**1. Archivo PHP: update\_pages.php**

**Ubicación:**  
En tu servidor, dentro de la carpeta de tu blog. Por ejemplo, si tu ruta principal es public\_html/blog/, crea una carpeta llamada api y coloca aquí el archivo:

public\_html/blog/api/update\_pages.php

**Código:**

<?php

// Ruta de la carpeta donde se encuentran las entradas (archivos HTML de cada entrada)

$blogDir = \_\_DIR\_\_ . "/../"; // Asume que las entradas están en public\_html/blog/

// Ruta al archivo entries.json

$entriesJsonPath = \_\_DIR\_\_ . '/entries.json';

$entriesPerPage = 6;

// Buscar todos los archivos HTML en la carpeta que no sean los de paginación ni el blog principal

// (Se asume que las entradas tienen nombres distintos a "blog.html" y "page\*.html")

$files = glob($blogDir . "\*.html");

$entries = [];

foreach ($files as $file) {

$filename = basename($file);

// Ignorar archivos de paginación o el principal

if (stripos($filename, 'blog.html') !== false || stripos($filename, 'page') === 0) {

continue;

}

$url = "/blog/" . $filename; // URL relativa de la entrada

// Obtener la fecha de creación (más estable que modification)

$creationTime = filectime($file);

// Leer el contenido del archivo HTML de la entrada

$htmlContent = file\_get\_contents($file);

// Extraer el primer <h1> para el título

preg\_match('/<h1[^>]\*>(.\*?)<\/h1>/si', $htmlContent, $titleMatch);

$title = isset($titleMatch[1]) ? trim($titleMatch[1]) : "Sin título";

// Extraer la primera imagen <img>

preg\_match('/<img[^>]+src=["\'](.\*?)["\']/si', $htmlContent, $imageMatch);

$image = isset($imageMatch[1]) ? trim($imageMatch[1]) : "/images/default.jpg";

// Extraer el primer párrafo <p> para el extracto

preg\_match('/<p[^>]\*>(.\*?)<\/p>/si', $htmlContent, $excerptMatch);

$excerpt = isset($excerptMatch[1]) ? strip\_tags($excerptMatch[1]) : "Sin descripción";

// Agregar la entrada al array, incluyendo la fecha de creación para ordenar

$entries[] = [

"image" => $image,

"title" => $title,

"link" => $url,

"excerpt" => $excerpt,

"created\_at" => $creationTime

];

}

// Ordenar las entradas de más reciente a más antigua

usort($entries, function ($a, $b) {

return $b['created\_at'] - $a['created\_at'];

});

// Eliminar el campo created\_at antes de guardar en JSON

foreach ($entries as &$entry) {

unset($entry['created\_at']);

}

// Guardar la lista de entradas en entries.json

file\_put\_contents($entriesJsonPath, json\_encode($entries, JSON\_PRETTY\_PRINT | JSON\_UNESCAPED\_UNICODE));

// Calcular el total de páginas necesarias

$totalPages = ceil(count($entries) / $entriesPerPage);

// Generar o actualizar las páginas de archivo: page1.html, page2.html, etc.

for ($i = 1; $i <= $totalPages; $i++) {

$pageName = \_\_DIR\_\_ . "/../page{$i}.html"; // Se crean en la carpeta del blog

$pageEntries = array\_slice($entries, ($i - 1) \* $entriesPerPage, $entriesPerPage);

// Plantilla base para cada página generada automáticamente

$content = '<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Blog-de Ayurveda-Tradicional - Página ' . $i . '</title>

<link rel="stylesheet" href="/css/styles.css">

<link rel="stylesheet" href="/css/blog.css">

<link href="https://cdnjs.cloudflare.com/ajax/libs/tailwindcss/2.2.19/tailwind.min.css" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Libre+Baskerville:wght@300;400;700&display=swap" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Merriweather:wght@300;400;500;600;700&display=swap" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Cormorant+Garamond:wght@300;400;500;600;700&display=swap" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Lora:wght@300;400;500;600;700&display=swap" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&display=swap" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Inter:wght@300;400;500;600;700&display=swap" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Noto+Sans+SC:wght@100..900&family=Roboto:ital,wght@0,100..900;1,100..900&display=swap" rel="stylesheet">

</head>

<body>

<main>

<div class="cabecera-paginas">

<img id="cabecera-img" src="images/blog-de-ayurveda-tradicional.webp" alt="Cabecera">

<h1>Blog de Ayurveda Tradicional - Página ' . $i . '</h1>

</div>

<div class="content-container">';

foreach ($pageEntries as $entry) {

$content .= '<div class="content-box">

<img src="' . $entry['image'] . '" alt="' . $entry['title'] . '">

<h2>' . $entry['title'] . '</h2>

<p>' . $entry['excerpt'] . '</p>

<a href="' . $entry['link'] . '" class="entry-button">Leer más</a>

</div>';

}

$content .= '

</div>

<div id="pagination-container" class="pagination-container"></div>

<script src="/js/selector.js"></script>

</main>

<!-- Pie de página -->

<footer>

<p>© 2025 Mi Sitio Web. Todos los derechos reservados.</p>

</footer>

</body>

</html>';

file\_put\_contents($pageName, $content);

}

echo json\_encode(["status" => "success", "message" => "Entradas detectadas y páginas actualizadas."]);

?>

OSEA MIENTRAS TENGA ESTOS DOS DIV   
<div class="content-container">';   
<div id="pagination-container" class="pagination-container"> Puedo crear la estructura hmlt que deseo

## ****2. Archivo JSON: entries.json****

**Ubicación:**  
En la misma carpeta que update\_pages.php:  
public\_html/blog/api/entries.json

**Contenido inicial (ejemplo):**

[

{

"image": "/images/entrada1.jpg",

"title": "Mi Primer Post",

"link": "/blog/gengibre.html",

"excerpt": "Resumen de mi primer post..."

},

{

"image": "/images/entrada2.jpg",

"title": "Nueva Receta Ayurveda",

"link": "/blog/dolor-de-espalda.html",

"excerpt": "Descubre esta receta saludable..."

}

]

## ****3. Archivo JavaScript: selector.js****

**Ubicación:**  
En la carpeta js dentro de tu directorio de blog, por ejemplo:  
public\_html/blog/js/selector.js

**Código:**

const entriesPerPage = 6;

// Extrae el número de página de la URL usando el formato "pageX.html"

function getCurrentPageFromUrl() {

const match = window.location.href.match(/page(\d+)/);

return match ? parseInt(match[1], 10) : 1;

}

// Renderiza la paginación en el div con id "pagination-container"

function renderPagination(totalPages) {

const blockSize = 5; // Número de botones visibles por bloque

const currentPage = getCurrentPageFromUrl();

const blockIndex = Math.ceil(currentPage / blockSize);

const startPage = (blockIndex - 1) \* blockSize + 1;

const endPage = Math.min(totalPages, blockIndex \* blockSize);

const container = document.getElementById("pagination-container");

container.innerHTML = '';

const nav = document.createElement('nav');

nav.className = 'pagination';

// Botón "Anterior": enlaza a la última página del bloque anterior

if (blockIndex > 1) {

const prevLink = document.createElement('a');

prevLink.className = 'page-link';

prevLink.innerHTML = "&laquo;";

prevLink.href = `page${startPage - 1}.html`;

nav.appendChild(prevLink);

}

// Botones de página para el bloque actual

for (let i = startPage; i <= endPage; i++) {

const link = document.createElement('a');

link.className = 'page-link' + (i === currentPage ? ' active' : '');

link.textContent = i;

link.href = `page${i}.html`;

nav.appendChild(link);

}

// Botón "Siguiente": enlaza a la primera página del siguiente bloque

if (endPage < totalPages) {

const nextLink = document.createElement('a');

nextLink.className = 'page-link';

nextLink.innerHTML = "&raquo;";

nextLink.href = `page${endPage + 1}.html`;

nav.appendChild(nextLink);

}

container.appendChild(nav);

}

// Obtener el total de páginas consultando entries.json

fetch("/blog/api/entries.json")

.then(response => response.json())

.then(data => {

const totalPages = Math.ceil(data.length / entriesPerPage);

renderPagination(totalPages);

})

.catch(error => console.error("Error al cargar la paginación:", error));

## ****4. Archivo principal: blog.html****

**Ubicación:**  
En la raíz de tu carpeta del blog, por ejemplo:  
public\_html/blog/blog.html

**Código (ejemplo de plantilla):**

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<title>Blog - Página Principal</title>

<link rel="stylesheet" href="/css/blog.css">

</head>

<body>

<header>

<h1>Blog - Página Principal</h1>

</header>

<div class="content-container">

<!-- Aquí se mostrarán las 6 entradas más recientes -->

<!-- Estas se pueden generar manualmente o mediante un proceso similar al de update\_pages.php -->

</div>

<div id="pagination-container" class="pagination-container"></div>

<script src="/js/selector.js"></script>

</body>

</html>

## ****5. Instrucciones de implementación en tu servidor (Sered)****

1. **Sube los archivos al servidor:**
   * Coloca blog.html y los archivos generados (page1.html, page2.html, etc.) en la carpeta raíz de tu blog, por ejemplo, public\_html/blog/.
   * Sube el archivo update\_pages.php y entries.json dentro de la carpeta public\_html/blog/api/.
   * Sube el archivo selector.js dentro de public\_html/blog/js/.
   * Asegúrate de que tu hoja de estilos blog.css esté en public\_html/blog/css/.
2. **Crear las entradas:**
   * Sube tus archivos de entrada (por ejemplo, gengibre.html, dolor-de-espalda.html, etc.) a la carpeta public\_html/blog/.
   * Estos archivos deben contener al menos un <h1>, una <img> y un <p> para que el script los pueda procesar.
3. **Actualizar automáticamente las páginas:**
   * Cada vez que subas una nueva entrada, añade al final de cada entrada el siguiente fragmento de código para activar la actualización:

<script>

fetch("/blog/api/update\_pages.php")

.then(response => response.json())

.then(data => console.log(data.message))

.catch(console.error);

</script>

* + Alternativamente, puedes llamar manualmente a https://tu-dominio.com/blog/api/update\_pages.php (reemplaza "tu-dominio.com" por tu dominio) para actualizar las páginas.

1. **Verifica la correcta generación:**
   * Abre en tu navegador la URL https://tu-dominio.com/blog/api/update\_pages.php para comprobar que se generan/actualizan las páginas (deberías ver un mensaje JSON indicando éxito).
   * Luego, revisa blog.html y los archivos page1.html, page2.html, etc., para confirmar que se muestran las entradas y la paginación correcta.

Con estos archivos y pasos, tu blog se organizará automáticamente de la siguiente forma:

* **blog.html** muestra las entradas más recientes.
* **page1.html, page2.html, ...** contienen las entradas más antiguas, 6 por página.
* La **paginación** se genera y actualiza automáticamente mediante JavaScript.
* Al subir una nueva entrada (sin importar el nombre del archivo), se extrae la información necesaria y se distribuye en la página disponible.

¡Éxito con tu blog!