,

GRADO EN INGENIERÍA INFORMÁTICA

TRABAJO DE FINAL DE GRADO

DESARROLLO DE UN PLUGIN DE WORDPRESS PARA LA CREACIÓN DE ZONAS PRIVADAS

Autor:
David Ribes Benet

Tutor académico:
Sven Casteleyn

Curso académico 2020/2021
Fecha de lectura: 15 de Septiembre de 2021

RESUMEN

En este proyecto se expondrá el diseño, planificación e implementación del trabajo de Ingeniería Informática llevado a cabo por el alumno David Ribes Benet como trabajador indefinido de la empresa IMK Digital Multimedia, localizada en Castellón de la Plana.

El proyecto que se va a llevar a cabo es el desarrollo de un plugin interno para Wordpress para la creación de zonas privadas en páginas web. Con este plugin se pretende agilizar la creación de zonas privadas en las páginas web que se desarrollan para clientes y evitar tener que repetir el mismo código en diferentes webs.

El desarrollo del plugin se ha realizado con éxito dentro de los plazos establecidos en la planificación y actualmente se encuentra en uso en diferentes webs.

Palabras clave

Plugin, Wordpress, Zona Privada, Página Web

Keywords

Plugin, Wordpress, Private Area, Webpage

Índice general

Índice de figuras

Índice de Tablas

Capítulo 1

Introducción

1.1. Contexto y motivación del proyecto

En esta sección, se presenta tanto la empresa en la que se ha desarrollado el proyecto como la motivación que ha llevado al desarrollo del mismo.

1.1.1. La empresa

IMK Digital Multimedia es una agencia de consultoría y desarrollo de servicios en Internet para empresas: Web, eCommerce, Video, Social Media, iOS & Android apps, B2B… [1]

La gran variedad de servicios que se ofrece hace que la empresa cuente con diferentes departamentos especializados en cada uno de ellos, siendo estos:

- Diseño: Se encuentran los diseñadores web y diseñadores gráficos.
- Programación: Se encuentran tanto los programadores web como los desarrolladores de Apps.
- SEO & SEM: Especialistas en posicionamiento y analítica web.
- Social Media: Todo lo relacionado con redes sociales y community management.
- Comercial: Captación de clientes e intermediario con los diferentes departamentos.

Este plugin ha sido desarrollado para los departamentos de diseño y programación web, reduciendo así la dependencia entre ambos departamentos para algunos proyectos determinados.

1.1.2. Contexto

Actualmente hay una gran demanda de empresas que necesitan una web corporativa donde además de enseñar al público sus servicios, puedan gestionar de forma interna recursos con sus propios empleados. Una especie de intranet donde poder compartir con ellos información confidencial, enlaces de interés y archivos laborales.
Pese a las grandes diferencias de necesidades entre empresas, al final siempre hay una serie de funcionalidades comunes entre todas y suelen tener un perfil final muy parecido.

En IMK se reciben este tipo de solicitudes muy habitualmente. Aunque en el repositorio de plugins existen muchos plugins capaces de crear funcionalidades parecidas, no existe ninguno que reúna

todas juntas, lo que hace tener que utilizar varios de ellos y tener que coordinarlos o incluso modificar su comportamiento para cumplir los requisitos que se piden.

1.1.3. Breve descripción del proyecto

Con este proyecto, se busca crear un plugin interno para IMK en el que unificar todas las funcionalidades necesarias para la creación de una zona privada en una página web, agilizando así el tiempo de producción, minimizando la repetición de una misma tarea entre diferentes webs, y reduciendo la dependencia entre los departamentos de programación y diseño.

Se busca unificar en un solo plugin la creación de una zona front-end para empleados de la empresa para la que se hace la web, la asignación de permisos de usuario a *posts* o noticias de Wordpress, un sistema de notificaciones pensado para que la empresa pueda enviar notificaciones individuales o grupales por departamentos o roles y finalmente un sistema de intercambio de archivos por FTP donde se podrán compartir con el empleado de forma privada todo tipo de archivos personales (como por ejemplo nóminas).

Cuando el plugin se active, automáticamente se crearán todos los recursos necesarios para que la zona privada funcione (tablas en la base de datos, página front-end..) y se dispondrá de una página de administración para que desde IMK se puedan configurar todos los términos de la zona privada.

1.2. Objetivos del proyecto

1.2.1. Objetivo principal

El objetivo principal de este proyecto es crear un plugin propio para IMK con el que se agilizará y simplificará la creación de zonas privadas para futuros clientes en sus páginas web, evitando repetir el mismo proceso ya desarrollado para otros clientes anteriores, siendo idéntico la mayoría de veces, o en su defecto, muy parecido y con código muy reciclable.
Además, gracias a este plugin se busca reducir la dependencia en estos casos entre los departamentos de programación y diseño, ya que se requiere la participación de ambos departamentos y esto hace que se ralentice mucho el tiempo de producción de la web.

1.2.2. Objetivos secundarios

Además de los comentados objetivos principales, con este proyecto se trata de alcanzar varios objetivos secundarios:

- Liberar la carga de los programadores, permitiendo que sean los diseñadores mediante el plugin quienes se encarguen de crear y configurar la zona privada sin tener que recurrir a ellos para determinadas funcionalidades.
- Crear un sistema fácil de configurar para cualquier departamento de la empresa.
- Crear un sistema escalable que con el tiempo se pueda ir mejorando y ampliando funcionalidades.
- Dotar al cliente final de un sistema de notificaciones vía mail integrado en su Wordpress donde se pueda elegir a qué usuarios notificar o a qué departamentos (roles de usuario).
- Dotar al cliente final de un sistema en su Wordpress que permite compartir noticias, posts, archivos u otros, seleccionando los usuarios o departamentos implicados y siendo automática la forma en la que aparecerá en la vista de cada usuario.
- Dotar al cliente final de un sistema FTP para volcado de documentos y compartición de archivos confidenciales.
- Reducir notablemente el tiempo de producción de las webs.
- Atraer nuevos clientes ofreciéndoles este sistema como extra a sus páginas web.

Capítulo 2

El proyecto

En este capítulo se hará una descripción más detallada del proyecto. Primero, se explicará la situación actual de la empresa cuando tiene que hacer una web con zona privada y posteriormente se detallará lo que se pretende conseguir con el plugin.
Finalmente, se indicará el alcance del proyecto a nivel funcional, informático y organizativo así como las restricciones a las que estará sometido.

2.1. Descripción del proyecto

En la actualidad, cuando una empresa contacta con IMK para hacer una página web con una sección privada para sus empleados, hay que instalar muchos plugins externos y comunicarlos entre ellos, teniendo la mayoría de las veces que implementar mucho código para conseguir cumplir los requisitos. Esto hace que en cada web de este estilo que entra se tenga que repetir el proceso, y no solo eso, sino que al contar con tantos plugins externos siempre se acaba siendo dependiente de los mismos y puede ocurrir que con las actualizaciones que van surgiendo algunas cosas dejen de funcionar. Todo esto hace que se incrementen los tiempos de entrega y los posibles futuros fallos.

Por todo esto, se desea desarrollar un plugin interno que permita por un lado ayudar a los programadores y diseñadores de IMK a crear y configurar rápida y fácilmente toda la zona privada, y por otro lado dotar al administrador final de la web de un sistema interno en el Wordpress con diferentes funcionalidades que serán explicadas más adelante.

2.2. Alcance

2.2.1. Alcance funcional

Este plugin se podrá instalar en cualquier plantilla de Wordpress, y automáticamente creará todos los componentes necesarios para que la plantilla disponga de una nueva zona privada. De forma automática, en el momento de la activación, se crearán todas las tablas necesarias en la base de datos, así como la página de configuración del plugin en el panel del administrador de Wordpress y la página front-end que cargará el contenido para los diferentes usuarios.

A la página de administración se podrá acceder desde el panel de administración del wordpress [2]. Desde aquí, se podrá configurar el plugin según convenga para los diferentes clientes.

Se podrá gestionar tanto la creación, modificación y borrado de usuarios y roles de usuario, así como la asignación de los usuarios a los diferentes roles de usuario. Además, contendrá un formulario para poder configurar algunos de los ajustes del plugin como por ejemplo si será necesario que el usuario inicie sesión o no para acceder al contenido.

También contará con un sistema de notificaciones que permitirá al administrador del sitio enviar notificaciones vía correo electrónico. Este sistema permitirá elegir si la notificación es para un solo usuario, para varios o para todos los usuarios de uno o varios roles determinados, simplificando así la gestión de comunicados a los diferentes departamentos (cada rol de usuario corresponde con un departamento).

Para difundir noticias o enlaces de interés, se recurrirá al blog por defecto de Wordpress [3]. Lo que hará el plugin es añadir un "*metabox*" o zona de asignación de permisos en los posts, de forma que cada vez que se vaya a crear un *post* nuevo, se dispondrá de esa zona creada por el plugin desde donde se podrá asignar quién tendrá acceso a ese *Post* [4]. Se podrán asignar los permisos tanto a nivel de rol (por ejemplo, todos los de administración) como a nivel de usuario si se quiere que solamente ciertos usuarios puedan ver ese *post*.
En base a esos permisos de visualización, se programará de forma que ni aun teniendo el acceso al link del *post* se pueda visualizar si no inicias sesión y tienes acceso al contenido.

Además, se notificará a los usuarios de que se ha creado un nuevo post de su interés. Los usuarios con los que se ha compartido la noticia recibirán un enlace a la misma en su correo electrónico.

En cuanto a la página front-end del usuario, variará en función de cada usuario. El contenido se cargará de forma dinámica según el rol o roles a los que pertenezca un usuario, haciendo solamente accesible el contenido que se ha especificado que puede ver, y clasificado por las distintas temáticas o categorías que se hayan establecido desde la administración.
Además, dispondrá de un apartado exclusivo para cada usuario, donde se le podrán compartir vía FTP archivos confidenciales o privados como las nóminas.

El sistema FTP comentado, se programará de forma que en el sistema de archivos de la web, cada usuario disponga de su propia carpeta. Esta carpeta, será visible en la página del usuario de manera que desde administración se pueda compartir con el usuario cualquier tipo de archivo, y solamente él tendrá acceso pues a cada usuario le saldrá su carpeta cuando inicie sesión.

Todo esto será desarrollado desde cero, a excepción de de las páginas de inicio de sesión que serán creadas por la plantilla de Wordpress de la web. Únicamente habrá que indicarle la redirección a la página front-end de usuario. Además, se aprovechará la página de edición de datos de usuario por defecto de Wordpress, añadiendo simplemente un enlace al mismo para que el usuario pueda cambiar sus datos personales, y el blog de Wordpress, implementando únicamente la asignación de permisos de *posts* y las restricciones de acceso.

2.2.2. Alcance organizativo

Los departamentos afectados serán los departamentos de diseño y de programación web de IMK. Cuando se firma una web con un cliente, el diseñador se encarga de configurar la plantilla y de cargar el contenido, y los programadores de crear cualquier tipo de funcionalidad que no venga por defecto con la plantilla elegida.
En los casos en los que se requiere zona privada, es el programador quien tiene que desarrollar todas estas funcionalidades, y solamente cuando están operativas el diseñador puede montar el contenido. Con este plugin, se pretende que con la activación del mismo se creen todas estas funcionalidades y sea el propio diseñador quien lo configure sin la necesidad de requerir de un programador, que generalmente, tienen mucha más carga de trabajo. Además, se evita la dependencia entre los departamentos y se ahorra mucho tiempo de producción.

2.2.3. Alcance informático

Para el alcance informático, el plugin no se comunica con ningún subsistema propio de IMK pero sí con la base de datos de Wordpress, así como con todo el sistema de ficheros de la web, por lo que se requiere acceso al servidor en el que está montado el Wordpress. En el caso de IMK, al contar con servidores propios que auto-administran, se puede acceder a la base de datos mediante Plesk [5] , y al sistema de archivos tanto con Plesk como con cualquier programa de FTP. En el caso de este proyecto se ha hecho uso de FileZilla [6].

2.3. Restricciones

Para el desarrollo de este proyecto, se han impuesto ciertas restricciones o pautas a seguir.

En cuanto a las tecnologías a utilizar, se ha indicado que toda la parte del backend debe desarrollarse con PHP puesto que es el lenguaje de desarrollo de Wordpress. Por lado de la aplicación frontend, se ha indicado que debe realizarse utilizando HTML. El resto de tecnologías utilizadas durante el proyecto, se han aplicado como apoyo al considerarse necesarias.

Con lo que respecta al tiempo, pese a que la estimación es de 300 horas, no se considera crítico sobrepasar este límite ya que se da como hecho que se van a necesitar constantes modificaciones y cambios en función de las necesidades de la empresa y es posible que haya que rehacer algún apartado si el jefe de proyecto lo considera necesario.

En cuanto al presupuesto, no se ha proporcionado ninguno puesto que en IMK ya se disponen de todos los recursos necesarios para realizar el proyecto. Tanto las herramientas utilizadas, como los gastos de servidor, como la plantilla de Wordpress utilizada ya forman parte de los gastos estimados por IMK durante el día a día de la empresa, y no es necesario adquirir ningún software adicional.

En cuanto a los recursos humanos, el proyecto es realizado por una sola persona pero se obvia apoyo de compañeros en caso de necesitarla para desarrollar ciertas funcionalidades que ya han sido desarrolladas con anterioridad por alguno de ellos, bien de forma exacta o bien de forma similar.

Capítulo 3

Tecnologías y herramientas

En este capítulo se exponen las tecnologías y herramientas utilizadas durante todo el proyecto para el desarrollo del plugin.

3.1. CMS

3.1.1. Wordpress

WordPress es lo que se conoce como un gestor de contenidos o CMS (Content Management System), es decir, un sistema para crear y diseñar sitios web donde principalmente se van a publicar contenidos (páginas, artículos, vídeos, etc.) fácilmente sin tener que saber programación web ni nada por el estilo [7].

Wordpress es el eje central de este proyecto puesto que es la plataforma para la que se desarrollará el plugin. Será en Wordpress donde se instale y configure, y desde donde se podrá ir comprobando su funcionamiento, por lo tanto el proyecto tendrá que encajar perfectamente en la estructura de Wordpress.

A una instalación de wordpress se le añade siempre una “Plantilla” o “Tema” que es lo que le dará la apariencia principal a la página web. Además, desde su página de opciones del tema se podrá personalizar esta misma. En el caso de este proyecto se ha utilizado para el entorno de desarrollo una plantilla Kalium [8] ya que es la que se utiliza en IMK para la creación de la mayoría de las webs.

Puesto que el plugin es independiente de la plantilla o tema que esté activo en el Wordpress, no afectará en la instalación del mismo en otras webs que utilicen una plantilla diferente. Únicamente se tendrá que tener en cuenta que la versión del plugin que se instale sea compatible con la versión de Wordpress que esté instalada en la web, y adaptar las páginas de inicio de sesión puesto que no forma parte del alcance del plugin.

3.2. Lenguajes

3.2.1. PHP

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de programación de código abierto destinado a desarrollar aplicaciones para la web y crear páginas web, favoreciendo la conexión entre los servidores y la interfaz de usuario. Puede ser incrustado en HTML [9].

En este proyecto se ha utilizado PHP para desarrollar toda la parte del back-end.

3.2.2. HTML

HTML (HyperText Markup Language) es un lenguaje de marcado para la elaboración de páginas web. Es un estándar que define una estructura básica y un código para la definición de contenido de una página web [10].

Se ha utilizado HTML para el desarrollo de la parte front-end del proyecto. Cualquier visualización final por parte del usuario se ha realizado a través de este lenguaje, tanto para la página de configuración que está en la administración del Wordpress como para la página final del usuario en la que se vuelca todo el contenido.

3.2.3. CSS

Hojas de estilo en cascada (CSS, siglas en inglés de *Cascading Stylesheets*) es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de las páginas web, e interfaces de usuario escritas en HTML o XHTML [11].

CSS es una tecnología usada junto con muchas otras por muchos sitios web para crear páginas atractivas, interfaces de usuario para aplicaciones web e interfaces gráficas para muchas aplicaciones móviles .
En este proyecto, se ha utilizado junto a HTML para aplicar los estilos de las diferentes páginas creadas. Su papel más importante ha sido en la página de usuario, donde mediante CSS se han aplicado no solamente estilos visuales como fondos o colores, sino también para establecer el comportamiento de colocación de los contenedores, así como sus tamaños.

3.2.4. JavaScript

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico [12].

En el proyecto, JavaScript ha sido utilizado para emplear funcionalidades explícitas en todas las páginas front-end.

3.2.5. jQuery

jQuery es una biblioteca de JavaScript que permite interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos y agregar interacción con AJAX a páginas web [13].

En el proyecto se ha usado jQuery principalmente para captar eventos y modificar el DOM. Por ejemplo, cuando en la página de configuración del plugin se aprieta el botón de borrado de un Usuario, con jQuery se captura el click y se obtiene el id del botón seleccionado, y se modifica el DOM para que ya no aparezca.

3.2.6. Ajax

AJAX es un acrónimo de Asynchronous JavaScript And XML. Es un técnica de desarrollo para crear aplicaciones interactivas que se ejecutan en el navegador de los usuarios mientras se mantiene una comunicación asíncrona con el servidor en segundo plano. Por esto, es posible realizar cambios en las páginas sin tener que recargarlas [14].

Para el proyecto se ha utilizado Ajax para modificar el contenido de las páginas, la mayoría de veces tras capturar un evento de JavaScript.

3.3. Librerías

3.3.1. Hooks de Wordpress

Los hooks de WordPress podrían definirse como ganchos o puntos de acceso en determinadas partes o ubicaciones del proceso de ejecución donde podemos engancharnos con nuestras propias funciones y alterar o modificar el comportamiento por defecto de dicho proceso y/o función, o simplemente modificar alguna variable o dato o añadir una funcionalidad extra que queramos que se ejecute en un momento determinado [15].

Es decir, nos permiten modificar el código fuente de WordPress o de un plugin sin realmente modificar el código fuente. Podemos alterar el comportamiento de WordPress o de un plugin sin tocar su core.

Intervenir a través de los hooks es una buena práctica, ya que permite seguir actualizando el core de WordPress o los plugins sobre los que hayamos actuado sin perder la funcionalidad extra que hemos añadido.

En la siguiente figura (Figura 1) se observa una ilustración de cómo se puede enganchar código propio a código de Wordpress.

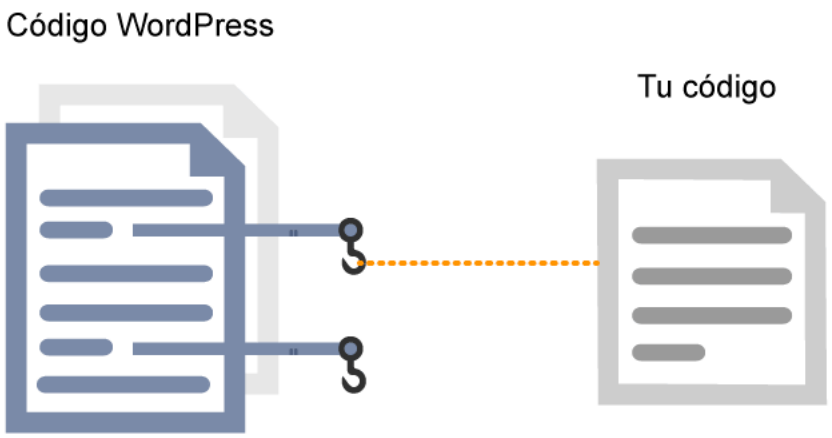

Figura 1: Hooks de Wordpress [16]

Existen dos tipos de hooks , los "*Action Hooks*" y los "*Filter Hooks*'' [16].

1. **Action Hooks:** Las acciones o *action hooks* ejecutan una función propia en un lugar preciso en un momento determinado de la línea de ejecución de WordPress.
2. **Filter Hooks:** Los filtros o *filter hooks* de WordPress manipulan la información en un punto concreto de la ejecución antes de que se muestre en la pantalla o se almacene en la base de datos.

Ambos tipos de hooks se han utilizado constantemente durante todo el desarrollo del proyecto para ejecutar determinadas acciones en momentos determinados, como por ejemplo en las acciones automáticas de preparación del entorno a realizar cuando se hace la activación del plugin, o en la acción de actualizar un post para aprovechar y enviar notificaciones a los usuarios afectados.

3.3.1.1 Hookr

Pese a que existen miles de portales desde donde acceder a las librerías y hooks de Wordpress, para la realización de este proyecto se ha utilizado en su mayoría el portal hookr.io como referencia [17].

3.4. Base de datos

3.4.1. MySQL

Para el desarrollo del plugin se interactúa constantemente con la base de datos de Wordpress.
Wordpress utiliza como base de datos MySQL, que es un sistema de gestión de bases de datos que cuenta con una doble licencia. Por una parte es de código abierto, pero por otra, cuenta con una

versión comercial gestionada por la compañía Oracle. Es una de las bases de datos de código abierto más famosas y utilizadas en el mundo entero [18].

Para el proyecto no se ha tenido que crear ninguna base de datos puesto que se utiliza la propia base de datos del Wordpress instalado, por lo tanto se crea durante la instalación del mismo y solamente es necesario añadir las tablas extra propias que se necesiten para el desarrollo del plugin.

3.5. Despliegue

3.5.1. Plesk

Plesk es un panel de administración que se instala en los servidores y cuyo principal propósito es el de facilitar las tareas de gestión y administración de un servicio de hosting a los usuarios. Una de sus principales ventajas es que permite a los usuarios una instalación a medida, totalmente personalizada. Además, es un software muy intuitivo y seguro; utiliza el estándar de seguridad adoptado por los servidores UNIX [5].

Desde este panel es desde donde se ha realizado la instalación del Wordpress, así como todo lo relacionado con la gestión del dominio y desde donde se puede acceder para visualizar la base de datos, así como otros servicios propios del hosting web como el acceso a los ficheros.

3.6. Gestión del proyecto

3.6.1. Tracking Time

Tracking Time es un software para la gestión de proyectos que permite a las empresas llevar un control detallado de proyecto activo en la empresa [19].

Pese a que esta herramienta tiene infinidad de funcionalidades, en IMK se gasta para el control del tiempo de los proyectos. Un mismo proyecto se descompone en varias tareas y se registra cuánto tiempo lleva cada una de ellas, de forma que se activa el temporizador cuando se empieza la tarea y se para cuando termina. De esta forma todo queda registrado y se lleva un control exhaustivo de cada tarea y se almacenan referencias para futuros proyectos.
Además, sirve para comparar entre una planificación inicial y los resultados finales o para poder comprobar si se ha excedido un plazo.

En la Figura 2, se aprecia una captura de pantalla de dicha herramienta hecha durante el desarrollo del proyecto.

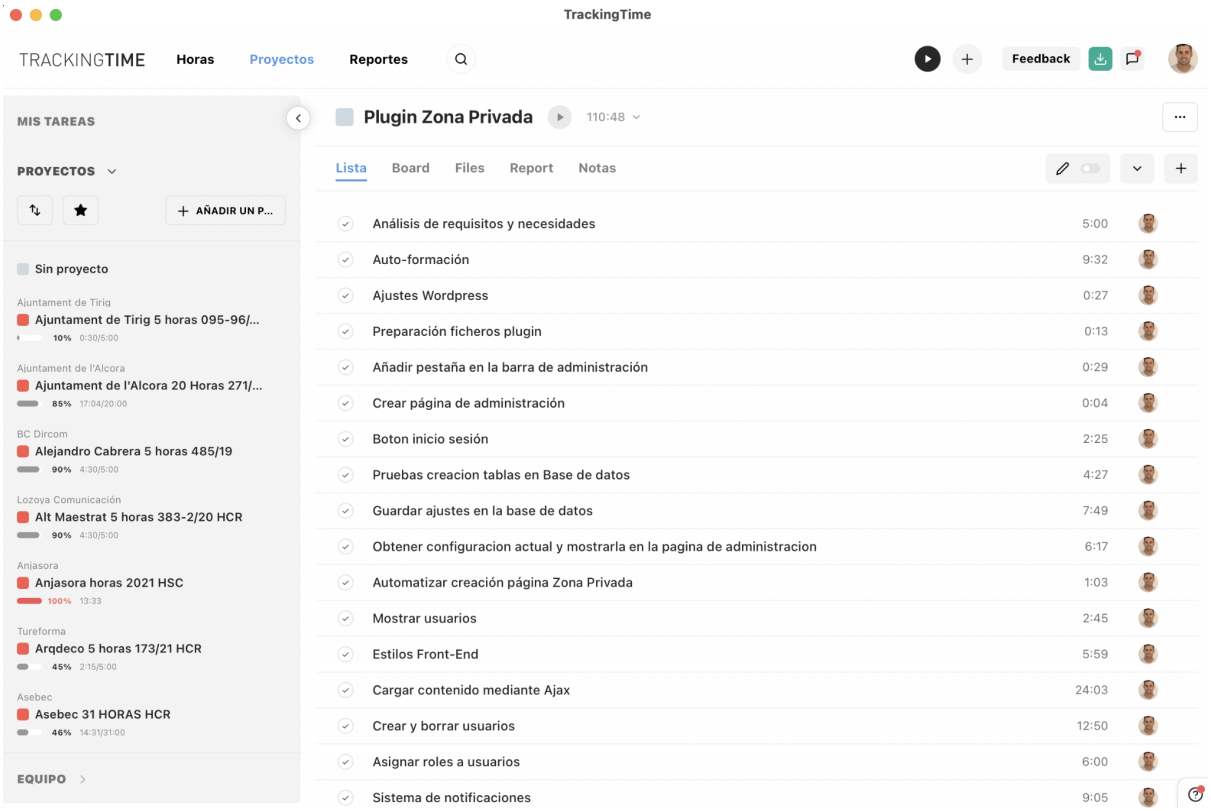

Figura 2: Captura de pantalla herramienta Tracking Time

3.6.2. Slack

Slack es un sistema de mensajería en tiempo real para la comunicación entre equipos que incluye todos los medios de comunicación en un mismo sitio y que además integra gran cantidad de herramientas (Dropbox, Google Drive, Twitter, MailChimp, Skype…) [20].

En IMK se gasta esta herramienta para realizar comunicaciones acerca de los diferentes proyectos, de forma que todo quede registrado y sirva para futuras referencias.

Se ha utilizado para realizar consultas determinadas a los programadores de la empresa durante el periodo de confinamiento, especialmente para dudas o errores en los que se necesitaba de un programador más experimentado. También para las comunicaciones con el responsable del proyecto acerca de revisiones o dudas.

En la figura 3 se aprecia una captura de pantalla de la herramienta en el entorno de IMK.

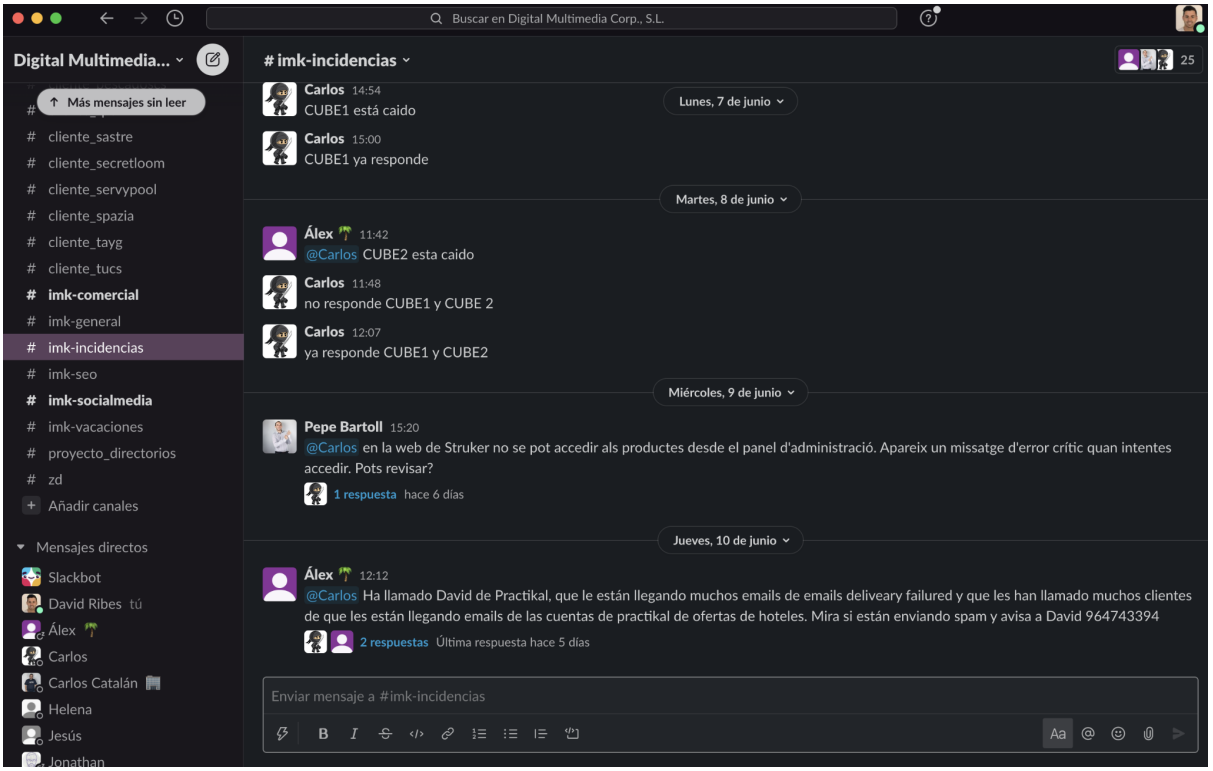

Figura 3: Captura de pantalla herramienta Slack

Capítulo 4

Planificación del proyecto

En este capítulo se explica la metodología utilizada durante el proyecto. Para ello, se comparará la metodología que se suele utilizar en IMK para el desarrollo de plugins con otras metodologías similares y estudiadas durante todo el grado.

A continuación, se explicará como se ha planificado el proyecto, así como el seguimiento del mismo y se debatirán los resultados en base a la planificación.

4.1. Metodología

Durante el desarrollo del plugin se ha utilizado una metodología iterativa muy similar a la metodología Scrum [21].
Pese a que no llega a ser exactamente igual, si que identificamos muchas similitudes entre la metodología Scrum y la metodología utilizada habitualmente en los proyectos de IMK. Por lo tanto, se puede decir que se ha utilizado una metodología Scrum adaptada a las necesidades de la empresa.

Generalmente, en IMK, al comienzo de un proyecto se crea en Tracking Time (*backlog* en Scrum) un listado con todas las tareas a realizar o casos de uso (historias de usuario en Scrum). El director del proyecto es quien crea estas tareas (Scrum Manager) tras hablar con la persona interesada en el desarrollo (Product Owner en Scrum). En este caso, la persona interesada es una puesta en común de las necesidades de todos los miembros del departamento para agilizar el desarrollo de zonas privadas.

Semanalmente, se realiza una reunión (Scrum review) con el equipo (Scrum Team) en la que se comenta toda la faena realizada durante la semana, así como los problemas surgidos. En base a los resultados de la semana, se organiza la siguiente semana, bien sea asignando nuevas tareas a los miembros del equipo o ampliando tareas de la semana actual en caso de no haberse finalizado las tareas.

En IMK, esta reunión se realiza los viernes a las 8:00h y es de carácter general, es decir, no solamente se trata un proyecto sino que se tratan todos los proyectos en marcha de cada miembro. Como el desarrollo de este plugin ha sido realizado en su totalidad por mí, solamente afecta a compañeros en caso de necesitar apoyo con alguna tarea específica.

Cada tarea del proyecto se puede identificar con un Sprint en Scrum. A cada sprint se le hace una estimación de tiempo y se asigna al miembro que tiene que realizarla en Tracking Time. Mediante el contador de esta herramienta, el miembro del equipo le da al play cuando empieza a realizarla y le da al stop cuando termina. De este modo se sabe exactamente el tiempo que ha llevado el desarrollo de esta tarea.
Solamente cuando la tarea se finaliza se puede pasar a la siguiente, es por eso que la realización de tareas se hace de forma iterativa siguiendo las indicaciones de funcionalidad del director del proyecto.

Se empieza una tarea, se finaliza y se prueba. Y solamente si pasa las pruebas, se pasa a la siguiente tarea.

Al igual que en la mayoría de proyectos de IMK, esta es la metodología que se utiliza para el desarrollo del plugin. Como se observa, presenta muchas similitudes con la metodología Scrum, pero también tiene notables diferencias.

La principal diferencia con Scrum es la duración de los Sprint. En IMK, el director de proyecto habitualmente asigna un tiempo a cada tarea basándose en tareas similares realizadas con anterioridad, ya que se puede revisar el histórico en Tracking Time, y la reunión se hace semanalmente independientemente de si se ha terminado la tarea o no.
Puesto que no se trabaja en este proyecto en exclusiva, la duración del sprint no se calcula en días sino en horas previstas, independientemente de en cuántos días se repartan estas horas, ya que dependerá de la carga de trabajo de otros proyectos.

4.2. Planificación

La planificación del proyecto se ha dividido en tres fases: investigación, planificación e implementación.

En la primera fase, se ha realizado un análisis de necesidades, requisitos y funcionalidades que tiene que tener el plugin. Para ello, se realizan reuniones individuales y colectivas con todos los miembros del equipo, y finalmente es el director del proyecto quien en base a estas reuniones y necesidades elabora una lista de tareas a desarrollar.

En la segunda fase, se ha realizado un estudio sobre todas las tecnologías que se van a utilizar, así como del funcionamiento de los hooks de Wordpress para saber qué comportamientos se pueden modificar o a cuáles podemos engancharnos añadiendo una funcionalidad.

En la tercera fase, se lleva a cabo todo el desarrollo del proyecto. Para ello, tras todas las reuniones realizadas, el director del proyecto elabora una lista con todas las tareas que hay que realizar y le asigna una estimación de tiempo. Estas tareas, serán creadas en el Tracking Time.

Pese a que las tareas iniciales son creadas y asignadas por el director del proyecto, la persona que tiene asignada la tarea la puede dividir en subtareas más exactas, para conocer con más exactitud en qué se emplea el tiempo. Es decir, si el director del proyecto crea una tarea llamada “Crear página de usuario” , el programador podrá crear subtareas llamadas “Mostrar noticias por rol” , “Mostrar noticias personales” que hace referencia a la parte de la página del usuario en la que se mostrarán las noticias del rol al que pertenezca y las noticias personales respectivamente.

Para este proyecto, en su inicio, se ha creado la siguiente estimación (Tabla 1) :

Tarea	**Horas Estimadas**
Fase 1: Análisis de requisitos y necesidades	5
Fase 2: Auto formación: Vídeos, tutoriales, preguntas...	25
Fase 3 : Primeros pasos	
- Creación y preparación de ficheros	5
-Instalación de plugin para que aparezca en el Wordpress	1
-Acciones activación del plugin	10
Fase 3: Funcionalidades administración Wordpress	
-Formulario de configuración	15
-Creación de usuarios	20
-Creación de roles	15
-Rol de usuario seleccionable en el formulario de registro	20
-*Metabox* en la página de edición del *post*	5
-Noticias por usuario	10
-Noticias por rol	25
- Sistema de notificaciones en la administración del Wordpress	20
- Zona FTP	40
Fase 3: Front End	
-Página de registro/inicio de sesión	5
-Página de usuario	20
-Mostrar posts de usuario	15
-Mostrar posts rol	15
-Mostrar carpeta FTP	10
-Testeo	15
-Documentación	5
TOTAL	301

Tabla 1: Planificación inicial del proyecto

En cuanto a las fechas, para este proyecto no se ha fijado ningún límite puesto que al ser un desarrollo para uso interno no hay ningún compromiso con ningún cliente, y por lo tanto se ha ido desarrollando en paralelo junto con otros proyectos.

En el caso de un estudiante en prácticas, el límite habría sido la finalización de las prácticas, pero al desarrollar yo el proyecto siendo miembro desde hace tiempo de la empresa, se ha ido desarrollando de forma adicional al trabajo habitual para otros clientes. Esto hacía difícil fijar un límite en el calendario ya que no siempre se podrá trabajar en el proyecto, sino que dependerá de la carga de trabajo de la empresa.
El director del departamento decide semanalmente cuánto tiempo se aplica a cada proyecto, siendo las semanas más liberadas de trabajo para clientes las que se dedican más tiempo al desarrollo del plugin.

Por lo tanto, tal y como se aprecia en la siguiente tabla, la fecha de inicio y finalización de una tarea puede no tener coherencia si se piensa que solamente se está trabajando en este proyecto, pero al haber semanas con más horas dedicadas al proyecto que otras, puede ocurrir que se tarde más días en hacer una tarea más rápida que otra.
Pero esto no supone un problema, ya que gracias a Tracking Time podemos saber exactamente cuántas horas se ha dedicado a cada punto de la planificación independientemente de los días usados.

En esta tabla se puede comparar la planificación inicial contra los resultados reales (Tabla 2) :

Tarea	**Horas Estimadas**	**Horas Dedicadas**	**Fecha Inicio**	**Fecha Fin**
Fase 1: Análisis de requisitos y necesidades	5	5	14/09/20	15/09/20
Fase 2: Auto formación: Vídeos, tutoriales, preguntas...	25	25	16/09/20	25/09/20
Fase 3 : Primeros pasos				
- Creación y preparación de ficheros	5	4	28/09/20	28/09/20
-Instalación de plugin para que aparezca en el Wordpress	1	1	29/09/20	29/09/20
-Acciones activación del plugin	10	13	05/10/20	12/10/20
Fase 3: Funcionalidades administración Wordpress				
-Formulario de configuración	15	20	13/10/20	22/10/20
-Creación de usuarios	20	18	22/10/20	02/11/20
-Creación de roles	15	10	17/11/20	23/11/20

-Rol de usuario seleccionable en el formulario de registro	20	19	07/12/20	23/12/20
-Metabox en la página de edición del post	5	8	04/01/21	11/01/21
-Noticias por usuario	10	12	14/01/21	18/01/21
-Noticias por rol	25	19	19/01/21	19/02/21
- Sistema de notificaciones en la administración del Wordpress	20	8	01/03/21	08/03/21
- Zona FTP	40	61	09/03/21	15/04/21
Fase 3: Front End				
-Página de registro/inicio de sesión	5	2	16/04/21	16/04/21
-Página de usuario	20	21	16/04/21	22/04/21
-Mostrar posts de usuario	15	12	23/04/21	28/04/21
-Mostrar posts rol	15	6	29/04/21	03/05/21
-Mostrar carpeta FTP	10	5	04/05/21	04/05/21
-Testeo	15	15	14/09/20	04/05/21
-Documentación	5	0	-	-
TOTAL	301	284		

Tabla 2: Planificación final del proyecto

Tal y como se aprecia, el proyecto se ha terminado en menos horas de las previstas. Por lo general, no ha habido grandes desviaciones en la estimación, y casi todas las tareas se han podido hacer en el tiempo previsto o incluso en menos. La única gran desviación de sobrepasar las horas ha sido en el desarrollo de la zona FTP.
Esto es debido a que se complicó un poco el tener que automatizar las copias de la carpeta del usuario en caso de borrado, y recuperarla en caso de volverse a crear el mismo usuario. Eso junto con la recursividad para mostrar las subcarpetas de todo el árbol de ficheros de la carpeta hizo que hubiera una desviación.

Por lo demás, el proyecto ha sido un éxito.

4.3. Estimación de recursos y costes

Para calcular el coste del proyecto, no se ha calculado como si fuera un trabajo habitual por horas a cualquier cliente. Generalmente, para este tipo de clientes , se suele cobrar un precio fijo por horas de 40€ ya que hay que sacar beneficio inmediato.

Pero en este caso, al ser un proyecto interno que será amortizado por la empresa con el tiempo en concepto de horas de trabajo, se han tenido en cuenta solamente los costes del operario que lo implementa, en este caso yo.

Para calcular el coste por hora de un operario de IMK se tienen en cuenta dos cosas:

1. En primer lugar, el coste de la hora trabajada del operario. Este coste es diferente para cada trabajador y se calcula dividiendo el sueldo bruto anual del empleado entre 1800h. En mi caso el coste de hora trabajada es de 11,39€.
2. En segundo lugar, se tienen que tener en cuenta los gastos directos atribuidos a la empresa. En estos gastos estarían todos los gastos que tiene la empresa a lo largo del año, como la luz, las licencias, gestoría… etc. Estos gastos se suman y se dividen entre todos, por lo que es un plus a la hora que se suma al coste del trabajador y que es el mismo para todos. Tras hablar con el departamento de contabilidad me dijeron que este plus es de 0,55 €/h.

Si sumamos ambos puntos, puesto que el proyecto únicamente está implementado por mí, cada hora trabajada del proyecto tendría un coste de:

11,39€ + 0,55€ = 11,94 €

Resumiendo un poco, en el primer punto se tiene en cuenta lo que paga IMK por cada hora trabajada mía, mientras que en el segundo punto se están teniendo en cuenta todos los gastos de la empresa.
Con este cálculo de costes no se están teniendo en cuenta únicamente los gastos directamente relacionados con el proyecto, ya que en los gastos del segundo punto se están teniendo en cuenta todos los gastos de la empresa, sean utilizados en el proyecto o no, pero es la forma que tiene IMK de calcular el coste de cada hora trabajada y este proyecto se calcula exactamente igual que los demás.

Para este proyecto, en los gastos atribuidos a la empresa, podríamos incluir el software utilizado, tanto Tracking Time, como Slack, como el alojamiento y la plantilla de Wordpress, pero tal y como se ha explicado, entran en el cómputo global de la empresa.

Por lo tanto, sabiendo el coste de cada hora , y sabiendo que proyecto está estimado en 300 horas, el coste total del proyecto sería:

11,94€ x 300h = 3582€

4.4. Seguimiento del proyecto

Tal y como se ha explicado en el apartado 4.1, se ha seguido una metodología ágil muy parecida a metodología Scrum, donde se lleva un control semanal del avance del proyecto.

Para hacer un seguimiento de las tareas, la empresa se apoya en las herramientas Tracking Time [19] y Slack [20]. Con Tracking Time se puede ver lo que está llevando de tiempo cada tarea y compararlo con la estimación realizada. En la reunión semanal, se observa el tiempo dedicado a cada tarea y se ve si se ha establecido en el tiempo previsto, o si por el contrario ha habido desviaciones. En caso de haber desviaciones se estudia por qué pueden haber sido y se busca una solución de cara a la siguiente semana. En slack, se queda registrada cualquier incidencia que haya ocurrido así como su posterior solución en caso de que haya ocurrido con anterioridad, además de conversaciones con compañeros acerca de problemas concretos.

En la reunión semanal, se hace una reflexión revisando toda la información de la que se dispone en estas dos herramientas y se debate con el director para comentar puntos de vista, opiniones y propuestas. En base a esto, el director reorganiza la semana siguiente y realiza las modificaciones que crea convenientes para tratar de enderezar el proyecto en caso de que se haya desviado o de mantener la misma línea si algo está funcionando bien.

Capítulo 5

Análisis y diseño del sistema

En este capítulo se hará un análisis del sistema y su diseño. Primero, se mostrará un diagrama de casos de uso y sus requisitos, se explicará la arquitectura y finalmente se mostrará el diseño establecido para cada apartado del proyecto.

5.1. Diagramas de casos de uso

En este punto del proyecto todavía no se han decidido exactamente qué partes tendrá el plugin, pero sí que se puede tratar de agrupar los casos de uso en función de quién tendrá que poder hacerlos o en base a con qué fin.

En la figura 4, se pueden observar los casos de uso que tendrá que ejercer el administrador del Wordpress en cuanto a la configuración del plugin y el sistema de envío de notificaciones se refiere. Generalmente será alguna persona del departamento de diseño de IMK o algún empleado de la empresa propietaria de la Web con conocimientos y formación en Wordpress.

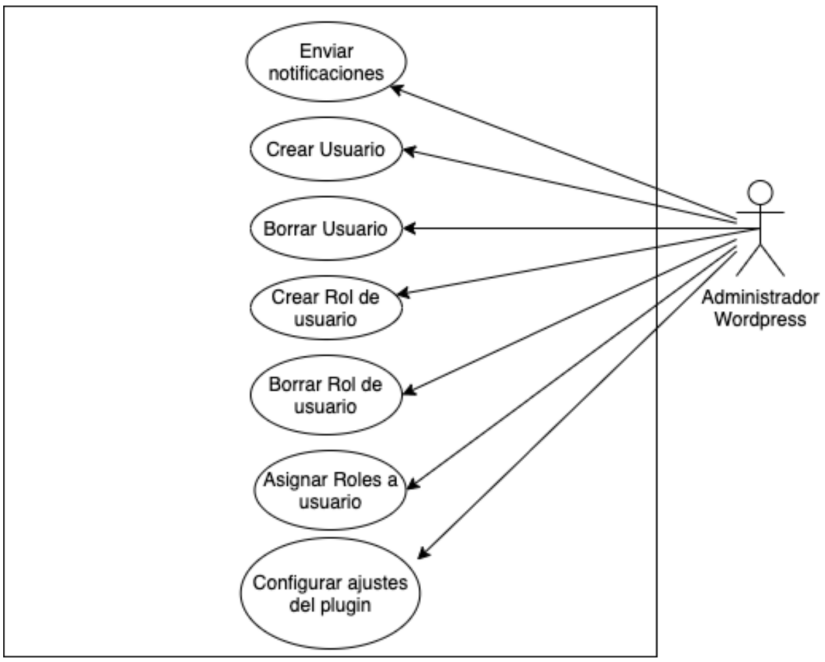

Figura 4: Diagrama de casos de uso configuración del plugin

En la figura 5, podemos observar el diagrama de casos de uso que tendrá que poder hacer la misma persona que en el diagrama anterior (Figura 4) pero referida al sistema asignación de permisos en las noticias o *posts*. En este caso es más probable que sea un empleado de la empresa propietaria de la web quien ejecute los casos, pero también podría ser un empleado de IMK si el cliente tiene contratado el mantenimiento de la web y la actualización de las noticias.
Además, también el caso de uso que ejecutará automáticamente el backend en cuanto el administrador del Wordpress termine de asignar los permisos.

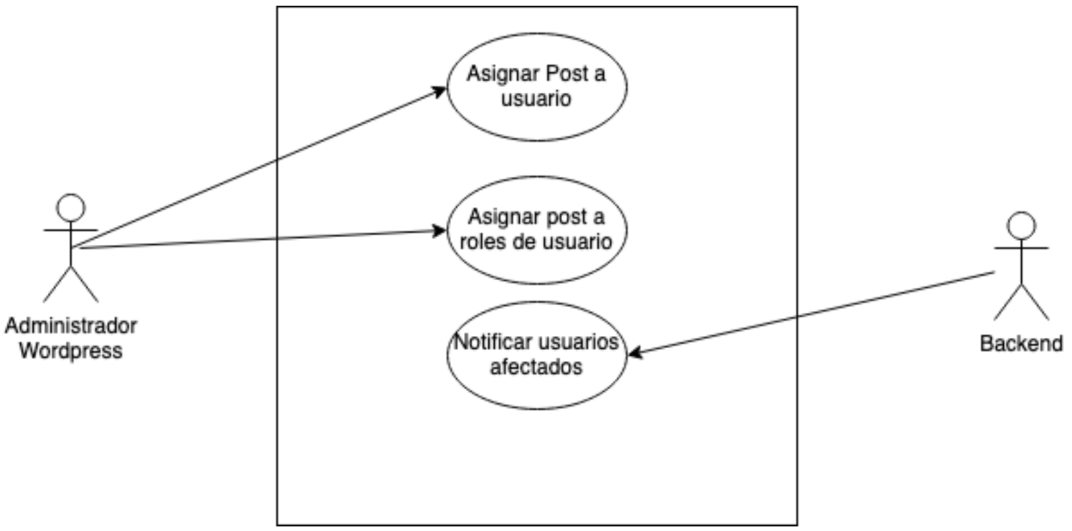

Figura 5: Diagrama de casos de uso asignación de permisos a *posts*

Por último, en la figura 6, se aprecian todos los casos de uso relacionados con el usuario final. Es decir, los empleados de la empresa o aquellos que no tienen acceso al Wordpress y que dispondrán de una zona front-end para poder acceder al contenido.
Igual que en la figura anterior (Figura 5), el backend también deberá trabajar para cumplir unos requisitos en base al tipo de usuario que esté accediendo a la página.

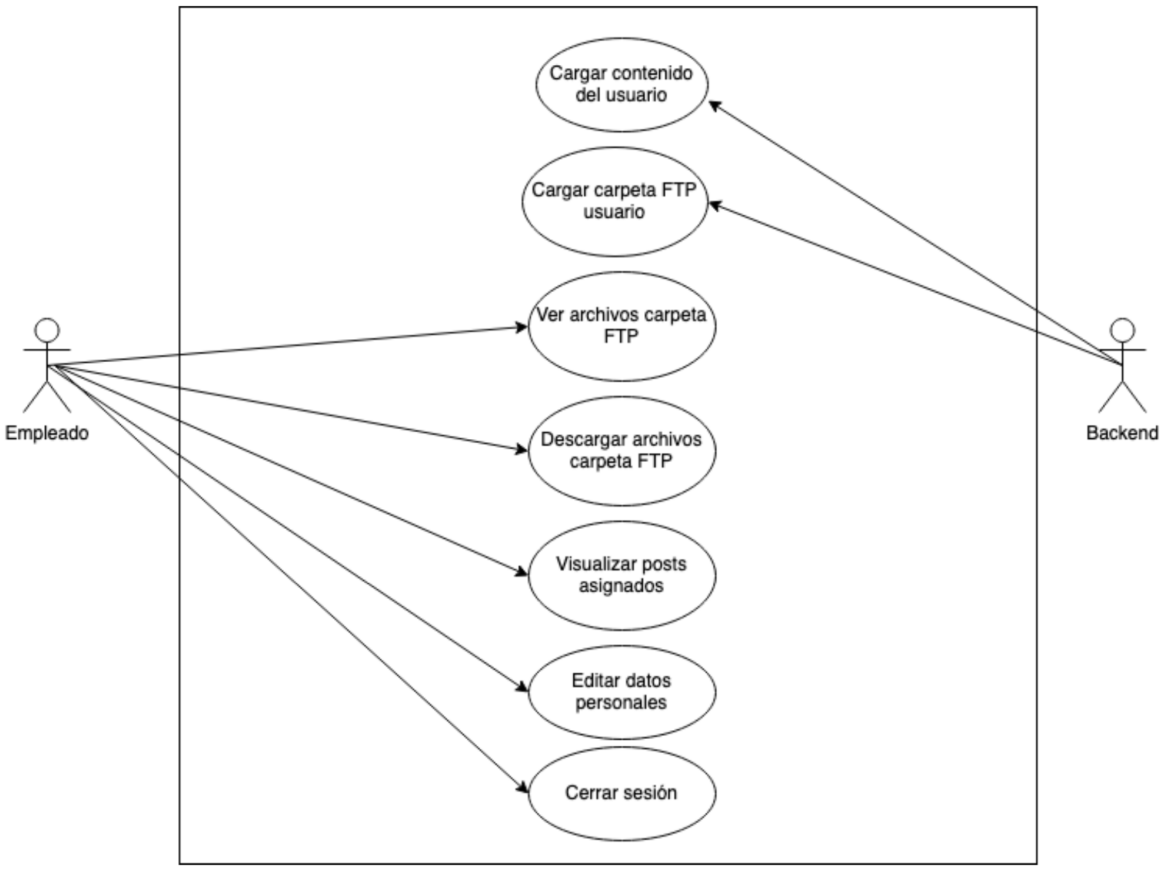

Figura 6: Diagrama de casos de uso del usuario

5.2. Requisitos funcionales del sistema

En este apartado, se especifican los requisitos de uso que se han mostrado en los diagramas del apartado 5.1.

En esta definición de requisitos, se ha eliminado la información innecesaria o irrelevante dejando solamente los campos importantes.

5.2.1. FR01 - Enviar notificaciones

Requisito funcional

ID	FR01
Nombre	Enviar notificaciones
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador del wordpress enviar notificaciones por email a los usuarios.
Precondiciones	Ninguna
Actor principal	Administrador Wordpress
Secuencia normal	1. El actor selecciona la pestaña notificaciones del panel de administración del plugin. 2. El actor selecciona los usuarios a los que quiere enviar la notificación. 3. El actor rellena los datos con el asunto y el cuerpo principal. 4. El actor le da al botón de enviar. 5. La aplicación *backend* envía todo el contenido a los usuarios afectados vía mail.
Postcondición	Si el correo se ha enviado con éxito, aparecerá el mensaje “Notificación enviada con éxito” subrayada en verde. Si el envío no se ha producido, se indicará con el mensaje “Ha ocurrido un error, inténtelo de nuevo” subrayado en rojo.
Comentarios	El actor podrá filtrar los usuarios por los diferentes roles de usuario creados en el sistema.

Tabla 3: Requisito funcional FR01

5.2.2. FR02 - Crear usuario

Requisito funcional	
ID	FR02
Nombre	Crear usuario
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador crear nuevos usuarios

Precondiciones	La opción “Habilitar registro usuarios” del formulario de configuración del plugin tiene que estar activada.
Actor principal	Administrador Wordpress
Secuencia normal	1. El actor selecciona la página de configuración del plugin. 2. El actor rellena los campos obligatorios en el formulario de creación de usuario. 3. El actor le da al botón “añadir usuario”. 4. La aplicación *backend* gestiona la creación del usuario.
Postcondición	Si el usuario se ha creado con éxito aparecerá en el listado de usuarios junto con todos los demás. Si el usuario no se ha creado o ya existe, se indicará con un mensaje de error.
Comentarios	Los únicos campos necesarios son el nombre, la contraseña y el correo.

Tabla 4: Requisito funcional FR02

5.2.3. FR03 - Borrar usuario

Requisito funcional	
ID	FR03
Nombre	Borrar usuario
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador borrar usuarios
Precondiciones	La opción “Habilitar registro usuarios” del formulario de configuración del plugin tiene que estar activada.
Actor principal	Administrador Wordpress
Secuencia normal	1. El actor selecciona la página de configuración del plugin. 2. El actor aprieta el botón de eliminar que aparece junto a cada usuario en el listado de usuarios. 3. La aplicación *backend* gestiona el borrado del usuario.
Postcondición	El listado de usuarios mostrará todos los usuarios sin el que se acaba de borrar.

Comentarios	-

Tabla 5: Requisito funcional FR03

5.2.4. FR04 - Crear rol de usuario

Requisito funcional	
ID	FR04
Nombre	Crear rol de usuario
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador crear roles de usuario
Precondiciones	La opción “Habilitar registro usuarios” del formulario de configuración del plugin tiene que estar activada.
Actor principal	Administrador Wordpress
Secuencia normal	1. El actor selecciona la página de configuración del plugin. 2. El actor introduce el nombre del nuevo rol en el campo. 3. El actor aprieta el botón de “Añadir Rol”. 4. La aplicación *backend* gestiona la creación del rol.
Postcondición	El listado de usuarios mostrará para cada usuario todos los roles incluido el nuevo. Si el rol ya existía, se indicará en un *pop-up*.
Comentarios	El rol se creará sin ningún usuario asociado.

Tabla 6: Requisito funcional FR04

5.2.5. FR05 - Borrar rol de usuario

Requisito funcional	
ID	FR05
Nombre	Borrar rol de usuario

Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador borrar roles de usuario
Precondiciones	La opción “Habilitar registro usuarios” del formulario de configuración del plugin tiene que estar activada.
Actor principal	Administrador Wordpress
Secuencia normal	1. El actor selecciona la página de configuración del plugin. 2. El actor aprieta el botón de eliminar que aparece junto a cada rol en el listado de roles. 3. La aplicación *backend* gestiona el borrado del rol.
Postcondición	El listado de roles mostrará todos los roles sin el rol que se acaba de borrar.
Comentarios	El sistema debe eliminar la asignación del rol a los usuarios en la base de datos.

Tabla 7: Requisito funcional FR05

5.2.6. FR06 - Asignar roles a usuarios

Requisito funcional	
ID	FR06
Nombre	Asignar roles a usuarios
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador añadir y quitar roles a los usuarios.
Precondiciones	La opción “Habilitar registro usuarios” del formulario de configuración del plugin tiene que estar activada.
Actor principal	Administrador Wordpress
Secuencia normal	1. El actor selecciona la página de configuración del plugin. 2. El actor marca o desmarca las casillas de cada usuario

	pertenecientes a cada rol en función del rol que quiere asignar o quitar. 3. El actor le da al botón de “Guardar”. 4. La aplicación *backend* gestiona la actualización de los roles de los usuarios.
Postcondición	La página se mostrará exactamente igual que cuando se selecciona el botón de guardado.
Comentarios	-

Tabla 8: Requisito funcional FR06

5.2.7. FR07 - Configurar ajustes del plugin

Requisito funcional	
ID	FR07
Nombre	Configurar ajustes del plugin
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador configurar los ajustes del plugin.
Precondiciones	Ninguna
Actor principal	Administrador wordpress
Secuencia normal	1. El actor selecciona la página de configuración del plugin. 2. El actor rellena el formulario seleccionando las opciones que necesita. 3. El actor le da al botón de “Guardar”. 4. La aplicación *backend* actualizará la base de datos con la configuración.
Postcondición	Si es la primera vez que se configura, se mostrará la configuración por defecto, sino, mostrará la configuración actual.

Comentarios	-

Tabla 9: Requisito funcional FR07

5.2.8. FR08 - Asignar post a usuarios

Requisito funcional	
ID	FR08
Nombre	Asignar post a usuarios
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador asignar el post a usuarios.
Precondiciones	Ninguna
Actor principal	Administrador Wordpress
Secuencia normal	1. El actor crea o edita un post. 2. El actor selecciona los usuarios a los que quiere asignar el permiso en el *metabox*. 3. El actor le da a “Publicar” o “Actualizar” el post. 4. La aplicación *backend* gestiona la asignación de permisos.
Postcondición	Se ejecuta el FR10
Comentarios	-

Tabla 10: Requisito funcional FR08

5.2.9. FR09 - Asignar post a roles de usuario

Requisito funcional	
ID	FR09
Nombre	Asignar post a roles de usuario

Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador asignar el post a roles de usuario.
Precondiciones	Ninguna
Actor principal	Administrador Wordpress
Secuencia normal	1. El actor crea o edita un post. 2. El actor selecciona los roles de usuario a los que quiere asignar el permiso en el metabox. 3. El actor le da a “Publicar” o “Actualizar” el post. 4. La aplicación *backend* gestiona la asignación de permisos.
Postcondición	Se ejecuta el FR10
Comentarios	El rol es más restrictivo que el usuario. Si un rol de usuario está seleccionado, automáticamente todos los miembros de ese rol tendrán acceso al post, aunque en la sección de usuarios no estén marcados.

Tabla 11: Requisito funcional FR09

5.2.10. FR10 - Notificar usuarios afectados

Requisito funcional	
ID	FR10
Nombre	Notificar usuarios afectados
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al administrador asignar el post a roles de usuario.
Precondiciones	Ninguna
Actor principal	*Backend*
Secuencia normal	1. Se ejecuta FR08 0 FR09 2. La aplicación *backend* envía a todos los usuarios a los que se ha asignado el permiso un correo con la URL del post.

Postcondición	-
Comentarios	-

Tabla 12: Requisito funcional FR10

5.2.11. FR11 - Cargar contenido del usuario

Requisito funcional	
ID	FR11
Nombre	Cargar contenido del usuario
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de cargar el contenido personalizado del usuario.
Precondiciones	Ninguna
Actor principal	Usuario/Empleado
Secuencia normal	1. El actor inicia sesión 2. La aplicación backend carga el contenido personalizado.
Postcondición	Se cargará la página con el contenido del usuario.
Comentarios	El contenido variará en función de la configuración del plugin y de los post asignados al usuario y a sus grupos.

Tabla 13: Requisito funcional FR11

5.2.12. FR12 - Cargar carpeta FTP del usuario

Requisito funcional	
ID	FR12
Nombre	Cargar carpeta FTP del usuario

Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de cargar la carpeta FTP del usuario.
Precondiciones	La opción “Habilitar zona FTP” del formulario de configuración tiene que estar activada.
Actor principal	*Backend*
Secuencia normal	1. El actor inicia sesión. 2. El sistema *backend* carga la carpeta FTP del usuario.
Postcondición	Se cargará la página con el contenido del usuario y la carpeta FTP
Comentarios	Si la opción “Habilitar zona FTP” está desactivada, el proceso será como en FR11.

Tabla 14: Requisito funcional FR12

5.2.13. FR13 - Ver archivos de la carpeta FTP

Requisito funcional	
ID	FR13
Nombre	Ver archivos de la carpeta FTP
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al usuario visualizar los documentos de su carpeta FTP.
Precondiciones	La opción “Habilitar zona FTP” del formulario de configuración tiene que estar activada.
Actor principal	Usuario/Empleado
Secuencia normal	1. El actor inicia sesión. 2. El actor selecciona la pestaña FTP. 3. El actor navega por el sistema de ficheros. 4. El actor selecciona el documento que quiere ver. 5. Se abre una pestaña con el documento.

Postcondición	-
Comentarios	-

Tabla 15: Requisito funcional FR13

5.2.14. FR14 - Descargar archivos de la carpeta FTP

Requisito funcional	
ID	FR14
Nombre	Descargar archivos de la carpeta FTP
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al usuario seleccionar y descargar documentos de su carpeta FTP.
Precondiciones	La opción “Habilitar zona FTP” del formulario de configuración tiene que estar activada.
Actor principal	Usuario/Empleado
Secuencia normal	1. El actor inicia sesión. 2. El actor selecciona la pestaña FTP. 3. El actor navega por el sistema de ficheros. 4. El actor marca los documentos y carpetas que quiere descargar. 5. El actor le da a descargar.
Postcondición	Se descarga un zip en el ordenador del actor con el contenido que ha seleccionado.
Comentarios	-

Tabla 16: Requisito funcional FR14

5.2.15. FR15 - Visualizar posts asignados

Requisito funcional

ID	FR15
Nombre	Visualizar posts asignados
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al usuario visualizar los posts que se le han asignado.
Precondiciones	-
Actor principal	Usuario/Empleado
Secuencia normal	1. El actor inicia sesión. 2. El actor selecciona pestaña de categoría de posts que quiere ver. 3. El actor selecciona el post que quiere ver.
Postcondición	Se abrirá el post.
Comentarios	Los post se podrán visualizar en las diferentes categorías dependiendo de las categorías que tenga asignadas el usuario.

Tabla 17: Requisito funcional FR15

5.2.16. FR16 - Editar datos personales

Requisito funcional	
ID	FR16
Nombre	Editar datos personales
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al actor editar sus datos.
Precondiciones	-
Actor principal	Usuario/Empleado
Secuencia normal	1. El actor inicia sesión. 2. El actor selecciona el botón editar datos personales.
Postcondición	Se abrirá la página por defecto de wordpress para que un usuario pueda cambiar sus datos.

Comentarios	Esta función únicamente tiene que redirigir a la página de wordpress que te permite hacer esto pues la funcionalidad ya está implementada.

Tabla 18: Requisito funcional FR16

5.2.17. FR17 - Cerrar sesión

Requisito funcional	
ID	FR17
Nombre	Cerrar sesión
Versión	1.1
Autor	David Ribes
Descripción	El sistema ha de permitir al actor cerrar sesión.
Precondiciones	-
Actor principal	Usuario/Empleado
Secuencia normal	1. El actor navega por su página de usuario. 2. El actor selecciona el botón de cerrar sesión. 3. La aplicación *backend* termina con la sesión.
Postcondición	Se redirigirá a la página de inicio de la web.
Comentarios	-

Tabla 19: Requisito funcional FR17

5.3. Requisitos de usabilidad del sistema

Además de los requisitos de funcionalidad, también se requieren ciertos requisitos en cuanto a la usabilidad del sistema.

5.3.1. UR01 - Interfaz de usuario

Requisito de uso	
ID	UR01
Nombre	Interfaz de usuario
Versión	1.1
Autor	David Ribes
Descripción	El sistema tiene que proporcionar una interfaz de usuario intuitiva para el front-end.
Precondiciones	-
Actor principal	Usuario/Empleado
Secuencia normal	1. El actor inicia sesión. 2. La aplicación backend carga la interfaz. 3. El usuario accede al contenido
Postcondición	-
Comentarios	Cualquier usuario tiene que poder acceder al contenido rápidamente y sin tener dudas independientemente de su nivel de conocimiento de informática.

Tabla 20: Requisito de uso UR01

5.3.2. UR02 - Interfaz de administrador

Requisito de uso	
ID	UR02
Nombre	Interfaz de administrador
Versión	1.1
Autor	David Ribes
Descripción	El sistema tiene que proporcionar una interfaz de administración sencilla y rápida

Precondiciones	-
Actor principal	Administrador del Wordpress
Secuencia normal	1. El actor accede a la administración del plugin 2. El actor configura sus necesidades
Postcondición	-
Comentarios	No se debe prestar especial atención a la estética de la interfaz pues el administrador que accede a ella tiene altos conocimientos de informática y se debe priorizar la rapidez con la que podrá realizar los cambios.

Tabla 21: Requisito de uso UR02

5.4. Requisitos de calidad del sistema

Finalmente, el sistema también tiene un requisito de calidad.

5.4.1. QR01 - Compatibilidad del sistema

Requisito de calidad	
ID	QR01
Nombre	Compatibilidad del sistema
Versión	1.1
Autor	David Ribes
Descripción	El sistema tiene que ser compatible con la versión del *core* del Wodrpress en el que se va a instalar.
Precondiciones	-
Actor principal	Backend
Postcondición	-
Comentarios	-

Tabla 22: Requisito de calidad UR01

5.5. Arquitectura del sistema

Cabe destacar que en este proyecto nos hemos tenido que adaptar a la arquitectura de Wordpress. Al no ser un proyecto independiente, puesto que es un plugin que únicamente funcionará con Wordpress pues interactúa con su código y está pensado para él exclusivamente, la arquitectura del proyecto ha tenido que adaptarse sobre la que ya había.

Wordpress es una aplicación Web que sirve como ejemplo perfecto de cómo distintas tecnologías vienen juntas para formar una aplicación web. Dispone de tres capas:

1. La Capa de la Base de datos, que en este caso es MySQL.
2. La capa de la Aplicación, que es el Wordpress en sí, y que está escrita en PHP y se encarga de muchas de las operaciones de lectura y escritura de datos a la vez que ofrece APIs para que los desarrolladores puedan aprovecharla aún más.
3. La capa de Presentación, que utiliza CSS y HTML.

Nuestro plugin se engancha al Wordpress mediante las APIs de la capa de aplicación, pudiendo añadir funcionalidades o modificaciones. Esto quiere decir que gracias a estas APIs podemos acceder y modificar cualquiera de las capas a nuestro gusto, interactuando con la base de datos y pudiendo crear y añadir páginas en la capa de presentación.

Si nos fijamos en la estructura de carpetas de la figura 7, que representa el sistema de ficheros de un Wordpress, podemos ver que nuestro nuevo plugin iría en la carpeta plugins de la carpeta wp-content, que es donde irían todos los plugins instalados y que junto el tema (carpeta themes) y la biblioteca multimedia (uploads) formarían la página web final.

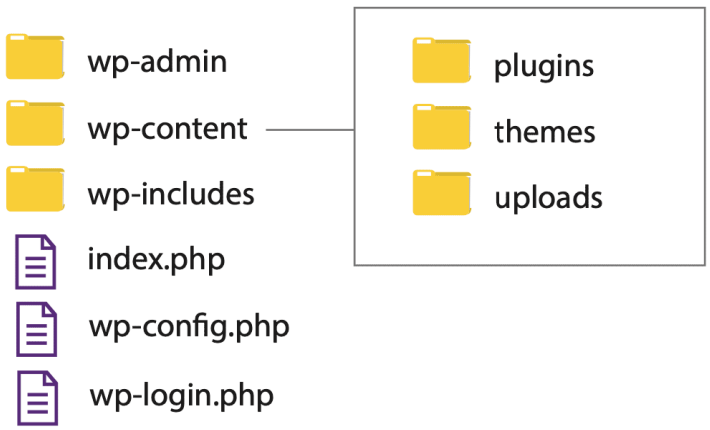

Figura 7: Estructura de carpetas de Wordpress [22]

Si nos centramos en la arquitectura del plugin, podemos apreciar en la figura 8 qué estructura tiene.

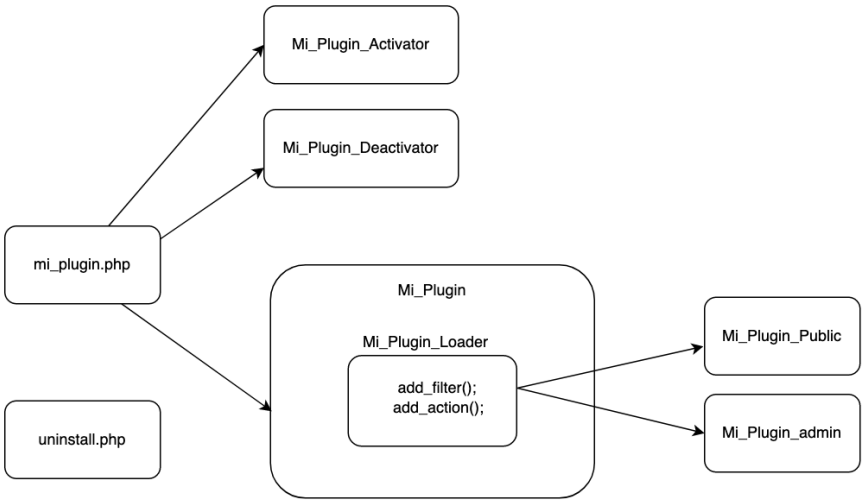

Figura 8: Arquitectura del plugin

Por un lado, tenemos Mi_Plugin_Activator y Mi_Plugin_Deactivator que harían referencia a las acciones que activan y desactivan el plugin.
Uninstall.php es el archivo que se ejecutaría cuando se desinstala el plugin. Esto quiere decir que cuando desactivas el plugin, se tiene que hacer cierta limpieza para no dejar rastros del mismo. Si el plugin ha creado tablas en la base de datos de Wordpress, hay que borrarlas, si el plugin ha creado páginas, hay que borrarlas. Es decir, dejar el Wordpress en el mismo estado que antes de instalarlo.

Después, tenemos los archivos del plugin que implementan todas las funciones y que surgen siempre a través de un filtro o un gancho (add_filter y add_action). Estas funcionalidades, pese a que se pueden clasificar de diferentes maneras, suelen separarse por las que actúan en la vista del administrador (Mi_Plugin_Admin) y las que actúan en el front-end final de la web (Mi_Plugin_Public).

5.6. Diseño de la interfaz

La interfaz del plugin la podemos diferenciar en cuatro partes. Las tres primeras, se encuentran en la administración del Wordpress. Para estas partes no se requería un diseño demasiado complejo, debía ser un diseño sencillo y útil pues las personas que lo van a utilizar tienen altos conocimientos de informática, ya que por lo general, serán empleados de IMK o responsables de la empresa de la web quienes lo administrarán.
La cuarts parte, es la interfaz front-end que tendrá el usuario/empleado final cuando inicie sesión. Para esta parte se ha intentado crear un diseño intuitivo y muy sencillo, que permita al usuario visualizar lo que quiere haciendo muy pocos clicks.

La primera parte, es la página de administración y configuración del plugin. Se encuentra en la administración del Wordpress y tal y como se aprecia en la siguiente figura, es un simple formulario de configuración.
Únicamente están las tres preguntas que se necesitan para poder configurar las partes, y además, si la opción de registro de usuarios está activa se despliega por Ajax un apartado desde donde se puede gestionar el alta y borrado de usuarios y de roles, así como la asignación de roles a los diferentes usuarios.

Esta es una de las grandes diferencias con los plugins de terceros que se venían utilizando hasta ahora, que se puede gestionar todo esto desde el mismo panel sin necesidad de ir combinando diferentes plugins.

En la siguiente figura (Figura 9), se puede apreciar el *Mockup* que se realizó para orientar cómo debía quedar la interfaz.

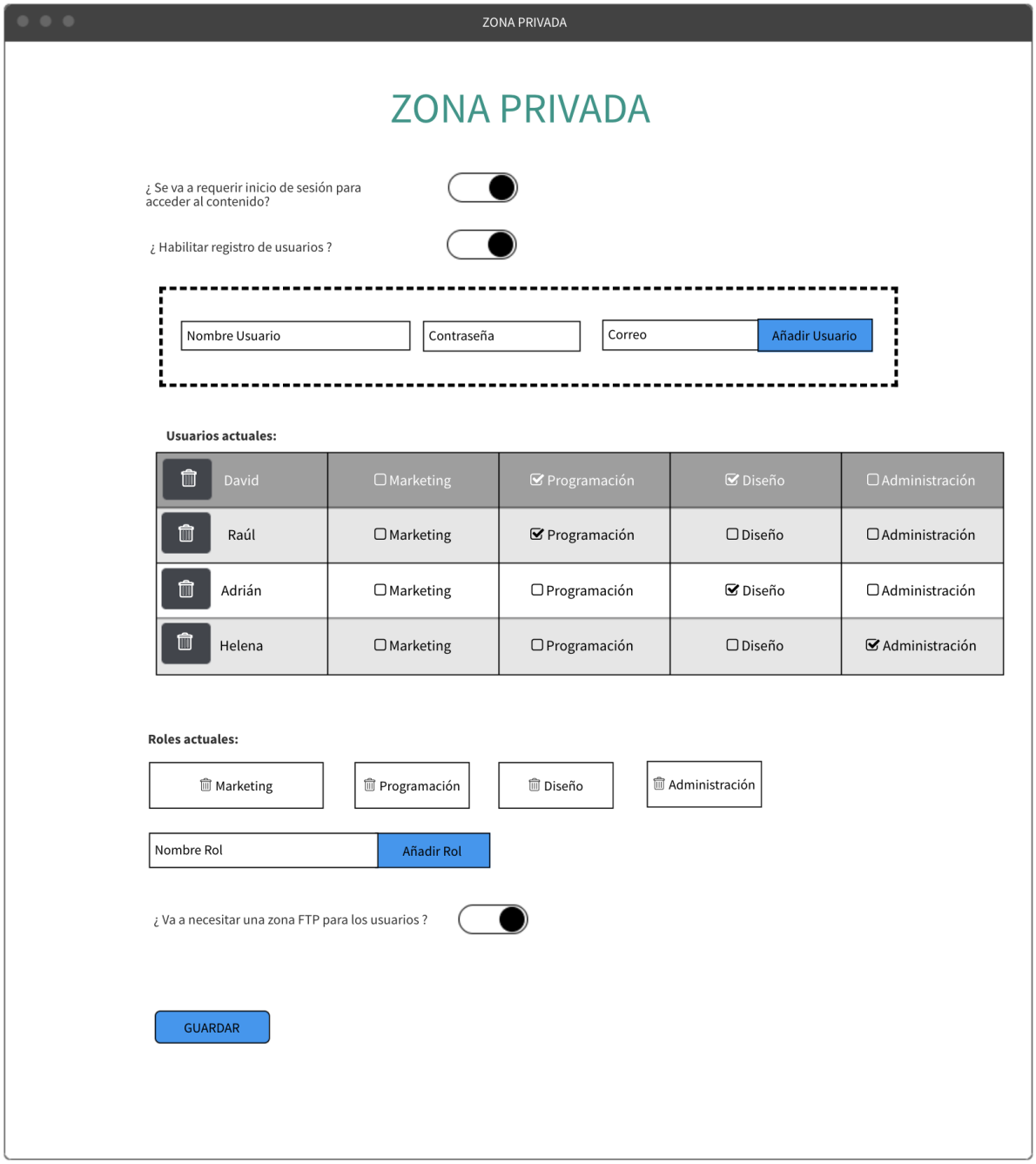

Figura 9: Mockup página configuración del plugin

La segunda pantalla, también se encuentra dentro de la administración del Wordpress y se trata del sistema de notificaciones. De nuevo no se necesita una interfaz demasiado bonita ni atractiva, sino útil, ya que será utilizada por la persona responsable de la web.

Únicamente consta de un pequeño filtro de usuarios desde el que se pretende poder seleccionar o bien de uno en uno o bien seleccionando un rol de usuario entero, de forma que los usuarios que se seleccionen serán quienes reciban la notificación en su correo electrónico.

Este sistema está pensado para poder enviar notificaciones a todos los roles de usuario concretos para informar por ejemplo de una reunión, o de los resultados semanales del departamento.
En la figura 10, se puede ver la idea de visualización que se hizo para la interfaz del sistema de notificaciones.

Figura 10: Mockup sistema de notificaciones

La última interfaz dentro de la administración de Wordpress no es una pantalla entera. Se trata únicamente de un *metabox* o contenedor que se incrusta en la pantalla de creación y edición de un *post* de Wordpress. Por lo tanto, aprovecha la interfaz de creación de *posts* y únicamente se tiene que introducir un contenedor que permita asignar los permisos del *post* y se lo transmita al backend.

Tal y como se aprecia en la siguiente imagen (Figura 11), la interfaz permite seleccionar permisos a nivel de grupo y a nivel individual. El usuario únicamente tendrá que marcar las opciones que interese para el post que se está editando y el sistema hará el resto.

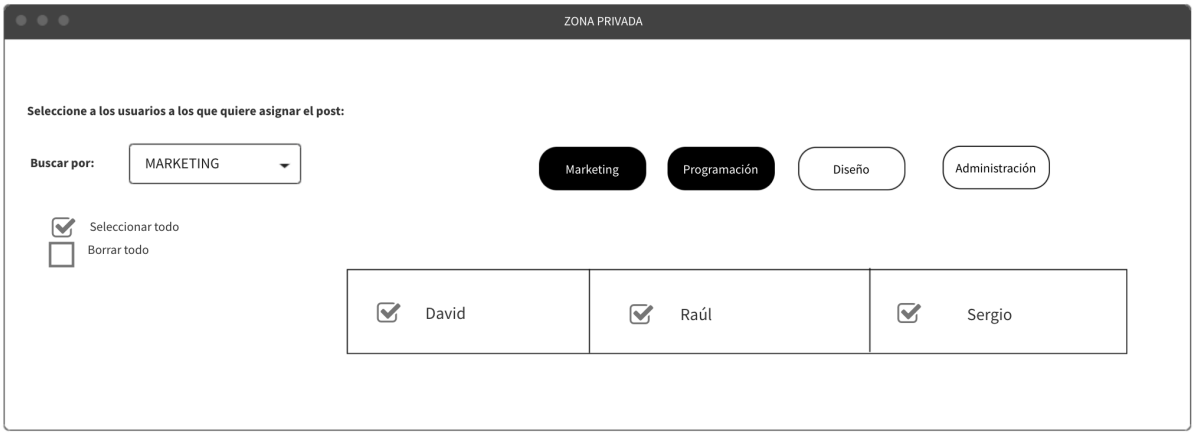

Figura 11: Mockup contenedor de asignación de permisos a *posts*

Finalmente, tenemos la pantalla de usuario. En esta interfaz si que se ha tenido más cuidado a la hora de diseñarla porque es la que está en el front-end de la web y tendrán acceso a ella todo tipo de usuario con diferentes conocimientos de informática, y se necesita que todos ellos puedan acceder al contenido de forma sencilla, intuitiva y rápida.

Simplemente se ha hecho una página con el nombre del usuario y justo bajo dos botones que te permiten cerrar la sesión o ir a la página por defecto de Wordpress para modificar tus datos.

A continuación, se muestran por orden:

Contenido personal - Contenido del Rol - Carpeta personal FTP

Son sencillas anclas que en cuanto las seleccionas cargan bajo el contenido correspondiente vía Ajax. Vamos a suponer un ejemplo para un usuario. El usuario, pertenece al rol de usuario “Marketing”. Por un lado, saldrá su contenido personal o sus “Novedades”. Este nombre, se podrá cambiar por otros en función del tipo de cliente. Por otro lado, el contenido de su tipo de usuario “Marketing” y finalmente su carpeta FTP. En la siguiente figura (Figura 12) se aprecia este ejemplo.

Figura 12: Mockup página front-end del usuario

En este ejemplo, el usuario tiene activa en la configuración la opción de FTP y solamente tiene asignado un rol de usuario, pero en caso de tener más, aparecería con otro icono junto con la imagen anterior. En caso de no tener activa la opción de FTP, la opción no aparecerá.

La idea es que conforme se cambie de opción seleccionada, el contenido de abajo cambie con Ajax.

5.7. Sitemap

Con el objetivo de que se entienda mejor cómo encaja el plugin en la arquitectura de Wordpress, se ha diseñado un sitemap. En el sitemap que se observa en la figura 13, se contemplan en color azul los elementos que vienen por defecto con Wordpress (no todos, solamente aquellos en los que nuestro plugin actúa o se consideran necesarios para entender el contexto) y en verde aquellos elementos y funcionalidades creados a partir del plugin.

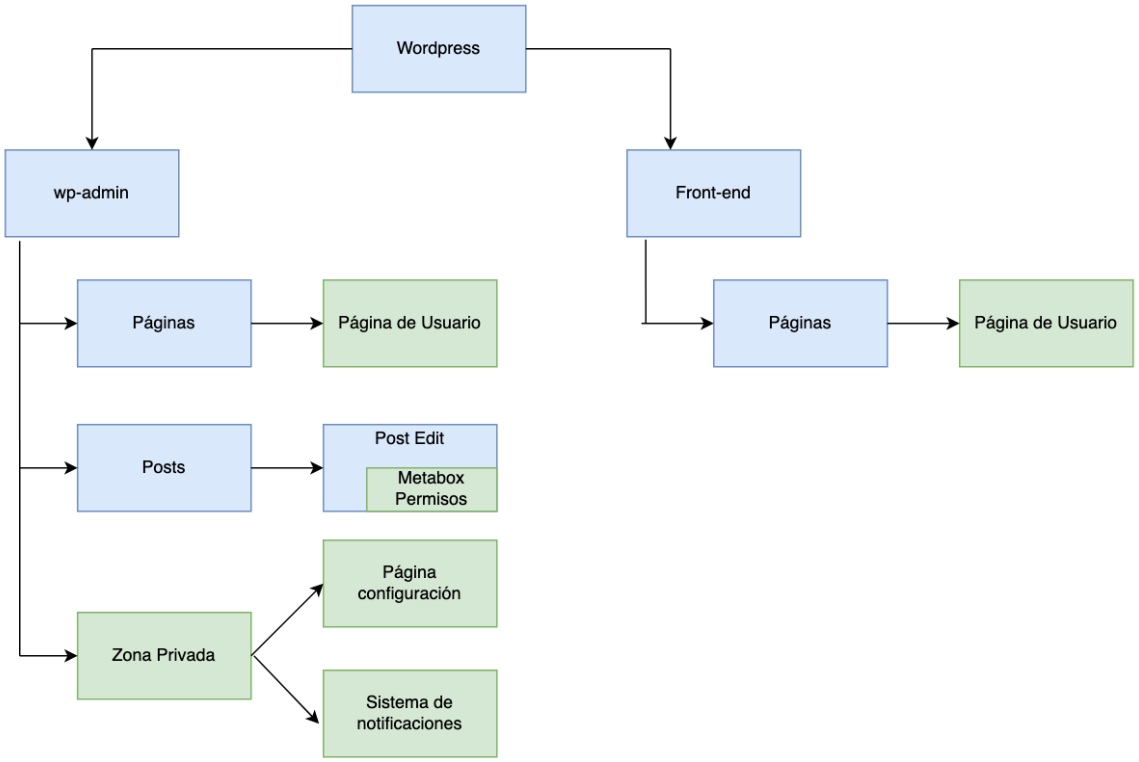

Figura 13: Sitemap del plugin

Se observa que el la página web se puede dividir en dos partes: El front end, que será la visualización normal de la web a través de un navegador, y la administración, que es donde está todo el panel de administración del Wordpress y que se accede añadiendo wp-admin al final de la url del dominio de tu sitio.

En el panel de administración, aparecen todos los elementos por defecto de los Wordpress (páginas, posts, ajustes, herramientas…). En este panel es donde aparecerá nuestro plugin de zona privada, tanto la página de configuración como el sistema de notificaciones.
Además, en las páginas de Wordpress, será necesario crear una página que será la que posteriormente mostrará el contenido de cada usuario en el front-end.
Finalmente, en la sección por defecto de *posts* o entradas, si entramos a crear o editar cualquier post, tendrá que aparecer una sección o *metabox* que nos permita asignar permisos para ese post.

En cuanto a la parte front-end, únicamente será necesario mostrar la página creada con anterioridad en la administración. El lugar de la web en el que se coloque dependerá de la estructura de la página web y de las necesidades del cliente.

Capítulo 6

Implementación y pruebas

En este capítulo se detallarán las estrategias seguidas para el desarrollo de cada funcionalidad del plugin, así como los problemas surgidos y la solución aplicada.

6.1. Situación inicial

Para poder empezar la implementación del plugin, únicamente se necesita la instalación de un Wordpress en un servidor desde el cual se tenga acceso tanto a los ficheros de configuración como a la base de datos. A partir de aquí, se puede empezar con la preparación del sistema de ficheros básico del plugin y realizar la instalación del mismo en el Wordpress. Una vez instalado, se pueden empezar a implementar las funcionalidades.

En este caso, para poder desarrollar el plugin se creó un subdominio del dominio de la empresa desde el que se desarrollan normalmente los proyectos (imkclientes.es). Se creó un hosting para este subdominio en uno de los servidores y en él se instaló Wordpress.

La forma en la que se pueden editar los ficheros dependerá de la persona que los edite. Para este proyecto se ha hecho vía FileZilla.

6.2. Detalles de implementación

Tal y como se ha explicado en el Capítulo 3, para la implementación del plugin se han utilizado varias tecnologías. El hecho de que un plugin sea un complemento para Wordpress, limita mucho a la hora de elegir lenguajes de programación puesto que está implementado con PHP y por lo tanto todas las funcionalidades de backend se tienen que desarrollar con este lenguaje.

Tal y como acabo de comentar, para realizar toda la implementación que afecta tanto al backend como a la página de administración, se ha utilizado PHP.
Además, para mostrar los diferentes elementos se ha utilizado HTML junto con CSS para dotarlos de estilos.
La funcionalidad de los distintos elementos de la página a nivel del cliente se han implementado usando Javascript junto con su framework jQuery. Además, para la comunicación con el servidor sin tener que someter a la página a una recarga se ha usado la tecnología Ajax.

En cuanto al patrón de diseño, no se ha seguido ningún patrón en concreto. Más bien se ha tratado de separar la implementación de los diferentes componentes del sistema y simplemente conectarlos entre ellos.

Estos componentes, se han separado según la parte del Wordpress en la que actúan y la finalidad que tienen:

1. Sistema de notificaciones
2. Página de configuración del plugin
3. Metabox para la asignación de permisos en posts
4. Sistema FTP
5. Página front-end de usuario

En estos cinco elementos se ha separado la implementación. Los tres primeros puntos son independientes entre ellos por lo que se han podido implementar e ir haciendo pruebas sin depender de otros puntos. El cuarto y quinto punto, depende de que las demás implementaciones estén terminadas y funcionando correctamente pues simplemente muestra toda la información asociada a un usuario determinado.

Entrando en detalle de cada punto, en el primer punto referido al sistema de notificaciones se optó por crear una página muy sencilla completamente con PHP y HTML. Básicamente lo que hace es listar todos los usuarios con un checkbox para que el administrador del wordpress pueda seleccionar a quienes quiere mandar el correo. Además, se puede filtrar por su rol de usuario. También se facilitó un botón de seleccionar todos para no tener que marcar uno a uno en caso de haber muchos usuarios, o poder mandar directamente a todo un departamento (rol).

En este apartado surgieron las primeras dudas en la implementación. Las dudas surgieron en cuál era la mejor forma de realizar la selección de usuarios y que no se perdieran las selecciones cuando se cambiaba el filtro del rol. Finalmente se optó por utilizar jQuery, de forma que simplemente ocultaba o mostraba los dispositivos mediante .show() o .hide() en función del filtro activo, sin perderse la selección que tenían hecha. Gracias a esto, pude conservar los usuarios marcados aun cambiando el filtro, pudiendo enviar así la notificación a varios departamentos a la vez.

En la figura 14, se aprecia la interfaz final de este apartado.

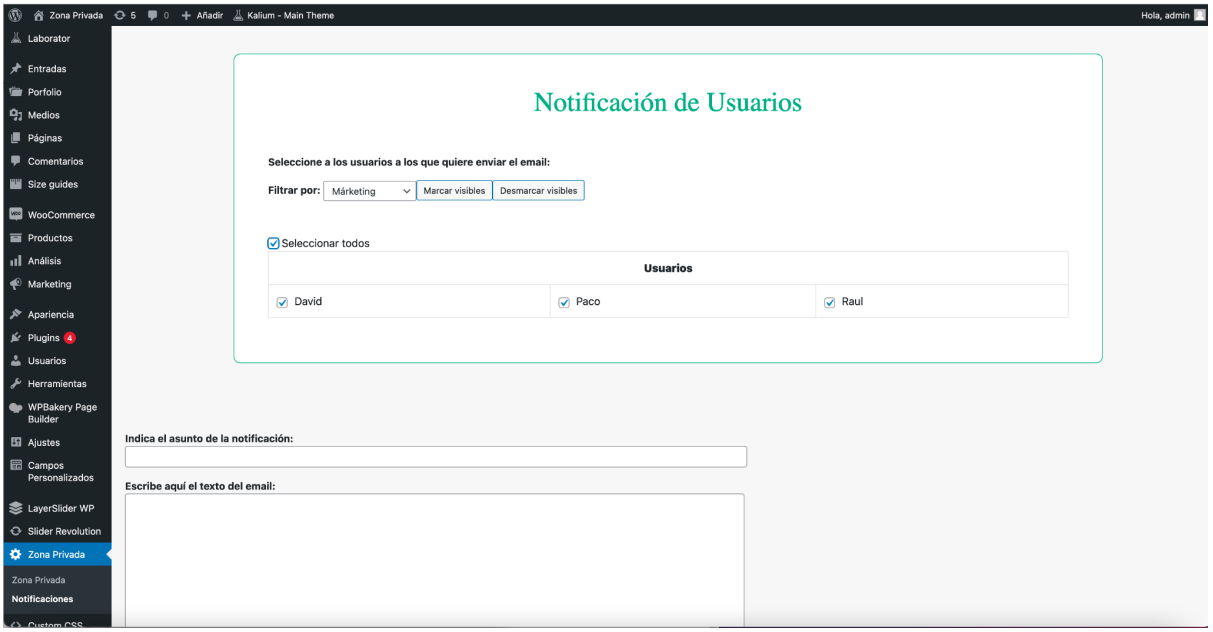

Figura 14: Sistema de notificaciones

En el segundo punto, que se refiere a la página de configuración del plugin, se implementó un simple formulario que almacenaba en la base de datos la configuración para que pudiera ser consultada por otros puntos del sistema.
Algunos de los puntos, requerían el uso de la tecnología Ajax puesto que requería una visualización u otra en función de si estaba activo o no. Es el caso del punto de registro de usuarios, que en caso de estar activo tenía que listar todos los usuarios junto con sus roles y no era muy eficiente tener que recargar la página para mostrarlos.

Una vez conseguido que funcionara el Ajax listar los usuarios fue muy fácil. Utilizando las funciones de la biblioteca de wordpress se podían conseguir listar fácilmente y únicamente tuve que ingeniarmelas para listarlos de la forma más adecuada posible.

Las complicaciones vinieron a la hora de automatizar la creación y borrado de usuarios. Wordpress permite hacer esto fácilmente desde el plugin por defecto que traen todos los wordpress, pero interesaba poder hacerlo de forma personalizada desde dentro del plugin añadiendo únicamente los campos que nos serían útiles más adelante.
Puesto que el plugin está pensado para uso de diseñadores y programadores de IMK con alto conocimiento del funcionamiento de Wordpress, bastó con crear un simple formulario para introducir un nombre, una contraseña y un correo, y no fué necesario implementar las típicas medidas de seguridad como pedir un formato concreto en la contraseña o tener que introducirla dos veces. Puesto que los usuarios se crearán exactamente igual que si se hubiera hecho con las opciones por defecto de wordpress, se podrá realizar posteriormente cualquier modificación de estos aspectos por parte del usuario.

El siguiente paso fue habilitar la creación y el borrado de roles, así como poder asignarlos a los diferentes usuarios. Se decidió que la mejor forma de hacerlo era aprovechando el listado realizado para los usuarios de manera que se apreciara cada usuario qué roles tenía marcados.
Además, añadir un nuevo campo para poder crear un rol nuevo. Aquí se hizo la comprobación de que si el nombre de rol ya existía, saltara un aviso.

En este apartado hubo complicaciones al borrar un rol. El problema que había era que cuando se borraba un rol, si posteriormente se volvía a crear con el mismo nombre, automáticamente se asignaba el rol a los usuarios que lo tenían con anterioridad.
Descubrí que era porque al borrar un rol, se borraba de la tabla de roles pero en la tabla de los usuarios seguía teniendo el rol asignado aunque no existiera. Entonces, al volver a crearlo, conservaba la memoria.
La solución que implementé fue que al borrar un rol, recorriera todas las tablas de usuario y eliminara la asignación de ese rol si lo tenía asignado.

En la figura 15 se puede apreciar la interfaz final de la página de configuración.

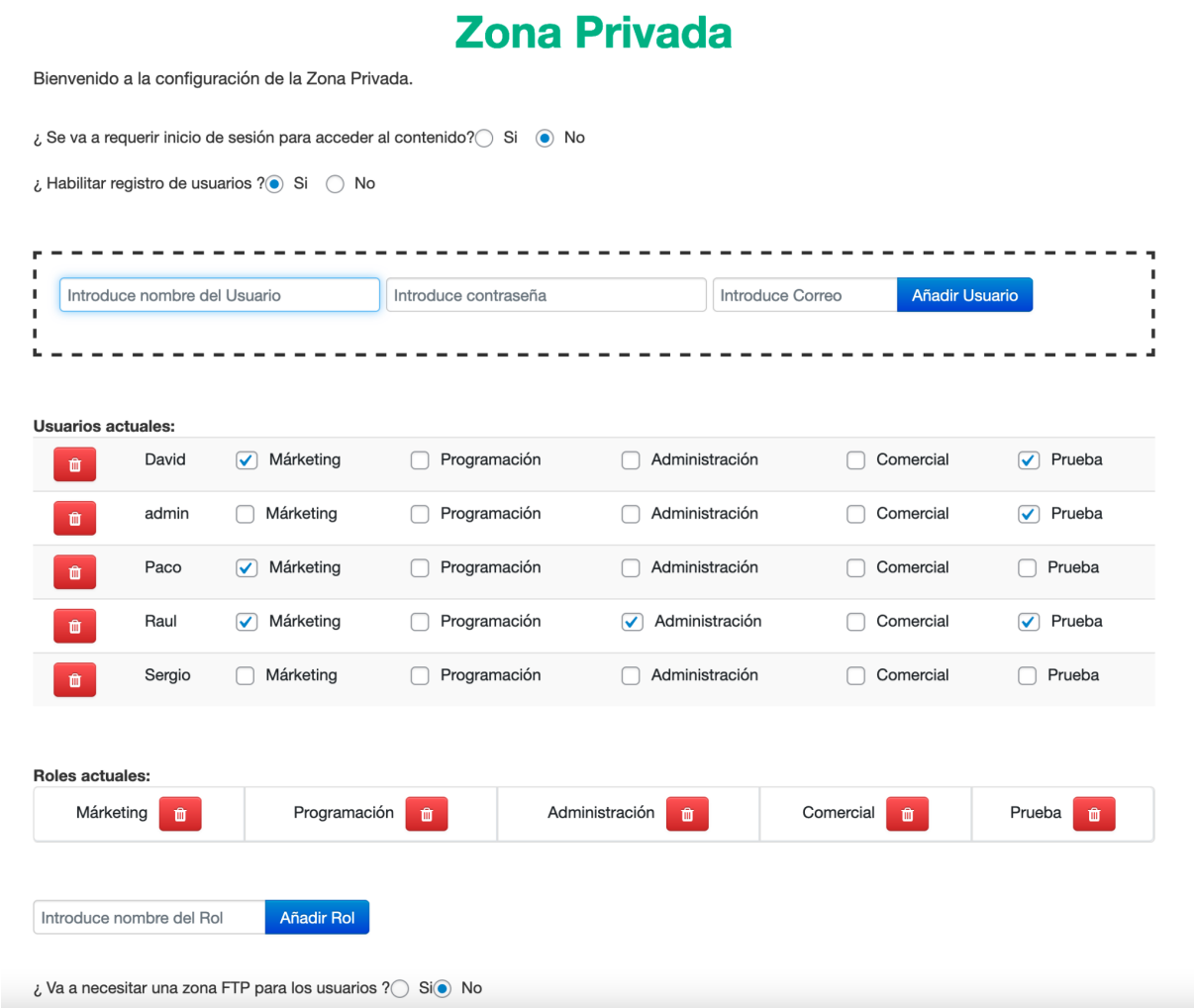

Figura 15: Página de configuración del plugin

Una vez tenía implementados los dos primeros puntos, pude empezar con el tercer punto. Este punto consistía en añadir una zona en la página de edición de un post que permitiera asignar permisos de ese post a los diferentes usuarios y roles.

El primer paso fue sencillo, simplemente tuve que añadir una acción que creara un metabox en la página de edición de los post. Una vez logrado esto, tuve que listar los usuarios y los roles de una forma parecida a como lo hice en el apartado de notificaciones, aunque de forma más concentrada puesto que el metabox era mucho más pequeño.

Las complicaciones vinieron a la hora de asignar los permisos de acceso. Tras varias pruebas e intentos, se llegó a la conclusión de que la mejor forma de hacerlo era almacenar en la tabla del post en la base de datos el id de los usuarios y roles que se habían marcado.

Se le dio prioridad al rol, de forma que si se marcaba un rol de usuario automáticamente todos los usuarios de ese rol tendrían permiso, aún quitando manualmente el permiso a ese usuario. En caso de querer solamente marcar algunos usuarios, bastaba con no marcar ningún rol y seleccionar el usuario concreto.

Cuando pude comprobar que se asignaban correctamente los permisos del post y que se almacenaba en la base de datos, se añadió una funcionalidad extra en la creación del post. La idea era que , en el momento en el que se crea el post, todas las personas a las que se le había asignado permiso para ver el contenido, recibieran un correo electrónico con el enlace a la noticia de interés.

Esto fue relativamente fácil de implementar. Únicamente tuve que engancharme a la acción que guardaba el post para que, en el momento en el que se hiciera la publicación del post enviara el correo a los afectados.

Puesto que se seleccionaba el id del usuario o rol de usuario afectado, mediante el id pude sacar fácilmente su correo e ir añadiéndolos en un array, y mediante PHP enviar un correo a todos estos usuarios indicando el nombre y el enlace del post.

Surgió un problema que me costó encontrar el fallo. Algunas veces, en el correo que llegaba al usuario llegaba un enlace a la noticia erróneo. Tras comentarlo con algunos compañeros, nos dimos cuenta que esto ocurría cuando en la publicación del post, en vez de darle a “actualizar” o a “publicar” se daba a la opción “vista previa” que ofrece wordpress para previsualizar el contenido antes de publicarlo. Ésto es porque el post todavía no estaba publicado pero igualmente intentaba generarse el enlace y se enviaba el correo igualmente. Finalmente decidimos que el correo únicamente se enviara cuando el botón que se seleccionaba era el de “publicar” .

Con este apartado funcionando, pude ponerme con los apartados 4 y 5.
Primero que nada, empecé con el desarrollo de la zona FTP. La idea era crear un *shortcode* que si la opción estaba activada en la página de configuración, se pusiera al final de la página de usuario y cargara contenido personalizado.

A cada usuario debía aparecerle automáticamente una carpeta personal donde la empresa podía compartirle cualquier tipo de documento personal y confidencial, y pudiera descargarlo. Algo parecido a una carpeta compartida en drive, donde si la empresa añade desde el FTP cualquier contenido a su carpeta, bien sean más carpetas o bien sean documentos, al usuario debía aparecerle.

El primer paso era automatizar la creación de la carpeta de usuario. Para ello, creé una acción que en el momento en el que se crea un usuario, coge su id y crea una carpeta llamada con el id+nombre y la metía en una carpeta donde estarían las carpetas de cada usuario llamada usuarios_activos. Ésta es la carpeta que se cargaría en la página del usuario junto con el resto del contenido en caso de estar la opción activa en la configuración del plugin.

Lo siguiente era mostrar el contenido de la carpeta en la página front-end del usuario. Esto no fue muy complicado ya que mediante PHP se podía recorrer la carpeta del usuario fácilmente e ir mostrando de forma recursiva su contenido, y después crear un zip con los archivos que el usuario seleccione para su descarga.
Lo complicado fue hacer un sistema de seguridad que en caso de borrar un usuario, se creara un backup de su contenido por si alguna vez se volvía a crear no se perdiera su contenido. Esto fue petición explícita del director de proyecto ya que no podíamos permitir que si se borraba un usuario equivocado por error se perdiera su contenido.

Lo que hice fue engancharme al hook del borrado de usuario para que cuando un usuario se borre, previamente creara una carpeta duplicada con el contenido de la carpeta de usuario y se guardara en una carpeta llamada usuarios_eliminados. A continuación, tuve que implementar esto a la inversa. Creé una acción que se ejecuta cuando se crea un usuario, en la que comprueba si ese usuario existía con anterioridad, y en caso afirmativo coge la carpeta del backup y la mueve a la carpeta con todas las demás carpetas de usuario.

Con la zona FTP terminada, solo faltaba trabajar el front-end del usuario final. El usuario tenía que poder acceder a una página en la que se cargara todo su contenido (noticias personales, noticias de los roles a los que pertenece, archivos personales en el FTP...), además de las opciones habituales de poder modificar sus datos y poder cerrar sesión.

Sin duda alguna esta fue la parte más costosa y complicada de implementar. Se tenía que buscar la forma de que el contenido que se mostrara fuera diferente en función del tipo de usuario y de la configuración que se había hecho del plugin en la página de administración.

El primer paso, fue hacer que en el momento de la activación del Wordpress automáticamente se creara una página con el nombre de “zona-privada” . En ésta página sería en la que se mostrará todo el contenido personalizado tras el en inicio de sesión.

Puesto que el inicio de sesión se maneja con otros plugins habituales y no forma parte de este proyecto el tener que desarrollarlo, lo único que había que indicarle al sistema era que tras el inicio de sesión te redireccionase a ésta página. Y después, mediante programación se cargaría el contenido.

El contenido se iba cargando mediante *shortcodes* en función de lo que se quería mostrar. La página contiene los shortcodes de los botones de modificación y cierre de sesión, otro para mostrar las noticias que tiene asignadas y finalmente otro para la zona de archivos FTP personales. Éste último shortcode solo se carga en función de si tiene esta opción activa en la configuración del plugin.

Lo primero era crear los dos botones de modificación de datos y cierre de sesión. Puesto que la labor de un programador es la de aprovechar al máximo los recursos existentes, decidimos aprovechar para esto las opciones por defecto que trae Wordpress para la gestión de usuarios, por lo que en el botón de modificar los datos únicamente pusimos un enlace a la página por defecto de wordpress en la que un usuario puede modificar los campos, y lo mismo para el cierre de sesión.

Una vez estaba hecha la parte más simple, únicamente había que cargar el contenido.
Puesto que el contenido en su mayor parte son post, únicamente teníamos que averiguar el id del usuario para ver qué noticias tiene asignadas. El id podía averiguarse fácilmente mediante las funciones de la librería de wordpress que te permiten interactuar con la sesión. Una vez obtenido el id, bastaba con recorrer en la base de datos las tablas de cada post y ver si tenía asignado ese id. En caso de tenerlo, guardaba el id del post en un array. Una vez los tenía todos, bastaba con clasificarlos en función de las necesidades del cliente (si se quiere mostrar por rol, por categorías…) e ir imprimiendolos en el código con la ayuda de las funciones de wordpress.

6.3. Verificación y validación

Por el tipo de proyecto del que se trata no se ha realizado un código de batería de pruebas. Las pruebas se han hecho de forma manual, y deben comprobar que se cumplen todos los requisitos establecidos.

El testeo del plugin se ha ido llevando a cabo durante todo el proyecto. Conforme se iban implementando las diferentes funcionalidades se iban haciendo las pruebas convenientes para comprobar que todo iba según lo previsto y que no había ningún comportamiento extraño.

Si bien es cierto que el plugin tiene muchas funcionalidades dependientes que no han podido probar hasta la finalización del mismo, se intentaba descomponer las tareas en subtareas muy pequeñas que sí que se podían ir probando.

La forma de realizar los testeos fue probando cada funcionalidad que se hacía al momento, y una gran prueba al final de cada uno de los cinco apartados que se han explicado en el punto anterior.

En el primer punto del sistema de notificaciones, las pruebas que se hicieron fueron muy simples. Se comprobó que se listaran todos los usuarios correctamente. Una vez se comprobó que se listaban todos los usuarios, se probó que el filtro funcionara bien y no sacara ningún usuario en un rol equivocado. Además, una de las pruebas más importantes fué comprobar que al cambiar de filtro no

se desmarcaran los usuarios que ya se habían seleccionado. Finalmente, se comprobó que el correo llegara únicamente a los usuarios que se habían seleccionado además de que no se dejara a ninguno.

Para el segundo punto, referido a la página de configuración del plugin, la comprobación principal fué que almacenara correctamente en la base de datos los datos que se configuraban, pues si no se guardaban correctamente cuando se refrescaba la página aparecía una configuración errónea.

Para ello, tenía abierto en otra pantalla la base de datos e iba comprobando que cada vez que guardaba un contenido, se actualizara correctamente en la base de datos.
Otra comprobación importante fué comprobar que en todo momento debía aparecer la misma configuración en la base de datos que en la página de configuración. Cualquier discordancia entre ambas indicaba un error bien en la inserción de datos o bien en la lectura. Para ello, cambié muchas veces la configuración en diferente orden, refrescaba, volvía a cambiar sin refrescar… etc. Todo esto siempre desde la propia página que había construido.

En caso de haber algún error que no conseguía localizar a simple vista, realizaba trazas en el código haciendo salir a pantalla datos mediante var_dump() o console.log() en función de donde se estaba ejecutando el código que quería depurar.

Para el tercer punto, que se refiere a la zona de asignación de permisos, únicamente tuve que comprobar que en el momento de actualizar el post se almacenaba correctamente en la base de datos los usuarios a los que había asignado el *post.* Además, igual que en el punto anterior, debía mostrar correctamente los usuarios que tenían el permiso en caso de volver a entrar a el post a hacer una actualización.

La siguiente prueba fué comprobar que llegara a los usuarios afectados un correo con el enlace de la noticia. En caso afirmativo, comprobaba que no se podía acceder a ella sin iniciar sesión o iniciando sesión desde un usuario sin permiso. Además, el enlace tenía que ser el correcto.

Al igual que antes, en caso de no encontrar algún fallo únicamente se depuraba el código.

Para la página front-end de usuario, se realizaron algunas pruebas más exhaustivas pues era la más compleja.

Primero que nada, se comprobaron las funcionalidades básicas (Redirecciona bien a la página de edición de usuario y se cierra correctamente la sesión).
La siguiente prueba fue comprobar que clasificaba bien las noticias, es decir, aparecían clasificadas en noticias personales y en su rol o roles asignados. Después, activaba y desactivaba la opción del FTP desde la página de configuración para comprobar que salía o dejaba de salir en función de la configuración.
Por último, comprobé que si una noticia se asignaba al usuario, le aparecía en la página front end, además de comprobar que no aparecía ninguna noticia a la que no tuviera permiso. Desde la administración, hice todas las combinaciones de pruebas posibles para comprobar que siempre se comportaba correctamente el sistema (Asignaba y quitaba permisos para comprobar que aparecía y

desaparecía la noticia, quitaba un rol de usuario y le asignaba otro… etc). Todo siempre desde la propia interfaz y comprobando la base de datos.

Finalmente, para el sistema FTP, las pruebas que hice fueron simples. Únicamente tuve que comprobar que en la página de cada usuario únicamente le apareciera su carpeta personal.
Y en cuanto a los backups, comprobé que cuando se creaba un usuario nuevo automáticamente se creara su carpeta en el FTP. Que si se borraba un usuario se duplicara su contenido en una carpeta de backup, y que si se creaba un usuario nuevo que existía con anterioridad cogiera su carpeta del backup y la pusiera con las demás carpetas de usuario. Todo esto lo iba comprobando con FileZilla.

Capítulo 7

Un caso real

En la actualidad, el plugin ya está funcionando en una web de un cliente real de IMK. En este capítulo, se mostrará este caso y se explicará como prácticamente no se ha tenido que adaptar nada.

7.1. El plugin en funcionamiento

Tras la finalización del proyecto se ha tardado poco en ponerlo en funcionamiento en un web real. El cliente necesitaba una zona privada con características muy similares a las del plugin desarrollado, por lo que fue una buena ocasión para probarlo.

El cliente es una asociación que necesitaba compartir con sus miembros documentos de diferentes categorías (Novedades, Circulares, Archivos particulares), además de compartirles de forma masiva las nóminas (sistema FTP).

En este caso, los roles del plugin se refieren al tipo de empresa asociada que es: Cooperativas, comercio privado, y gente de la oficina. Mediante el plugin, el administrador de asociex únicamente tiene que crear un post de wordpress y asignar a quién quiere compartirlo y de que categoría es, y en el front-end aparecerá automáticamente donde corresponde.

En las figuras 16.1 y 16.2 se aprecian capturas de pantalla del front-end de un usuario perteneciente al grupo “oficinas”. El usuario se creó a mi nombre para poder hacer comprobaciones mientras está en funcionamiento, pero se ve contenido real de cualquier miembro del mismo rol.

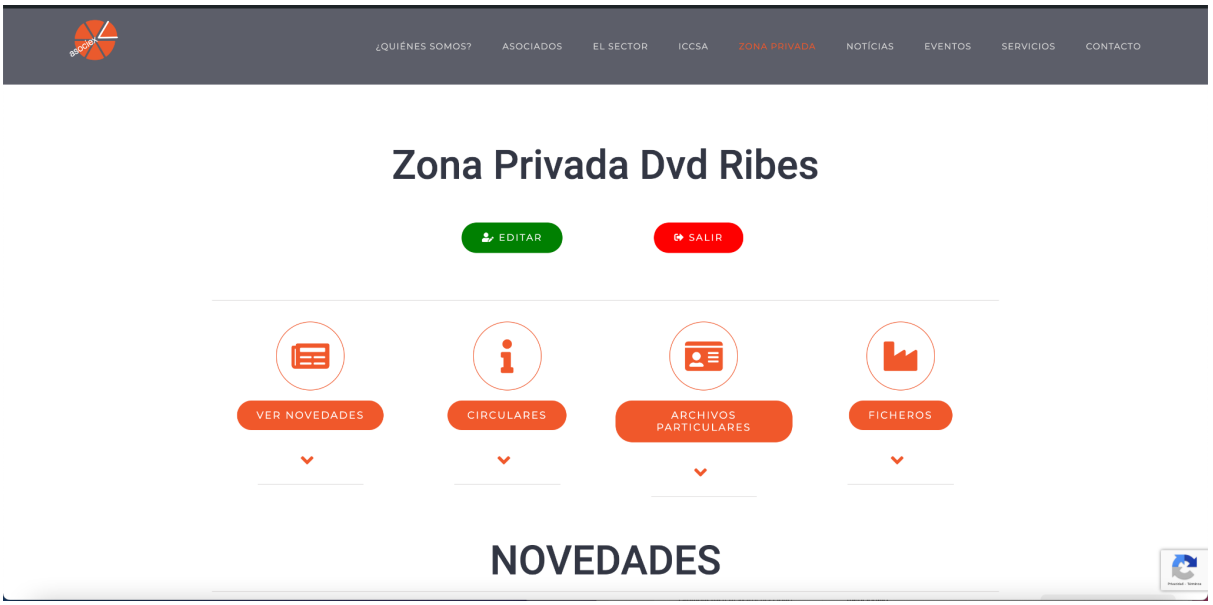

Figura 16.1: Ejemplo página front-end de usuario

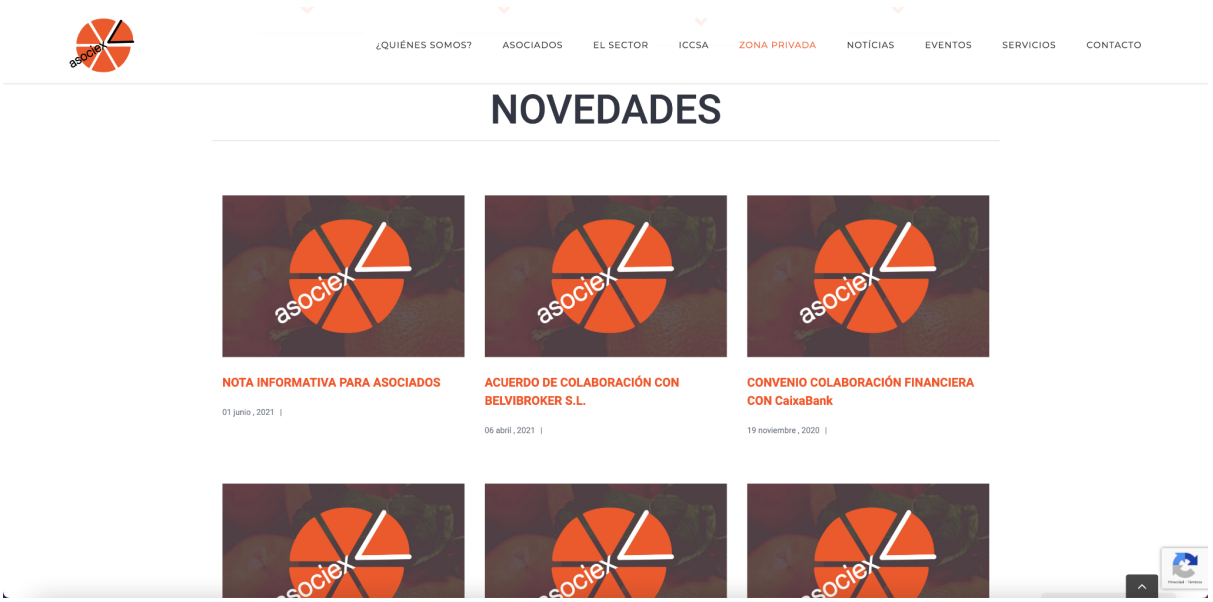

Figura 16.2: Ejemplo página front-end de usuario

Además, la siguiente imagen (Figura 17) muestra la zona de asignación de permisos en un post.

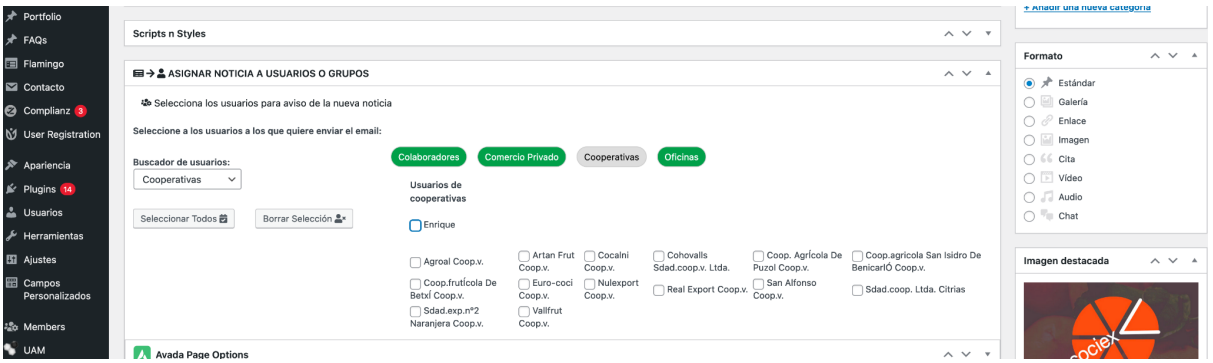

Figura 17: Ejemplo página edición de post

7.2. Resultados sobre la puesta en práctica

En general los resultados con esta puesta en práctica han sido bastante positivos.
Desde el punto de vista de IMK, se ha conseguido implementar toda la zona privada en una cantidad de tiempo muy por debajo de lo habitual y sin apenas necesitar la participación de ningún programador.
Todas las acciones implementadas se activaron correctamente con la instalación del plugin y no se tuvo ningún imprevisto.

El feedback del cliente también ha sido bastante positivo. Por lo general se han adaptado rápido al sistema , y únicamente han experimentado dificultades durante los primeros días con la asignación de los permisos a los posts. Pese a que han aprendido a utilizarlo correctamente, esto nos ha hecho replantearnos re-diseñar este apartado de cara al futuro para que sea un poco más intuitivo o más sencillo de comprender. El requisito de uso era que para la parte de administración se debía tener en cuenta solamente la eficacia y rapidez con la que se podía hacer una tarea, dando por hecho que la persona que administraría el blog tendría buenos conocimientos de informática, pero con el feedback ahora sabemos que habrá que re-plantearse este apartado.

A día de hoy, el plugin no ha experimentado ningún problema técnico, de incompatibilidad o de funcionalidad que haya requerido de una actualización crítica de la primera versión. Está previsto empezar a trabajar en una actualización en breve.

Capítulo 8

Conclusiones

En este capítulo, se expondrán las conclusiones técnicas a nivel técnico y a nivel personal. Además, se comentarán algunas de las mejoras que han ido surgiendo como idea a medida que iba avanzando el proyecto.

8.1. Conclusiones técnicas

El resultado del proyecto ha sido un éxito. Si recapitulamos sobre los objetivos que se plantearon al inicio del proyecto , se han ido cumpliendo todos con éxito:

Primero que nada, se ha conseguido reducir muchísimo el tiempo de producción de una página web de estas características, y se ha podido crear a través del plugin sin necesidad de recurrir a ninguno de los programadores, por lo que se ha dotado de independencia entre los departamentos. Además, puesto que ya está en funcionamiento y tenemos garantía de su funcionamiento, lo podemos ofrecer a otros clientes para atraerlo o como extra de lo que tienen contratado.
En cuanto a los objetivos orientados a la satisfacción del cliente, se le ha proporcionado con éxito las exigencias que se demandaban: un sistema desde el que poder compartir todo tipo de posts clasificados por roles de usuario y un sistema de notificaciones de correo desde el que poder notificar desde el mismo Wordpress a todos los usuarios.

Durante todo el desarrollo del proyecto no se han experimentado problemas críticos destacables, y las tareas que han superado la estimación temporal han sido compensadas rebajando el tiempo previsto para otras gracias a la experiencia en otras webs.

Tal y como se ha mostrado en el punto anterior, el plugin actualmente se encuentra en funcionamiento en una web, y se prevé que en los próximos meses se instale en varios clientes nuevos.

Aunque en sus primeras semanas el plugin ha funcionado correctamente, será durante los próximos meses cuando nosotros mismos obtendremos un *feedback* con posibles mejoras o incluso errores, ya que la mejor prueba posible es el uso del mismo.

8.2. Conclusiones personales

A nivel personal, puesto que ya llevaba más de un año trabajando en esta empresa, la experiencia no ha sido muy diferente a lo que venía siendo. Sí que es cierto que he tenido la oportunidad de trabajar en un producto interno de la empresa, algo que no es muy habitual puesto que generalmente siempre trabajamos para otros clientes. Esto me ha permitido hacer algo diferente a lo que acostumbro a hacer en la empresa, a la vez que lo he compaginado con mi trabajo habitual.

Creo que el hecho de no tener la presión de terminar el proyecto en un plazo concreto me ha ayudado mucho y me ha quitado mucha presión. El intercalar el proyecto con otras tareas, me ha permitido aprovechar cada rato que tenía destinado a implementar el plugin, y me ha permitido dar lo mejor de mí.

En cuanto al proyecto, considero que es un avance importante para el departamento en cuanto a productividad y agilización de las tareas. Además, tal y como se explica en el siguiente punto, creo que puede seguir creciendo e ir añadiendo muchas funcionalidades, y hacer que un primer producto simple se convierta en uno mucho más grande y elaborado.

Creo que el haber desarrollado el plugin para la empresa me convierte en un activo importante dentro de IMK, ya que se tiene previsión de seguir haciéndolo grande y de seguir usándolo en muchas webs, y al haberlo implementado yo tengo muchos puntos para ser la persona o una de las personas que sigan trabajando en él.

8.3. Posibles mejoras

Una de las grandes ventajas es que un proyecto así no tiene límites para seguir creciendo. Éste tipo de proyecto puede ser mejorado de muchas formas e incluso ir añadiendo cada vez más funcionalidades para ir haciéndolo cada vez mejor.

Durante el desarrollo del proyecto fueron surgiendo nuevas ideas que a fecha de hoy ya se están implementando como mejora. El hecho de usarlo en una web real nos confirmó que estas nuevas funcionalidades son necesarias.

La primera de ellas, es para el apartado de creación de posts o noticias.
Nos dimos cuenta de que habíamos desarrollado una función que cuando se creaba una nueva noticia avisaba enviándole un correo a todos los afectados, pero no habíamos pensado en qué ocurriría si había que hacer una modificación sobre esa noticia ya creada. Tal y como estaba implementado, tras la actualización del post, volvía a enviar la notificación a todos los afectados. Ésto no es nada óptimo, y lo que estamos haciendo es añadir un checkbox que permita desactivar la notificación automática tras crear un post, de forma que el creador sea quien decida si quiere que se envíen las notificaciones o no.

Además, también nos dimos cuenta de que si se añadía un usuario nuevo a la noticia, no se podía notificarle a él solamente sin notificar a los demás. La solución que pensamos y que se encuentra en desarrollo, es almacenar un histórico de usuarios a los que ya se ha enviado el correo con el enlace, de forma que cada vez que se actualice, mire si hay algún usuario nuevo y solamente se le notifique a él.

Además, tras varias reuniones, se está pensando de cara al futuro desarrollar también un sistema de gestión de vacaciones, en la que dentro de la página de usuarios el usuario pueda solicitar sus vacaciones , ver las que ya ha disfrutado y cuántas le quedan, y si las que ha solicitado se le han concedido o no.

Bibliografía

[1] IMK. Información de la empresa. https://www.imk.es/servicios/. [Consulta 19 de Julio de 2021]

[2] Dario BF. Información panel de administración de Wordpress. https://www.dariobf.com/panel-administracion-wordpress/#:~:text=El%20Panel%20de%20Administraci%C3%B3n%20es,ext%2Fwp%2Dadmin. [Consulta 19 de Julio de 2021]

[3] Instituto Cajasol. Información sobre qué son los Blog de Wordpress. https://institutocajasol.com/que-es-wordpress-y-como-funciona/ . [Consulta 19 de Julio de 2021]

[4] Raiola Networks. Información sobre los posts de Wordpress . https://raiolanetworks.es/blog/post-wordpress/ . [Consulta 19 de Julio de 2021]

[5] Profesional Hosting. ¿Qué es plesk? . https://www.profesionalhosting.com/soporte-en-linea/panel-de-control-plesk/qu-es-plesk.html . [Consulta 19 de Julio de 2021]

[6] Wikipedia . Filezilla https://es.wikipedia.org/wiki/FileZilla . [Consulta 19 de Julio de 2021]

[7] Web Empresa. Wordpress . https://www.webempresa.com/wordpress/que-es-wordpress.html. [Consulta 19 de Julio de 2021]

[8] Themeforest. Kalium Theme . https://themeforest.net/item/kalium-creative-theme-for-professionals/10860525 [Consulta 19 de Julio de 2021]

[9] PHP. Información sobre PHP. https://www.php.net/manual/es/intro-whatis.php. [Consulta 19 de Julio de 2021]

[10] Wikipedia. HTML. https://es.wikipedia.org/wiki/HTML. [Consulta 19 de Julio de 2021]

[11] Wikipedia. CSS. https://es.wikipedia.org/wiki/Hoja_de_estilos_en_cascada. [Consulta 19 de Julio de 2021]

[12] Wikipedia. Javascript. https://es.wikipedia.org/wiki/JavaScript). [Consulta 19 de Julio de 2021]

[13] Wikipedia. JQuery. https://es.wikipedia.org/wiki/JQuery [Consulta 19 de Julio de 2021]

[14] Wikipedia. Ajax. https://es.wikipedia.org/wiki/AJAX [Consulta 19 de Julio de 2021]

[15] Raiola Networks. Hooks de Wordpress. https://raiolanetworks.es/blog/hooks-wordpress/ [Consulta 19 de Julio de 2021]

[16] Web Empresa . Tipos de hooks. https://www.webempresa.com/blog/hooks-wordpress-como-utilizarlos-guia-rapida.html [Consulta 19 de Julio de 2021]

[17] Hookr. Listado de Hooks. http://hookr.io/all/ . [Consulta 19 de Julio de 2021]

[18] Wikipedia . MySQL. https://es.wikipedia.org/wiki/MySQL . [Consulta 19 de Julio de 2021]

[19] Tracking Time . Herramienta Tracking Time. https://trackingtime.co/ . [Consulta 19 de Julio de 2021]

[20] Slack . Información herramienta Slack. https://slack.com/intl/es-es/help/articles/115004071768-%C2%BFQu%C3%A9-es-Slack- . [Consulta 19 de Julio de 2021]

[21] Wikipedia. Scrum. https://es.wikipedia.org/wiki/Scrum_(desarrollo_de_ software). [Consulta 19 de Julio de 2021]

[22] Web Empresa. Principales archivos en la raíz. https://www.webempresa.com/blog/estructura-general-archivos-carpetas-wordpress.html . [Consulta 19 de Julio de 2021]