



Trabajamos con WordPress

PHP



Qualentum Lab

Trabajamos con WordPress



En el mundo del desarrollo web, es muy importante la fácil integración de webs, blogs o tiendas online. En este contexto, WordPress, un gestor de contenido (CMS), destaca por su capacidad para facilitar la interacción entre los usuarios y su contenido.

¡Empezamos!

Autor: Jesús Donoso

 [Introducción a WordPress](#)

 [Instalación de WordPress](#)

 [Estructura](#)

 [Hooks](#)

 [Loops](#)

 [Páginas y plantillas](#)

 [Creación de plugins personalizados](#)

 [Cómo trabajar con las bases de datos en WordPress](#)

 [Seguridad y buenas prácticas](#)

 [Conclusiones](#)

Introducción a WordPress



WordPress es un sistema de gestión de contenido de código abierto muy popular por su relevancia en el ámbito del desarrollo web.

Ampliamente utilizado, ofrece una **interfaz fácil** de usar que permite a los usuarios, incluso aquellos con habilidades técnicas limitadas, crear y administrar sitios web y blogs de manera eficiente.

La plataforma ha evolucionado significativamente. Desde su inicio como un sistema de blogs ha ido convirtiéndose en una **herramienta versátil** adaptable a diversas necesidades web. Su importancia radica en la capacidad que tiene para **facilitar la creación y personalización de sitios** a través de temas y plugins, permitiendo a los usuarios adaptar el diseño y la funcionalidad según sus requerimientos específicos.

La comunidad activa de usuarios y desarrolladores hacen de WordPress un entorno excepcional.

Esta comunidad contribuye con **temas, plugins y soluciones**, brindando un respaldo continuo y asegurando que la plataforma esté actualizada con las últimas tendencias y estándares tecnológicos.

WordPress también ha subido al pódium de la fama gracias a la **integración de prácticas de SEO**, a su capacidad para gestionar contenido multimedia y a sus constantes actualizaciones que mejoran la seguridad y añaden nuevas funcionalidades.

Como decíamos, esto ha convertido a WordPress en una opción popular para sitios web de diversos tamaños y propósitos, desde blogs personales hasta complejas plataformas de comercio electrónico.

Instalación de WordPress



En el procedimiento de instalación manual de WordPress, se siguen varios pasos que requieren atención y precisión. Estos pasos proporcionan **un mayor control sobre la configuración y son fundamentales** si queremos comprender a fondo la estructura interna de WordPress.

A continuación, entramos a detallar cada etapa del proceso. ¡Lee con atención!

Descarga de WordPress



Accede al sitio web oficial de WordPress en wordpress.org y descargar la versión más reciente del software.

Esto asegura que estás utilizando **la versión más actualizada y segura**.

Subida de archivos al servidor

Utilizando un cliente FTP, como FileZilla, procede a **cargar los archivos de WordPress en el servidor**. Se puede optar por colocarlos en el directorio raíz o en un subdirectorio, según las preferencias y la estructura deseada para el sitio web.

Creación de la base de datos

En el panel de control del servidor, se lleva a cabo **la creación de una base de datos MySQL y un usuario asociado**. Es crucial tomar nota de la información de la base de datos, incluyendo el nombre de la base de datos, el nombre de usuario y la contraseña, ya que se requiere durante el proceso de instalación.

Configuración del archivo 'wp-config.php'

El siguiente paso es **renombrar el archivo 'wp-config-sample.php' como 'wp-config.php'**. Este archivo es esencial para la conexión entre WordPress y la base de datos. Al abrirlo, introduce los detalles de la base de datos que se generaron en el paso anterior.

A continuación, puedes observar un ejemplo de configuración de este fichero:

```
<?php

// ** Configuración de MySQL - Puedes obtener esta información de tu
proveedor de alojamiento ** //
define('DB_NAME', 'nombre_de_tu_base_de_datos');
define('DB_USER', 'nombre_de_usuario');
define('DB_PASSWORD', 'tu_contraseña');
define('DB_HOST', 'localhost');
define('DB_CHARSET', 'utf8');
define('DB_COLLATE', '');

// ** Claves únicas de autenticación y sal: cambia estos valores a algo
único y seguro ** //
define('AUTH_KEY', 'coloca_tu_propia_clave_aqui');
define('SECURE_AUTH_KEY', 'coloca_tu_propia_clave_aqui');
define('LOGGED_IN_KEY', 'coloca_tu_propia_clave_aqui');
define('NONCE_KEY', 'coloca_tu_propia_clave_aqui');
define('AUTH_SALT', 'coloca_tu_propia_clave_aqui');
define('SECURE_AUTH_SALT', 'coloca_tu_propia_clave_aqui');
define('LOGGED_IN_SALT', 'coloca_tu_propia_clave_aqui');
define('NONCE_SALT', 'coloca_tu_propia_clave_aqui');

// ** Prefijo de la tabla de la base de datos para instalaciones múltiples
** //
$table_prefix = 'wp_'; // Cambia "wp_" a tu propio prefijo si es
necesario.

// ** Para desarrolladores: Modo de depuración de WordPress. ** //
define('WP_DEBUG', false);

/* ¡Eso es todo, deja de editar! ¡Feliz blogging! */
```

Inicio de la instalación

Al abrir un **navegador web** y **acceder a la URL del sitio**, se activa automáticamente el proceso de instalación de WordPress. Aquí, es necesario completar la información crucial, como el nombre del sitio, un nombre de usuario y una contraseña para el administrador, así como una dirección de correo electrónico.

Finalización de la instalación

Al hacer clic en 'Instalar WordPress', se ejecuta **el proceso final de instalación**. El sistema procesa la información proporcionada y, una vez completado, recibirás las indicaciones para acceder al panel de administración del sitio.

Este método manual, aunque implica seguir una serie de pautas, permite un mayor entendimiento de la infraestructura de WordPress y proporciona un control más preciso sobre la configuración del sitio web. Es el mejor procedimiento si buscamos personalizar la instalación y comprender mejor el funcionamiento interno del gestor.

Estructura



Qualentum Lab

La estructura de WordPress se encuentra **organizada en torno al lenguaje de programación PHP**, que desempeña un papel fundamental en la generación dinámica de contenido y la funcionalidad del sistema.

A continuación, vamos a estudiar **la relación entre la estructura de WordPress y el uso de PHP**.

Directorio raíz

Este directorio contiene archivos esenciales como '**index.php**', '**wp-config.php**', entre otros. El archivo '**index.php**' funciona como punto de entrada, iniciando la carga del núcleo de WordPress.

wp-admin

Resguarda archivos y recursos destinados al área de administración del sitio.

wp-includes

Contiene archivos críticos, como clases y funciones esenciales de WordPress.

wp-content

En este directorio se almacenan temas, plugins y archivos multimedia, desempeñando un papel vital en la personalización del sitio.

wp-content/themes

Contiene temas de WordPress, cada uno con su propia estructura de archivos, incluyendo archivos PHP para distintas plantillas.

wp-content/plugins

Alberga plugins de WordPress, cada uno en su propio directorio con archivos PHP específicos del plugin.

Archivos PHP clave

wp-config.php

En el directorio raíz, este archivo contiene configuraciones críticas, como la conexión a la base de datos y claves de seguridad.

functions.php

Ubicado en temas y en el directorio raíz del tema activo, permite la inclusión de funciones personalizadas y modificaciones al comportamiento del tema.

header.php, footer.php...

En el directorio del tema, estos archivos definen la estructura de la cabecera, el pie de página y otras partes del diseño del sitio.

single.php, page.php...

También en el directorio del tema, estos archivos determinan la apariencia de entradas individuales, páginas y otros tipos de contenido.

wp-login.php

Gobierna la página de inicio de sesión de WordPress y reside en el directorio raíz.

En definitiva, la estructura de WordPress de la mano de PHP nos brinda un desarrollo web dinámico y altamente personalizable.

La modularidad de los temas y plugins, junto con la potencia de PHP, ofrece a los desarrolladores una amplia flexibilidad para crear sitios web personalizados, únicos.

Hooks



Qualentum Lab

En el contexto de WordPress, **los hooks son elementos cruciales** en cuanto a personalizar y ampliar la funcionalidad del sistema de manera elegante y modular.

Los *hooks*, o ganchos, son puntos estratégicos a lo largo de la ejecución de WordPress donde se puede incorporar código personalizado.

Existen **2 tipos**:

- Action hooks:** estos eventos específicos permiten la ejecución de código en momentos clave del proceso.
- Filter hooks:** gracias a ellos, podemos modificar datos antes de que se utilicen.

En los siguientes apartados vamos a detallar cada tipo de *hook*.

Action hooks

Los ganchos de acción en WordPress son eventos específicos que ocurren en puntos clave durante la ejecución del sistema.

Estos eventos permiten a los desarrolladores **ejecutar su propio código** en respuesta a acciones específicas. A continuación, vamos a describir los *action hooks* más comunes.

1

Init

Este *hook* se dispara cuando WordPress ha terminado de cargar, pero antes de que se envíe cualquier contenido.

Es útil para inicializar variables y configurar funcionalidades adicionales.

```
add_action('init', 'mi_funcion_personalizada');

function mi_funcion_personalizada() {
    // Código a ejecutar en la inicialización
}
```

2

wp_head

Se activa en la sección `<head>` del HTML en cada página.

Es útil para agregar scripts, estilos y otros elementos a la cabeza del documento.

```
add_action('wp_head', 'mi_funcion_personalizada');

function mi_funcion_personalizada() {
    // Código para agregar elementos en la cabeza del documento
}
```

3

wp_footer

Es similar a `wp_head`, pero se activa en la sección `<footer>` del HTML al final de cada página.

Puede utilizarse para **agregar scripts y contenido al pie de página**.

```
add_action('wp_footer', 'mi_funcion_personalizada');

function mi_funcion_personalizada() {
    // Código para agregar elementos en el pie de página
}
```

4

save_post

Este *hook* se activa cuando se guarda una entrada o una página. Puede utilizarse para **realizar acciones específicas** después de que se ha guardado el contenido.

```
add_action('save_post', 'mi_funcion_personalizada');

function mi_funcion_personalizada($post_id) {
    // Código a ejecutar al guardar una entrada o página
}
```

5

admin_menu

Se dispara cuando se construye el menú de administración. Es útil para **agregar elementos de menú personalizados** en el panel de administración.

```
add_action('admin_menu', 'mi_funcion_personalizada');

function mi_funcion_personalizada() {
    // Código para agregar elementos al menú de administración
}
```

Como advertíamos, estos son solo algunos ejemplos de *action hooks*, pero existe una amplísima variedad que nos permite a los desarrolladores intervenir en diferentes etapas del ciclo de vida de WordPress y personalizar el comportamiento del sistema según nuestras necesidades.

Filter hooks

En WordPress, los filtros son *hooks* que nos sirven para **modificar o filtrar datos** antes de que se presenten en la pantalla o de que se almacenen en la base de datos. Gracias a ellos, por tanto, podemos personalizar la salida de datos en diversas partes del sistema.

Un filtro en WordPress tiene una estructura básica compuesta por dos elementos principales: el nombre del filtro y su función.

EL NOMBRE DEL FILTRO

LA FUNCIÓN DEL FILTRO

Es un identificador único para el filtro. Estos filtros en WordPress están asociados con eventos o lugares específicos en los que se aplican.

EL NOMBRE DEL FILTRO

LA FUNCIÓN DEL FILTRO

Esta se ejecutará cuando se aplique el filtro. Esta función toma un valor como argumento, realiza alguna manipulación en ese valor y luego devuelve el resultado.

Supongamos que queremos modificar el contenido del título de una publicación antes de que se presente en la pantalla. En ese caso, utilizaremos el filtro *the_title*.

```
// Definir la función de filtro
function modificar_titulo($titulo) {
    // Realizar alguna manipulación en el título
    return 'Prefijo: ' . $titulo;
}

// Agregar la función de filtro al hook the_title
add_filter('the_title', 'modificar_titulo');
```

En este ejemplo, *the_title* es el nombre del filtro al que se conecta la aplicación. Este filtro se aplica al título de una publicación. *Modificar_titulo* es la función que se ejecuta cuando se aplica el filtro. Toma el título actual como argumento, agrega un prefijo y devuelve el nuevo título.

Ahora, vamos a estudiar otras **dos aplicaciones básicas de los filtros**.

1

Filtros en plantilla

Podemos utilizar filtros en un tema de WordPress para **modificar dinámicamente la salida de datos**. Por ejemplo, es posible cambiar el formato de las fechas, agregar enlaces adicionales a las publicaciones, o incluso modificar el contenido del cuerpo de una publicación.

```
// Modificar la salida de las fechas en los comentarios
function modificar_formato_fecha($fecha) {
    return date('d-m-Y H:i:s', strtotime($fecha));
}

add_filter('get_comment_date', 'modificar_formato_fecha');
```

Como vemos en este ejemplo, se utiliza el filtro `get_comment_date` para cambiar el formato de las fechas en los comentarios.

2

Filtros anidados

En WordPress, también es común **encontrar filtros anidados donde varios filtros pueden aplicarse en cadena** para modificar datos de manera sucesiva. Podemos ver su aplicación en el siguiente ejemplo.

```
Copy code
function filtro_anidado_1($contenido) {
    // Manipulación del contenido aquí
    return $contenido;
}

function filtro_anidado_2($contenido) {
    // Manipulación adicional del contenido
    return $contenido;
}

add_filter('nombre_del_filtro', 'filtro_anidado_1');
add_filter('nombre_del_filtro', 'filtro_anidado_2');
```

Recuerda: a la hora de modificar datos antes de almacenarlos o presentarlos, tus mejores amigos en WordPress son los filtros. Y es que tener el poder de personalizar la salida de datos, según el contexto, nos otorga a los desarrolladores un control granular sobre el contenido de su sitio.

Loops



El *loop* en WordPress es un componente central que orquesta la presentación dinámica del contenido en un sitio web.

Este ciclo iterativo permite la recuperación y exhibición de entradas almacenadas en la base de datos del gestor, siguiendo una estructura definida en las plantillas de temas. ¿Y cómo es ese proceso?

Lo estudiamos paso a paso...

Inicio del *loop*

El *loop* se inicia con **la evaluación de la existencia de entradas** disponibles mediante código (como el siguiente), generalmente, ubicado en archivos como *index.php*.

```
<?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
```

Este bloque verifica la presencia de entradas y, de ser así, inicia un bucle que recorre cada una de ellas.

Contenido del *loop*

Dentro del *loop*, la función *the_post()* avanza al siguiente elemento, permitiendo el acceso a los datos de la entrada actual. Las funciones de WordPress, como *the_title()* y *the_content()*, son utilizadas para **mostrar información específica** de la entrada.

```
// Muestra el título de la entrada  
the_title();  
  
// Muestra el contenido de la entrada  
the_content();
```

Este segmento ofrece flexibilidad al desarrollador para personalizar la presentación según requisitos específicos.

Fin del loop

Tras presentar el contenido de la entrada actual, **se cierra el bucle** mediante el siguiente código:

```
<?php endwhile; endif; ?>
```

Esta estructura verifica la existencia de más entradas y concluye el bucle, en el caso contrario.

Reseteo del loop

En situaciones donde **se anidan bucles adicionales**, se puede requerir el reinicio del *loop* para evitar interferencias. Esto se realiza mediante la función `wp_reset_postdata()`.

```
<?php wp_reset_postdata(); ?>
```

Esto asegura que funciones subsiguientes que dependan del loop no se vean afectadas por ejecuciones previas.

¡Quédate con esto! El *loop* en WordPress es la herramienta clave para la presentación coherente de contenido. Permite que los sitios web actualicen automáticamente su apariencia a medida que se añaden nuevas entradas. Esto nos ayuda a los desarrolladores a personalizar la visualización de cada entrada según las necesidades particulares de nuestros proyectos.

Páginas y plantillas



Las páginas y plantillas son elementos fundamentales de WordPress. La creación de estos elementos implica seguir un conjunto de pasos para personalizar la apariencia y estructura de un sitio web.

Creación de páginas

En WordPress, **una página se define como un tipo de contenido estático** destinado a mostrar información que generalmente no experimenta cambios frecuentes. A diferencia de las entradas, que suelen utilizarse para contenido dinámico y con fecha, las páginas son ideales para la información más permanente.

Verás que crear una nueva página es bastante sencillo, ¡apunta!

1

Inicia sesión en el panel de administración de WordPress utilizando las credenciales correspondientes.

- 2 Accede al menú lateral del panel de administración y selecciona **la sección 'Páginas'**.
- 3 Selecciona **'Añadir Nueva'** para comenzar la creación de una nueva página.
- 4 **Edita un título** para la página en el campo respectivo. Después agrega el contenido deseado mediante el uso del editor visual o de texto.
- 5 Al completar la edición, haz **clic en 'Publicar'** para que la página esté disponible en el sitio.

Creación de plantillas

Una plantilla en WordPress se define como un archivo o conjunto de archivos en los que se especifica la estructura y el diseño de una página particular en un sitio web.

Son elementos esenciales para la presentación visual del contenido en un sitio WordPress, dictando cómo se mostrará la información generada dinámicamente por el sistema.

Existen varios tipos de plantillas en WordPress, cada una diseñada para **manejar un tipo específico de contenido o página en el sitio**. Algunos ejemplos comunes incluyen plantillas para la página principal, publicaciones individuales, páginas individuales, archivos de categorías, archivos de etiquetas, archivos de autor y más.

Ahora veamos los pasos necesarios para la creación de una plantilla...

1

Determina la estructura: decide la estructura y el diseño deseados para la página. Puede tratarse de una plantilla completamente nueva o basada en una existente.

2

Crea un archivo de plantilla: utilizando un editor de texto, se crea un nuevo archivo PHP para la plantilla, asignándole un nombre según la preferencia (por ejemplo, *plantilla-personalizada.php*).

3

Encabezado de la plantilla: al inicio del archivo de la plantilla, se agrega un bloque de código para indicar que se trata de una plantilla de WordPress. El nombre puede ajustarse según las necesidades o preferencias.

```
<?php
/*
Template Name: Plantilla Personalizada
*/
?>
```

4

Codificación de la plantilla: se desarrolla el código HTML y PHP para la plantilla, incorporando funciones de WordPress como el Loop y las llamadas a funciones, según sea necesario.

```
<?php get_header(); ?>

<!-- Contenido de la plantilla -->

<?php get_footer(); ?>
```

5

Guardar y subir: el archivo de la plantilla se guarda y se carga en el directorio del tema de WordPress, ya sea a través de FTP o del administrador de archivos proporcionado por el proveedor de alojamiento.

Asignación de una plantilla a una página

Ahora, necesitamos **asignar una plantilla a una página**, ¿y cómo es el procedimiento? Vamos a verlo también.

- Editar la página:** se accede a la sección de páginas y se selecciona la página específica que se desea modificar.
- Ajustes de página:** en la barra lateral, se busca la sección 'Atributos de Página' u opción similar.
- Seleccionar plantilla:** en la opción 'Plantilla', se elige la plantilla deseada.
- Actualizar:** por último, se guardan los cambios y se actualiza la página.

Siguiendo este proceso, las páginas hacen usos de plantillas personalizadas y los desarrolladores podemos controlar su apariencia y estructura.

Creación de plugins personalizados



Un plugin se define como un complemento de software que se puede agregar al núcleo de WordPress para ampliar sus capacidades y funcionalidades.

Ofrecen a los usuarios y desarrolladores la capacidad de **personalizar y mejorar sus sitios de WordPress sin tener que modificar directamente el código fuente** del sistema.

Los plugins, como bien sabemos, nos proporcionan funcionalidades adicionales a un sitio web. Pueden abarcar desde la incorporación de nuevas características, como galerías de imágenes, formularios de contacto o integraciones con redes sociales, hasta la mejora del rendimiento del sitio mediante la optimización de imágenes, el almacenamiento en caché y la gestión de la velocidad de carga.



La creación de plugins personalizados es una práctica común en WordPress para **extender y personalizar la funcionalidad del sistema**. Y aquí vamos a aprender el proceso general para crear un plugin personalizado.

Crear un directorio para el plugin

En el directorio `wp-content/plugins/` de la instalación de WordPress, se crea un nuevo directorio con el nombre del plugin en minúsculas y sin espacios.

```
wp-content/plugins/mi-plugin/
```

Crear el archivo principal del plugin

Dentro del directorio del plugin, **se crea un archivo principal** con una extensión '.php', por ejemplo, *mi-plugin.php*.

Encabezado del plugin

Se agrega un encabezado al archivo principal del plugin, incluyendo al menos el nombre del mismo.

```
<?php
/*
Plugin Name: Mi Plugin Personalizado
Description: Descripción del plugin.
Version: 1.0
Author: Tu Nombre
*/
```

Definir la funcionalidad del plugin

Se añade el código necesario para la funcionalidad del plugin, incluyendo funciones, ganchos (*hooks*), filtros y cualquier otra lógica específica.

```
// Ejemplo de función simple en el plugin
function mi_funcion_personalizada() {
    // Código de la función
}

// Enganchar la función a un gancho (hook)
add_action('init', 'mi_funcion_personalizada');
```

Activar el plugin

En el panel de administración de WordPress, se accede a 'Plugins' y se activa el nuevo plugin.

Test del plugin

Se realiza una prueba del plugin para garantizar su correcto funcionamiento. Y si es necesario, realizamos ajustes y mejoras según sea necesario.

Documentación

Se incluyen **comentarios descriptivos en el código** para que otros desarrolladores entiendan el plugin o a modo de recordatorio para nosotros mismos de cara a un futuro (y los consecuentes olvidos).

Empaquetar y distribuir (opcional)

Si se planea compartir el plugin, se considera **empaquetarlo adecuadamente y distribuirlo** a través del repositorio oficial de WordPress u otros medios.

Ahora compartimos un ejemplo básico de un plugin que agrega un *shortcode* para mostrar un saludo en una página o entrada.

```
<?php
/*
Plugin Name: Saludo Personalizado
Description: Agrega un shortcode para mostrar un saludo.
Version: 1.0
Author: Tu Nombre
*/

// Función que retorna el saludo
function saludar_shortcode() {
    return '|Hola, este es un saludo personalizado!';
}

// Registra el shortcode
add_shortcode('saludo', 'saludar_shortcode');
```

Fíjate bien: este plugin define una función `saludar_shortcode` que se ejecuta cuando se encuentra el shortcode `[saludo]` en una página o entrada, mostrando el mensaje especificado.

Cómo trabajar con las bases de datos en WordPress



Qualentum Lab

El trabajo con bases de datos en WordPress implica una serie de operaciones esenciales para gestionar datos de manera eficiente. Aquí vamos a ofrecer una visión general de cuáles son estas operaciones y de su manejo.

Conexión a la base de datos

WordPress maneja la conexión a la base de datos automáticamente a través del archivo de configuración **wp-config.php**, que contiene detalles como el nombre de la base de datos, el usuario y la contraseña.

Recuperar datos

Para recuperar datos, se utilizan **funciones** como `wpdb::get_results` y `wpdb::get_var`.

Echa un vistazo a este ejemplo sobre cómo obtener todas las entradas de la tabla de entradas.

```
global $wpdb;
$resultados = $wpdb->get_results("SELECT * FROM {$wpdb->prefix}posts WHERE
post_type = 'post'");
```

Insertar datos

La función `wpdb::insert` se utiliza para **insertar datos**. Fíjate en el siguiente ejemplo para aprender a agregar una nueva entrada.

```
global $wpdb;
$wpdb->insert(
    $wpdb->prefix . 'posts',
    array(
        'post_title' => 'Nuevo Título',
        'post_content' => 'Contenido de la nueva entrada',
        'post_status' => 'publish',
        'post_type' => 'post',
    )
);
```

Actualizar datos

La función `wpdb::update` se emplea para **actualizar datos**. Por ejemplo, para modificar el título de una entrada sería así:

```
global $wpdb;
$wpdb->update(
    $wpdb->prefix . 'posts',
    array('post_title' => 'Nuevo Título Modificado'),
    array('ID' => 1)
);
```

Eliminar datos

Lo haremos con la función `wpdb::delete` y si queremos eliminar una entrada, podemos ejecutarlo así:

```
global $wpdb;
$wpdb->delete(
    $wpdb->prefix . 'posts',
    array('ID' => 1)
);
```

Consultas personalizadas

Con la función `wpdb::query` podemos ejecutar consultas SQL personalizadas. Sin embargo, se recomienda utilizar las funciones de WordPress siempre que sea posible para garantizar compatibilidad y seguridad.

```
global $wpdb;  
$wpdb->query("DELETE FROM {$wpdb->prefix}posts WHERE post_type = 'post' AND  
post_status = 'draft'");
```

¡Ojo! Es fundamental tener precaución al interactuar directamente con la base de datos, garantizando la seguridad y procesamiento adecuado de los datos. La clase `wpdb` de WordPress ofrece métodos seguros para ejecutar operaciones en la base de datos, siguiendo las mejores prácticas y asegurando la integridad del sistema.

Seguridad y buenas prácticas



El mantenimiento de la seguridad y la adopción de buenas prácticas en WordPress son fundamentales para **garantizar la integridad y protección del sitio web**. Tan fundamentales son también llevar a cabo las siguientes recomendaciones:

- **Mantener nuestro WordPress actualizado.** Lo ideal es utilizar la versión más reciente, ya que las actualizaciones periódicas suelen abordar vulnerabilidades de seguridad.
- **Actualizar los plugins y temas.** Es crucial mantener actualizados todos los plugins y temas instalados para beneficiarse de correcciones de seguridad proporcionadas por los desarrolladores.
- **¿Un must? Crear contraseñas robustas.** Se aconseja el uso de contraseñas sólidas tanto para la cuenta de WordPress como para la base de datos, evitando contraseñas predecibles.
- **Una buena gestión de roles y permisos.** Asignar roles y permisos adecuados a los usuarios es esencial para evitar privilegios excesivos y garantizar la seguridad.
- **Asegúrate de la configuración del servidor.** Se recomienda una configuración segura del servidor, incluyendo firewalls y ajustes seguros de PHP.

- **Haz copias de seguridad regulares.** La realización periódica de copias de seguridad del sitio y la base de datos proporciona una medida de seguridad en caso de incidentes.
- **Limitación de intentos de acceso.** Implementar medidas para limitar intentos de acceso, como bloquear direcciones IP después de varios intentos fallidos, refuerza la seguridad.
- **Implementación de SSL.** Utilizar un certificado SSL es fundamental para cifrar la comunicación entre el servidor y los usuarios, protegiendo la información transmitida.
- **Monitorización de actividades.** Herramientas de monitorización y plugins de seguridad pueden ayudar a detectar y responder a actividades sospechosas.
- **Eliminación de elementos no utilizados.** Desactivar y eliminar elementos no utilizados, como plugins y temas, reduce las posibles vulnerabilidades.
- **Validación y escape de datos.** Al mostrar datos dinámicamente, el uso de funciones de validación y escape en WordPress previene inyecciones de código.
- **Evita el usuario admin.** La evitación del nombre de usuario predeterminado 'admin' y la elección de un nombre único refuerzan la seguridad.
- **Seguridad de htaccess y wp-config.php.** Asegurar correctamente el archivo .htaccess y wp-config.php protege contra accesos no autorizados.
- **Realizar auditorías del sitio.** Estas auditorías deben ser periódicas, y escanear el sitio en busca de vulnerabilidades es una práctica muy recomendable.
- **Conocimiento de mejores prácticas de desarrollo.** Para garantizarnos un desarrollo seguro, validar y sanitizar datos es esencial.

Siguiendo estas recomendaciones, fortalecemos significativamente la seguridad del sitio WordPress y reducimos los riesgos potenciales de amenazas y ataques. Además, como buenos profesionales, debemos mantenernos siempre informados sobre las últimas prácticas y vulnerabilidades, apostar por un enfoque proactivo hacia la seguridad del sitio nos evitará situaciones críticas y peligros.

Conclusiones



A lo largo de este fastbook, hemos visto qué es WordPress recorriendo todo su potencial de esta popular plataforma de gestión de contenido. Se han detallado **aspectos fundamentales de este gestor y se han proporcionado pautas** sobre su instalación y configuración.

Al mismo tiempo se han tratado conceptos elementales como la **creación y configuración de páginas y plantillas**, el desarrollo de *hooks*, la interacción con una base de datos y la implementación de plugins.

De esta forma, ya albergas **todos los conceptos básicos y buenas prácticas** para el desarrollo de sitios web con Wordpress.



No obstante, te sugiero que amplíes información [en su página oficial](#).

¡Enhорabuena! Fastbook superado



Qualentum.com