



Plugin de reservas en Wordpress

Fernando Bravo Sánchez

Índice

Objetivos	3
Introducción	4
¿Qué es Wordpress ?	4
¿Que realiza un plugin en Wordpress?	4
Escenario	5
Configuración e instalación	6
Elaboración del plugin de Reservas	7
Configuración Inicial	7
Creación de la estructura básica	7
Cabecera del plugin	8
Proteger archivo php de accesos fuera del entorno Wordpress	8
Registrar Plugin	9
Activar Plugin	10
Desactivar Plugin	10
Implementación formulario de reservas	11
Creación del formulario	11
Procesar formulario	12
Guardar los datos en la base de datos	13
Enviar correo de confirmacion al usuario	14
Guardamos las reservas en un archivo log para el Administrador	14
Mostrar un mensaje al realizar la reserva	17
Mostrar un mensaje de error al realizar una reserva	17
Carga del procesamiento de datos con INIT	18
Creación menú en el panel de administración de Wordpress	19
Agregar un Menú en el panel de administración de wordpress	20
Mostrar las reservas en el panel de administración	21
Vista Previa del plugin básico funcionando	23
Formulario desde la página de wordpress	23
Vista de las reservas realizadas desde el administrador	25
Opciones adicionales añadidas	26
Plugin SMTP de Wordpress para enviar correo	26
Formulario horas y fecha posterior a 7 dias	27
Funcion rp_mostrar_formulario_reservas	27
Funcion rp_procesar_formulario_reserva	29
Visualizar CSV en el panel de administración	30
Creacion del submenú	30
Creación de la función	31
Dar estilo al formulario	35
Vista del formulario con CSS	39
Activación Plugin y Desactivación Plugin	41
Seguridad Plugin aplicada	44
Comprobación ABSPATH	44
Comprobación Sanitize	44

Objetivos

Este plugin consistirá en permitir a los usuarios realizar reservas desde el sitio web . El plugin está dirigido para empresas que requieran gestión de reservas como por ejemplo un restaurante . El plugin ofrecerá una interfaz agradable a la vista permitiendo a los usuarios seleccionar fechas y horarios disponibles en la empresa . El administrador tendrá la vista de las reservas realizadas a los usuarios en el panel de administración , tanto la vista de un archivo csv como la de un archivo log .

Introducción

¿Qué es Wordpress ?

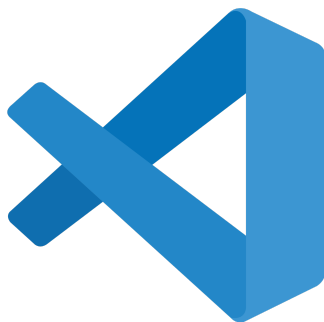
WordPress es una plataforma de gestión de contenidos, representa un software de código abierto altamente versátil que permite a los usuarios crear y desarrollar sitios web personalizados de manera sencilla, adaptable y profesional, con una muy buena experiencia de usuario.

¿Que realiza un plugin en Wordpress?

Los plugins extienden las capacidad de WordPress con utilidades, que van desde algo muy simple, a cambiar completamente el funcionamiento de la web. Existen plugins para compartir en redes sociales, para tener una tienda online, para mejorar el rendimiento del sitio, para crear widgets...

Escenario

El escenario que he elegido para realizar este proyecto de FCT ha sido alojar una máquina virtual con Debian 12 . Esta máquina está alojada en un servidor Proxmox del **les Ciudad Jardín** . A esta máquina se instalará Wordpress , mysql y visual studio Code para poder realizar el código del plugin .



Configuración e instalación

La instalación de wordpress y mysql no la comentaré ya que es irrelevante , continuaré con la explicaciones del código e imágenes de este más la configuración del mysql para poder enlazarla con el plugin , también estará subido el código del proyecto una vez esté finalizado del en mi [github](#) . Como ya vemos aquí ya tenemos instalado wordpress en la máquina.



También tenemos instalado una base de datos mariadb

```
root@fernando:~# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.6-MariaDB-0+deb12u1 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases ;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| wordpress |
+-----+
5 rows in set (0.003 sec)
```

Elaboración del plugin de Reservas

Primero empezaremos por la configuración inicial del plugin, es decir el desarrollo básico para empezar a hacer funcionar este plugin .

Configuración Inicial

Para ello primero debemos crear una carpeta para guardar el plugin y que wordpress lo reconozca.

Creación de la estructura básica

Vamos a la ruta donde se instaló wordpress y buscamos la carpeta **plugins** . Dentro de esta carpeta creamos otra carpeta y la llamamos como en mi caso **Reservas** .

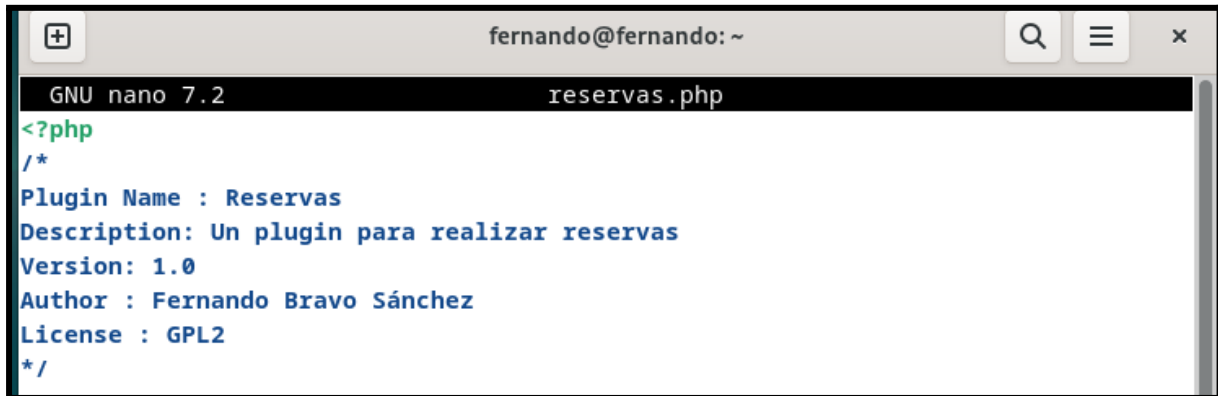
```
root@fernando:/var/www/html/wordpress/wp-content/plugins# mkdir reservas
root@fernando:/var/www/html/wordpress/wp-content/plugins# ls
akismet hello.php index.php reservas
root@fernando:/var/www/html/wordpress/wp-content/plugins# █
```

Ahora dentro de la carpeta **Reservas** crearemos un archivo con la extensión php el cual reconocerá wordpress para utilizar el plugin en este . En mi caso lo llamo **reservas.php** .

```
root@fernando:/var/www/html/wordpress/wp-content/plugins/reservas# ls
reservas.php ←
```

Cabecera del plugin

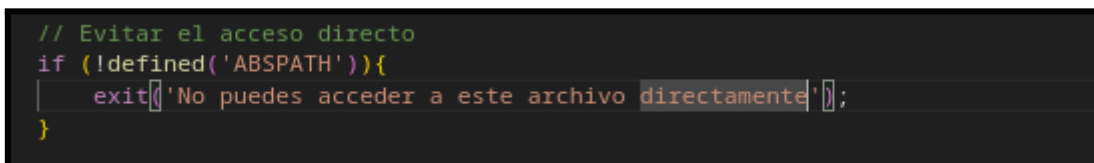
Dentro del archivo **reservas.php** , añadiremos la cabecera del plugin que es lo que reconocerá wordpress para leer la información del plugin desde el panel de administración .



```
fernando@fernando: ~
GNU nano 7.2 reservas.php
<?php
/*
Plugin Name : Reservas
Description: Un plugin para realizar reservas
Version: 1.0
Author : Fernando Bravo Sánchez
License : GPL2
*/
```

Proteger archivo php de accesos fuera del entorno Wordpress

Para ello debemos definir la ruta **ABSPATH** , se utiliza para proteger nuestro archivo php de que se intente ejecutar desde el entorno fuera de Wordpress . Con esto nos aseguramos de que el archivo se cargue desde el núcleo de Wordpress . [ABSPATH](#) es una ruta absoluta al directorio raíz de Wordpress y siempre está definida cuando Wordpress esté ejecutándose.



```
// Evitar el acceso directo
if (!defined('ABSPATH')){
    exit('No puedes acceder a este archivo directamente');
}
```


Registrar Plugin

Para asegurarnos de que el plugin se registre correctamente dentro de Wordpress. Vamos a utilizar un hook para registrar funciones , estilos etc.. cuando el plugin se active. También crearé una función para las funciones básicas del plugin. Lo cual lo haremos más adelante.

```
//Hook para iniciar el plugin
add_action('init','rp_inicializar_reservas_del_plugin');

function rp_inicializar_reservas_del_plugin(){
    // Aqui van las funciones básicas del plugin
}
```

Activar Plugin

Crearemos una función que al [activar el plugin](#) se creará las tablas en la base de datos . Crearemos un hook que activará esta función .

```
24 //Funcion al activar el plugin
25 function rp_activar_reservas_plugin(){
26     //Crearemos las tablas
27 }
28 registrar_activacion_hook(__FILE__, 'rp_activar_reservas_plugin');
29
```

Desactivar Plugin

Crearemos una función que al [desactivar el plugin](#) borre la tabla de la base de datos . Crearemos un hook que activará esta función.

```
//Función al desactivar el plugin
function rp_desactivar_reservas_plugin(){

    //Borramos los archivos temporales
}
registrar_desactivacion_hook(__FILE__, 'rp_desactivar_reservas_plugin');
|
```

Implementación formulario de reservas

Ahora crearemos un formulario el cual verán los usuarios a la hora de realizar la reserva.

Creación del formulario

Para ello crearemos una función en la cual guardaremos el formulario , utilizaremos el comando [ob_start](#) el cual guardará el formulario en el buffer . También utilizaremos el método POST para la creación del formulario. Una vez creado el formulario devolveremos el formulario capturado en el buffer con el comando [ob_get_clean](#) . Y por último añadiremos un shortcode a la función para darle otro nombre ya que lo utilizaremos más adelante , en mi caso lo llamo **reserva_formulario**

```
41 // Creamos una funcion para mostrar el formulario
42
43 function rp_mostrar_formulario_reservas() {
44     ob_start();
45
46     ?>
47     <form method="post" action="">
48         <label for="nombre">Nombre : </label>
49         <input type="text" id="nombre" name="nombre" required>
50
51
52         <label for="email">Email : </label>
53         <input type="email" id="email" name="email" required>
54
55         <label for="fecha">Fecha : </label>
56         <input type="date" id="fecha" name="fecha" required>
57
58
59         <label for="hora">Hora : </label>
60         <input type="time" id="hora" name="hora" required>
61     </form>
62     <?php
63
64     return ob_get_clean();
65 }
66
67 add_shortcode('reserva_formulario' , 'rp_mostrar_formulario_reservas');
```

Procesar formulario

Ya que el formulario es enviado , necesitamos manejar estos datos para ello lo podemos hacer en PHP . Por lo que crearé una función que realizará el manejo de datos y almacenará la reserva en la base de datos creada para guardar estos datos. Utilizaremos el método [Post](#) ya que esto asegura que solo se procesa el formulario cuando los datos necesarios en mi caso (**nombre , email , fecha , hora**) estén rellenos . A las variables (**nombre , email , fecha , hora**) le validan los datos con **\$POST** y le añado un formato de texto (**sanitize_text_field**) a las variables **nombre fecha , hora** para eliminar caracteres no deseados . A la variable email le añado un formato de texto (**sanitize_email**) para que valide que es un email .

```
// Procesar formulario.php

function rp_procesar_formulario_reserva(){

    if (($_SERVER['REQUEST_METHOD'] == 'POST' && isset($_POST['nombre'])
    && isset($_POST['email']) && isset($_POST['fecha']) && isset($_POST['hora'])) {

        $nombre = sanitize_text_field($_POST['nombre']);
        $email = sanitize_email($_POST['email']);
        $fecha = sanitize_text_field($_POST['fecha']);
        $hora = sanitize_text_field($_POST['hora']);
```

Guardar los datos en la base de datos

Una vez los datos están procesados estos se guardan en la tabla de **reservas** de la base de datos . Para ello debemos hacer un insert con la clase **\$wpdb** . Utilizamos el prefijo de la tabla con **\$wpdb->prefix** . Insertamos los valores con un **array** .

```
// Aquí podemos almacenar la reserva en la base de datos
global $wpdb;
$tabla_reservas = $wpdb->prefix . 'reservas';
$wpdb->insert($tabla_reservas, array(
    'nombre' => $nombre,
    'email' => $email,
    'fecha' => $fecha,
    'hora' => $hora
));
```

Enviar correo de confirmacion al usuario

Para ello crearemos dos variables una llamada **asunto** que será el título del mensaje y otra que será el **mensaje** que le diremos cuando realizó la reserva y su hora . Para enviar el email utilizaremos la función **wp_mail** . [Esto da error ya que no dispongo de un servidor SMTP .](#)

```
//enviar correo de confirmación al usuario
$asunto = 'Confirmación de reserva';
$mensaje = "Hola $nombre , tu reserva se realizó para la fecha del $fecha a las $hora";
wp_mail($email,$asunto,$mensaje);
```

Guardamos las reservas en un archivo log para el Administrador

Cuando un usuario realice una reserva se guardará en un archivo log para que el administrador vea las reservas y tenga una copia de éstas . Para ello primero creamos el archivo log en el directorio donde se guarda el plugin para ello utilizamos [plugin_dir_path](#) . Luego creamos las entradas al archivo con las variables donde se guarda el nombre email fecha y hora con saltos de línea con `\n` . Utilizamos [file_put_contents](#) para escribir en el archivo del log . Esta parte del código la situamos en la función `procesar_formulario_reserva` , ya que queremos que guarde la reserva una vez se realice .

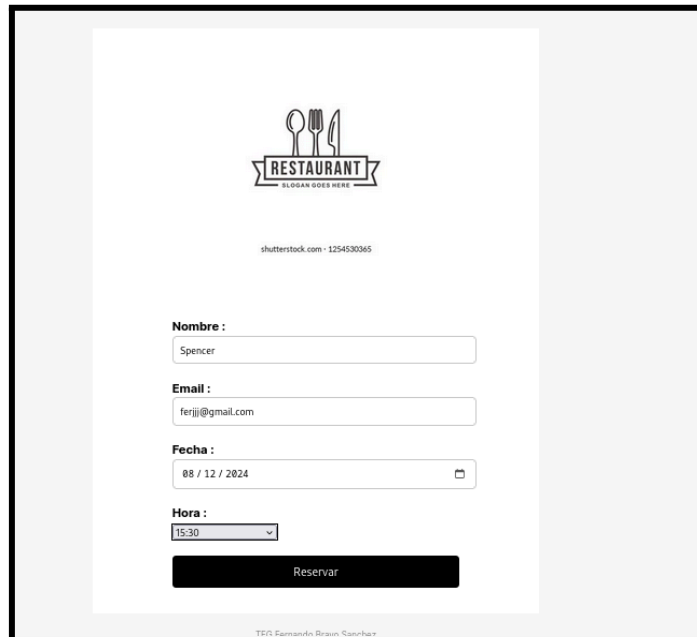
```
//Notificar al administrador
$ARCHIVO_LOG = plugin_dir_path(__FILE__) . 'reservas-admin.log';
$entrada = "Reserva realizada:\n";
$entrada .= "_____\n";
$entrada .= "Nombre: $nombre\n";
$entrada .= "Email: $email\n";
$entrada .= "Fecha: $fecha\n";
$entrada .= "Hora: $hora\n";
$entrada .= "_____\n";

//Escribir en el archivo

file_put_contents($ARCHIVO_LOG, $entrada , FILE_APPEND);

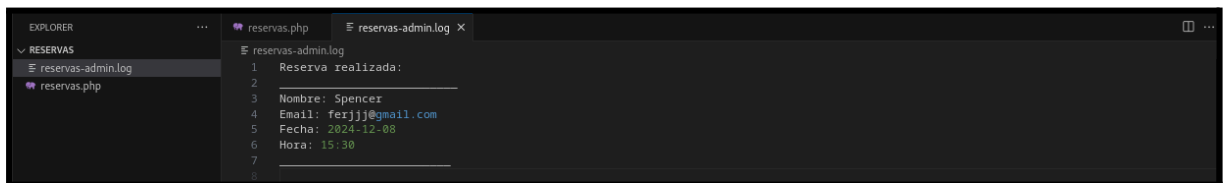
echo '<div class="reserva-mensaje reserva-exito"> ¡Reserva Realizada! El administrador ha sido notificado</div>';
```

Vamos a realizar una reserva



A screenshot of a web form for a restaurant reservation. At the top is a logo with a fork and knife and the word "RESTAURANT". Below the logo is a placeholder text "shutterstock.com - 1254530065". The form contains four input fields: "Nombre:" with the value "Spencer", "Email:" with the value "ferjjj@gmail.com", "Fecha:" with the value "08 / 12 / 2024", and "Hora:" with a dropdown menu showing "15:30". A black button labeled "Reservar" is at the bottom.

Vamos a mirar el archivo log , como vemos se ha realizado correctamente



A screenshot of a code editor showing a log file named "reservas-admin.log". The log contains the following text:

```
1 Reserva realizada:
2
3 Nombre: Spencer
4 Email: ferjjj@gmail.com
5 Fecha: 2024-12-08
6 Hora: 15:30
7
8
```

Ahora vamos a eliminar el archivo log una vez desactivemos el plugin , para ello nos vamos a la función [desactivar reservas plugin](#) , Utilizamos la función [unlink](#) para borrar el archivo .

```
//Función al desactivar el plugin
function rp_desactivar_reservas_plugin(){

    global $wpdb;
    $tabla_reservas = $wpdb->prefix . 'reservas';

    // Eliminar tabla
    $wpdb->query("DROP TABLE IF EXISTS $tabla_reservas");

    //Eliminar archivo log

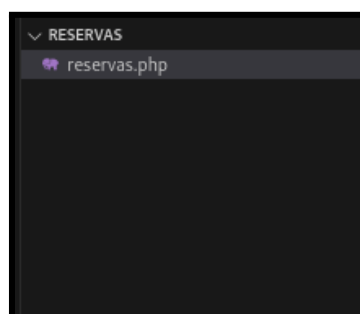
    $ARCHIVO_LOG = plugin_dir_path(__FILE__) . 'reservas-admin.log';

    if(file_exists($ARCHIVO_LOG)){
        unlink($ARCHIVO_LOG);
    }

}

register_deactivation_hook(__FILE__, 'rp_desactivar_reservas_plugin');
```

Cuando desactivamos el plugin se borra el archivo log , como vemos aquí



Mostrar un mensaje al realizar la reserva

Una vez realicemos la reserva , queremos que nos salga un mensaje para ver que la reserva ha sido realizada correctamente . Para ello realizaremos un **echo** . Este mensaje es cuando la reserva se realice con exito

```
echo '<div class="reserva-mensaje reserva-exito"> ¡Reserva Realizada! El administrador ha sido notificado</div>';
```

Mostrar un mensaje de error al realizar una reserva

La reserva al realizarla puede dar error ya que la hora seleccionada ya ha sido adjudicada a otra persona para ello realizamos otro mensaje cuando esto suceda .

```
echo '<div class="reserva-mensaje reserva-error"> La reserva con la hora seleccionada no esta disponible</div>';
```

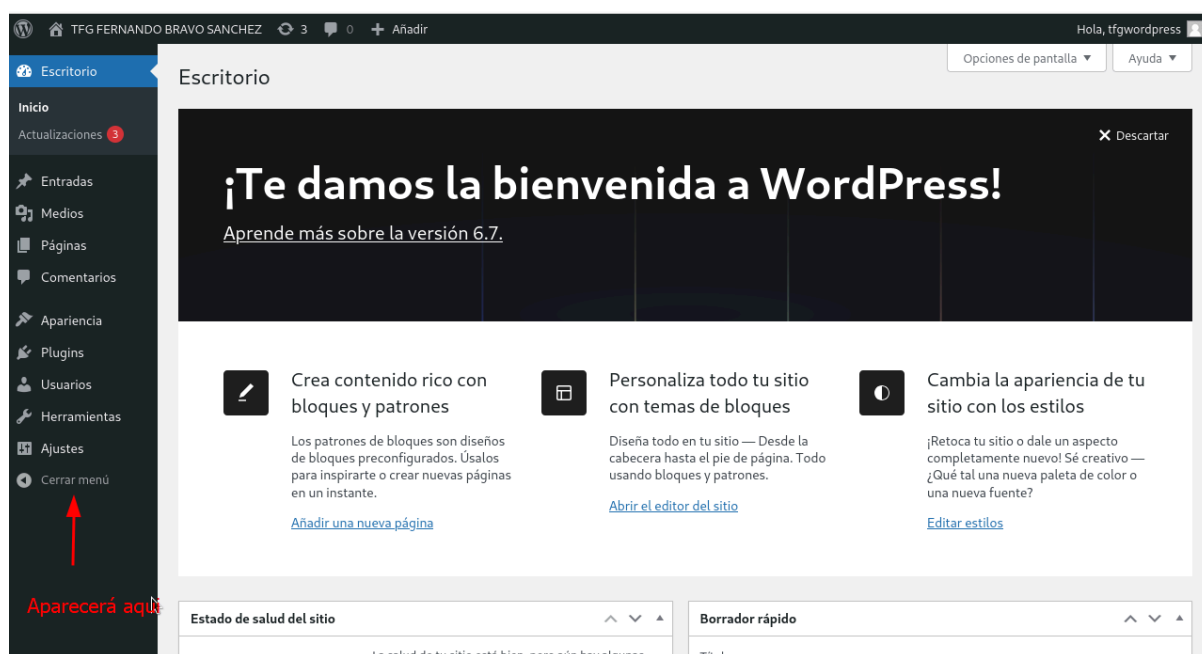
Carga del procesamiento de datos con INIT

Para ello la función creada en este caso **rp_procesar_formulario_reserva()** lo activamos con el hook **init** , lo que hacemos con esto integramos el procesamiento de datos esté bien integrado con Wordpress . Si el formulario se envía , la función **rp_procesar_formulario_reserva** procesa los datos , si no se envía no hará nada , ya que no tiene datos para procesar.

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

Creación menú en el panel de administración de Wordpress

Crearemos un menú en el panel de administración de Wordpress el cual aparecerá en la barra lateral .



Al añadirlo aquí será una vista cómoda para que el administrador de la página vea una lista de reservas que han sido realizadas .

Agregar un Menú en el panel de administración de wordpress

Usaremos el hook **admin_menu** con la función creada para el menú del panel de administración **rp_agregar_menu_reservas** .

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

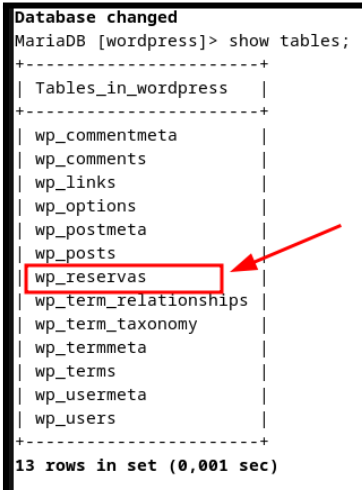
Para ello crearemos la función mencionada anteriormente (**rp_agregar_menu_reservas**) también utilizaremos la función ya incorporada en wordpress llamada [add_menu_page](#) . Aquí añadiremos el título de la página , el nombre que aparecerá en el menú del panel , capacidad para acceder al menú , slug que identifica al menú , Función que se ejecuta para mostrar el contenido de la página , el icono y la posición

```
function rp_agregar_menu_reservas() {  
    add_menu_page(  
        'Gestión de Reservas', //Título de la página  
        'Reservas FCT',        //Nombres del menú  
        'manage_options',      // Capacidad para acceder al menú  
        'reservas-plugin',      // Slug que identifica al menú  
        'rp_mostrar_reservas',  // Funcion para mostrar el contenido de la página  
        'dashicons-calendar-alt', // Icono  
        6                       // Posición  
    );  
}
```

Mostrar las reservas en el panel de administración

Para ello vamos a crear una función la cual consta de realizar un select a la base de datos en la cual se guardan las reservas realizadas. Para esto utilizamos la variable [global \\$wpdb](#) . Declaramos la variable **\$tabla_reservas** para que guarde la tabla de la base de datos **wp_reservas** . Para ello utilizamos la variable **\$wpdb→ prefix** la cual nos da el prefijo de la tabla la cual es **wp** , luego llamamos . **'reservas'** ya que es como se llama la tabla . Luego declaramos una variable llamada **\$reservas** la cual se encarga de realizar un select a toda la tabla . Dado esto , luego realizamos una tabla en html para poder ver desde el panel de administración las reservas. Para verificar si no está vacía la tabla de reservas , realizaré un **bucle foreach** . Para ello empezamos con un **if(!empty(\$reservas))** para decir que si no está vacía la tabla de la base de datos realice el bucle foreach . Si esta está vacía que muestre un texto en pantalla (No hay reservas registradas) . Ahora en caso de que haya datos en la tabla realizamos un bucle foreach dentro de la tabla , lo mostramos en una fila con **<tr>** y en cada celda con **<td>** . Utilizamos la función incorporada de Wordpress llamada [esc_html](#) , la cual evita problemas de seguridad .

```
Database changed
MariaDB [wordpress]> show tables;
+-----+
| Tables_in_wordpress |
+-----+
| wp_commentmeta      |
| wp_comments         |
| wp_links            |
| wp_options          |
| wp_postmeta         |
| wp_posts            |
| wp_reservas         |
| wp_term_relationships |
| wp_term_taxonomy    |
| wp_termmeta         |
| wp_terms            |
| wp_usermeta         |
| wp_users            |
+-----+
13 rows in set (0,001 sec)
```



```

//Mostrar las reservas en el panel de administraci3n
function rp_mostrar_reservas() {
    global $wpdb;
    $tabla_reservas = $wpdb->prefix . 'reservas';
    $reservas = $wpdb->get_results("SELECT * FROM $tabla_reservas");

    echo '<div class="wrap">';
    echo '<h1> Reservas Realizadas</h1>';
    echo '<table class="wp-list-table widefat fixed striped">';
    echo '<thead><tr><th>ID</th><th>Nombre</th><th>Email</th><th>Fecha</th><th>Hora</th><th>Acciones</th></tr></thead>';
    echo '<tbody>';

    if (!empty($reservas)){
        foreach ($reservas as $reserva){
            echo '<tr>';
            echo '<td>' . esc_html($reserva->id) . '</td>';
            echo '<td>' . esc_html($reserva->nombre) . '</td>';
            echo '<td>' . esc_html($reserva->email) . '</td>';
            echo '<td>' . esc_html($reserva->fecha) . '</td>';
            echo '<td>' . esc_html($reserva->hora) . '</td>';
            // echo '<td><a href=?'
            echo '</tr>';
        }
    } else {
        echo '<tr><td colspan="6">No hay reservas registradas.</td></tr>';
    }

    echo '</tbody>';
    echo '</table>';
    echo '</div>';
}

?>

```

Vista Previa del plugin básico funcionando

Ya con todo este código explicado ya tendríamos el plugin funcionando , ahora lo que haremos será añadir cosas adicionales lo cual mejorará la interacción con la persona que utilice este plugin . Las cosas adicionales las iré explicando como va quedando de un antes y después , lo que voy a hacer es mostrar una vista previa de como se ve el plugin funcionando actualmente

Formulario desde la página de wordpress

Antes de todo para poder inyectar el formulario en la página de wordpress introducimos el shortcode declarado en el código anteriormente .

```
add_shortcode('reserva_formulario' , 'rp_mostrar_formulario_reservas');
```

Creamos una nueva página en wordpress y le introducimos el shortcode tal que así



Así es como se ve el formulario el cual vé el usuario desde la página de wordpress .

Reservas Fernando FCT

Nombre :

Email :

Fecha :

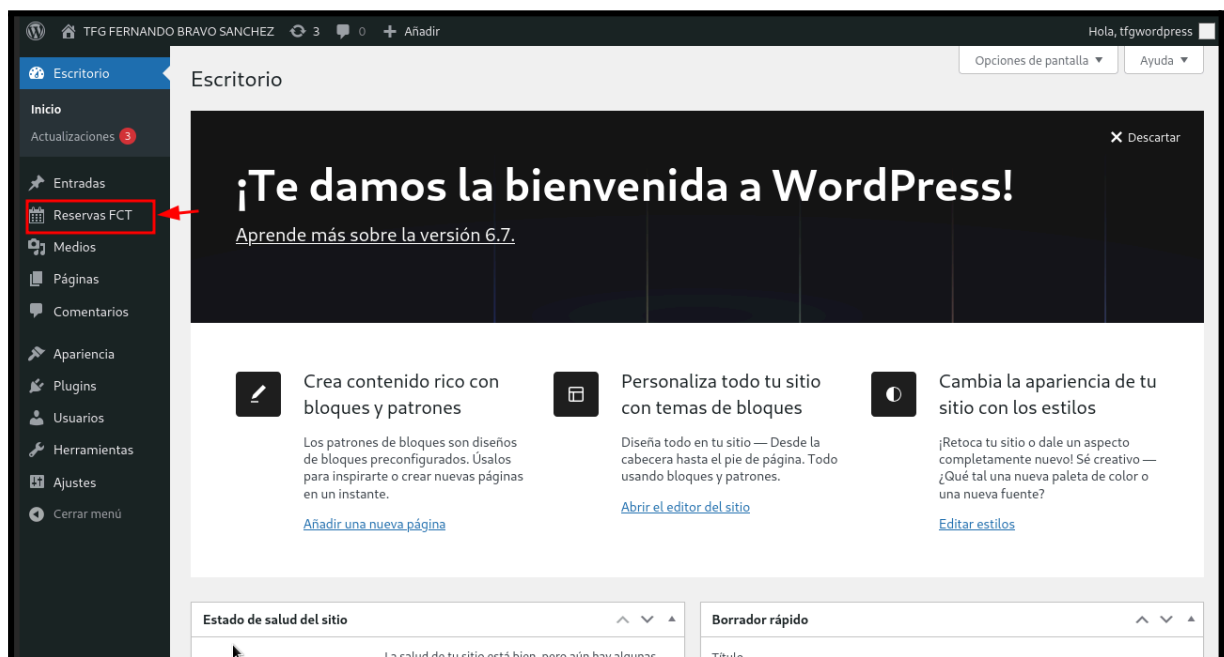
dd / mm / aaaa

Hora : -- : --

Reservar

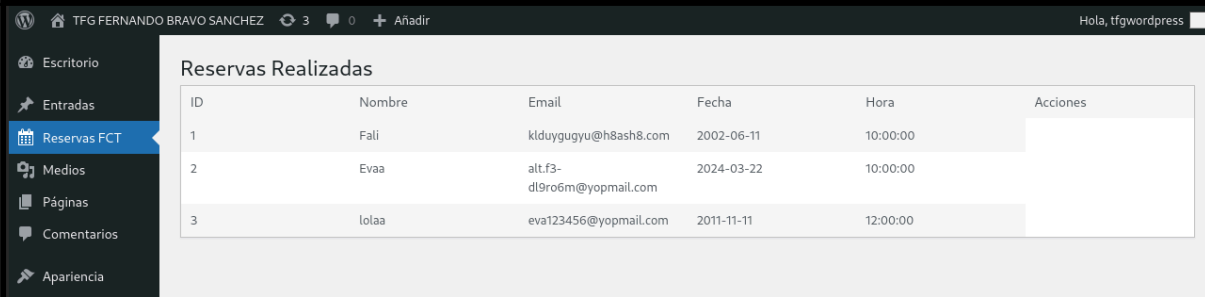
Menú desde panel de administración

Así es como se ve el menú desde el panel de administración el cual solo puede acceder el administrador



Vista de las reservas realizadas desde el administrador

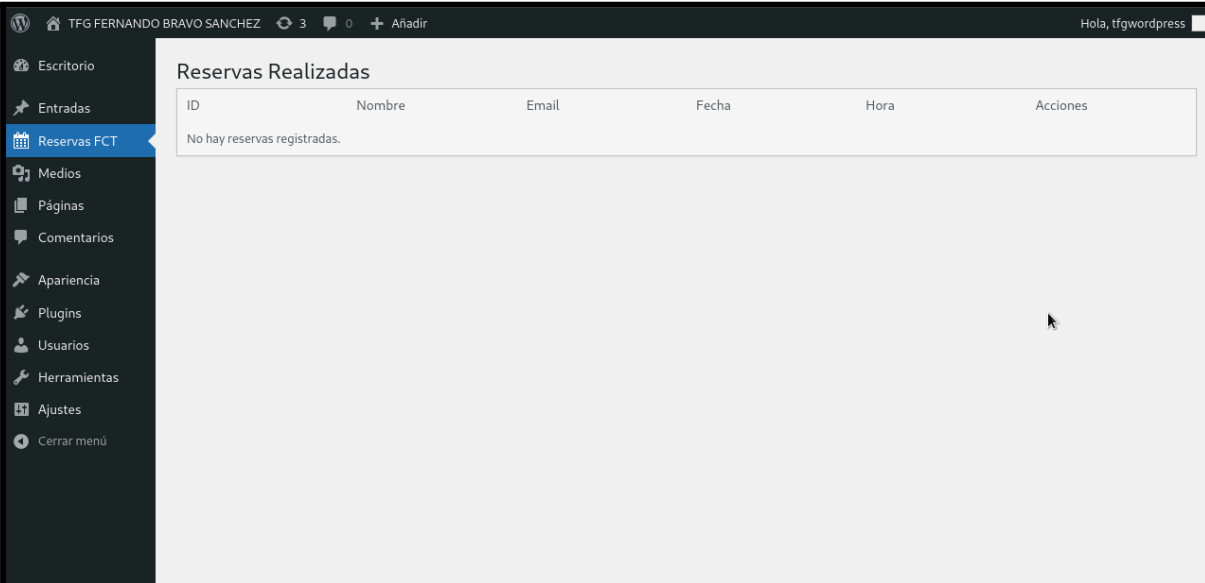
Para ello realizamos las reservas con diferentes nombres para ver como se ve desde la vista del administrador . Así es como se ve teniendo en cuenta que se han realizado las reservas correctamente



The screenshot shows the WordPress admin interface. The left sidebar has a menu with 'Reservas FCT' highlighted. The main content area is titled 'Reservas Realizadas' and contains a table with the following data:

ID	Nombre	Email	Fecha	Hora	Acciones
1	Fali	klduygugyu@h8ash8.com	2002-06-11	10:00:00	
2	Evaa	alt.f3-dl9ro6m@yopmail.com	2024-03-22	10:00:00	
3	lolaa	eva123456@yopmail.com	2011-11-11	12:00:00	

Así es como se ve , teniendo en cuenta que no se ha realizado ninguna reserva .



The screenshot shows the same WordPress admin interface, but the 'Reservas Realizadas' table is now empty. The message 'No hay reservas registradas.' is displayed at the top of the table area.

ID	Nombre	Email	Fecha	Hora	Acciones
No hay reservas registradas.					

Opciones adicionales añadidas

Plugin SMTP de Wordpress para enviar correo

Para esto cuando enviamos la reserva el correo no se envía ya que necesitamos un servidor smtp . Utilizamos un plugin de Wordpress llamado **WP Mail SMTP**. Con esto podemos ver que se intenta enviar el correo



Al realizar una reserva nos lanzará este mensaje dentro del plugin **WP Mail SMTP**. Esto significa que funciona pero no es posible enviarlo ya que necesitamos un servidor SMTP .

```
¡Atención! El último correo electrónico que trató de enviar tu sitio no se envió correctamente.

Origen del correo electrónico: WP Mail SMTP
Mailer: Google / Gmail
{
  "error": {
    "code": 401,
    "message": "Request is missing required authentication credential. Expected OAuth 2 access token, login cookie or other valid authentication credential. See https://developers.google.com/identity/sign-in/web/devconsole-project.",
    "errors": [
      {
        "message": "Login Required.",
        "domain": "global",
        "reason": "required",
        "location": "Authorization",
        "locationType": "header"
      }
    ],
    "status": "UNAUTHENTICATED",
    "details": [
      {
        "@type": "type.googleapis.com/google.rpc.ErrorInfo",
        "reason": "CREDENTIALS_MISSING",
        "domain": "googleapis.com",
        "metadata": {
          "method": "caribou.api.proto.MailboxService.SendMessage",
          "service": "gmail.googleapis.com"
        }
      }
    ]
  }
}

For more details please try running an Email Test or reading the latest error event.
```



Formulario horas y fecha posterior a 7 dias

Funcion rp_mostrar_formulario_reservas

Ya que está enfocado a la reserva de un restaurante , voy a crear dos variables las cuales nos dirán la fecha de hoy y la fecha 7 días más tarde , para realizar un rango de reservas en esos 7 días .

Creamos la variable [\\$fecha_actual](#) y [\\$fecha_maxima](#)

```
global $wpdb; // Acceder base de datos
$fecha_actual = date('Y-m-d'); // Fecha de hoy
$fecha_maxima = date('Y-m-d', strtotime("+ 1 week")); // fecha de 7 días posteriores
```

Luego de esto creamos un array con las horas que están disponibles para realizar la reserva

```
$intervalos_hora = [
    '13:00' , '13:15' , '13:30' , '13:45' , '14:00' , '14:15' , '14:30' , '14:45' , '15:00' , '15:15' , '15:30' ,
    '20:00' , '20:15' , '20:30' , '20:45' , '21:00' , '21:15' , '21:30' , '22:00' , '22:15' , '22:30'];
```

Ahora creamos la variable [\\$fecha_adjudicada](#) la cual se encarga de comprobar si la fecha que ha introducido el usuario esta presente , si esta presente se enviara como parte de la solicitud POST , si el usuario no ha elegido ninguna fecha le seleccionamos automáticamente la fecha actual con la variable creada anteriormente .

```
// Obtenemos la fecha seleccionada por el usuario o si no elige le ponemos la fecha actual
$fecha_adjudicada = isset($_POST['fecha']) ? sanitize_text_field($_POST['fecha']): $fecha_actual;
// Comprobamos las horas en la base de datos
$tabla_reservas = $wpdb->prefix . 'reservas';
$horas_reservadas = $wpdb->get_col($wpdb->prepare("SELECT hora FROM $tabla_reservas WHERE fecha = %s" , $fecha_adjudicada));
```

Ahora llamamos a la base de datos wp_reservas y realizamos una [consulta](#) la cual nos da toda la columna de horas .

Ahora en el formulario realizamos un [select](#) para crear un menú desplegable , tras esto abrimos las etiquetas para poder escribir en php y realizamos un foreach que recorre la lista creada anteriormente dentro de este foreach realizamos un echo \$hora para que nos muestres todas las horas escritas en la lista . También hacemos un if para que en caso de que no haya ninguna hora reservada lo omitimos . Acabamos los bucles if y foreach .También en el apartado value llamamos a la variable \$hora la cual se enviara cuando el formulario se envíe .

```
<label for="hora">Hora : </label>
<select id="hora" name="hora" required>
    <option value="" disabled selected> Selecciona una hora </option>
    <?php foreach ($intervalos_hora as $hora): ?>
        <?php if (!in_array($hora, $horas_reservadas)): ?>
            <option value="<?php echo $hora; ?>"><?php echo $hora;?></option>
        <?php endif; ?>
    <?php endforeach; ?>
</select>
```

Funcion rp_procesar_formulario_reserva

Vamos a verificar si la hora está reservada , para ello accedemos a la base de datos con la función [\\$wpdb](#) , creamos una consulta la cual cuenta cuantas veces esta la misma hora en esa fecha , en caso de que haya 1 nos saldrá un mensaje de que la reserva con la hora seleccionada no está disponible , en caso de que no haya 0 la reserva se realizará

```
//verificamos si la hora ya esta reservada

global $wpdb;
$tabla_reservas = $wpdb->prefix . 'reservas';
$reserva_existente = $wpdb->get_var($wpdb->prepare("SELECT COUNT(*) FROM $tabla_reservas WHERE fecha = %s AND hora = %s", $fecha , $hora));

if ($reserva_existente > 0) {
    echo '<p> La reserva con la hora seleccionada no esta disponible</p>';
    return;
}
```

Visualizar CSV en el panel de administración

Creacion del submenú

Para ello vamos a crear un [submenú](#) en el panel de administración .

'Reservas-plugin' -> Menú principal en el que se añade este submenú es decir se añade en el apartado de RESERVAS-FCT

'Exportar reservas a CSV' -> Es el título que se muestra en el navegador

'Exportar CSV' -> Título del submenú que se añade en el apartado de RESERVAS-FCT .

'manage_options' -> Solo accede el administrador

'exportar-csv' -> Slug que se utiliza como parte de la URL

'rp_exportar_csv' -> Función que se ejecuta para mostrar el contenido del submenu

```
//submenu
add_submenu_page(
    'reservas-plugin' ,
    'Exportar reservas a CSV' ,
    'Exportar CSV',
    'manage_options',
    'exportar-csv',
    'rp_exportar_csv'
);
```

Creación de la función

Primero verificamos los permisos del usuario si este no es administrador se le denegará el acceso a este submenú , esto lo conseguimos con [current_user_can](#) le añadimos la exclamación delante para decir que si no es el administrador que salga el mensaje 'No tienes permisos para acceder a esta página'.

```
//funcion csv

function rp_exportar_csv(){
    // Verificar permisos
    if(!current_user_can('manage_options')){
        wp_die('No tienes permisos para acceder a esta pagina.');
```

Comprobamos si el usuario ha solicitado la descarga con **isset** y **verificamos** si el índice 'exportar_csv' existe en el array [\\$_GET](#) . En caso de ser verdadero mostrará el csv en caso de ser falso nos saldrá el mensaje No hay reservas para exportar. Accedemos a la base de datos con \$wpdb y llamamos como siempre a la tabla .

```
if (isset($_GET['exportar_csv'])){
    global $wpdb;
    $tabla_reservas = $wpdb->prefix . 'reservas';
```

Realizamos una consulta la cual obtendremos todas las reservas y la mostraremos en array

```
$reservas = $wpdb->get_results("SELECT * FROM $tabla_reservas", ARRAY_A);
```

Ahora creamos un if para que si la tabla de reservas de la base de datos no está vacía , utilizamos [header](#) para que nos intente descargar el archivo csv con el nombre **reservas.csv** , forzando la descarga con **attachment**

```
if(!empty($reservas)) {  
    header('Content-Type: text/csv; charset=utf-8');  
    header('Content-Disposition: attachment; filename="reservas.csv"');
```

Abrimos el archivo con fopen

```
$salida = fopen('php://output', 'wb');
```

Agregamos los encabezados del csv con fputcsv , utilizamos la variable \$salida para abrir el archivo y añadimos un array con los nombres de los encabezados

```
//Agregar los encabezados al csv  
  
fputcsv($salida,['ID', 'Nombre', 'Email', 'Fecha', 'Hora']);
```

Escribimos las reservas en cada fila , para no hacerlo uno a uno y que se vaya actualizando cuando hagamos más reservas , utilizamos un bucle para ello foreach , dentro de este bucle utilizamos de nuevo fputcsv para añadir las reservas

```
//escribir la reservva en cada fila  
foreach ($reservas as $reserva) {  
    fputcsv($salida, $reserva);  
}
```


Cerramos el archivo csv con `fclose` y `exit` para terminar el script en este caso la escritura del csv para que después no interfiera en la escritura del archivo . Realizamos un `else` para que en caso de que este vacío la tabla reservas , nos muestre un mensaje de No hay reservas para exportar.

```
//Cerramos
fclose($salida);
exit;
} else {
    echo '<p> No hay reservas para exportar.</p>';
}
```

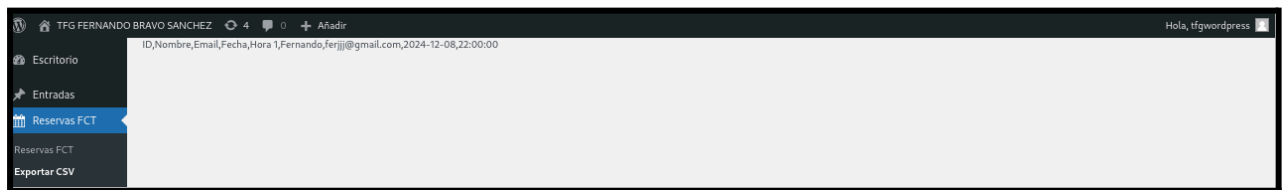
Ahora para la interfaz del administrador en el panel de administración vamos a crear un botón de exportar para que sea visible y pueda ver el csv . Para crearlo utilizamos un href el cual es el enlace que direcciona para poder ver el csv , también utilizamos la class [button-primary](#) para mostrar el botón .

localhost/wordpress/wp-admin/admin.php?page=exportar-csv&exportar_csv=1

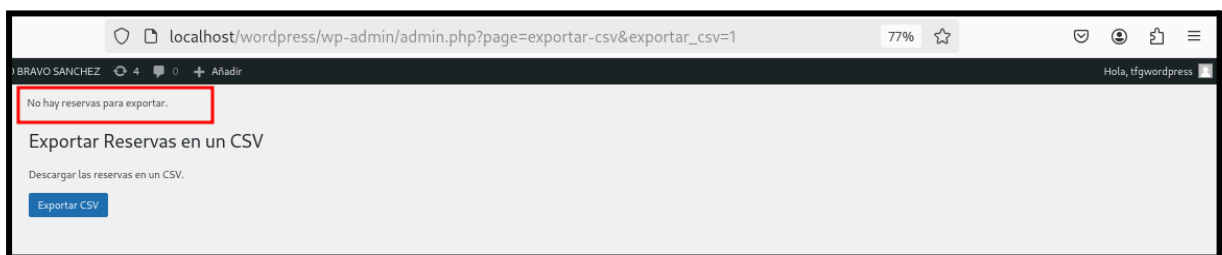
```
//Boton de exportar csv

echo '<div class="wrap">';
echo '<h1> Exportar Reservas en un CSV</h1>';
echo '<p> Descargar las reservas en un CSV.</p>';
echo '<a href="?page=exportar-csv&exportar_csv=1" class="button-primary">Exportar CSV</a>';
echo '</div>';
}
```

Una vez realizado todo esto nos muestra el csv en modo array en caso de haber reservas en la tabla de la base de datos .



En caso de no haber realizado ninguna reserva a la hora de exportar el csv nos saldrá el mensaje No hay reservas para exportar.



Dar estilo al formulario

Para ello no vamos a la función `mostrar_formulario` y le añadimos clases con `css` guiándonos por la página [w3school](https://www.w3school.com/).

Primero creamos el contenedor el cual lo centramos y lo fijamos

```
<style>
.reserva-contenedor {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 80vh;
  background-color: #fff
}
```

Creamos un estilo al formulario dándole a este un borde para que sea más inmersivo

```
.reserva-formulario{
  background-color: #fff;
  padding: 30px;
  border-radius: 8px;
  max-width: 400px;
  width: 100%;
}
```

Creamos un estilo para el `h1` el cual es el título para el formulario

```
}
.reserva-formulario h1 {
  text-align: center;
  margin: 0px 0 12vh;

  color: #333;
}
```

Creamos un estilo al label añadiendo la letra en negrita y que se muestre en bloque , dejando un margin top y color negro .

```
.reserva-formulario label {  
  display: block;  
  margin-top : 2vh;  
  font-weight: bold;  
  color:#000000;  
}
```

Creamos un estilo a la casilla donde el usuario ingresa los datos , añadiendo un borde

```
.reserva-formulario input[type="text"],  
.reserva-formulario input[type="email"],  
.reserva-formulario input[type="date"],  
.reserva-formulario input[type="time"] {  
  width: 100%;  
  padding: 10px;  
  margin-bottom: 30 px;  
  border: 1px solid #ccc;  
  border-radius: 5px;  
}
```

Creamos un estilo al botón de enviar para que cuando el usuario posicione el ratón encima del botón sea de color azul , para esto utilizamos [:hover](#)

```
.reserva-formulario input[type="submit"]:hover {  
  background-color: #005983;  
}
```

Creamos también un estilo para los mensajes de reserva con éxito y error . La posición la fijamos , le damos un borde , le damos [sombra](#) , lo alineamos en el centro etc..

```
}  
.reserva-mensaje {  
  position: fixed;  
  top: 50%;  
  left: 50%;  
  transform: translate(-50%, -50%);  
  z-index: 9999;  
  padding: 20px 40px;  
  border-radius: 8px;  
  text-align: center;  
  font-size: 18px;  
  font-weight: bold;  
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);  
  color: white;  
}
```

Ahora cuando la reserva sea confirmada le damos un color verde de fondo

```
.reserva-exito {  
  background-color: green;  
}
```

Cuando la reserva muestra un error le damos el color rojo

```
.reserva-error {  
  background-color: red;  
}
```

Creamos un estilo al footer el cual se muestra debajo del formulario

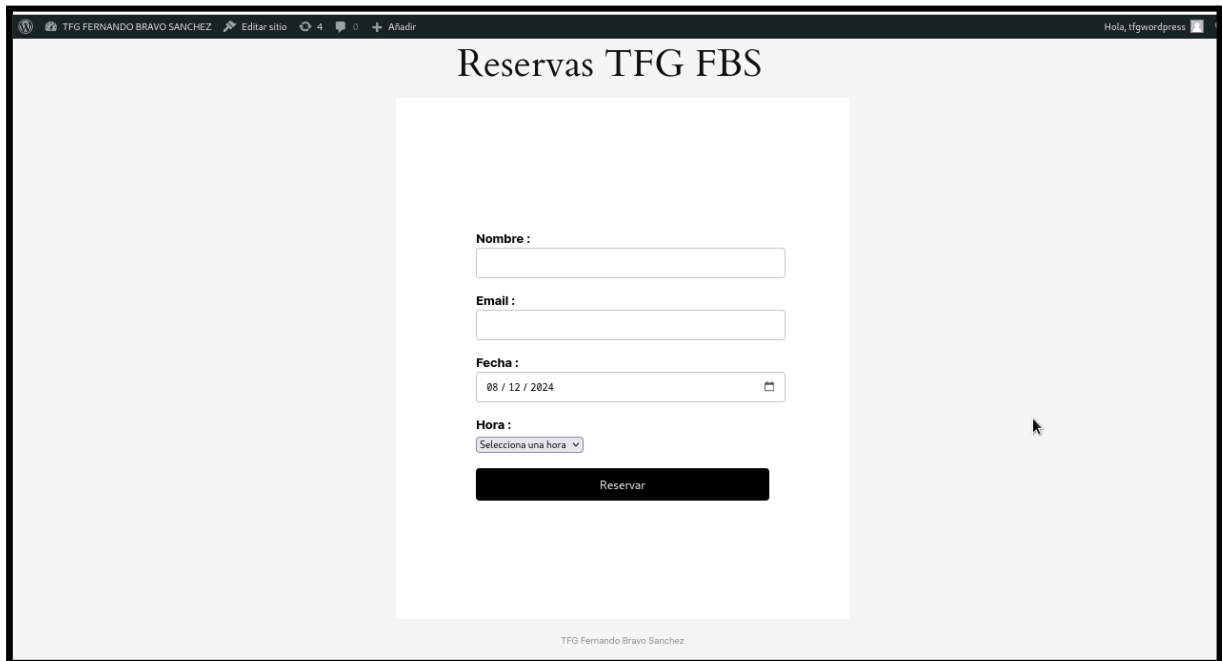
```
.footer-text {  
    text-align:center;  
    margin-top:20px;  
    font-size:12px;  
    color: #888;  
}  
</style>
```

El css ya estaría listo , ahora lo añadimos al formulario con el [atributo class](#) es decir creamos un div con la clase reserva formulario

```
<div class="reserva-contenedor">  
  <form method="post" action="" class="reserva-formulario">  
    <label for="nombre">Nombre : </label>  
    <input type="text" id="nombre" name="nombre" required>  
  
    <label for="email">Email : </label>  
    <input type="email" id="email" name="email" required>  
    <br>  
    <label for="fecha">Fecha : </label>  
    <input type="date" id="fecha" name="fecha" min="<?php echo $fecha_actual; ?>" max="<?php echo $fecha_maxima; ?>">  
  
    <label for="hora">Hora : </label>
```

Vista del formulario con CSS

Una vez creado los estilos css y añadidos al html se ve tal que así

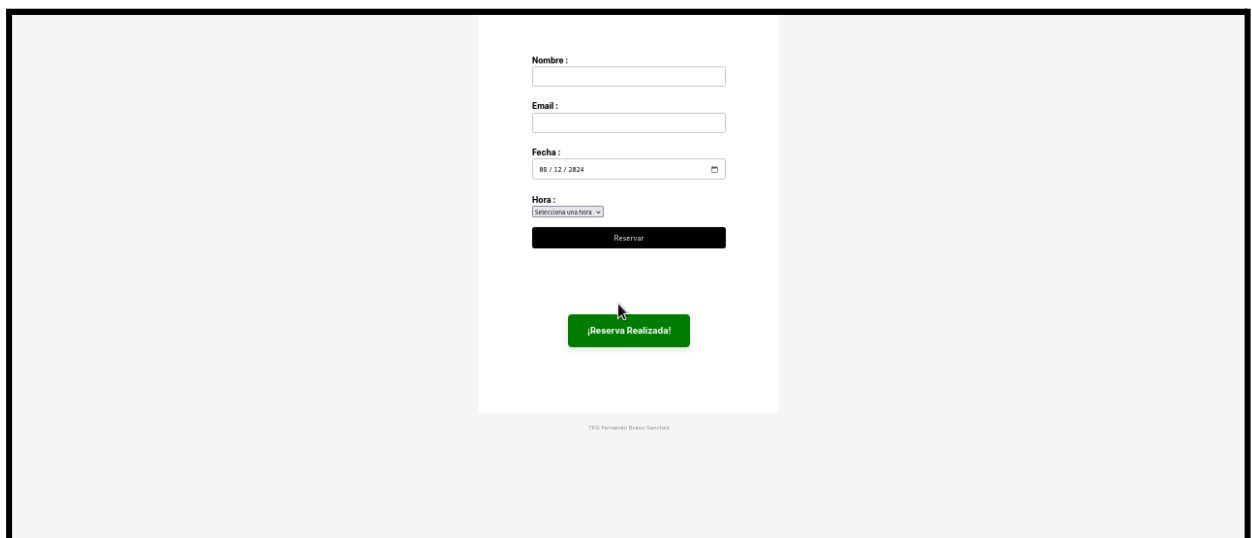


The screenshot shows a web browser window with the title 'Reservas TFG FBS'. The browser's address bar shows 'Hola, tfgwordpress'. The form is centered on a light gray background. It contains the following fields:

- Nombre :** A text input field.
- Email :** A text input field.
- Fecha :** A date picker showing '08 / 12 / 2024'.
- Hora :** A dropdown menu with the text 'Selecciona una hora'.
- Reservar** : A black button with white text.

At the bottom of the page, there is a small footer that reads 'TFG Fernando Bravo Sanchez'.

Cuando realizamos una reserva saldrá el mensaje en verde



This screenshot shows the same reservation form as the previous one, but with a green success message displayed below the 'Reservar' button. The message is '¡Reserva Realizada!' in white text on a green background. The form fields and the 'Reservar' button remain visible above the message.

Cuando la reserva da un error nos saldrá el botón en rojo

The screenshot shows a web browser window with the address bar displaying 'localhost/wordpress/'. The page content includes a reservation form with the following fields:

- Nombre :
- Email :
- Fecha :
- Hora :

Below the form is a black button labeled 'Reservar'. A red error message box is displayed below the button, containing the text: 'La reserva con la hora seleccionada no esta disponible'.

The footer of the page contains the text 'TFG FERNANDO BRAVO SANCHEZ' and links for 'Acerca de', 'Privacidad', and 'Social'.

Activación Plugin y Desactivación Plugin

Creamos la función activar reservas el cual se ejecuta una vez el plugin se active .Utilizamos \$wpdb para acceder a la base de datos , llamamos a la tabla wp_reservas ya que utilizamos el prefijo de wordpress(wp) con \$wpdb->prefix . Luego utilizamos [get_charset_collate](#) para garantizarnos que la tabla que se va a crear utilice la misma configuración que hay en las demás tablas de la base de datos de Wordpress , luego lanzamos una consulta para crear la tabla , en este caso utilizo id , nombre , email , fecha y hora , añadiendo a cada una que sea not null para que no este vacia . Añadimos el archivo upgrade.php en el cual se encuentra la función [dbdelta](#) la cual gestiona la creacion y actualizacion de las tablas ,

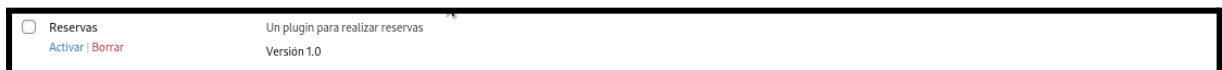
```
function rp_activar_reservas_plugin(){
    //Crearemos las tablas para guardar y gestionara las reservas
    global $wpdb;
    $tabla_reservas = $wpdb->prefix . 'reservas';
    $charset_collate = $wpdb->get_charset_collate();

    $sql = "CREATE TABLE $tabla_reservas (
        id mediumint(9) NOT NULL AUTO_INCREMENT ,
        nombre tinytext NOT NULL ,
        email varchar(100) NOT NULL ,
        fecha date NOT NULL ,
        hora time NOT NULL ,
        PRIMARY KEY (id)
    ) $charset_collate;";

    require_once(ABSPATH . 'wp-admin/includes/upgrade.php');
    dbDelta($sql);

    register_activation_hook(__FILE__, 'rp_activar_reservas_plugin');
```

Os muestro el ejemplo , una vez activamos el plugin se creará la tabla, ahora mismo el plugin no esta activado por lo cual la tabla wp_reservas no existe en la base de datos.



Como vemos no se encuentra en la base de datos wordpress en la cual se guarda la tablas que vamos a crear

```

+-----+
| Tables_in_wordpress |
+-----+
| wp_actionscheduler_actions |
| wp_actionscheduler_claims |
| wp_actionscheduler_groups |
| wp_actionscheduler_logs |
| wp_commentmeta |
| wp_comments |
| wp_links |
| wp_options |
| wp_postmeta |
| wp_posts |
| wp_term_relationships |
| wp_term_taxonomy |
| wp_termmeta |
| wp_terms |
| wp_usermeta |
| wp_users |
| wp_wpmailsmtp_debug_events |
| wp_wpmailsmtp_tasks_meta |
+-----+
18 rows in set (0,001 sec)

```

Ahora vamos a activar el plugin

<input type="checkbox"/> Reservas Desactivar	Un plugin para realizar reservas Versión 1.0
--	---

Como vemos la tabla se ha creado automáticamente

```

+-----+
| Tables_in_wordpress |
+-----+
| wp_actionscheduler_actions |
| wp_actionscheduler_claims |
| wp_actionscheduler_groups |
| wp_actionscheduler_logs |
| wp_commentmeta |
| wp_comments |
| wp_links |
| wp_options |
| wp_postmeta |
| wp_posts |
| wp_reservas |
| wp_term_relationships |
| wp_term_taxonomy |
| wp_termmeta |
| wp_terms |
| wp_usermeta |
| wp_users |
| wp_wpmailsmtp_debug_events |
| wp_wpmailsmtp_tasks_meta |
+-----+

```

Cuando desactivemos el plugin he añadido que se borre la tabla de la base de datos .

Accedemos a la base de datos con \$wpdb y llamamos a la tabla , luego lanzamos una consulta la cual ordena que si la tabla existe la elimine . Esto se ejecutará una vez desactivemos el plugin

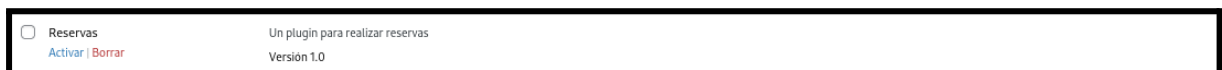
```
//Función al desactivar el plugin
function rp_desactivar_reservas_plugin(){

    global $wpdb;
    $tabla_reservas = $wpdb->prefix . 'reservas';

    // Eliminar tabla
    $wpdb->query("DROP TABLE IF EXISTS $tabla_reservas");

}
register_deactivation_hook(__FILE__, 'rp_desactivar_reservas_plugin');
```

Ahora vamos a desactivar el plugin



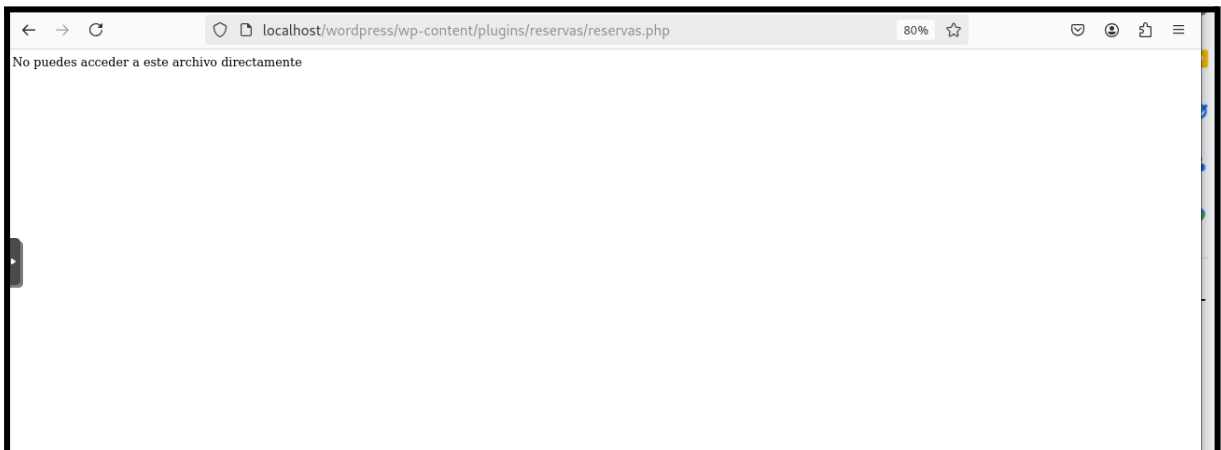
Por lo cual se ha borrado la base de datos

Tables_in_wordpress	
wp_actionscheduler_actions	
wp_actionscheduler_claims	
wp_actionscheduler_groups	
wp_actionscheduler_logs	
wp_commentmeta	
wp_comments	
wp_links	
wp_options	
wp_postmeta	
wp_posts	
wp_term_relationships	
wp_term_taxonomy	
wp_termmeta	
wp_terms	
wp_usermeta	
wp_users	
wp_wpmailsmtp_debug_events	
wp_wpmailsmtp_tasks_meta	

Seguridad Plugin aplicada

Comprobación ABSPATH

Para evitar el acceso directo a un archivo php utilizamos [ABSPATH](#) . Si intentamos acceder a un archivo php directamente desde el navegador web nos sale un mensaje de que no puedes acceder al archivo php



Comprobación Sanitize

Para ello utilizamos este formato `sanitize_text_field` que se encarga de limpiar las etiquetas html , esto lo colocamos en la funcion `procesar_formulario` .

```
$nombre = sanitize_text_field($_POST['nombre']);  
$email = sanitize_email($_POST['email']);  
$fecha = sanitize_text_field($_POST['fecha']);  
$hora = sanitize_text_field($_POST['hora']);
```

También he añadido un [pattern](#) en el nombre formulario , el cual solo deja escribir letras y espacios

```
<input type="text" id="nombre" pattern="[A-Za-záéíouÁÉÍóúñÑ ]+" title=" Solo se permiten nombres" name="nombre" required>
```

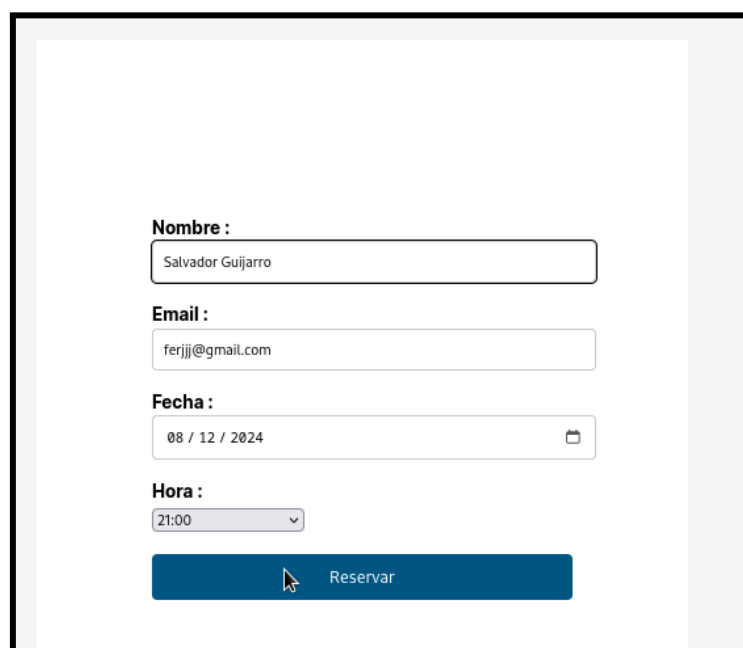
Si escribimos etiquetas html nos saldrá un mensaje de que no es válido



The screenshot shows a reservation form with the following fields and elements:

- Nombre :** A text input field containing the HTML script `<script>alert('hack')</script>`.
- Error Message:** A light gray box with rounded corners displays the message: "Ajústese al formato solicitado: Solo se permiten nombres."
- Fecha :** A date input field showing "08 / 12 / 2024" with a calendar icon on the right.
- Hora :** A dropdown menu showing "21:00" with a downward arrow.
- Reservar:** A blue button with a white mouse cursor icon pointing at it.

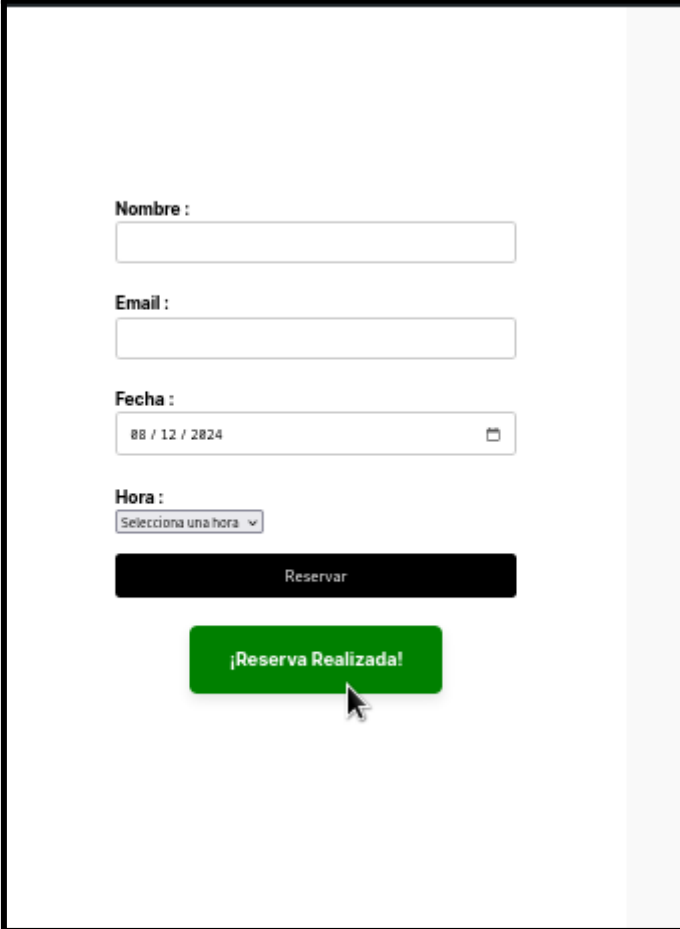
Solo aceptara letras y espacios por ejemplo voy a reservar a nombre de Salvador Guijarro



The screenshot shows the same reservation form, but with valid input:

- Nombre :** The text input field now contains "Salvador Guijarro".
- Email :** A new text input field contains "ferjji@gmail.com".
- Fecha :** The date input field remains "08 / 12 / 2024".
- Hora :** The dropdown menu remains "21:00".
- Reservar:** The blue button remains, with the mouse cursor still pointing at it.

La reserva se ha realizado como podemos ver



Nombre :

Email :

Fecha :

Hora :

Reservar

¡Reserva Realizada!

Lo vemos en el menú de administración

11	Salvador Guajardo	ferj@gmail.com	2024-12-08	21:00:00
----	-------------------	----------------	------------	----------

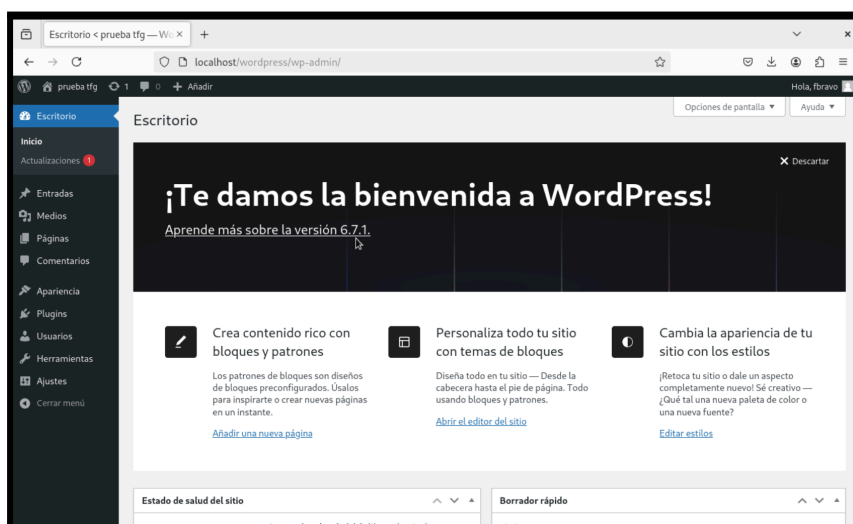
Comprobación y funcionamiento del plugin en otra máquina

Vamos a probar la descarga e instalación del plugin en otra máquina. Para ello vamos a utilizar otra máquina con mariadb y wordpress en proxmox que es lo necesario para que el plugin funcione .

5061 (asir2-fbravo-TFG-Prueba)

Una vez creada la máquina e instalada con wordpress y mariadb vamos a descargarnos el php de mi [github](#) .

Una vez instalado un LAMP y wordpress



```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 51
Server version: 10.11.6-MariaDB-0+deb12u1 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [wordpress]>
```

Para la instalación del plugin es necesario crear una carpeta (llámala como quieras)

```
root@fernandoprueba:/var/www/html# cd wordpress/  
root@fernandoprueba:/var/www/html/wordpress#
```

Creamos la carpeta dentro de la carpeta plugins , yo llamare a la carpeta reservas , importante dar permisos **www-data** a la carpeta , si no nos puede proporcionar un error al crear el log de las reservas .

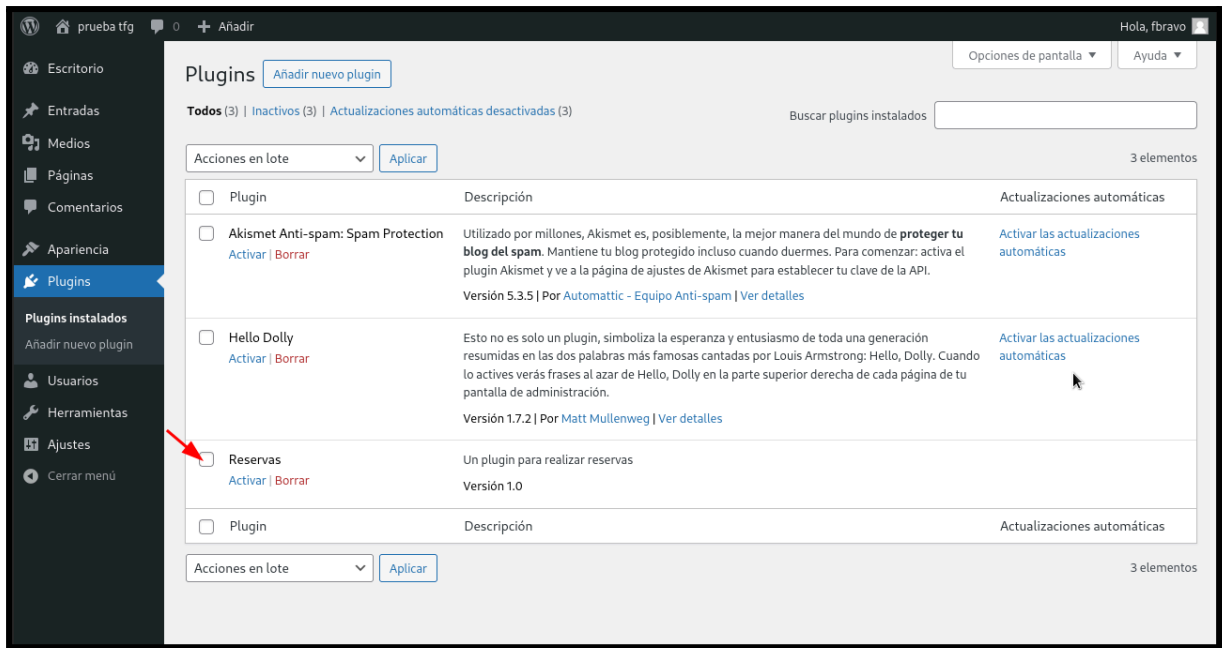
```
root@fernandoprueba:/var/www/html/wordpress/wp-content/plugins#
```

```
root@fernandoprueba:/var/www/html/wordpress/wp-content/plugins# ls  
akismet hello.php index.php reservas  
root@fernandoprueba:/var/www/html/wordpress/wp-content/plugins#
```

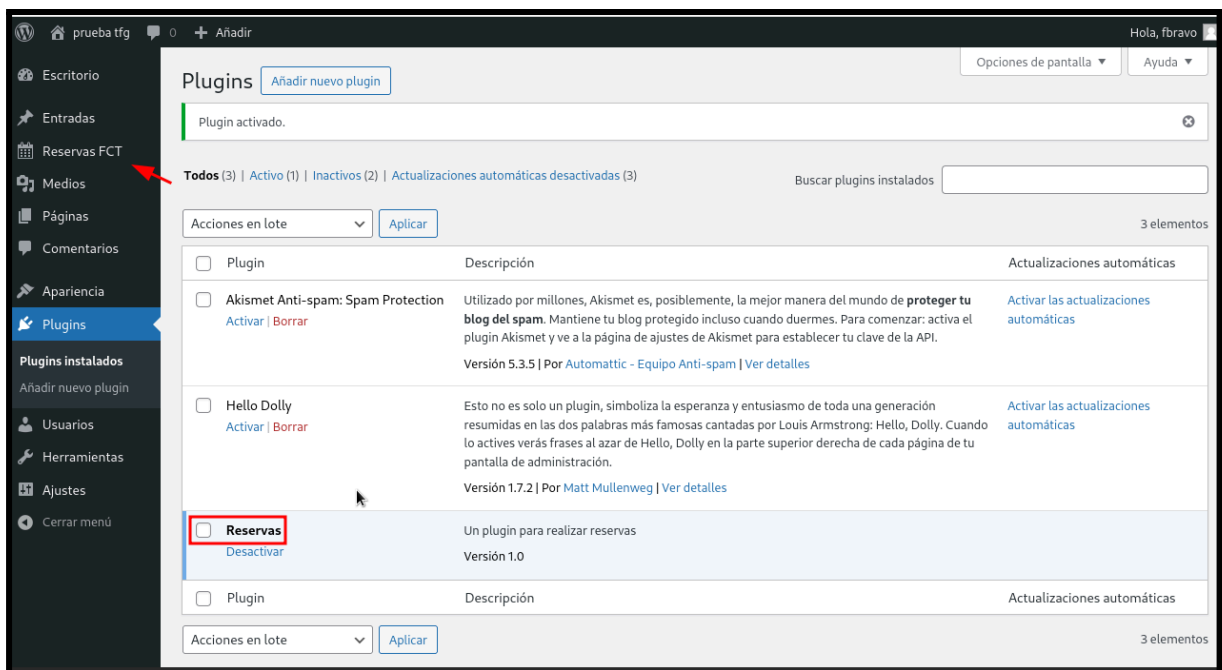
Dentro de esta carpeta almacenamos el php descargado de github ,

```
root@fernandoprueba:/home/fernando/Descargas# mv reservas.php /var/www/html/wordpress/wp-content/plugins/reservas/  
root@fernandoprueba:/home/fernando/Descargas#
```

Nos vamos al menú de administración y al apartado plugins nos encontramos con el plugin de reservas



Vamos a activarlo para comprobar que funciona correctamente



Como vemos se ha creado la base de datos wp reservas

```
Database changed
MariaDB [wordpress]> show tables;
+-----+
| Tables_in_wordpress |
+-----+
| wp_commentmeta      |
| wp_comments         |
| wp_links            |
| wp_options          |
| wp_postmeta         |
| wp_posts            |
| wp_reservas         |
| wp_term_relationships |
| wp_term_taxonomy    |
| wp_termmeta         |
| wp_terms            |
| wp_usermeta         |
| wp_users            |
+-----+
13 rows in set (0,000 sec)
```

Añadimos el shortcode el cual nos muestra el formulario



prueba tfg Editar sitio + Añadir Hola, bravo

Prueba TFG

shutterstock.com - 1254530365

Nombre :

Email :

Fecha :

Hora :

Añadimos algunas reservas para ver si todo está correcto

ID	Nombre	Email	Fecha	Hora
1	Fernando	ferji@gmail.com	2024-12-09	20:15:00
2	Fran	francis@yopmail.com	2024-12-09	22:30:00

Visualizamos el csv

ID,Nombre,Email,Fecha,Hora 1,Fernando,ferji@gmail.com,2024-12-09,20:15:00 2,Fran,francis@yopmail.com,2024-12-09,22:30:00

Vemos el archivo log si se ha creado correctamente

```
reservas-admin.log reservas.php
root@fernandoprueba:/var/www/html/wordpress/wp-content/plugins/reservas# cat reservas-admin.log
Reserva realizada:
-----
Nombre: Fernando
Email: ferjj@gmail.com
Fecha: 2024-12-09
Hora: 22:30
-----
Reserva realizada:
-----
Nombre: enrique garcia
Email: ferjj@gmail.com
Fecha: 2024-12-09
Hora: 13:00
-----
root@fernandoprueba:/var/www/html/wordpress/wp-content/plugins/reservas#
```

Desactivamos el plugin para ver que se ha borrado tanto la base de datos wp reservas como el archivo log



Como vemos se han borrado

```
root@fernandoprueba:/var/www/html/wordpress/wp-content/plugins/reservas# ls  
reservas.php
```

```
Database changed  
MariaDB [wordpress]> show tables;  
+-----+  
| Tables_in_wordpress |  
+-----+  
| wp_commentmeta      |  
| wp_comments         |  
| wp_links            |  
| wp_options          |  
| wp_postmeta         |  
| wp_posts            |  
| wp_term_relationships |  
| wp_term_taxonomy    |  
| wp_termmeta         |  
| wp_terms            |  
| wp_usermeta         |  
| wp_users            |  
+-----+  
12 rows in set (0,001 sec)  
  
MariaDB [wordpress]>
```

Por lo tanto el plugin es totalmente funcional

Bibliografía

Instalación de LAMP y Wordpress

LAMP →  LAMP en Debian 12

Wordpress instalación →  Instalación de Wordpress

Código fuente del proyecto

<https://www.pontikis.net/blog/what-is-abspath-in-wordpress-and-how-to-use-for-security>

https://wp-kama.com/function/register_activation_hook

https://wp-kama.com/function/register_deactivation_hook

<https://stackoverflow.com/questions/38532256/use-of-ob-start-and-ob-get-clean>

https://www.w3schools.com/php/php_superglobals_post.asp

https://www.w3schools.com/php/func_filesystem_unlink.asp

<https://stackoverflow.com/questions/38852041/how-to-append-data-to-file-using-file-put-contents>

<https://stackoverflow.com/questions/20780422/wordpress-get-plugin-directory>

https://developer.wordpress.org/reference/functions/add_menu_page/

<https://andres-dev.com/utilizando-la-clase-wpdb-de-wordpress/>

<https://www.freecodecamp.org/espanol/news/etiqueta-select-de-html-como-hacer-un-menu-desplegable-o-lista-combinada/>

<https://developer.wordpress.org/reference/classes/wpdb/prepare/>

<https://stackoverflow.com/questions/13045279/if-isset-post>

<https://www.php.net/manual/en/function.strtotime.php>

<https://www.php.net/manual/es/function.date.php>

https://developer.wordpress.org/reference/functions/esc_html/

https://www.w3schools.com/bootstrap/bootstrap_buttons.asp

<https://stackoverflow.com/questions/217424/create-a-csv-file-for-a-user-in-php>

<https://www.php.net/manual/es/reserved.variables.get.php>

https://developer.wordpress.org/reference/functions/current_user_can/

https://developer.wordpress.org/reference/functions/add_submenu_page/

<https://developer.mozilla.org/es/docs/Web/CSS/:hover>

https://www.w3schools.com/css/css_intro.asp

<https://developer.mozilla.org/en-US/docs/Web/CSS/box-shadow>

https://www.w3schools.com/html/html_classes.asp

<https://stackoverflow.com/questions/70528728/is-using-wpdb-get-charset-collate-necessary-when-creating-tables-in-wordpress>

<https://stackoverflow.com/questions/9957836/creating-table-in-wordpress-with-dbdelta-function>

<https://stackoverflow.com/questions/56057913/html-input-pattern-for-alphanumeric-underscore-dash-and-dot>

Conclusión

El plugin de reservas es una herramienta práctica y básica para gestionar reservas en wordpress . Mi idea ha sido realizarlo enfocado a un restaurante , El usuario tiene la vista de un formulario simple y limpio , tambien el administrador tiene la vista de ver las reservas realizadas por los usuarios desde su menú del panel de administración , visualizar un csv , y ver un archivo logs con las reservas realizadas ya que he intentado realizar el envío de emails pero no ha sido posible por la falta de un servidor SMTP . También he reforzado la seguridad ante el formulario y accesos a archivos internos de wordpress . Los horarios han sido ajustados a mi idea de la apertura de un restaurante , he evitado la duplicación de reservas a la misma hora . El plugin logra el objetivo de realizar reservas en wordpress por lo cual estoy contento con el resultado.