

AI-4. Diseño aplicación web y API Rest



Despliegue de Aplicaciones

José Ignacio Gutiérrez Cerrato 2-DAW

<https://github.com/duactive/DespliegueAplicaciones.git>

Requerimiento 1

Crear una guía de un site, para desarrollo web.

1.INTRODUCCIÓN

El cliente tiene una empresa de turismo ecuestre, donde realizan rutas a caballo, senderismo, campamentos y terapias con caballos.

El cliente necesita una web para mostrar sus servicios y también se pueda usar como un blog al uso.

La web tendrá una parte como una web para mostrar y contratar sus servicios y otra parte para poder comentar y mostrar las diferentes actividades realizadas por los diferentes grupos o personas.

1.1 CMS

Dado que el cliente quiere compartir contenido como artículos, fotos de las actividades, etc y a la vez quiere que la gente participe en ese contenido se decide implementar wordpress , dado que contempla todas las necesidades del cliente.

El posicionamiento web lo contrata el cliente con una agencia especializada, aunque le dejamos el plugin instalado.

1.2 TECNOLOGÍAS UTILIZADAS

Wordpress es un CMS que está desarrollado con lenguajes front HTML5,CCS3 y Javascript , como lenguaje de back PHP y base de datos MySQL.

El servidor hosting debe soportar MySQL + PHP + APACHE para poder instalar wordpress en dicho servidor, actualmente la mayoría de hosting soportan dicha tecnología.

Para este caso usaremos el hosting de **SiteGround**.

1.3 CONFIGURACIÓN DE CMS

Indicamos los puntos de configuración prioritarios para el site:

Configuración de usuarios

Creamos distintos usuarios con roles diferentes:

- Autores para escribir contenido nuevo que mejore la vida de las personas.
- Editores para corregir todos los errores en los contenidos de los autores.
- Administradores para mantener todo actualizado y funcionando sin problemas
- Colaboradores para ayudar a los editores a editar publicaciones
- Suscriptores que pueden o no habernos pagado dinero, pero al menos se han registrado con nosotros y nos han proporcionado una dirección de correo electrónico. (eso tiene un valor)

Tienda online:

La página web tiene la posibilidad de realizar comprar de las distintas actividades, no todas las actividades son de pago.

Para ello instalamos y configuramos el plugin woocommerce.

Seguridad:

Para el apartado de seguridad instalamos y configuramos el plugin All in one Wp Security , unos de los mejores plugins para gestionar la seguridad del site.

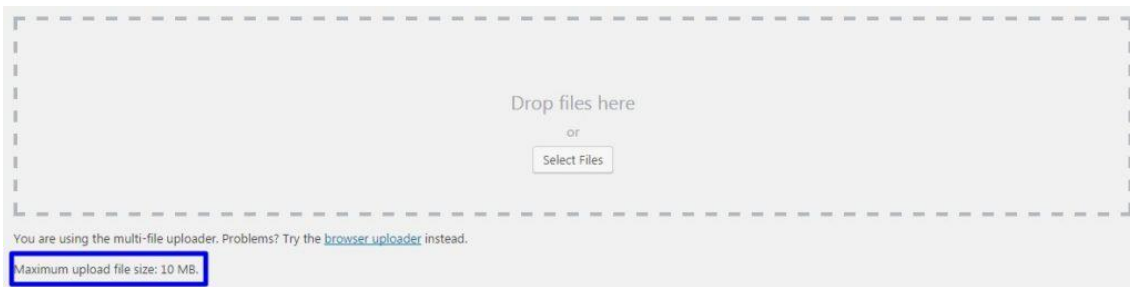
Formularios:

Esta web requiere formularios de contactos específicos por actividad, para ello usamos el plugin Contact form 7 , fácil de configurar y maquetar.

Imágenes:

Al ser una web como mucho contenido multimedia necesitamos comprimir la imágenes y gestionar el lacy load , para ello usamos el plugin Optimole .

Para este caso es necesario aumentar el tamaño máximo de subida en WordPress o en tu servidor (o en ambos).



Opción 1 →Actualizando el archivo .htaccess

Puedes añadir directamente en tu .htaccess nuevas líneas con las directivas que controlan el límite de subida, por ejemplo:

```
Apache
1php_value upload_max_filesize 128M
2php_value post_max_size 256M
3php_value memory_limit 512M
```


Opción 2 →Editando uno de los archivos functions.php o wp-config.php

Añadir en uno de los archivos mencionados el siguiente código:

```
PHP
1@ini_set('upload_max_size','128M');
2@ini_set('post_max_size','256M');
3@ini_set('memory_limit','512M');
```

Posicionamiento SEO

Para el posicionamiento seo instalamos y configuramos el plugin Yoast SEO.



Yoast SEO

Improve your WordPress SEO: Write better content and have a fully optimised WordPress site using the Yoast SEO plugin.

By Team Yoast

[More Details](#)

★★★★★ (27,090)

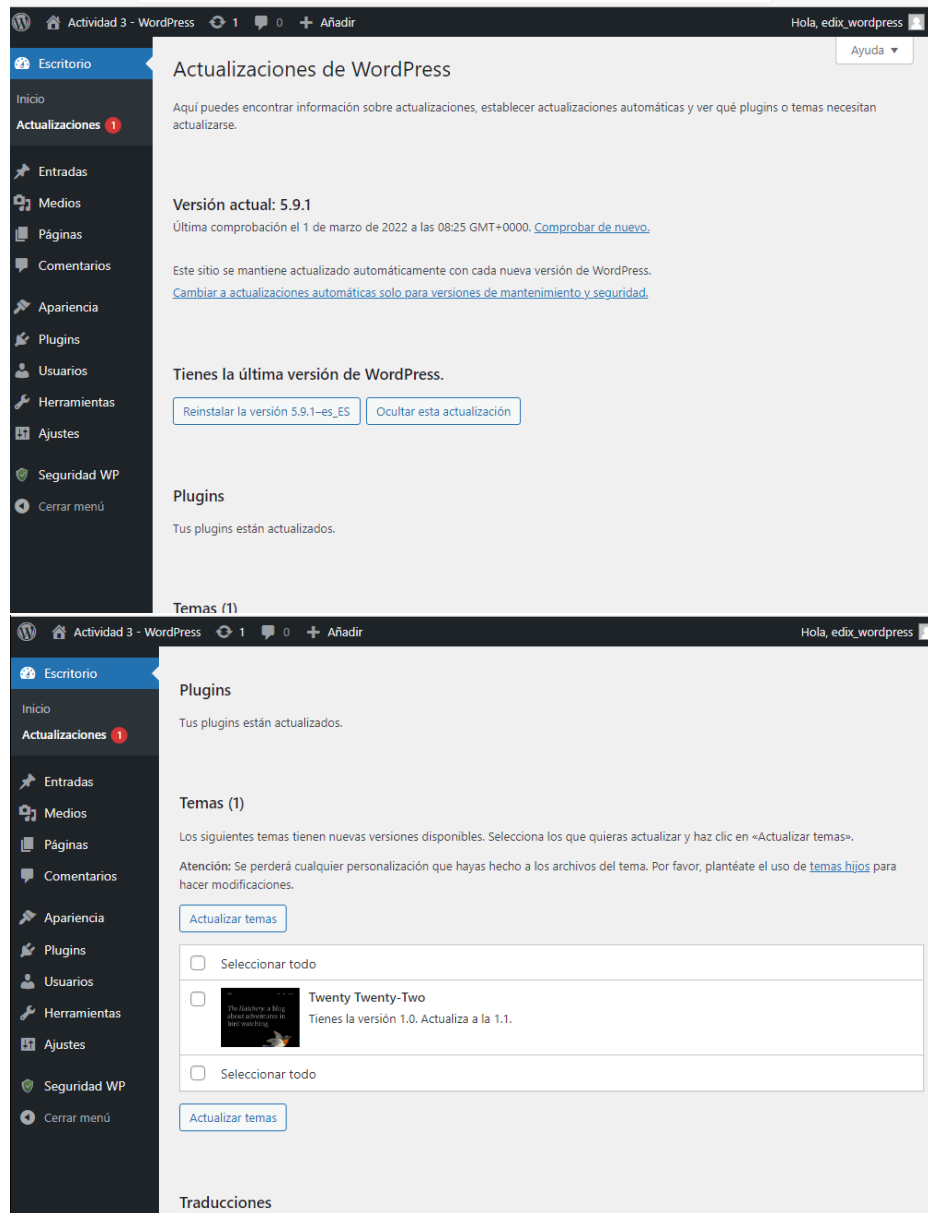
1+ Million Active Installations

Last Updated: 24 hours ago

✖ Incompatible with your version of WordPress

1.4 MANTENIMIENTO DE CMS

CMS, plugins y temas podemos actualizar de manera automática, pero es recomendable gestionar las actualizaciones para confirmar la compatibilidad y el correcto funcionamiento.



Recomendamos actualizar la web y sus componentes mínimo cada 15 días.

La mayoría de las empresas en alojar webs con wordpress te facilitan entorno de pruebas en prácticamente todos los planes de hosting , por ejemplo **SiteGround** en nuestro caso..

2. DISEÑO GRÁFICO

Tenemos dos grandes áreas diferenciadas, por un lado, tenemos la web para realizar la venta y comunicación de las distintas actividades y por otro lado tenemos el lado dedicado al blog.

En el área de Web debemos tener visible en todo momento la opción de compra y acceso al carrito para las distintas actividades publicitadas.

Lo ideal es que sea como tienen Amazon , que es visible en todo momento desde que seleccionas un producto para la cesta.

El proceso de compra tiene que estar disponible en todo momento.

En el área de blog debemos hacer hincapié en la visualización de los artículos, fotos, comentarios, etc. ... debe ser un blog vivo y atractivo para el usuario indicando claramente los temas destacados como por ejemplo las últimas actividades realizadas, campamentos, rutas ...

Tanto para la web como para el blog las diferentes categorías están identificadas perfectamente: campamentos, senderismo, rutas a caballo ,... y desde el propio blog tenemos que tener acceso a la opción de compra de alguna de las actividades así como al carrito.

2.1 LOGOS, LETRAS

El logo nos lo facilita el cliente.

Utilizamos fuentes **WOFF2** , al estar comprimidas mejora la carga del site.

La manera mas sencilla es agrega y edita fuentes a través de temas y plugins.

Desde el propio tema podemos indicar la fuente con la que queremos trabajar o usando el plugin **Fonts Plugin**.

Edit Font

Name

The name of the font as it appears in the customizer options.

Font Fallback

Add the font's fallback names with comma(,) separator. eg. Arial, Serif

Font Display

Font weight

Font .woff2

Upload the font's woff2 file or enter the URL.

Font .woff

Upload the font's woff file or enter the URL.

Font .ttf

Upload the font's ttf file or enter the URL.

Font .eot

Upload the font's eot file or enter the URL.

Font .svg

Upload the font's svg file or enter the URL.

Font .otf

Upload the font's otf file or enter the URL.


2.2 IMÁGENES

Las imágenes nos las proporciona el cliente ya que tiene multitud de imágenes categorizadas por las distintas actividades que realiza.

Como usamos el plugin Optimole , realizamos la compresión de las imágenes para mejorar el rendimiento del site y favorecer el espacio utilizado.

3. POSICIONAMIENTO SEO.

El cliente lo contrata con una agencia especializada en posicionamiento SEO , por nuestra parte dejamos instalado el plugin Yoast SEO.



Yoast SEO

[More Details](#)

Improve your WordPress SEO: Write better content and have a fully optimised WordPress site using the Yoast SEO plugin.

By Team Yoast

★★★★★ (27,090)

Last Updated: 24 hours ago

1+ Million Active Installations

✖ Incompatible with your version of WordPress

Requerimiento 2

Crear una API web.

El cliente necesita una API Rest para la gestión de eventos que luego será explotada por el front-end específico.

Para ello desarrollamos la parte backend con JAVA para realizar el CRUD con la base de datos.

Dicha base de datos está implementada con MySQL8.

Utilizamos una **estructura modelo controlador**, definiendo en la parte modelo las distintas clases por cada una de las tablas de la base de datos y su relación.

En la parte **modelo** indicamos las clases necesarias, por ejemplo “**eventos**”

*****Begin Ejemplo modelo eventos

@Entity

@Table(name="eventos")

@NamedQuery(name="Evento.findAll", query="SELECT e FROM Evento e")

public class Evento implements Serializable {

private static final long serialVersionUID = 1L;

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="ID_EVENTO")

private int idEvento;

@Column(name="AFORO_MAXIMO")

private int aforoMaximo;

private String descripcion;

private String destacado;

private String direccion

private int duracion;

private String estado;

@Temporal(TemporalType.DATE)

@Column(name="FECHA_INICIO")

```

private Date fechaInicio;

@Column(name="MINIMO_ASISTENCIA")

private int minimoAsistencia;

private String nombre;

private BigDecimal precio;

//uni-directional many-to-one association to Tipo

@ManyToOne

@JoinColumn(name="ID_TIPO")

private Tipo tipo;

public Evento() {

}

*****End Ejemplo modelo eventos

```

Para el **controlador**, donde definimos url's que consumirá el front-end en función de la funcionalidad definida.

```

@RestController

@RequestMapping("/rest/eventos")

```

Es decir, <https://xxxx.com/rest/eventos> , será el path principal de nuestra API Rest.
De ese path dependerán las distintas URL's y sus peticiones.

```

@GetMapping("/todos")

Mediante el método GET Obtenemos todos los eventos.

https://xxxx.com/rest/eventos/todos

La respuesta es en formato JSON.

```

```

@GetMapping("/activos")

Mediante el método GET obtenemos los eventos que estén activos en el momento de la petición.

https://xxxx.com/rest/eventos/activos

La respuesta es en formato JSON.

```

@GetMapping("/desactivos")

Mediante el método GET obtenemos los eventos que estén Inactivos en el momento de la petición.

<https://xxxx.com/rest/eventos/desactivos>

La respuesta es en formato JSON.

@GetMapping("/destacados")

Mediante el método GET obtenemos los eventos que estén destacados en el momento de la petición.

<https://xxxx.com/rest/eventos/destacados>

La respuesta es en formato JSON.

@GetMapping("/verUno/{idEvento}")

Mediante el método GET y con PathVariable pasamos el idEvento para obtener un evento específico.

<https://xxxx.com/rest/eventos/verUno/22>

La respuesta es en formato JSON.

@GetMapping("/buscarEventos/{cadena}")

Mediante el método GET y con PathVariable pasamos la cadena de caracteres que queremos buscar en el título de los eventos.

<https://xxxx.com/rest/eventos/buscarEventos/Cas>

La respuesta es en formato JSON.

@GetMapping("/plazasQuedan/{idEvento}")

Mediante el método GET con PathVariable pasamos el idEvento para obtener las plazas disponibles de un evento dado en el momento de la petición.

<https://xxxx.com/rest/eventos/plazasQuedan/22>

La respuesta es en formato JSON.

@PostMapping("/alta")

Mediante el método POST, pasamos en el body en formato JSON el elemento que queremos dar de alta.

<https://xxxx.com/rest/eventos/alta>

La respuesta es en formato JSON.

@PutMapping("/modificar")

Mediante el método PUT pasamos en el body en formato JSON el elemento que queremos modificar

<https://xxxx.com/rest/eventos/modificar>

La respuesta es en formato JSON.

@DeleteMapping("/eliminar/{idEvento}")

Mediante el método DELETE con PathVariable pasamos el idEvento que queremos eliminar.

<https://xxxx.com/rest/eventos/eliminar/22>

La respuesta es en formato JSON.

Toda esta documentación se facilitará con JavaDoc o Swagger.

Para validar las distintas peticiones podemos usar postman o alguna herramienta similar.

