# Auditoría Técnica Forense y Hoja de Ruta Estratégica: Proyecto Anclora Press vs. Estándares del Mercado

## 1. Resumen Ejecutivo

El presente informe constituye una auditoría técnica y de mercado exhaustiva solicitada para el proyecto "Anclora Press", con un enfoque específico en la integración de modelos de lenguaje (GLM-4) y la optimización de costes mediante soluciones de código abierto. El análisis del panorama competitivo revela un mercado bifurcado y polarizado: **Vellum** domina el segmento premium con una arquitectura nativa para macOS que garantiza estabilidad absoluta pero excluye al ecosistema Windows/Linux 1; **Atticus**, el principal competidor web, ha capturado cuota de mercado mediante accesibilidad multiplataforma pero sufre de inestabilidad arquitectónica crítica, manifestada en pérdida de datos y latencia severa derivada de una sincronización defectuosa.3

Para que Anclora Press logre una ventaja competitiva sostenible con presupuesto limitado ("Zero Cost"), la estrategia no debe emular la paridad de características, sino resolver los fallos estructurales de los incumbentes. La arquitectura recomendada prioriza la **integridad epistémica del manuscrito** (arquitectura *Local-First*) y la **fidelidad de exportación** (paridad pantalla-impresión mediante Paged.js). Se propone un stack tecnológico basado en **Tiptap** (edición semántica), **Paged.js** (paginación CSS estándar), **Fabric.js** (diseño vectorial de portadas) y **Mammoth.js** (ingesta semántica), eliminando costes de licencias comerciales. La integración de la IA GLM-4 de ZhipuAI debe estratificarse, utilizando modelos "Flash" gratuitos para tareas volumétricas y reservando modelos complejos para asistencia editorial de alto valor.6

## 2. Análisis Forense del Ecosistema Competitivo

El mercado de software de autoedición ("indie publishing") se define actualmente por una tensión no resuelta entre la accesibilidad de las aplicaciones web y la robustez de las aplicaciones nativas. El análisis de las evidencias documentales permite deconstruir las capacidades y vulnerabilidades de los actores dominantes.

### 2.1. El Estándar de Oro: Vellum (La "Fortaleza Amurallada")

Posicionamiento: Líder indiscutible en calidad tipográfica y estabilidad para usuarios de macOS.

Modelo de Precios: Licencia perpetua de alto coste ($199 solo eBook, $249 Impresión+eBook).2

Arquitectura: Aplicación nativa (Objective-C/Swift) utilizando el motor de renderizado Quartz de Apple.

**Competencias Críticas:**

* **Fidelidad de Renderizado:** Al operar sobre motores gráficos nativos del sistema operativo, Vellum garantiza que la previsualización en pantalla sea idéntica al PDF exportado. Gestiona automáticamente micro-ajustes tipográficos complejos, como el control de líneas viudas y huérfanas, sin intervención del usuario.1
* **Interfaz de Usuario (UI) Curada:** Reduce la carga cognitiva del autor limitando las opciones. Ofrece aproximadamente 26 estilos predefinidos que han sido validados tipográficamente, evitando que usuarios inexpertos generen diseños visualmente pobres.2
* **Fiabilidad Offline:** La naturaleza local del software elimina la latencia de red y los vectores de fallo asociados a la sincronización en la nube, un punto crítico de venta frente a las soluciones SaaS.9

**Vulnerabilidades Explotables por Anclora:**

* **Exclusividad de Plataforma:** La restricción a macOS deja desatendido a todo el mercado de Windows y Linux. Los usuarios de PC se ven obligados a recurrir a servicios de "Mac-in-Cloud", que añaden fricción, latencia y coste recurrente.2
* **Rigidez en la Personalización:** La filosofía de "caja negra" impide ajustes granulares. Un usuario no puede modificar márgenes específicos para una sección o crear elementos de diseño no estándar (como cajas de texto personalizadas) fuera de los temas predefinidos.9

### 2.2. El Retador: Atticus (El "Híbrido Sobrestirado")

Posicionamiento: La alternativa principal para autores en Windows, comercializada como "Vellum para PC" + "Scrivener".

Modelo de Precios: Pago único de $147 (Lifetime).8

Arquitectura: Aplicación Web Progresiva (PWA) / Wrapper Electron (basada en React/Angular).

**Competencias Críticas:**

* **Universalidad:** Accesible desde cualquier navegador, permitiendo flujos de trabajo en Chromebooks, Windows y Linux.2
* **Personalización Granular:** A diferencia de Vellum, Atticus permite un control detallado sobre fuentes (más de 1.500 opciones), tamaños y espaciados, apelando a usuarios que desean control total sobre la estética.2
* **Convergencia Funcional:** Intenta unificar la fase de escritura (procesador de textos) y la de maquetación en una sola herramienta, reduciendo la fragmentación del flujo de trabajo.1

**Fallos Sistémicos y Puntos de Dolor (La Oportunidad):**

* **Corrupción de Datos y Sincronización:** Existe evidencia sustancial de usuarios reportando la eliminación de texto recién escrito, saltos de cursor y "pantallas blancas de la muerte". Estos fallos sugieren una implementación defectuosa de la lógica de colaboración (posiblemente conflictos en Transformación Operativa o CRDTs) donde el estado local y el del servidor no logran reconciliarse correctamente.3
* **Latencia de Rendimiento:** Al ser una SPA (Single Page Application) que maneja documentos masivos en el DOM, los usuarios experimentan retardos significativos al escribir o navegar entre capítulos extensos. La carga cognitiva de esperar a que la interfaz responda rompe el "estado de flujo" del escritor.11
* **Inconsistencia en la Exportación ("WYSI-NOT-WYG"):** Se han documentado discrepancias críticas entre el editor visual y el archivo final exportado, tales como márgenes invertidos, páginas en blanco inexplicables y enlaces de Tabla de Contenidos (TOC) rotos.12

### 2.3. Herramientas de Nicho y Tradicionales

* **Adobe InDesign:** Estándar industrial para maquetación compleja. Ofrece control a nivel de píxel pero presenta una curva de aprendizaje vertical y un modelo de suscripción costoso (~$20-$60/mes). Es excesivo ("overkill") para narrativa basada en texto.1
* **Scrivener:** Herramienta de escritura y organización estructural superior, pero carente de un motor de maquetación final competente. Genera archivos que requieren post-procesamiento en otras herramientas para ser publicables profesionalmente.1

### 2.4. Tabla Comparativa de Capacidades

| **Dimensión** | **Vellum (Incumbente)** | **Atticus (Retador)** | **Anclora Press (Objetivo)** |
| --- | --- | --- | --- |
| **Arquitectura** | Nativa (macOS) | Web/PWA (React) | **Web/PWA (Local-First)** |
| **Coste Licencia** | $249 | $147 | **$0 - $49 (Low Cost)** |
| **Estabilidad de Datos** | Crítica (Alta) | Comprometida (Baja) | **Crítica (Alta)** |
| **Motor de PDF** | Apple Quartz | Desconocido (Backend) | **Paged.js (Cliente/CSS)** |
| **Tipografía** | Curada (~26 fuentes) | Google Fonts (1.500+) | **Híbrida (Curada + Upload)** |
| **Diseño de Portadas** | Inexistente | Básico | **Avanzado (Canvas/Fabric)** |
| **Integración IA** | Nula | Superficial | **Nativa (GLM-4)** |

## 3. Diagnóstico de Puntos de Dolor Críticos

La investigación cualitativa en foros especializados y reseñas de usuarios permite aislar tres vectores de frustración ("Pain Clusters") que Anclora debe resolver arquitectónicamente para desplazar a la competencia.

### 3.1. Punto de Dolor A: El "Terror" a la Pérdida de Datos

La confianza del usuario en Atticus se ha erosionado debido a fallos de sincronización. Comentarios como "vi literalmente cómo borraba frases que acababa de escribir" 3 indican un fallo catastrófico en la capa de persistencia. En herramientas creativas, la integridad de los datos es el requisito no funcional más importante.

* **Causa Raíz Hipotética:** Dependencia de una conexión constante y una gestión optimista de la UI sin un mecanismo robusto de resolución de conflictos (como CRDTs) ante latencias de red.
* **Solución para Anclora:** Implementar una arquitectura **"Local-First"**. La aplicación debe escribir primero en una base de datos local en el navegador (IndexedDB/RxDB) y sincronizar con la nube en segundo plano. El estado de verdad debe residir siempre en el dispositivo del usuario, eliminando el riesgo de pérdida por desconexión.9

### 3.2. Punto de Dolor B: La Brecha "Previsualización vs. Impresión"

La promesa de "What You See Is What You Get" (WYSIWYG) se rompe frecuentemente en las aplicaciones web competidoras. Los usuarios reportan márgenes que se invierten al exportar o elementos que se desplazan.13

* **Causa Raíz Hipotética:** Uso de motores de renderizado distintos para la visualización en pantalla (HTML/CSS del navegador) y para la generación del PDF (librerías backend como wkhtmltopdf o PrinceXML). Esta discrepancia introduce errores de interpretación de estilos.
* **Solución para Anclora:** Unificar el motor de renderizado utilizando **Paged.js**. Esta librería permite renderizar el PDF directamente en el navegador utilizando los estándares CSS de medios paginados. Lo que el usuario ve en la "Vista de Impresión" es técnicamente el mismo objeto DOM que se imprimirá, garantizando una paridad del 100%.15

### 3.3. Punto de Dolor C: La Fricción de Rendimiento

La percepción de "pesadez" y lentitud en Atticus 5 degrada la experiencia de escritura.

* **Causa Raíz Hipotética:** Renderizado de documentos monolíticos donde cambios pequeños provocan re-calculos de diseño costosos en todo el documento, y carga de recursos innecesarios.
* **Solución para Anclora:** Implementar **Virtualización** de listas para la navegación de capítulos y carga diferida ("Lazy Loading") de instancias del editor. Solo el capítulo activo debe residir en la memoria de renderizado inmediato.

## 4. Arquitectura Técnica "Zero-Cost" Recomendada

Para cumplir con el mandato de "poco presupuesto" y priorizar soluciones "open source", se ha diseñado un stack tecnológico que evita licencias comerciales costosas sin sacrificar la calidad profesional.

### 4.1. Editor de Contenido: Tiptap (Headless Wrapper de ProseMirror)

Elección: Tiptap.17

Justificación Técnica:

A diferencia de Lexical (Meta), que aunque performante presenta una curva de aprendizaje pronunciada y una documentación fragmentada ("construir tu propio cargador") 19, y de CKEditor 5, cuya arquitectura monolítica dificulta la personalización profunda sin licencias comerciales 21, Tiptap ofrece el equilibrio óptimo. Se basa en ProseMirror, el estándar industrial para edición de texto rico (usado por New York Times, Atlassian), pero envuelto en una API amigable para Vue/React.

**Estrategia de Implementación:**

* **Arquitectura Headless:** Tiptap no impone una interfaz de usuario. Anclora puede construir una barra de herramientas minimalista y personalizada que imite la elegancia de Vellum, manteniendo el control total sobre la experiencia visual.
* **Semántica Estricta:** Permite definir "Nodos" personalizados. Por ejemplo, en lugar de que un usuario formatee un salto de escena con asteriscos (\*\*\*), Anclora puede implementar un nodo <SceneBreak /> que se renderice visualmente como un ornamento gráfico pero se guarde semánticamente, asegurando una exportación limpia a formatos accesibles o audio.17
* **Colaboración Gratuita:** La versión open-source de Tiptap es compatible con **Y.js**, una librería de CRDTs. Esto permite implementar colaboración en tiempo real alojando un servidor WebSocket propio (Hocuspocus) sin pagar las tarifas del servicio SaaS de Tiptap.18

### 4.2. Motor de Maquetación y Exportación: Paged.js

Elección: Paged.js.15

Justificación Técnica:

La mayoría de las soluciones web fallan al imprimir porque los navegadores están diseñados para el scrolling continuo, no para la paginación física. Paged.js actúa como un "polyfill" para las especificaciones W3C de Paged Media. Fragmenta el contenido HTML en páginas discretas basándose en dimensiones físicas (ej. 6x9 pulgadas) directamente en el cliente.

**Ventajas Competitivas:**

* **Control Tipográfico Avanzado:** Soporta propiedades CSS críticas como widows y orphans para evitar líneas sueltas al inicio o final de página, un requisito indispensable para la maquetación profesional.22
* **Contenido Generado Dinámicamente:** Permite crear encabezados y pies de página vivos (ej. mostrar el título del capítulo actual en el encabezado) mediante CSS (content: string(chapter-title)), eliminando la necesidad de scripts complejos de post-procesamiento.23
* **Coste Cero:** Es una librería de código abierto mantenida por la comunidad (Coko Foundation), sin costes de licencia por volumen de exportación como ocurre con APIs de conversión PDF comerciales (DocRaptor).24

### 4.3. Editor de Portadas: React Design Editor (Fabric.js)

Elección: React Design Editor (basado en Fabric.js).25

Justificación Técnica:

El diseño de portadas requiere manipulación de lienzos (canvas), no de DOM. Herramientas como Puck son excelentes constructores de páginas web (drag-and-drop de componentes React) pero carecen de las capacidades de composición de imagen (capas, filtros, mezcla) necesarias para una portada de libro.27 Soluciones comerciales como Polotno o IMG.LY son potentes pero costosas.29

Fabric.js proporciona una API de objetos sobre el canvas HTML5, permitiendo redimensionado, rotación y capas. El repositorio "React Design Editor" ofrece una implementación de referencia que ahorra cientos de horas de desarrollo en la creación de manejadores de selección y guías de alineación.26

**Estrategia de Implementación:**

* Implementar soporte para zonas de sangrado (*bleed*) críticas para impresión.
* Configurar la exportación del canvas con un multiplicador de escala para asegurar una resolución de 300 DPI (píxeles por pulgada), ya que los canvas web operan por defecto a 72/96 DPI.

### 4.4. Ingesta y Migración: Mammoth.js

Elección: Mammoth.js.31

Justificación Técnica:

La importación de archivos .docx es históricamente problemática, resultando en "sopa de HTML" llena de estilos en línea sucios. Mammoth.js se distingue por enfocarse en la estructura semántica en lugar de la apariencia visual.

* **Mapeo Semántico:** Convierte estilos de Word (ej. "Título 1") en etiquetas HTML semánticas (<h1>), ignorando el tamaño de fuente específico o el color usado en Word. Esto asegura que el manuscrito importado adopte limpiamente el sistema de diseño de Anclora sin arrastrar "deuda técnica" visual del archivo original.31

### 4.5. Integración Estratégica de IA (GLM-4)

Contexto: El repositorio del usuario sugiere el uso de modelos ZhipuAI (GLM-4).

Análisis de Costes:

* **GLM-4.6:** Alto rendimiento, coste significativo ($2.2/1M tokens).6
* **GLM-4-Flash:** Gratuito (o de coste extremadamente bajo) para tareas rápidas.6

Estrategia Diferencial:

Anclora puede posicionarse no solo como herramienta de formato, sino como "Asistente Editorial".

* Utilizar **GLM-4-Flash** para características de "baja latencia y alto volumen" sin coste marginal: corrección gramatical en tiempo real, sugerencias de sinónimos, y análisis de legibilidad.
* Reservar **GLM-4.6** para características "Premium": análisis de trama, generación de ideas para portadas (vía CogView si está disponible), y edición de desarrollo profunda. Esto permite un modelo de negocio freemium sostenible.

## 5. Hoja de Ruta de Mejoras Priorizada (Roadmap)

Esta hoja de ruta ordena las fases de desarrollo desde la estabilización fundamental hasta la creación de ventajas competitivas ("Moats").

### Fase 1: El "Núcleo de Hierro" (Estabilidad e Ingesta)

*Objetivo: Garantizar que la herramienta sea segura para escribir y capaz de recibir manuscritos externos.*

1. **Persistencia Local-First:** Implementar **RxDB** o **IndexedDB** para guardar cada pulsación de tecla localmente. La sincronización con la nube debe ser un proceso secundario que no bloquee la interfaz ni arriesgue datos si falla la red.9
2. **Integración de Tiptap:** Configurar el editor con extensiones básicas (Historial, Párrafos, Encabezados).
3. **Pipeline de Importación Mammoth:** Crear una interfaz de "Mapeo de Estilos" donde el usuario confirme la conversión de sus estilos de Word a la estructura de Anclora (ej. "¿'Estilo Personalizado 1' equivale a 'Cita en Bloque'?").
4. **Validación de Exportación EPUB:** Generar EPUBs válidos (ePubCheck) es más sencillo que PDF y permite validar la estructura semántica tempranamente.

### Fase 2: El "Motor Visual" (Impresión y Paginación)

*Objetivo: Igualar la calidad de salida de Vellum.*

1. **Implementación de Paged.js:** Crear una vista de "Previsualización de Impresión" que inyecte el contenido del editor en un iframe controlado por Paged.js.
2. **Sistema de Temas CSS:** En lugar de estilos rígidos, utilizar **Variables CSS** (--font-header, --margin-inner, --line-height) para controlar el diseño. Esto permite crear un "Constructor de Temas" infinito (ventaja sobre Vellum) con un coste de desarrollo casi nulo.
3. **Control de Viudas/Huérfanas:** Configurar las reglas CSS de Paged.js para forzar break-inside: avoid en párrafos cortos y asegurar cortes de página limpios.
4. **Motor de PDF:** Utilizar la API de impresión nativa del navegador (window.print() a PDF) desde la vista de Paged.js. Esto garantiza que el PDF sea *byte a byte* idéntico a la previsualización.

### Fase 3: El "Estudio Creativo" (Portadas y Metadatos)

*Objetivo: Eliminar la dependencia de herramientas externas como Canva.*

1. **Integración Fabric.js:** Incrustar el editor de canvas para la creación de portadas completas (frontal, lomo, contraportada).
2. **Calculadora de Lomo Dinámica:** Desarrollar una utilidad que calcule el ancho del lomo basándose en el recuento de páginas final (derivado de Paged.js) y el gramaje del papel seleccionado (ej. Papel Crema = 0.0025" por página). Este es un dolor crítico para autores en Amazon KDP que los competidores suelen ignorar.
3. **Biblioteca de Plantillas SVG:** Ofrecer plantillas vectoriales editables para géneros populares, reduciendo la barrera de entrada al diseño.

### Fase 4: El "Co-Piloto IA" (Ventaja Competitiva)

*Objetivo: Activar los activos GLM del repositorio.*

1. **Menú Contextual IA:** Implementar un menú flotante en Tiptap (estilo Notion/Medium) que ofrezca "Reescribir", "Expandir" o "Acortar" usando la API de GLM-4-Flash.
2. **Análisis de Tono:** Un dashboard que visualice el arco emocional del libro capítulo a capítulo.
3. **Generación de Prompts para Portadas:** Utilizar el modelo de lenguaje para ayudar al usuario a describir su escena visualmente y generar un prompt optimizado para herramientas de generación de imágenes.

## 6. Detalles de Implementación y Especificaciones Funcionales

### 6.1. Mejora del Editor: Estructura sobre Texto Plano

Problema: Los competidores a menudo tratan el texto como cadenas simples, lo que dificulta el formato avanzado.

Solución Anclora: Uso de Nodos Tiptap.

* **Saltos de Escena Semánticos:** Crear un nodo <SceneBreak> que sea un objeto inmutable en el editor. El usuario puede cambiar globalmente el icono (de una estrella \* a una hoja 🍃) con un solo clic en la configuración del tema, sin necesidad de "Buscar y Reemplazar" en todo el texto.
* **Páginas de Legales:** Un nodo bloqueado que autocompleta ISBN y año, evitando errores humanos en los datos críticos de publicación.

### 6.2. Proceso de Exportación: "Targeting" por Dispositivo

Problema: Vellum genera archivos genéricos que a veces fallan en dispositivos específicos.

Solución Anclora: Consultas de Medios CSS (Media Queries).

* Utilizar *hooks* de Paged.js para inyectar CSS específico según el destino:
  + @media print: Imágenes de alta resolución (300 DPI), perfil de color CMYK (vía simulación CSS), fuentes con serifa para legibilidad en papel.
  + @media screen: Imágenes RGB optimizadas para web, fuentes sans-serif o seleccionadas por el usuario para lectura en pantalla.

## 7. Análisis de Presupuesto y Licencias (Zero-Cost)

| **Componente** | **Tecnología Recomendada** | **Licencia** | **Coste** | **Notas Estratégicas** |
| --- | --- | --- | --- | --- |
| **Frontend** | React + Vite | MIT | $0 | Estándar industrial, alta velocidad de desarrollo. |
| **Editor Texto** | **Tiptap (Core)** | MIT | $0 | Usar el núcleo open-source; construir UI propia para evitar pagar extensiones "Pro". |
| **Canvas Portada** | **React Design Editor** | MIT | $0 | Wrapper de Fabric.js. Ahorra semanas de desarrollo base. |
| **Motor PDF** | **Paged.js** | MIT | $0 | Polyfill estándar W3C. Sin coste por documento generado. |
| **Iconografía** | Phosphor / Lucide | MIT | $0 | Iconos modernos y consistentes. |
| **Tipografía** | Google Fonts | OFL | $0 | Cachear localmente para soportar el modo offline. |
| **IA Backend** | **GLM-4-Flash** | Comercial | Gratis\* | \*Aprovechar el tier gratuito/promocional actual de ZhipuAI. |

## 8. Matriz de Confianza

| **Categoría** | **Nivel de Confianza** | **Justificación de la Evaluación** |
| --- | --- | --- |
| **Características de Competidores** | **Alta** | Documentación directa de precios y listas de características oficiales.2 |
| **Puntos de Dolor de Usuarios** | **Alta** | Corroboración múltiple en hilos de Reddit independientes y reseñas de terceros.3 |
| **Stack Tecnológico** | **Alta** | Paged.js y Tiptap son estándares industriales con repositorios activos y maduros. |
| **Precios API GLM** | **Media** | Los precios de IA son volátiles; el modelo "Flash" gratuito podría cambiar condiciones.6 |
| **Integración Fabric.js** | **Media** | El repositorio "React Design Editor" tiene 4 años de antigüedad 25; requerirá refactorización para React 18/19. |

## 9. Manejo de Incertidumbre y Conflictos de Datos

Conflicto 1: Fiabilidad de la Colaboración en Atticus.

Fuente A (Marketing de Atticus): Promociona la colaboración como característica clave.

Fuente B (Reportes de Usuarios): Documenta corrupción de datos severa.4

Resolución: Este informe prioriza la Fuente B como evidencia forense del estado actual del software, recomendando a Anclora evitar la colaboración en tiempo real en la Fase 1 para garantizar la estabilidad.

Conflicto 2: Soporte de Vellum para Windows.

Fuente A: Sugiere "soluciones" como Mac-in-Cloud.

Fuente B: Afirma que Vellum es exclusivo de Mac.

Resolución: Técnicamente, Vellum es exclusivo de Mac. Los servicios en la nube son soluciones de hosting, no compatibilidad de software. El informe trata a Vellum como "Solo Mac" para resaltar el vacío de mercado.

Conflicto 3: "Gratuidad" de la IA.

Fuente A: Lista GLM-4-Flash como gratuito.

Fuente B: Nota "Gratis por tiempo limitado".6

Resolución: Se recomienda construir un backend de IA modular que permita cambiar de modelo si ZhipuAI elimina la capa gratuita, asegurando que el modelo de negocio de la app no se rompa por subidas de precios de API.

## 10. Conclusión

Anclora Press se encuentra ante una ventana estratégica única. El líder del mercado (Vellum) ignora deliberadamente a los usuarios fuera del ecosistema Apple, y el principal retador (Atticus) ha comprometido su estabilidad técnica al priorizar la colaboración en la nube sobre la integridad local de los datos.

Al adoptar una doctrina de **"Estabilidad Primero"** —apoyada en **Tiptap** para la edición estructurada, **Paged.js** para un renderizado PDF fiable y CSS-compliant, y **Fabric.js** para el diseño gráfico— Anclora puede entregar una experiencia de "Vellum para Windows" con un coste operativo cercano a cero. La integración de **GLM-4** proporciona un diferenciador moderno que ataca el problema de la "Hoja en Blanco", posicionando a Anclora no solo como una herramienta de formato, sino como un entorno integral de autoría.

La prioridad inmediata debe ser construir un Producto Mínimo Viable (MVP) capaz de ingerir un documento de Word y producir un PDF perfecto mediante Paged.js, verificando así la promesa de fidelidad que Atticus está fallando en cumplir actualmente.

#### Obras citadas

1. The Best Book Formatting Software for Authors - Barker Books Publishing, fecha de acceso: diciembre 13, 2025, <https://barkerbooks.com/best-book-formatting-software/>
2. Atticus vs Vellum: A Side-by-Side Comparison [2025] - Kindlepreneur, fecha de acceso: diciembre 13, 2025, <https://kindlepreneur.com/atticus-vs-vellum/>
3. Atticus software : r/selfpublish - Reddit, fecha de acceso: diciembre 13, 2025, <https://www.reddit.com/r/selfpublish/comments/1iw8yc7/atticus_software/>
4. New to this... PSA: don't use Atticus.io : r/selfpublish - Reddit, fecha de acceso: diciembre 13, 2025, <https://www.reddit.com/r/selfpublish/comments/1k8eeda/new_to_this_psa_dont_use_atticusio/>
5. Atticus Formatting Software Unusable : r/selfpublish - Reddit, fecha de acceso: diciembre 13, 2025, <https://www.reddit.com/r/selfpublish/comments/1iq6ejf/atticus_formatting_software_unusable/>
6. Pricing - Z.AI DEVELOPER DOCUMENT, fecha de acceso: diciembre 13, 2025, <https://docs.z.ai/guides/overview/pricing>
7. Product Pricing - ZHIPU AI OPEN PLATFORM, fecha de acceso: diciembre 13, 2025, <https://bigmodel.cn/pricing>
8. Best Book Formatting Software: 2025 Update [+ Discounts] - Kindlepreneur, fecha de acceso: diciembre 13, 2025, <https://kindlepreneur.com/book-formatting-software/>
9. Vellum vs. Atticus for Non-Fiction Interior Book Design - Jeremy B. Shapiro, fecha de acceso: diciembre 13, 2025, <https://www.jeremyshapiro.com/2025/06/vellum-vs-atticus-for-non-fiction-interior-book-design/>
10. Top 10 Book Formatting Services in 2025 (Free & Paid) - PaperTrue, fecha de acceso: diciembre 13, 2025, <https://www.papertrue.com/blog/book-formatting-services/>
11. Atticus, Vellum, or Reedsy? : r/selfpublish - Reddit, fecha de acceso: diciembre 13, 2025, <https://www.reddit.com/r/selfpublish/comments/1ici9cb/atticus_vellum_or_reedsy/>
12. What is the best tool for book formatting ? Vellum vs Atticus : r/KDP - Reddit, fecha de acceso: diciembre 13, 2025, <https://www.reddit.com/r/KDP/comments/1i5sixl/what_is_the_best_tool_for_book_formatting_vellum/>
13. Is Atticus worth the money? : r/selfpublish - Reddit, fecha de acceso: diciembre 13, 2025, <https://www.reddit.com/r/selfpublish/comments/15yc7c1/is_atticus_worth_the_money/>
14. Top Essential Tools for Self-Published Authors in 2025 - Spines, fecha de acceso: diciembre 13, 2025, <https://spines.com/top-essential-tools-for-self-published-authors-in-2024/>
15. pagedjs and react - CodeSandbox, fecha de acceso: diciembre 13, 2025, <https://codesandbox.io/s/pagedjs-and-react-xlod3>
16. Paged.js - a free and open source JavaScript library that paginates content in the browser to create PDF output from any HTML content. This means you can design works for print (eg. books) using HTML and CSS - Reddit, fecha de acceso: diciembre 13, 2025, <https://www.reddit.com/r/javascript/comments/f5syqi/pagedjs_a_free_and_open_source_javascript_library/>
17. CKEditor vs. Tiptap: Built for What's Next, fecha de acceso: diciembre 13, 2025, <https://tiptap.dev/alternatives/ckeditor-vs-tiptap>
18. Tiptap vs Lexical: Which Rich Text Editor Should You Pick for Your Next Project? - Medium, fecha de acceso: diciembre 13, 2025, <https://medium.com/@faisalmujtaba/tiptap-vs-lexical-which-rich-text-editor-should-you-pick-for-your-next-project-17a1817efcd9>
19. Tiptap Editor: The Ultimate AI-Powered Toolkit for Modern Content Creation, fecha de acceso: diciembre 13, 2025, <https://skywork.ai/skypage/en/Tiptap-Editor-The-Ultimate-AI-Powered-Toolkit-for-Modern-Content-Creation/1972858043986800640>
20. Why I chose Lexical over Tiptap - DEV Community, fecha de acceso: diciembre 13, 2025, <https://dev.to/codeideal/why-i-chose-lexical-over-tiptap-38nd>
21. CKEditor vs Tiptap: Choose the editor that's built to scale, fecha de acceso: diciembre 13, 2025, <https://ckeditor.com/ckeditor-vs-tiptap/>
22. How to use Paged.js, fecha de acceso: diciembre 13, 2025, <https://pagedjs.org/en/documentation/>
23. Using PagedJS with React - doppio.sh API - HTML to PDF, fecha de acceso: diciembre 13, 2025, <https://doc.doppio.sh/article/using-pagedjs-with-react.html>
24. DocRaptor vs. WeasyPrint: A PDF Export Showdown - DEV Community, fecha de acceso: diciembre 13, 2025, <https://dev.to/thawkin3/docraptor-vs-weasyprint-a-pdf-export-showdown-34f>
25. GitHub - bharathreddyza/react-design-editor, fecha de acceso: diciembre 13, 2025, <https://github.com/bharathreddyza/react-design-editor>
26. GitHub - salgum1114/react-design-editor, fecha de acceso: diciembre 13, 2025, <https://github.com/salgum1114/react-design-editor>
27. Puck - Open Source Alternatives, fecha de acceso: diciembre 13, 2025, <https://www.opensourcealternatives.to/item/puck>
28. Puck - Agentic visual editor for React, fecha de acceso: diciembre 13, 2025, <https://puckeditor.com/>
29. Modern image and video editing platform for creators and developers, fecha de acceso: diciembre 13, 2025, <https://polotno.com/>
30. Polotno SDK vs IMG.LY – a detailed head to head comparison, fecha de acceso: diciembre 13, 2025, <https://polotno.com/sdk/product/compare/polotno-sdk-vs-imgly>
31. mwilliamson/mammoth.js: Convert Word documents (.docx files) to HTML - GitHub, fecha de acceso: diciembre 13, 2025, <https://github.com/mwilliamson/mammoth.js/>