**INSTITUTO POLITÉCNICO NACIONAL**

CENTRO DE ESTUDIOS CIENTÍFICOS Y TECNOLÓGICOS No. 3
"ESTANISLAO RAMÍREZ RUÍZ"

# DESARROLLO DE UNA PLATAFORMA WEB PARA LA CREACIÓN Y PUBLICACIÓN TEMPORAL AUTOMATIZADA DE CONTENIDO EDUCATIVO Y CREATIVO

**TESIS**

QUE PARA OBTENER EL TÍTULO DE:
**TÉCNICO EN COMPUTACIÓN**

**PRESENTAN:**

- **CASTRO AGUILAR EDER JOEL**
- **HERNÁNDEZ TELLEZ HÉCTOR FIDEL**
- **VALENCIA OROPEZA ANGEL YAHIR**

**ASESORES:**
ING. JOSÉ ERWIN RODRÍGUEZ PACHECO

ECATEPEC DE MORELOS, 14 DE MAYO DE 2025

# Índice general

# INTRODUCCIÓN

La creación de contenido didáctico que permita facilitar el aprendizaje o la difusión de mensajes mediante el uso de herramientas digitales es una necesidad fundamental en distintos ámbitos como el académico y en la cotidianidad misma. En la actualidad existe una variedad de herramientas que facilitan la creación de todo tipo de materiales visuales y funcionales como lo son infografías, collages y presentaciones, y que pueden ser útiles para los usuarios que se especializan en el ámbito del diseño. Sin embargo, la mayor limitante es el enfoque con que se utiliza cada herramienta.

Muchas de estas opciones cuentan con un desequilibrio, un impedimento técnico, ya que algunas están dedicadas a un público amplio, y cuentan con funcionalidades básicas y generales que limitan a los usuarios expertos. Por otra parte, otras herramientas son altamente complejas y dificultan el uso de los principiantes debido a la curva de aprendizaje que implican.

Este proyecto tiene como objetivo general desarrollar una plataforma web que combine lo mejor de ambos tipos de herramientas: la simplicidad y la funcionalidad avanzada para crear, editar y publicar contenidos útiles. De esta manera se puede utilizar por todo tipo de usuarios sin que represente un desafío para alguno de los dos sectores.

Se trata de una iniciativa accesible para todas las personas, que permite a todo tipo de personas crear diseños fácilmente, y que cuenta con documentación clara que invita a los usuarios a aprender a utilizar la plataforma para explotar su potencial.

Su principal ventaja, al momento de crear un diseño, es que cuenta con una interfaz dual. Esto es, incluye un editor que puede utilizarse en dos modos: el básico, que cuenta con una interfaz sencilla y automatización, y el avanzado, que ofrece personalización profunda y control granular. De esta manera, cada usuario puede ajustar su experiencia a su nivel de habilidad.

Así mismo, este proyecto tiene una aplicación fundamental en el ámbito de la educación, ya que los profesores y alumnos de una escuela cuentan con la ventaja de poder crear y difundir contenidos que resumen tópicos académicos y organizan visualmente la información, con el fin de facilitar el aprendizaje.

La opción de publicar los contenidos elaborados dentro de la misma plataforma es esencial, ya que permite que todos los usuarios interesados puedan acceder a los diseños que otros miembros han creado con anterioridad. Así es como esta herramienta se convierte

en una que, además de contar con una gran utilidad como editor, tiene una comunidad que la mantiene con dinamismo.

Así también, los diseños se comparten de forma segura mediante enlaces únicos, logrando que los usuarios puedan reutilizar los contenidos como base de otros diseños futuros, al implementar la exportación sencilla de estos y su consulta en las bases de datos.

La integración de Inteligencia Artificial dentro de la aplicación es otra de las ventajas para los que la utilizan. La capacidad de sugerir y generar contenido que se basa en diseños anteriores permite que todos los usuarios puedan aprovechar las funcionalidades de la herramienta mediante un asistente que los apoya al momento de crear un diseño funcional.

Es entonces que este proyecto satisface la necesidad de todo tipo de personas de crear, diseñar, consultar y difundir contenidos gráficos útiles de forma accesible, sin sacrificar la eficiencia que otras herramientas brindan.

# Capítulo 1

# MARCO TEÓRICO

El presente marco teórico sienta las bases conceptuales y referenciales que sustentan el desarrollo de una plataforma web para la creación, edición y publicación de contenido digital, con un enfoque dual que atiende tanto a usuarios principiantes como avanzados. Se explorarán los conceptos clave relacionados con la creación de contenido digital, la inteligencia artificial en este ámbito, las tecnologías de desarrollo web, la seguridad en la publicación y la experiencia de usuario.

## 1.1. Creación de contenidos digitales

Hoy en día, es esencial la utilización de medios digitales para transmitir ideas. El desarrollo de contenidos digitales hace referencia al proceso de planificar, generar y difundir contenidos mediante medios digitales. Estos contenidos pueden ser expuestos en varios formatos, tales como textos, imágenes, videos, audios, etc. El objetivo esencial de este proceso es cautivar a determinada audiencia mediante el contenido que se expone.

### 1.1.1. Importancia en entornos educativos

La implementación de contenidos digitales ha jugado un papel fundamental en el sector educativo contemporáneo, pues su importancia se ha visto reflejada en diversas actividades propias de ambos entornos.

En la actualidad, la creación de contenidos digitales revolucionó la forma en que se relacionan la enseñanza y el aprendizaje, pues ofrece una amplia serie de beneficios significativos, entre los cuales se encuentran:

1. **Mayor accesibilidad y flexibilidad:** Los medios digitales permiten a los estudiantes el acceso a material educativo en cualquier momento y lugar, lo que permite que estos obedezcan su propio ritmo de aprendizaje.

2. **Aprendizaje interactivo:** Gracias a materiales gráficos de apoyo como videos, animaciones, simuladores, infografías, presentaciones electrónicas y demás, el aprendizaje ha tomado un giro más dinámico que lo vuelve visualmente atractivo, atrayendo más la atención y compromiso de los estudiantes.

3. **Trabajo colaborativo:** Los medios digitales facilitan la realización de contenidos digitales de manera conjunta, lo que conlleva a una mejora en la comunicación y las habilidades de trabajo colaborativo de los estudiantes.

### 1.1.2. Herramientas digitales: ventajas y limitaciones

Hoy en día existe una amplia gama de herramientas para la creación de medios digitales, cada una con sus propias ventajas y limitaciones. Algunas plataformas, como Canva o Adobe Express sobresalen por su facilidad de uso, siendo herramientas intuitivas en las que la comunidad juega un papel importante aportando herramientas de apoyo como plugins, plantillas prediseñadas para diferentes medios, etc.

Por otro lado, tenemos el software especializado como Adobe Photoshop, Adobe Illustrator o Figma que son herramientas dirigidas a la producción de medios de manera profesional, siendo una de sus características principales la curva de aprendizaje pronunciada y costos fijos de suscripción mensual, lo que puede resultar en una barrera que limite a usuarios que comienzan a adentrarse en la creación de contenido digital.

## 1.2. Desarrollo web de la plataforma

El desarrollo web hace referencia al conjunto de procesos involucrados en la formación, construcción y mantenimiento de sitios web. Requiere esencialmente de los siguientes procesos:

1. **Planificación:** Consiste en la definición de los objetivos del sitio web, identificar la audiencia a la que se dirige el mismo y estructurar los contenidos e información que este contendrá.

2. **Diseño:** Durante este proceso se definirá la apariencia visual del sitio, incluyendo la disposición de elementos, la paleta de colores, tipografías y experiencia del usuario.

3. **Implementación:** Consiste en la ejecución mediante codificación para generar un sitio web funcional. Este proceso se divide en dos partes:

    a. **Front-end (lado del cliente):** Conforma la parte interactiva del sitio web. Se enfoca en la interfaz de usuario (UI) y la experiencia del usuario (UX), se utilizan tres lenguajes esencialmente para su desarrollo:

i. **HTML (HyperText Markup Language):** HyperText Markup Language (HTML) es un lenguaje muy sencillo que permite describir hipertexto, es decir, texto presentado de forma estructurada y agradable, con vínculos o enlaces (hyperlinks) que conducen a otros documentos o fuentes de información relacionadas y con inserciones multimedia (gráficos, sonido, etc.).

ii. **CSS (Cascading Style Sheets):** MDN Web Docs (2024) define CSS como el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML, este describe cómo debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios.

iii. **JavaScript:** Es el lenguaje de programación de secuencia de comandos de tipo interpretado que permite dar dinamismo a las páginas web [...]. Es un lenguaje orientado a documento, esto último indica que todos los componentes del DOM (Document Object Model) que constituyen a las ventanas de un navegador son accesibles por el lenguaje, el cual mediante fragmentos de código (funciones) permite aplicar diversas actividades sobre ellos.

b. **Back-end (lado del servidor):** Se encarga de la lógica que ocurre de manera "oculta", entre sus funciones principales se encuentran la gestión de bases de datos, la autenticación de usuarios y la comunicación entre el front-end y el servidor. Utiliza diversos lenguajes y frameworks como Python, Java, PHP, Node.js, Ruby, etc.

4. **Pruebas:** Durante esta etapa se verifica que el sitio funcione correctamente en distintos navegadores, dispositivos y condiciones, con el fin de identificar y corregir cualquier error (bug) que pueda estar presente.

5. **Despliegue:** Es la etapa final del proceso, en la cual se publica el sitio web a través de un servidor para que sea accesible a través de internet.

### 1.2.1. Herramientas para el desarrollo web de la plataforma

El desarrollo de un sitio web robusto, escalable, eficiente e interactivo como lo es la propuesta de este proyecto requiere de la correcta y adecuada combinación de tecnologías modernas. Para mayor eficiencia, hemos optado por el uso de frameworks que nos permitirá disminuir el uso de recursos y dar mayor dinamismo a la herramienta que produciremos.

Un framework (en el contexto del desarrollo web) es una estructura predefinida que funge como cimiento para la construcción de sitios web de manera rápida, eficiente y organizada.

**React**

Para el lado del cliente, optamos por el uso de una herramienta muy popular: React. React es una biblioteca de JavaScript free-code creada por Meta en 2013 que se utiliza para estructurar interfaces de usuario dinámicas e interactivas. Se caracteriza por su enfoque en optimizar el rendimiento de cualquier aplicación web mediante el uso de componentes, que son fragmentos de código reutilizables los cuales en conjunto construyen la interfaz de usuario, además, para mejorar el React utiliza un DOM virtual que agiliza los cambios de estado de la aplicación.

**Node.js**

Por su parte, para el lado del servidor haremos uso de Node.js el cual es un entorno de ejecución de JavaScript multiplataforma, free-code y gratuito que permite la creación de servidores, aplicaciones web, herramientas de línea de comando y scripts. Se caracteriza principalmente por su diseño asíncrono y no bloqueante, aunado a su ecosistema robusto: npm (Node Package Manager) que ofrece un extenso catálogo de módulos y bibliotecas que facilitan el desarrollo y la reutilización de código.

### 1.2.2. Bases de datos para el back-end

Una base de datos es una recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático. Normalmente, una base de datos está controlada por un sistema de gestión de bases de datos (DBMS). En conjunto, los datos y el DBMS, junto con las aplicaciones asociadas a ellos, reciben el nombre de sistema de bases de datos, abreviado normalmente a simplemente base de datos.

La gran parte de bases de datos utilizan un lenguaje de consulta estructurada (SQL) para leer y escribir datos. El SQL es un lenguaje de programación usado en la mayoría de bases de datos relacionales principalmente para consultar, editar y estructurar los datos.

Existe una variedad de tipos diferentes de bases de datos, entre los cuales destacan las siguientes:

1. **Bases de datos relacionales:** Organiza la información en tablas interconectadas, las cuales se asemejan a una hoja de cálculo, con filas que fungen como registros individuales y columnas que representan los atributos de cada entidad.

2. **Bases de datos NoSQL:** Representan una alternativa a las bases de datos relacionales con el fin de adaptarse a las necesidades de las aplicaciones modernas cuyo volumen de datos es superior a una convencional. Este tipo de bases de datos no utilizan un modelo relacional de tablas con relaciones, en cambio, utilizan un

esquema flexible que facilita el manejo de datos variables. Además, están orientadas a operaciones rápidas de consulta y manipulación.

Para nuestro proyecto, hemos optado por el uso de una de las bases de datos NoSQL más populares a nivel global: Cloud Firestore. Cloud Firestore es una base de datos remota NoSQL alojada en la nube que puede ser conectada a distintas aplicaciones a la vez (Apple, Android y sitios web). Permite almacenar, sincronizar y consultar datos de forma fácil y eficaz. Se caracteriza por su flexibilidad y escalabilidad, además de la sincronización de los datos en tiempo real (incluso sin contar con una conexión a internet) lo que facilita el desarrollo de aplicaciones multiusuario y colaborativas.

## 1.3. Inteligencia Artificial en la generación de contenido

La Inteligencia Artificial (IA) está emergiendo como una tecnología transformadora en el campo de la creación de contenido. Sus aplicaciones van desde la automatización de tareas repetitivas hasta la generación asistida o autónoma de elementos gráficos, textuales y audiovisuales. La IA Generativa, en particular, se refiere a modelos capaces de crear nuevo contenido original, como texto, imágenes, audio o código, a partir de los datos con los que fueron entrenados.

### 1.3.1. Panorama Actual de Modelos de IA Generativa para la Creación de Contenido

En los últimos años, ha habido una proliferación de modelos de IA generativa, cada uno con fortalezas y especializaciones particulares. Entre los más destacados se encuentran:

**Modelos de Lenguaje Grandes (LLMs) para Texto:**

- **Serie GPT de OpenAI (tales como GPT-3.5, GPT-4):** Conocidos por su capacidad avanzada para comprender y generar texto coherente y contextualmente relevante, responder preguntas, resumir información y realizar tareas de escritura creativa.
- **Claude de Anthropic:** Diseñado con un enfoque en ser útil, honesto e inofensivo, Claude destaca en tareas de conversación, resumen y análisis de texto, con un énfasis en la seguridad y la ética.
- **Gemini de Google:** Un modelo multimodal que puede procesar y generar información a través de diferentes tipos de datos como texto, código, imágenes y video, ofreciendo una comprensión y razonamiento más holísticos.

- **DeepSeek (especialmente DeepSeek V2):** Desarrollado por DeepSeek AI, este conjunto de modelos incluye modelos de lenguaje generales que han ganado atención por su rendimiento competitivo y por ser de código abierto.
- **Grok de xAI:** Presentado como un LLM con la capacidad de acceder a información en tiempo real a través de la plataforma X (anteriormente Twitter) y diseñado para responder preguntas con un toque de ingenio y una perspectiva menos restrictiva.

**Modelos de Generación de Imágenes:**

- **DALL-E de OpenAI:** Capaz de crear imágenes y arte a partir de descripciones textuales (prompts), permitiendo la generación de visuales únicos.
- **Midjourney:** Un laboratorio de investigación independiente que produce un programa de IA que crea imágenes a partir de descripciones textuales, conocido por su estilo artístico distintivo.
- **Stable Diffusion de Stability AI:** Un modelo de texto a imagen de código abierto que permite una mayor personalización y la ejecución en hardware local, democratizando el acceso a la generación de imágenes.
- **Gemini de Google (Capacidades de Imagen):** Como modelo multimodal, Gemini también incluye la capacidad de generar imágenes a partir de texto, integrando esta funcionalidad dentro de su arquitectura más amplia.

Estos modelos varían en sus arquitecturas subyacentes (e.g., Transformers), los datos con los que fueron entrenados, sus capacidades específicas (texto, imagen, multimodalidad), la calidad de sus resultados, la disponibilidad de APIs para desarrolladores y sus modelos de costos.

### 1.3.2. Análisis Comparativo y Justificación de la Selección de Gemini

Para la selección del modelo de IA a integrar en la presente plataforma, se realizó un análisis comparativo de las principales opciones disponibles en el mercado, considerando criterios clave para los objetivos del proyecto. Estos criterios incluyeron:

- Calidad y versatilidad en la generación de contenido textual.
- Capacidad y calidad en la generación de elementos visuales.
- Flexibilidad y potencia de la API, especialmente la disponibilidad de funcionalidades como “function calling” para la interacción con otras herramientas de la plataforma.

- Capacidades multimodales que permitan una interacción más rica y natural.
- Soporte y documentación para desarrolladores.
- Consideraciones de escalabilidad y costos.

Tras este análisis, se ha seleccionado el modelo Gemini de Google para su integración en la plataforma. Esta elección se fundamenta en su gran y diversa capacidad multimodal, que permite no solo la generación de texto e imágenes de alta calidad, sino también una comprensión y razonamiento más profundos al poder procesar diferentes tipos de información de manera integrada. Un factor decisivo fue la robustez de su API y la disponibilidad de la funcionalidad "function calling", la cual es crucial para permitir que la IA interactúe de forma programática con las herramientas del editor de la plataforma, facilitando una automatización más profunda y contextualizada de las tareas. Además, el respaldo de Google y su continuo desarrollo en el campo de la IA ofrecen perspectivas prometedoras para futuras mejoras y expansiones de la plataforma.

### 1.3.3. Implementación Específica de Gemini en la Plataforma

En el contexto de esta plataforma, la implementación de Gemini se centrará en potenciar la experiencia del usuario, especialmente en el "modo básico", mediante las siguientes funcionalidades clave:

- **Generación Asistida de Contenido Textual:** Utilizando las capacidades de procesamiento de lenguaje natural de Gemini, la plataforma podrá generar borradores de texto, sugerir títulos, crear descripciones o incluso desarrollar secciones de contenido a partir de indicaciones (prompts) del usuario.
- **Creación de Elementos Visuales mediante IA:** La plataforma integrará la capacidad de Gemini para generar imágenes originales basadas en descripciones textuales proporcionadas por el usuario.
- **Automatización de Acciones mediante Interacción Inteligente (Function Calling):** A través de la funcionalidad de "function calling" de Gemini, la IA no solo generará contenido, sino que también podrá interactuar de forma programática con las herramientas internas del editor de la plataforma.

### 1.3.4. Contextualización: Aplicaciones de IA en Plataformas de Diseño Existentes

La integración de IA en herramientas de diseño no es un concepto nuevo, y diversas plataformas ya han incorporado capacidades inteligentes para mejorar la experiencia del

usuario y la eficiencia del flujo de trabajo. Ejemplos de estas aplicaciones incluyen la eliminación inteligente de fondos en imágenes, la mejora automática de la calidad visual, la sugerencia de plantillas o diseños basada en el contenido del usuario, y, de manera creciente, la generación de texto e imágenes a partir de prompts.

## 1.4. Tecnologías para el Desarrollo Web de la Plataforma

El desarrollo de una plataforma web robusta, escalable, eficiente e interactiva, como la propuesta en este proyecto, requiere una cuidadosa selección y combinación de tecnologías modernas. Se ha optado por un stack tecnológico que no solo facilita el desarrollo ágil sino que también habilita funcionalidades avanzadas como la integración de Inteligencia Artificial y la colaboración en tiempo real.

### 1.4.1. Desarrollo del Frontend: Interfaz de Usuario Dinámica con React

La interfaz de usuario (frontend) es el principal punto de interacción para los usuarios de la plataforma, por lo que su diseño y funcionalidad son críticos para una experiencia de usuario óptima. Para su desarrollo, se ha seleccionado React, una biblioteca de JavaScript de código abierto ampliamente utilizada para construir interfaces de usuario interactivas y reutilizables.

- **Componentización y Reusabilidad:** React permite descomponer la interfaz de usuario en componentes independientes y reutilizables.
- **Virtual DOM para Rendimiento Eficiente:** React utiliza un DOM (Document Object Model) virtual para optimizar las actualizaciones de la interfaz.
- **Ecosistema y Comunidad:** React cuenta con un vasto ecosistema de bibliotecas y herramientas complementarias.
- **Gestión del Estado:** Para manejar el estado de la aplicación se pueden utilizar soluciones como el propio gestor de estado de React o bibliotecas más avanzadas como Redux o Zustand.

### 1.4.2. Backend y Base de Datos: Potenciando la Plataforma con Firebase

Para el backend y la gestión de datos, se ha elegido Firebase, una plataforma de desarrollo de aplicaciones móviles y web propiedad de Google. Firebase ofrece un conjunto

de herramientas y servicios que simplifican el desarrollo de backend, la gestión de bases de datos y la implementación de funcionalidades complejas.

- **Firebase Realtime Database y/o Firestore:** Una de las características más destacadas de Firebase es su capacidad para sincronizar datos en tiempo real.
- **Firebase Authentication:** Simplifica la implementación de sistemas de autenticación seguros.
- **Firebase Hosting:** Proporciona un servicio de alojamiento rápido y seguro para aplicaciones web.
- **Firebase Functions:** Permite ejecutar código de backend sin necesidad de gestionar servidores (serverless).

### 1.4.3. Integración de Inteligencia Artificial con la API de Gemini

La funcionalidad de Inteligencia Artificial se integrará utilizando la API de Gemini proporcionada por Google.

1. **Comunicación API:** La aplicación frontend (React) realizará solicitudes a la API de Gemini para las tareas de generación de texto, generación de imágenes y ejecución de "function calling".
2. **Flujo de Interacción:**
   a. El usuario interactúa con la interfaz de la plataforma.
   b. La aplicación React formula una solicitud a la API de Gemini.
   c. Gemini procesa la solicitud y devuelve el contenido generado.
   d. La aplicación React recibe la respuesta y actualiza la interfaz de usuario.

# Capítulo 2

# ANÁLISIS DE REQUERIMIENTOS Y ARQUITECTURA DEL SISTEMA

En este capítulo se presenta un análisis detallado de los requerimientos necesarios para el desarrollo de la plataforma web, y también se incluye un desglose de la estructura y arquitectura técnica que lo conforma. Esto con el objetivo de entender la funcionalidad de la aplicación, así como lograr comprender el diseño del sistema.

## 2.1. Levantamiento de requerimientos

Para garantizar que la aplicación cumpla con su propósito, debemos asegurar que esta satisfaga alguna necesidad que tengan los usuarios objetivos. De esta manera se plantea el levantamiento de requerimientos, que consistió en la observación de la utilización de otras herramientas con propósitos similares y de materiales creados por educadores.

Durante esta observación se destacó la dificultad de los docentes para crear materiales educativos con fluidez y que tuvieran un diseño atractivo. Para el caso de algunas herramientas, su utilización resulta poco intuitiva o requiere de una inversión adicional de tiempo para aprender a utilizarlas, por lo que representa un reto para todos aquellos usuarios no especializados en el diseño gráfico y provoca que crear materiales visuales adecuados es ineficiente.

## 2.2. Requerimientos del sistema

El desarrollo de la plataforma se basó en cubrir con los requerimientos funcionales y no funcionales que satisfacen las necesidades observadas.

### 2.2.1. Requerimientos funcionales

Los requerimientos funcionales describen las acciones concretas que el sistema debe ser capaz de realizar:

- **Registro de usuarios:** El sistema debe permitir que nuevos usuarios se registren ingresando su nombre, correo electrónico y contraseña.
- **Autenticación segura:** El inicio de sesión de todos los usuarios debe estar autenticado para que se pueda acceder a los contenidos de la plataforma de forma segura.
- **Creación de contenido educativo y creativo:** El sistema debe permitir a los usuarios generar materiales en distintos formatos (texto, imágenes, PDF).
- **Asistencia mediante inteligencia artificial:** El sistema debe ofrecer sugerencias y plantillas automáticas para facilitar la creación de materiales atractivos cuando el usuario solicite.
- **Programación de publicación temporal:** Los usuarios deben poder definir fechas de inicio y expiración para cada contenido publicado.
- **Visualización del estado del contenido:** Cada entrada debe mostrar claramente si está activa, pendiente o expirada.
- **Generación de enlaces temporales:** El sistema debe generar enlaces únicos para cada material y que funcionen dentro del periodo definido.
- **Edición y eliminación de contenido:** El usuario debe poder modificar o eliminar sus contenidos fácilmente desde su panel.
- **Notificaciones automáticas:** El sistema debe enviar recordatorios cuando un contenido esté próximo a expirar o haya expirado.

### 2.2.2. Requerimientos no funcionales

Los requerimientos no funcionales establecen las condiciones de calidad, rendimiento, accesibilidad y seguridad del sistema:

- **Compatibilidad:** La plataforma debe ser accesible desde cualquier navegador moderno como Google Chrome, Firefox, Edge y Safari.
- **Interfaz responsiva:** El diseño debe adaptarse correctamente a distintos tamaños de pantalla (PC, tablet, smartphone).

- **Rendimiento:** El sistema debe ser capaz de atender al menos 50 usuarios concurrentes sin degradar la experiencia.
- **Seguridad:** Los datos de usuarios y contenidos deben almacenarse en servidores protegidos mediante cifrado y prácticas de seguridad.
- **Disponibilidad:** La plataforma debe estar disponible y en línea durante la mayoría del tiempo a lo largo de un mes.
- **Escalabilidad:** La arquitectura del sistema debe permitir su ampliación futura.
- **Tiempo de respuesta:** Las acciones básicas como iniciar sesión, guardar contenido o compartir enlaces no deben superar los 2 segundos de tiempo de espera.

## 2.3. Modelo de interacción y experiencia de usuario

La experiencia de usuario y el diseño de la interfaz son elementos fundamentales para que el uso de la plataforma sea fluido y se puedan crear contenidos creativos y educativos fácilmente.

### 2.3.1. Principios de diseño aplicados

Los principios de diseño son las directrices que permiten que una composición visual sea funcional y se mantenga estética:

- **Simplicidad:** La plataforma evita mostrar al usuario una gran cantidad de opciones innecesarias.
- **Consistencia:** Se emplean patrones visuales y de navegación coherentes en todas las vistas.
- **Accesibilidad:** La aplicación debe estar adaptada para todo tipo de dispositivos.
- **Retroalimentación inmediata:** Las acciones del usuario generan respuestas visuales claras y notificaciones de estado.

### 2.3.2. Flujo general y mejora de la experiencia

Se diseñó el flujo de interacción de tal forma que el usuario puede llevar a cabo sus objetivos en la menor cantidad de pasos. El flujo básico contempla:

- Inicio de sesión o registro, con opciones de inicio rápido y recuperación.

- Acceso al panel principal, donde se muestran todos los contenidos que ha creado el usuario.
- Creación de nuevos contenidos, con las sugerencias asistidas por IA.
- Definición de los atributos del contenido, como su fecha de inicio/expiración o su visibilidad.
- Generación del enlace temporal para compartir.
- Seguimiento de los contenidos creados, y acceso a otros contenidos.

## 2.4. Arquitectura del sistema

La arquitectura de la plataforma fue diseñada bajo un enfoque modular, escalable y orientado a servicios, integrando tecnologías tanto en el cliente como en el servidor.

### 2.4.1. Componentes Principales

a. **Frontend (Cliente):** Desarrollado con React. Será una Single Page Application (SPA) responsable de toda la interfaz de usuario.

b. **Backend (Servicios en la Nube):** Se utilizará Firebase como Backend as a Service (BaaS). Esto incluye:

   - Firebase Authentication
   - Firestore (o Realtime Database)
   - Firebase Storage
   - Firebase Functions (Cloud Functions)

c. **API de Inteligencia Artificial Externa:** Se integrará la API de Gemini de Google para las funcionalidades de generación de texto, generación de imágenes y "function calling".

### 2.4.2. Flujo de Datos e Interacciones

- El usuario interactúa con la interfaz React en su navegador.
- Las acciones del usuario desencadenan eventos en el frontend.
- Para operaciones de datos, React se comunicará directamente con Firestore/Realtime Database y Firebase Storage.
- Para la autenticación, React utilizará los servicios de Firebase Authentication.

- Para las funcionalidades de IA, React enviará solicitudes a la API de Gemini.
- Firebase Functions manejará la lógica para la validación y caducidad de los enlaces de publicación temporal.

## 2.5. Distribución y gestión recursos del sistema

### 2.5.1. Recursos del lado del cliente

La plataforma, diseñada como una Single Page Application (SPA), descarga e inicializa la mayor parte del código de la interfaz en el navegador del usuario:

- **Procesamiento local:** La renderización del contenido, navegación entre pantallas, edición visual y la gestión básica del estado de la aplicación se llevan a cabo directamente en el navegador.
- **Consumo de memoria:** La aplicación está optimizada para consumir recursos mínimos.
- **Requerimientos mínimos:** Se requiere únicamente de un navegador moderno y una conexión a internet estable.

### 2.5.2. Recursos del lado del servidor

La parte del servidor se basa en servicios proporcionados por Firebase y APIs externas:

- **Escalabilidad automática:** Firebase Functions y Firestore escalan de forma automática según el número de usuarios activos.
- **Procesamiento intensivo:** Las tareas que requieren recursos significativos se procesan en el backend.
- **Gestión de almacenamiento:** Los archivos generados o cargados por los usuarios se almacenan de forma eficiente en Firebase Storage.

### 2.5.3. Balance de carga y eficiencia

La plataforma sigue un modelo donde las tareas de interacción, edición visual y navegación permanecen en el cliente, aprovechando los recursos locales del dispositivo. Las operaciones críticas, costosas o que implican seguridad se ejecutan en la nube.

# Capítulo 3

# DISEÑO E IMPLEMENTACIÓN

Este capítulo se adentra en el proceso práctico de construcción de la plataforma web para la creación, edición y publicación de contenido digital.

## 3.1. Desarrollo del editor de contenido

El editor de contenido es el núcleo de la plataforma. Se diseñará para ser intuitivo y potente, ofreciendo dos modos de operación para adaptarse a diferentes niveles de habilidad.

### 3.1.1. Modo básico vs. modo avanzado

**Modo Básico:**

- **Interfaz:** Simplificada, con herramientas esenciales y opciones preconfiguradas. Se priorizará la facilidad de uso y un flujo de trabajo guiado mediante una interfaz visual (WYSIWYG - What You See Is What You Get).
- **Funcionalidades:** Énfasis en plantillas, elementos de arrastrar y soltar, y fuerte asistencia de la IA. Las opciones de personalización serán limitadas a las más comunes accesibles a través de controles visuales.
- **Objetivo:** Permitir a usuarios novatos crear contenido atractivo rápidamente y sin una curva de aprendizaje pronunciada.

**Modo Avanzado:**

- **Interfaz:** En lugar de una sobrecarga de botones y paneles visuales complejos, el modo avanzado se centrará en proveer áreas de entrada de código dedicadas.
- **Funcionalidades:**

- **Control Directo mediante Código:** Los usuarios avanzados podrán definir la estructura del contenido utilizando HTML, aplicar estilos precisos con CSS, y potencialmente añadir interactividad básica con fragmentos de JavaScript.
- **Personalización sin Límites Visuales:** Este enfoque elimina las restricciones impuestas por una interfaz gráfica.
- **Integración con IA para Código:** La Inteligencia Artificial seguiría estando disponible, orientada a generar fragmentos de código HTML/CSS/JS basados en prompts del usuario.
- **Reutilización de Snippets:** Los usuarios podrían guardar y reutilizar sus propios fragmentos de código.

- **Objetivo:** Ofrecer a usuarios con conocimientos de desarrollo web la máxima flexibilidad y control granular sobre sus diseños.

### 3.1.2. Implementación de plantillas y personalización

**Sistema de Plantillas:**

- Se creará una biblioteca de plantillas prediseñadas para diversos propósitos (infografías, presentaciones, posts para redes sociales, etc.).
- Las plantillas se almacenarán en Firestore, incluyendo su estructura (JSON o similar) y los assets asociados.
- Los usuarios podrán seleccionar una plantilla para iniciar un nuevo diseño.

**Herramientas de Personalización:**

- **Manipulación de Objetos:** Selección, movimiento, redimensionamiento, rotación, alineación, distribución y agrupación de elementos.
- **Texto:** Edición de contenido, selección de fuentes, tamaño, color, estilo, alineación, espaciado.
- **Imágenes:** Carga de imágenes propias, búsqueda e inserción de imágenes de stock o generación mediante IA.
- **Formas y Gráficos:** Herramientas para añadir formas básicas, líneas y gráficos simples.
- **Colores y Fondos:** Paletas de colores, selectores de color, capacidad de aplicar colores sólidos, gradientes o imágenes de fondo.

## 3.2. Integración de inteligencia artificial

La IA, a través de la API de Gemini, se integrará para asistir al usuario en diversas etapas del proceso de creación.

- **Acceso a Funcionalidades IA:** Se dispondrán puntos de entrada intuitivos en la interfaz del editor para las funciones de IA.
- **Generación de Texto:** El usuario podrá ingresar prompts para generar títulos, párrafos, descripciones, o ideas de contenido.
- **Generación de Imágenes:** El usuario podrá describir la imagen deseada mediante un prompt.
- **Function Calling para Automatización:** Se explorará el uso de "function calling" de Gemini para tareas más complejas.

### 3.2.1. Implementación Técnica de la Integración de IA con Gemini API

La integración de la Inteligencia Artificial con la API de Gemini se centrará en una robusta implementación del lado del servidor utilizando Firebase Functions.

#### Implementación del Backend (Firebase Functions con API de Gemini)

Se crearán Firebase Functions (Node.js) dedicadas para cada tipo de interacción con la API de Gemini:

- **Recepción y Validación de Solicitudes del Frontend:** Cada función escuchará en un endpoint HTTPS específico y validará los datos de entrada.
- **Gestión Segura de Claves API de Gemini:** La clave API se almacenará como una variable de entorno en la configuración de Firebase Functions.
- **Interacción con la API de Gemini:** Se utilizará el SDK de Google AI para Node.js dentro de las Firebase Functions.

```
const { GoogleGenerativeAI } = require("@google/generative-ai");
const genAI = new GoogleGenerativeAI(process.env.GEMINI_API_KEY);

async function handleTextGeneration(prompt) {
    const model = genAI.getGenerativeModel({ model: "gemini-pro" });
    const result = await model.generateContent(prompt);
    const response = await result.response;
```

```
    return response.text();
}
```

Listing 3.1: Ejemplo conceptual en una Firebase Function

**Comunicación Frontend (React) con el Backend (Firebase Functions)**

La comunicación entre la aplicación React y las Firebase Functions se establecerá mediante solicitudes HTTPS:

- **Definición de Endpoints API:** Se establecerá una convención clara para los endpoints del backend.
- **Realización de Solicitudes desde React:** Se utilizará la API fetch nativa del navegador para realizar solicitudes POST a los endpoints.

```
async function callGenerateTextAPI(prompt) {
    const idToken = await firebase.auth().currentUser.getIdToken();

    const response = await fetch('/api/generateText', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json',
            'Authorization': `Bearer ${idToken}`
        },
        body: JSON.stringify({ prompt: prompt })
    });

    if (!response.ok) {
        throw new Error(`API call failed: ${response.status}`);
    }

    return response.json();
}
```

Listing 3.2: Ejemplo conceptual en React

## 3.3. Implementación del sistema de publicación temporal

Esta funcionalidad permitirá a los usuarios compartir sus diseños con una validez limitada:

- **Generación de Enlaces Únicos:** Al publicar, el sistema generará un enlace único y difícil de adivinar para cada diseño compartido temporalmente.
- **Opciones de Temporalidad:** El usuario podrá configurar fecha y hora de expiración y número máximo de visualizaciones.
- **Almacenamiento de Configuración:** Los metadatos de la publicación temporal se almacenarán en Firestore.
- **Validación de Enlaces:** Una Firebase Function interceptará las solicitudes y verificará la validez del enlace.

## 3.4. Gestión de base de datos y almacenamiento de proyectos

Firebase será la piedra angular para la persistencia y gestión de todos los datos de la aplicación.

### 3.4.1. Modelo de Datos en Firestore/Realtime Database

- **users:** Información de perfil de usuario (ID de Firebase Auth, nombre, email, preferencias).
- **projects:** Documentos que representan los diseños de los usuarios. Cada proyecto contendrá:
  - ownerId: ID del usuario propietario
  - title: Título del proyecto
  - description: Descripción
  - canvasData: Objeto JSON que describe todos los elementos en el lienzo
  - templateId (opcional): Si se basó en una plantilla
  - createdAt, updatedAt: Marcas de tiempo
  - collaborators (opcional): Lista de IDs de usuarios colaboradores
  - temporalPublishSettings (opcional): Configuración de publicación temporal
- **templates:** Colección de plantillas prediseñadas, con su canvasData y metadatos.

### 3.4.2. Almacenamiento de Archivos (Firebase Storage)

- Se crearán carpetas estructuradas para almacenar imágenes y otros assets.
- Los proyectos en Firestore contendrán referencias (URLs) a estos archivos almacenados.

### 3.4.3. Operaciones CRUD y Sincronización

- El frontend (React) interactuará con Firebase para crear, leer, actualizar y eliminar proyectos.
- Se aprovecharán las capacidades de escucha en tiempo real de Firebase para la colaboración.

## 3.5. Seguridad y protección de datos

La seguridad será una consideración primordial en todas las etapas del diseño y la implementación:

- **Autenticación:** Se utilizará Firebase Authentication para manejar de forma segura los procesos de registro e inicio de sesión.
- **Autorización:** Se definirán reglas de seguridad en Firestore y Firebase Storage para asegurar que los usuarios solo puedan acceder a sus propios datos.
- **Protección de Datos en Tránsito:** Toda la comunicación se realizará a través de HTTPS.
- **Protección de Datos en Reposo:** Firebase encripta los datos en reposo por defecto.
- **Seguridad de APIs Externas:** Las claves de API para Gemini se gestionarán de forma segura.
- **Validación de Entradas:** Se implementará validación tanto en el frontend como en el backend.
- **Gestión de Enlaces Temporales Seguros:** Los enlaces temporales serán generados con suficiente aleatoriedad.
- **Privacidad de Datos:** Se desarrollará una política de privacidad clara y se obtendrá el consentimiento del usuario.

# Bibliografía


[1] Anthropic. (2023). Introducing Claude. https://www.anthropic.com/news/introducing-claude

[2] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., & Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.

[3] Cooper, A., Reimann, R., Cronin, D., & Noessel, C. (2014). *About face: The essentials of interaction design* (4th ed.). Wiley.

[4] Duckett, J. (2011). *HTML and CSS: Design and build websites*. Wiley.

[5] Elgammal, A., Liu, B., Elhoseiny, M., & Mazzone, M. (2017). CAN: Creative adversarial networks, generating "art" by learning about styles and deviating from style norms. *arXiv*. https://arxiv.org/abs/1706.07068

[6] Flanagan, D. (2020). *JavaScript: The definitive guide* (7th ed.). O'Reilly Media.

[7] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

[8] Google. (2023). Introducing Gemini: Our largest and most capable AI model. https://blog.google/technology/ai/google-gemini-ai/

[9] HolonIQ. (2023). Artificial intelligence in education: 2023 survey insights. https://www.holoniq.com/notes/artificial-intelligence-in-education-2023-survey-insights

[10] Jenkins, H. (2006). *Convergence culture: Where old and new media collide*. New York University Press.

[11] Kaplan, A., & Haenlein, M. (2019). Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence. *Business Horizons*, 62(1), 15–25.

[12] Krug, S. (2014). *Don't make me think, revisited: A common sense approach to web usability* (3rd ed.). New Riders.

[13] Lessig, L. (2008). *Remix: Making art and commerce thrive in the hybrid economy*. Penguin Press.

[14] Luckin, R., Holmes, W., Griffiths, M., & Forcier, L. B. (2016). *Intelligence unleashed: An argument for AI in education*. Pearson Education.

[15] Martin, R. C. (2008). *Clean code: A handbook of agile software craftsmanship*. Prentice Hall.

[16] Mertala, P. (2020). Low-code development platforms: A systematic literature review. *Journal of Systems and Software*, 166, 110607.

[17] Nielsen, J. (1994). *Usability engineering*. Morgan Kaufmann.

[18] Norman, D. (2013). *The design of everyday things: Revised and expanded edition*. Basic Books.

[19] Paar, C., & Pelzl, J. (2010). *Understanding cryptography: A textbook for students and practitioners*. Springer.

[20] Prensky, M. (2010). *Teaching digital natives: Partnering for real learning*. Corwin Press.

[21] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., & Sutskever, I. (2021). Zero-shot text-to-image generation. *Proceedings of the 38th International Conference on Machine Learning*, 139, 8821–8831.

[22] Silberschatz, A., Korth, H. F., & Sudarshan, S. (2010). *Database system concepts* (6th ed.). McGraw-Hill.

[23] Stallings, W. (2017). *Cryptography and network security: Principles and practice* (7th ed.). Pearson.

[24] UNESCO. (2018). Digital services for education in Africa. `https://www.unesco.org/en/articles/workshop-launch-digital-services-education-africa-publication`

[25] Voigt, P., & von dem Bussche, A. (2017). *The EU General Data Protection Regulation (GDPR)*. Springer.