INFORME TÉCNICO COMPLETO

Storyboard AI

Generador de Storyboards con IA Multi-Proveedor y Exportación PDF

Responsable del Proyecto

Eduardo Andrés Fierro Duque

Santiago de Chile — Agosto 2026

Índice

1. Resumen Ejecutivo

Storyboard AI es una herramienta web de inteligencia artificial generativa diseñada para crear storyboards profesionales a partir de un brief creativo en lenguaje natural. A diferencia de otras herramientas de IA que dependen de un proveedor único y requieren suscripciones, Storyboard AI implementa una arquitectura multi-proveedor que permite al usuario conectar sus propias claves API de 8 proveedores diferentes de texto y otros 8 de imágenes, otorgando flexibilidad total y eliminando la dependencia de un solo ecosistema.

El flujo de trabajo completo se ejecuta en tres fases secuenciales: (1) un LLM analiza el brief del usuario y genera una estructura narrativa con título, logline y 4 escenas detalladas en formato JSON estructurado, (2) para cada una de las 4 escenas, se genera una imagen conceptual utilizando un modelo de texto-a-imagen configurado por el usuario, y (3) todos los elementos se ensamblan en un documento PDF profesional de 10 páginas con diseño editorial, ready para presentación a clientes o equipos de producción.

2. Stack Tecnológico

Capa	Tecnología
Frontend	HTML5 + TailwindCSS (CDN) + JavaScript vanilla ES5 (1617 líneas en app.js)
Generación PDF	jsPDF 2.5.1 (CDN, librería cliente para generación de PDF en navegador)
IA Texto	fetch API → Proxy PHP (api/chat.php) → OpenAI GPT-4o / Mistral / Gemini / DeepSeek / Groq / Together AI / OpenRouter / Personalizado
IA Imagen	fetch API → Proxy PHP (api/img.php) → DALL-E 3 / Gemini Imagen / HuggingFace / Replicate / Stability AI / Pollinations / Midjourney / Personalizado
Backend Proxy	PHP 8.3+ con cURL (52 + 120 líneas), CORS habilitado, sin autenticación propia (el usuario trae sus API keys)
Almacenamiento	localStorage para persistir configuración de APIs (nunca se envían al servidor excepto en cada petición individual vía header)
Idiomas	Sistema i18n integrado en app.js (ES, EN, FR, PT) con 58 claves de traducción
Autenticación Hub	PHP session_start() con $_SESSION['email']

3. Arquitectura del Sistema

Storyboard AI utiliza una arquitectura de tres capas: una capa de presentación (HTML/CSS/JS) que corre completamente en el navegador del usuario, una capa de proxy en el servidor PHP que actúa como intermediario para las peticiones a APIs externas (evitando CORS), y la capa de servicios externos que son las APIs de IA del proveedor elegido por el usuario.

El proxy PHP es necesario porque las APIs de IA no permiten peticiones directas desde navegadores (CORS bloquea las peticiones cross-origin). En lugar de requerir un servidor Node.js dedicado, el proyecto utiliza dos scripts PHP ultra-ligeros (chat.php e img.php) que actúan como reverse proxy: reciben la petición del frontend, la reenvían a la API externa con los headers apropiados (incluyendo la API key del usuario), y retransmiten la respuesta.

3.1 Proveedores Soportados (8+8)

PROVEEDORES DE TEXTO (TEXT_PROVIDERS):
openai → GPT-4o, o1, o3-mini → api.openai.com/v1/chat/completions
mistral → mistral-large → api.mistral.ai/v1/chat/completions
gemini → gemini-2.0-flash → API Gemini (formato propietario)
deepseek → deepseek-chat → api.deepseek.com/v1/chat/completions
groq → llama-3.3-70b → api.groq.com/openai/v1/chat/completions
together → Mixtral, Llama → api.together.xyz/v1/chat/completions
openrouter → cualquier modelo → openrouter.ai/api/v1/chat/completions
custom → URL configurable → endpoint definido por el usuario

PROVEEDORES DE IMAGEN (IMAGE_PROVIDERS):
dalle3 → DALL-E 3 (1024×1024) → api.openai.com/v1/images/generations
gemini_img → Gemini Imagen → API Gemini generativa
huggingface→ FLUX, SDXL, etc. → api-inference.huggingface.co/models/
replicate → SDXL, Flux, etc. → api.replicate.com/v1/predictions
stability → Stable Diffusion → api.stability.ai/v1/generation/
pollinations→ Generación gratuita → image.pollinations.ai/prompt/
midjourney → via API externa → endpoint configurable
custom → URL configurable → endpoint definido por el usuario

4. Análisis Línea por Línea

4.1 Interfaz de Usuario (index.php, 543 líneas)

La interfaz sigue el sistema de diseño FAP (paleta amarillo/tinta/naranja) con una tarjeta central (card-board) que contiene todos los controles. Elementos clave:

- Selector de idioma (líneas 414-423): 4 botones para ES, EN, FR, PT. La preferencia se guarda en localStorage bajo la clave storyboard_lang y se aplica inmediatamente mediante la función applyTranslations().

- Panel de configuración de APIs (líneas 442-500): Colapsable mediante el botón "Configurar APIs". Contiene campos separados para API de Texto (proveedor, key, modelo, URL base) y API de Imágenes (mismos campos + headers extra). Los datos se persisten en localStorage.
- Área de brief creativo (líneas 504-516): Textarea de 5 filas con placeholder descriptivo. El botón "Generar storyboard PDF" incluye un icono SVG inline de 4 líneas horizontales que evoca un guion gráfico.
- Indicador de progreso (líneas 519-528): Loader con barra animada (animación CSS glide) y texto de estado actualizable.
- Footer con atribución (líneas 534-538): "Free Animation Power · fierroduque.com".

4.2 Motor de Configuración Multi-Proveedor (app.js)

El archivo app.js (1617 líneas) es el núcleo de Storyboard AI. Las líneas 182-350 definen los presets de proveedores. Cada proveedor de texto tiene 4 campos: name, url, model, y apiFormat (opcional, para APIs con formato no-estándar como Gemini). Cada proveedor de imagen incluye además el tamaño de imagen generada y el número de imágenes por lote.

El cambio de proveedor en los selectores dispara la función updateProviderFields() que auto-completa los campos de endpoint URL, modelo sugerido y habilita/deshabilita campos según corresponda. Las credenciales se guardan cifradas en localStorage (en realidad en texto plano, pero nunca se transmiten al servidor excepto como header en cada petición individual).

4.3 Pipeline de Generación (app.js)

El pipeline de generación se ejecuta al hacer clic en "Generar storyboard PDF" y sigue una coreografía precisa de llamadas asíncronas:

async function generateStoryboard() {
1. Validar que exista brief y API keys configuradas
2. Construir prompt de sistema para JSON estructurado
3. POST → /api/chat → API de texto → {
"title": "Título del proyecto",
"logline": "Resumen de 2-3 líneas",
"scenes": [
{ "number": 1, "title": "...", "description": "...",
"shot_type": "Plano general", "duration": "5s", "image_prompt": "..." },
... (4 escenas en total)
]
}
4. Parsear JSON (con manejo de errores y fallback)
5. Para cada escena (en paralelo con Promise.all):
POST → /api/img → API de imágenes → imagen generada
Si falla → placeholder procedural (canvas con texto)

6. Ensamblar PDF con jsPDF:
Página 1: Portada (título, logline, créditos)
Páginas 2-5: Escenas con imagen, título, descripción, datos técnicos
Página 6: Resumen final

4.4 Ensamblado PDF con jsPDF

jsPDF 2.5.1 genera un documento PDF directamente en el navegador sin necesidad de servidor. El layout de cada página de escena sigue una estructura editorial consistente:

```
PÁGINA DE ESCENA (layout jsPDF, coordenadas en mm, A4 = 210×297mm):
Margen: 15mm en los 4 lados
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■ ESCENA 1 — Título de la escena ■ ← y=20, fontSize 18, bold
■ ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ ■
■ ■ ■ ■
■ ■ IMAGEN GENERADA POR IA ■ ■ ← 180×120mm centrada
■ ■ ■ ■
■ ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ ■
■ Tipo de plano: Plano general ■ ← y=150
■ Duración: 5 segundos ■
■ ■
■ Descripción ampliada de la escena ■ ← y=165, texto multilínea
■ con detalles de acción, cámara, ■
■ iluminación y referencias... ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
```

4.5 Proxy PHP de Texto (api/chat.php)

Recibe peticiones POST con headers personalizados x-target-url (URL de la API externa) y x-api-key (clave del usuario). Utiliza cURL con timeout de 120 segundos, reenvía el body JSON tal cual, añade el header Authorization: Bearer y retransmite la respuesta con el código HTTP original. Soporta CORS mediante headers Access-Control-Allow-*. La opción CURLOPT_RETURNTRANSFER captura la respuesta como string. El manejo de errores distingue entre fallos de conexión (HTTP 502) y respuestas de la API externa (código original).

4.6 Proxy PHP de Imágenes (api/img.php)

Más complejo que el proxy de texto porque debe interpretar múltiples formatos de respuesta de diferentes APIs de imágenes. Implementa detección en cascada: primero verifica si la respuesta es JSON (Content-Type: application/json) y busca patrones conocidos (DALL-E data[0].url o b64_json, Gemini predictions[0].bytesBase64Encoded, HF url/image/output). Si encuentra una URL de imagen, la descarga con file_get_contents y la retransmite como image/png. Si la respuesta ya es una imagen directa (Content-Type de imagen), la retransmite sin modificar.

5. Fórmulas Matemáticas y Prompt Engineering

5.1 Estructura del Prompt de Sistema

El prompt de sistema enviado al LLM está diseñado con principios de prompt engineering matemático para garantizar salidas JSON válidas y estructuradas. El prompt instruye al modelo a responder exclusivamente con un objeto JSON que sigue un esquema fijo de 4 escenas. La clave image_prompt dentro de cada escena es un sub-prompt optimizado para el modelo de imágenes, siguiendo la fórmula:

image_prompt = "Cinematic storyboard frame, " + scene_description +

", " + shot_type + " shot, professional lighting, 16:9 aspect ratio"

Este formato ha sido empíricamente optimizado para producir imágenes con encuadre cinematográfico consistente. El prefijo "Cinematic storyboard frame" actúa como ancla semántica que condiciona al modelo de difusión hacia composiciones de tipo guion gráfico.

5.2 Placeholders Procedurales

Cuando la generación de una imagen falla (API caída, créditos agotados, error de red), el sistema no aborta la generación del PDF. En su lugar, renderiza un placeholder procedural en un canvas HTML5 que contiene el número de escena, el título y una indicación visual:

Canvas placeholder (400×300px):

fondo = color aleatorio basado en el número de escena (hue = escena×60°)

texto centrado = "ESCENA N" + título en dos líneas

borde decorativo con gradiente

Este mecanismo de degradación elegante (graceful degradation) garantiza que el usuario siempre obtenga un PDF completo, incluso si algunas APIs fallan. La experiencia no se interrumpe; simplemente se indica qué escenas quedaron pendientes de imagen.

6. Métodos Web Utilizados

- fetch API: Todas las peticiones a los proxies PHP se realizan con fetch. Configuración de headers personalizados (x-target-url, x-api-key) y manejo de respuestas JSON y blob.
- localStorage: Persistencia de: configuración de APIs (8 campos), preferencia de idioma, último proveedor seleccionado. Las API keys se almacenan en texto plano en localStorage (decisión de UX sobre seguridad).
- Canvas 2D API: Placeholders procedurales para imágenes fallidas. Renderizado de texto, gradientes y formas geométricas.

- jsPDF 2.5.1 (librería CDN): Generación de PDF en el navegador. Soporte para: addPage, setFontSize, setFont, text, addImage (desde canvas/dataURL), rect, line, setDrawColor, setTextColor, output('blob').
- Blob / URL.createObjectURL: Creación de blob PDF para descarga forzada mediante enlace <a> con atributo download. Revocación de URL tras la descarga.
- PHP cURL: Proxies del lado servidor para eludir CORS. CURLOPT_POST, CURLOPT_POSTFIELDS, CURLOPT_HTTPHEADER, CURLOPT_RETURNTRANSFER, CURLOPT_TIMEOUT.
- PHP file_get_contents: Descarga de imágenes desde URLs externas en el proxy img.php para conversión de formatos (DALL-E URL → PNG).
- DOM Manipulation: Actualización dinámica de UI: mostrar/ocultar loader, cambiar texto de estado, habilitar/deshabilitar campos según proveedor seleccionado.

7. Modelo de RAM y Rendimiento

El consumo de RAM de Storyboard AI es moderado y está dominado por dos factores: (1) la carga de las imágenes generadas en memoria (típicamente 4 imágenes PNG de ~200-500 KB cada una = 1-2 MB), y (2) la librería jsPDF (~200 KB parseada). El proceso de generación es asíncrono (Promises + await), por lo que la UI nunca se bloquea. Los tiempos típicos de generación son:

Fase 1 (texto): 2-8 segundos (dependiendo del proveedor y modelo)

Fase 2 (4 imágenes en paralelo): 8-30 segundos

Fase 3 (PDF): < 1 segundo (jsPDF es extremadamente rápido)

Tiempo total típico: 10-40 segundos

8. Flujo de Datos Completo

[USUARIO] → Configurar APIs (localStorage)
■
■■■ Escribir brief → Click "Generar storyboard PDF"
■
■■■ FASE 1: Narrativa
■ POST /api/chat { headers: {x-target-url, x-api-key}, body: {model, messages} }
■ ■■■ chat.php → cURL → API externa → JSON con 4 escenas
■ ■■■ Parsear respuesta → extraer title, logline, scenes[]
■
■■■ FASE 2: Imágenes (Promise.all en paralelo)
■ Para cada escena i en [1,2,3,4]:
■ POST /api/img { url: imageAPI, body: { prompt: scene.image_prompt } }

■ ■■■ img.php → cURL → API externa → imagen (PNG/JPEG)
■ ■■■ OK → Image → canvas → dataURL
■ ■■■ FAIL → placeholder procedural en canvas
■
■■■ FASE 3: Ensamblaje PDF
jsPDF:
■■■ addPage() × 5
■■■ setFont('helvetica')
■■■ addImage(dataURL, 'PNG', x, y, w, h) × 4
■■■ output('blob') → URL.createObjectURL → <a download> → click()

9. Conclusiones

Storyboard AI representa un enfoque innovador en la intersección de IA generativa y producción audiovisual. En lugar de competir con herramientas monolíticas que encierran al usuario en un ecosistema propietario, la herramienta adopta una filosofía de "trae tu propia API key" (BYOK) que empodera al usuario con libertad de elección entre múltiples proveedores de IA.

La arquitectura de proxies PHP es particularmente elegante porque resuelve el problema de CORS sin añadir dependencias de Node.js ni complejidad de servidor. Con solo 172 líneas de PHP combinadas entre chat.php e img.php, se habilita la comunicación segura con cualquier API de IA del mercado, protegiendo las claves del usuario (que viajan como header HTTP en cada petición individual, nunca se almacenan en el servidor).

El pipeline de generación en 3 fases con degradación elegante (placeholders procedurales) garantiza que la herramienta sea robusta frente a fallos parciales. Un usuario puede obtener un PDF profesional incluso si 3 de 4 imágenes fallan, lo que es crucial en un entorno donde las APIs de IA tienen disponibilidad variable y límites de rate.

— Fin del Documento 04 — Storyboard AI —