

PDR — Plataforma Web3 para artistas manga/webtoon con micropagos e IA

Resumen ejecutivo

Plataforma descentralizada donde artistas de manga/webtoon monetizan contenido directo, recibiendo pagos automáticos en cripto por capítulos anticipados, recomendaciones IA y premios automáticos, eliminando intermediarios y reduciendo comisiones.

Diferencial clave: Pagos directos a creadores sin fricciones, eliminando las comisiones de plataformas como Patreon (8-12%) y Paypal (2.9% + \$0.30).

Referencia base: Proyecto x402-demo de Coinbase Developer Platform y el MCP de coinbase que se tiene instalado.

1. Objetivo General

Desarrollar un MVP funcional que demuestre:

- ☒ Monetización de contenido digital mediante micropagos automáticos (x402 + CDP).
- ☒ Onboarding sin fricción para usuarios Web3 (Embedded Wallet).
- ☒ Recomendaciones IA pagadas y orquestadas automáticamente (n8n + OpenAI).
- ☒ Premiaciones automáticas a creadores destacados.

Contexto: Hackathon ETHGlobal Buenos Aires (track Coinbase Developer Platform).

Métrica de éxito: Demo funcional completamente, flujo end-to-end, <4 min pitch, documentación clara.




2. Roles y Experiencia MVP

Lector/Usuario

- ☒ Conecta wallet embebida (Coinbase CDP Embedded Wallet).
- ☒ Navega por series de manga/webtoon sin restricción.
- ☒ Accede a capítulos gratuitos sin fricción.
- ☒ Paga micropagos (0.01 USDC) por capítulos premium/anticipados.
- ☒ Solicita recomendaciones IA personalizadas (pagadas a 0.01 USDC).
- ☒ Ve su balance actualizado en tiempo real.
- ☒ Visualiza historial de pagos y acceso desbloqueado.
- ☒ Verifica transacciones en Basescan.










Artista (Simulado en MVP)

- ☒ Wallet asignada para recibir pagos (mock para demo).
- ☒ Acceso a ranking de visitas y recompensas semanales.








-  Recibe pagos automáticos por capítulos leídos (simulado, mock data).
-  Panel de gestión futuro (out of scope MVP).
-  Analytics y dashboard (out of scope MVP).

3. Funcionalidades del MVP






3.1 Lectura y Acceso



Función	Descripción	Status
Visualización de capítulos	Imágenes pre-cargadas, navegación fluida	MVP 
Capítulos gratuitos	Capítulos 1-4 disponibles sin restricción	MVP 
Capítulos premium	Capítulo 5 desbloqueado tras pago 0.01 USDC	MVP 
Liberación automática	Al subir capítulo 6, cap 5 pasa a gratis (simulado)	MVP 
Historial de acceso	Usuario ve qué capítulos ya leyó/pagó	MVP 
Indicador visual	 Premium,  Free,  Owned	MVP 

3.2 Micropagos y Monetización






Función	Descripción	Status
Pagos x402	Micropago automático protege capítulo	MVP 
CDP Embedded Wallet	Sin extensiones, onboarding express	MVP 
Faucet testnet	Recibir 0.1 USDC gratis en Base Sepolia	MVP 
Token Balances API	Ver balance actualizado en tiempo real	MVP 
Historial de pagos	Log de transacciones confirmadas	MVP 
Basescan integration	Verificar pagos en explorer testnet	MVP 
Error handling	Manejo de rechazos/errores de pago	MVP 

3.3 Recomendaciones IA (Pagadas)

Función	Descripción	Status
Petición IA	Usuario solicita recomendación	MVP 
Pago previo	0.01 USDC antes de generar	MVP 
Webhook a n8n	Backend dispara workflow automático	MVP 
Consulta OpenAI	n8n consulta GPT con contexto	MVP 
Entrega resultado	Recomendación mostrada en frontend	MVP 

Función	Descripción	Status
Transparencia	Se muestra prompt usado y costo	MVP 
Loader/UX	Spinner mientras procesa	MVP 

3.4 Premiaciones Automáticas

Función	Descripción	Status
Rankings	Top 3 artistas por visitas	MVP 
Payout semanal	Simulación de distribución de premios	MVP 
Historial	Log de pagos realizados	MVP 
Basescan	Verificación de transacciones	MVP 
Pool de premios	Visualización de fondos disponibles	MVP 

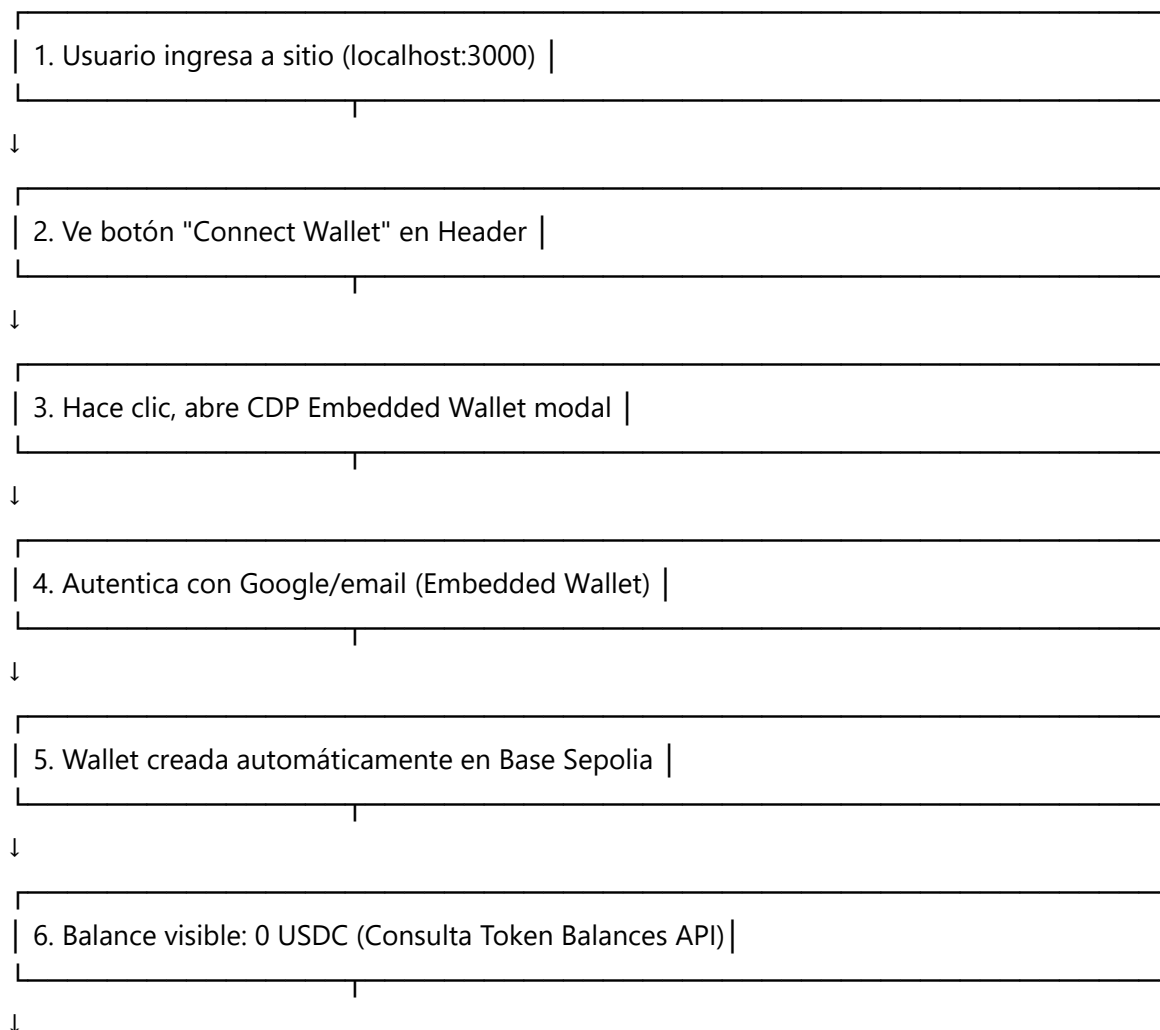
4. Stack tecnológico

Componente	Tecnología	Versión	Justificación
Frontend	Next.js	14+	Compatible x402-demo, SSR, routing sencillo
Framework JS	React	18+	Ecosystem robusto, componentes reutilizables
Estilos	Tailwind CSS	3.4+	UI rápida, responsive, utility-first, sin overhead
Backend	Node.js	18+	Integración perfecta x402-express, lightweight
Framework Backend	Express	4.18+	Middleware simple, routing clara, comunidad grande
Pagos	CDP Embedded Wallet	Latest	Onboarding sin fricción, sin extensiones
Protocolo pago	x402 Facilitator	Latest	Micropagos automáticos, verificación serverless
SDK CDP	@coinbase/coinbase-sdk	Latest	Testnet + balance + bloqueo contenido
Orquestación IA	n8n	Latest	Workflow visual, webhook, sin servidor
Modelo IA	OpenAI GPT-4o	Latest	Recomendaciones contextuales, rápidas

Componente	Tecnología	Versión	Justificación
HTTP Client	axios	1.6+	Request sencillos, interceptores
Validación	zod / yup	Latest	Type-safe schemas, backend validation
Data	JSON/arrays estáticos	N/A	Rápido MVP, sin DB overhead
Network	Base Sepolia testnet	N/A	Testnet oficial CDP, bajo costo
Explorer	Basescan	N/A	Ver transacciones confirmadas
Deploy	Vercel (frontend) + Railway/Render (backend)	N/A	Gratis, rápido, visible para jurado
CI/CD	GitHub Actions (opcional)	N/A	Tests automáticos, deploy workflow
Repo	GitHub	N/A	Control de versiones, commits ordenados

5. Flujos del usuario en el MVP

5.1 Onboarding y Conexión



7. Usuario ve botón "Get Testnet USDC" (Faucet) |

↓

8. Hace clic, recibe 0.1 USDC vía Faucet API |

↓

9. Balance actualizado a 0.1 USDC en tiempo real |

↓

10. Usuario listo para usar plataforma ☒ |

Puede navegar a "/series" y explorar contenido |

Tiempo esperado: <2 minutos.

Herramientas CDP usadas: Embedded Wallet, Faucet API, Token Balances API.

5.2 Lectura y pago de capítulo premium

1. Usuario en página de serie: /series/[id] |

↓

2. Ve listado de capítulos con imágenes pre-cargadas |

↓

3. Capítulos 1-4: "FREE" ☒ (sin lock) |

Capítulo 5: "PREMIUM" ☐ (0.01 USDC) |

↓

4. Hace clic botón "Read Chapter 5" |

↓

5. Frontend hace request: |

GET /api/chapters/5 (sin headers de pago) |

↓

| 6. Backend responde: HTTP 402 Payment Required |
| Body: { paymentRequired: true, amount: "0.01" } |

↓

| 7. Frontend muestra modal: "Paga 0.01 USDC para leer" |
| Botón: "Authorize Payment" |

↓

| 8. Usuario hace clic, CDP Wallet abre para confirmar |

↓

| 9. x402 Facilitator valida pago automáticamente |
| Transacción se procesa en Base Sepolia |

↓

| 10. Confirmación en Basescan (tx hash mostrado) |

↓

| 11. Backend guarda en historial (mock JSON) |
| { chapter: 5, paid: true, amount: "0.01", date } |

↓

| 12. Frontend recibe respuesta 200 OK con contenido |
| Capítulo 5 mostrado en pantalla (imágenes) |

↓

| 13. Balance actualizado: 0.09 USDC |
| Capítulo 5 marcado como "Owned" ☒ |

↓

| 14. Usuario puede releer sin volver a pagar |
| Request futuro retorna 200 directo |

5.3.3 Paso 3: Usuario hace clic en "Get Recommendation" - Alert de costo

1. Usuario hace clic en botón fixed en la esquina inferior derecha |
"Get Recommendation" -> se despliega un chat flotante que soporta markdown |

2. Frontend muestra MODAL DE CONFIRMACIÓN: |

Confirmation Required |
This recommendation will cost: |
0.01 USDC |
Current balance: 0.09 USDC ☒ |

[Cancel] [Proceed with Payment] |

3. Usuario tiene dos opciones: |
✓ Cancel: cierra modal, vuelve al formulario |
✓ Proceed: continúa al pago |

5.3.4 Paso 4: Usuario autoriza - x402 Payment Required

1. Usuario hace clic "Proceed with Payment" |

2. Frontend hace POST request al backend: |
POST /api/recommendations/request |
BODY (JSON): |
{ |
"userId": "0x1234567890abcdef", |
"genre": "action", |
"tone": "dark", |
"type": "manga", |

```
| "preferences": "With superpowers, protagonist..." |  
| } |
```

↓

```
| 3. Backend (Express) recibe request |  
| x402 middleware valida: ¿hay pago? |  
| Respuesta: 402 Payment Required |  
| |  
| RESPONSE: |  
| HTTP/1.1 402 Payment Required |  
| Content-Type: application/json |  
| { |  
| "paymentRequired": true, |  
| "amount": "0.01", |  
| "currency": "USDC", |  
| "paymentId": "rec_12345", |  
| "message": "Payment required to generate..." |  
| } |
```

↓

```
| 4. Frontend recibe 402, abre CDP Wallet |  
| para autorizar pago de 0.01 USDC |
```

5.3.5 Paso 5: Usuario autoriza pago en CDP Wallet

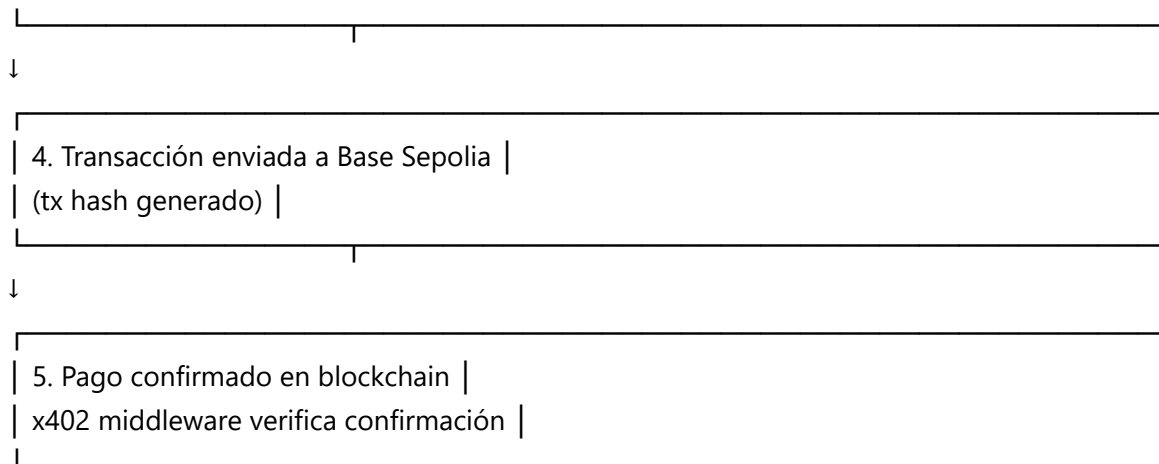
```
| 1. CDP Wallet modal abre automáticamente |  
| Mostrando confirmación de pago: |  
| "Authorize 0.01 USDC payment" |
```

↓

```
| 2. Usuario ve detalles: |  
| Destinatario: Platform Wallet (0xABC...) |  
| Cantidad: 0.01 USDC |  
| Red: Base Sepolia |  
| Botones: [Cancel] [Confirm] |
```

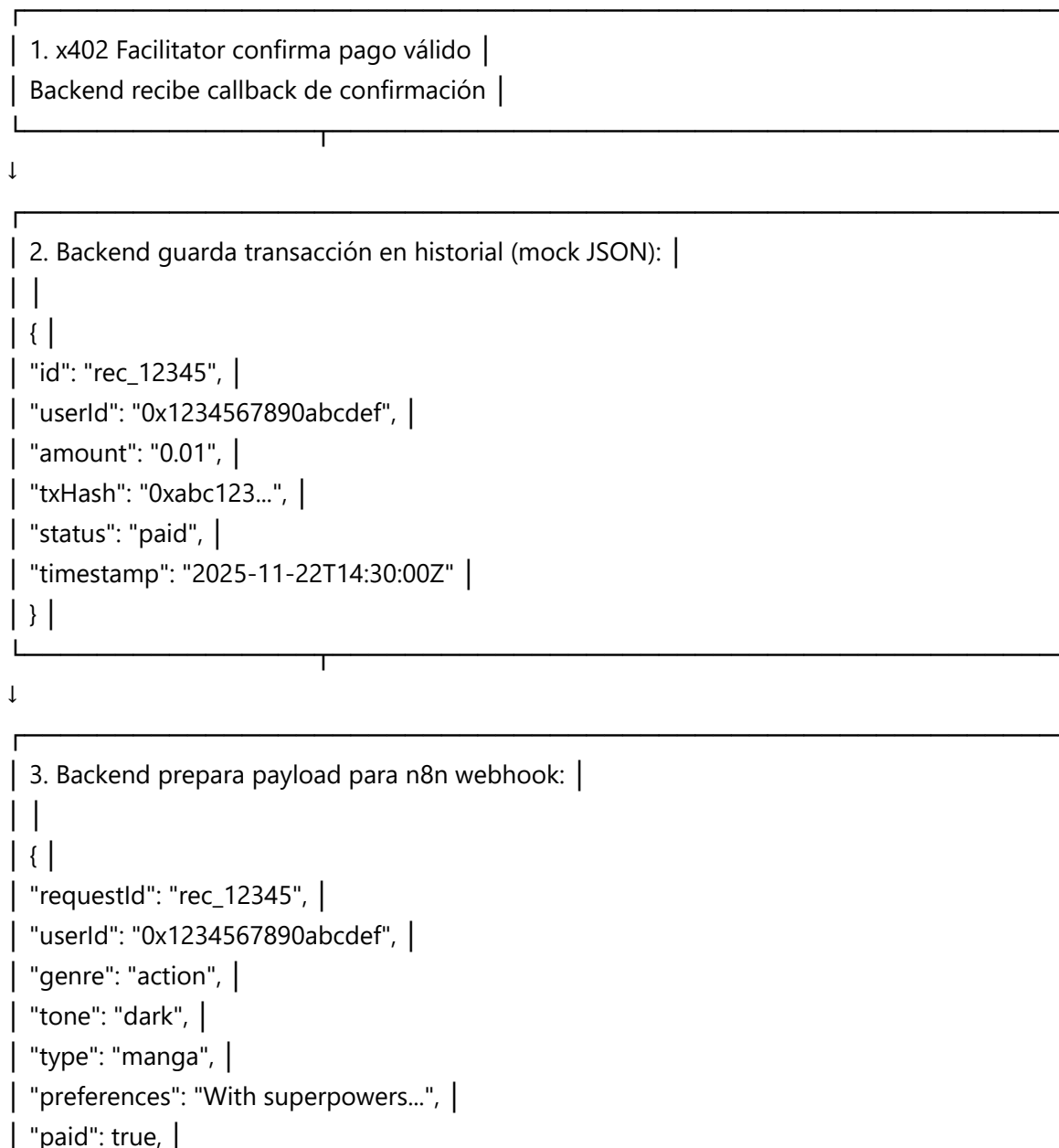
↓

```
| 3. Usuario hace clic "Confirm" |  
| x402 Facilitator procesa pago |
```

Herramientas CDP usadas: x402 Facilitator, Embedded Wallet.

5.3.6 Paso 6: Backend valida pago y gatilla webhook a n8n



```
| "amount": "0.01", |  
| "txHash": "0xabc123..." |  
| } |
```

↓

```
| 4. Backend envía POST a n8n webhook URL: |  
| POST https://n8n-instance/webhook/recommendation |  
| |  
| Headers: |  
| { |  
| "Content-Type": "application/json", |  
| "Authorization": "Bearer n8n-token-xyz" |  
| } |  
| |  
| Body: (payload arriba) |
```

↓

```
| 5. Backend responde al frontend: |  
| HTTP 200 OK |  
| { |  
| "success": true, |  
| "message": "Payment received. Processing...", |  
| "requestId": "rec_12345", |  
| "estimatedTime": "5-10 seconds" |  
| } |
```