurarse con un cron job en Vercel o con Supabase Edge Functions.

Asunto del email

```
Tu auditoría semanal está lista — {X} hallazgos en tus campañas 🔍
```

Estructura del email

- **Score de la semana con comparativa vs semana anterior**
- Top 3 hallazgos más urgentes en texto plano
- Botón CTA: 'Ver reporte completo →' que lleva al dashboard
- Pie: 'Próxima auditoría automática: lunes {fecha}'

Prompt para el asunto del email

```
A partir de este XML de auditoría, generá ÚNICAMENTE el asunto
del email de notificación. Máximo 60 caracteres. Debe mencionar
el número de hallazgos y generar curiosidad o urgencia.
Sin comillas. Sin puntos al final.

XML: {xml_auditoria}
```

08. Versionado y Mejora del Prompt

El prompt maestro va a mejorar con el tiempo a medida que ves cómo responden los usuarios a los reportes. Cada versión debe guardarse en GitHub con su número de versión.

Cuándo actualizar el prompt

- Cuando detectes que el LLM clasifica mal un tipo de campaña específico
- Cuando usuarios se quejan de que las recomendaciones son demasiado genéricas
- Cuando agregues nuevas métricas de Meta Ads API al análisis
- Cuando quieras ajustar los benchmarks para un mercado específico

Cómo testear cambios

8. Creá un dataset de prueba con 10 campañas ficticias de diferentes estados.
9. Corré el prompt actual y el nuevo prompt con el mismo dataset.
10. Compará los XML de output. El nuevo debe detectar más o mejor sin perder precisión.
11. Deployá solo si el nuevo supera al anterior en al menos 3 de los 10 casos.