Información

https://v0-website-clone-omega-liart.vercel.app/
https://github.com/Eduardo3382/v0-blog-personal/tree/main
https://v0.app/chat/website-clone-hyttsY3rXlP

OBJETIVO (confirmado)

Poder hacer un repositorio de notas a escribir en MD para que lo pueda mostrar la pagina

1. Escribir artículos **solo en Markdown**
2. Subirlos desde GitHub (web o local)
3. Cada artículo:
    - tiene **una imagen**
    - tiene **resumen**
4. En la home:
    - 🔹 **último artículo destacado** (foto a la izquierda, como en tu imagen)
    - 🔹 **3 o 4 artículos anteriores** debajo
    - 🔹 link “Ver todos los artículos”
5. Página `/blog` con el **listado completo**
6. **Sin crear páginas manualmente por artículo**

0️⃣ PRECONDICIÓN (ya la cumplís)

✔ Proyecto en GitHub
✔ Deploy en Vercel
✔ App Router
✔ Ya corriste (una sola vez):

```
npm install gray-matter
```

1️⃣ ESTRUCTURA DEFINITIVA DE CARPETAS

En la **raíz del proyecto**:

```
content/
  blog/
    comenzando-tu-viaje.md
    otro-articulo.md
    tercer-articulo.md
```

Para imágenes:

```
public/
  blog/
    comenzando-tu-viaje.jpg
    otro-articulo.jpg
    tercer-articulo.jpg
```

Reglas claras:

- **un `.md` = un artículo**
- **una imagen principal por artículo**
- no subcarpetas
- nombres simples con guiones

2️⃣ PLANTILLA REAL DE ARTÍCULO `.md`

Ejemplo real basado en lo que ya mostrás:

`content/blog/comenzando-tu-viaje.md`

```
---
title: "Comenzando tu viaje en el mundo de la programación"
date: "2026-01-15"
description: "La programación no es solo escribir código, es aprender a pensar de manera
lógica y resolver problemas de formas creativas."
image: "/blog/comenzando-tu-viaje.jpg"
---

Acá empieza el artículo completo.

Podés escribir tranquilo, largo, corto,
con párrafos, listas, reflexiones, etc.
```

Esto reemplaza por completo escribir en `.tsx`.

3️⃣ LECTOR CENTRAL DE ARTÍCULOS (lib)

Creamos / usamos:

```
lib/posts.ts
```

```
import fs from "fs";
import path from "path";
import matter from "gray-matter";
```

```
const postsDirectory = path.join(process.cwd(), "content/blog");

export function getAllPosts() {
  const files = fs.readdirSync(postsDirectory);

  const posts = files.map((file) => {
    const slug = file.replace(/\.md$/, "");
    const fullPath = path.join(postsDirectory, file);
    const fileContents = fs.readFileSync(fullPath, "utf8");

    const { data, content } = matter(fileContents);

    return {
      slug,
      content,
      ...data,
    };
  });

  return posts.sort(
    (a, b) => new Date(b.date).getTime() - new Date(a.date).getTime()
  );
}
```

Este archivo es **el corazón del blog**.

4️⃣ HOME: último artículo destacado + lista

En tu **home** (`app/page.tsx` o similar):

```
import { getAllPosts } from "@/lib/posts";

const posts = getAllPosts();

const featuredPost = posts[0];
const olderPosts = posts.slice(1, 5); // 4 anteriores
```

⭐ Último artículo destacado (como en tu imagen)

Concepto visual:

```
[ FOTO ]   Fecha
           Título
           Resumen
           Leer artículo →
```

Ejemplo JSX (simplificado):

```
<Image
  src={featuredPost.image}
  alt={featuredPost.title}
  width={240}
  height={160}
/>

<p>{featuredPost.date}</p>
<h2>{featuredPost.title}</h2>
<p>{featuredPost.description}</p>

<Link href={`/blog/${featuredPost.slug}`}>
  Leer artículo →
</Link>
```

📌 El diseño exacto lo mantenés como hoy:
solo cambió el origen de los datos.

📄 Lista de artículos anteriores

```
{olderPosts.map((post) => (
  <article key={post.slug}>
    <h3>{post.title}</h3>
    <p>{post.description}</p>
    <Link href={`/blog/${post.slug}`}>Leer</Link>
  </article>
))}
```

🔗 Link al listado completo

```
<Link href="/blog">
  Ver todos los artículos →
</Link>
```

5️⃣ PÁGINA `/blog` (listado completo)

Ruta:

```
app/blog/page.tsx
```

```
import { getAllPosts } from "@/lib/posts";

const posts = getAllPosts();
```

```
export default function BlogPage() {
  return (
    <>
      <h1>Artículos</h1>

      {posts.map((post) => (
        <article key={post.slug}>
          <h2>{post.title}</h2>
          <p>{post.description}</p>
          <Link href={`/blog/${post.slug}`}>Leer artículo</Link>
        </article>
      ))}
    </>
  );
}
```

📌 Acá ves **todo**, ordenado por fecha.

6️⃣ PÁGINA DE ARTÍCULO INDIVIDUAL

Ruta dinámica:

```
app/blog/[slug]/page.tsx
```

Concepto mínimo:

```
import { getAllPosts } from "@/lib/posts";
import { notFound } from "next/navigation";
```

```
const post = posts.find(p => p.slug === params.slug);
if (!post) notFound();
```

Render:

```
<h1>{post.title}</h1>

<Image
  src={post.image}
  alt={post.title}
  width={800}
  height={400}
/>

<article>
  {post.content}
</article>
```

📌 Después, si querés, se puede mejorar el render de Markdown.

7️⃣ FLUJO REAL DE TRABAJO (el que querías)

A partir de ahora:

1. Creás un archivo `.md`
2. Subís una imagen a `public/blog`
3. Commit desde GitHub web
4. Vercel rebuild
5. El artículo:
   - aparece solo
   - si es el más nuevo → queda destacado
   - empuja los otros hacia abajo
     ❌ No tocás React
     ❌ No copiás páginas
     ❌ No repetís código

🧠 RESUMEN EN UNA FRASE

> **El código no cambia más; solo crecen tus artículos en Markdown.**